5Міністерство освіти і науки України
Західноукраїнський національний університет

**О.М.Березький, В.М.Теслюк, Л.О.Дубчак,
Г.М.Мельник, Ю.М.Батько**

ДОСЛІДЖЕННЯ І ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

Навчальний посібник

**Тернопіль
ЗУНУ
2022**

УДК
ББК
Б

Березький О.М. Дослідження і проектування компютерних систем та мереж: навч.посіб. / Березький О.М., Теслюк В.М., Дубчак Л.О., Мельник Г.М., Батько Ю.М. – Тернопіль: ЗУНУ, 2022. – 252 с.

Рецензенти:

Говорущенко Т.О., д.т.н., професор, завідувачка кафедри компютерної інженерії та інформаційних систем, Хмельницький національний університет;

Литвиненко Я.В., д.т.н., професор, кафедра комп'ютерних наук, Тернопільський національний технічний університет ім.Івана Пулюя.

Цмоць І.Г., д.т.н., професор, кафедра автоматизованих систем управління, Національний університет «Львівська політехніка».

Рекомендовано до видання Вченою радою
Західноукраїнського національного університету
Протокол №9 від 15.06.2022 р.

ЗМІСТ

ВСТУП

Широке впровадження цифрових технологій зумовило розповсюдження цифрових комп’ютерів різного призначення, обчислювальної потужності, вартості, масо-габаритних розмірів і т.п. Розвиток локальних мереж сприяв виникненню глобальної мережі – INTERNET. Розвиток інформаційних технологій породив штучний інтелект, який став основою шостої промислової революції.

Отже, комп’ютерні системи та комп’ютерні мережі стали основними обчислювальними засобами в цифровому світі людства. Тому моделювання, та дослідження відомих комп’ютерних систем та мереж і проектування та оптимізація нових є архіважливою проблемою для підготовки нових і підвищення кваліфікації навчання діючих ІТ-фахівців.

Автори посібника мають досвід у проектуванні комп’ютерних систем для задач комп’ютерного зору та спеціалізованих комп’ютерних інформаційних систем із елементами штучного інтелекту медичного спрямування. Так у посібнику [1] розглянуто проектування комп’ютерних систем на сучасній елементні базі - програмованих логічних інтегральних схемах.

У ряді робіт розроблено алгоритми обробки зображень на комп’ютерних системах з графічними процесорами [2 - 5]

Розробці компонентів нейроорієнтованих комп’ютерних систем для робо технічних систем присвячені такі роботи [7-11].

Окремі алгоритми та реальні комп’ютерні системи для опрацювання біомедичних зображень висвітлені у роботах [12 -17].

Задачі розробки апаратних засобів аналізу та синтезу зображень розглянуті в роботах [18-21].

Методи розподілу доступу та захисту інформації описані у роботах [22-25].

Розв’язання задач управління ІТ проектами висвітлено у матеріалах [26, 27].

Досвід практичних розробок різного застосування показав, що задачі проектування, моделювання комп’ютерних систем та мереж є багатокритеріальними. В даному посібнику описані математичні структури для розв’язання цих задач.

Перший розділ навчального посібника присвячений класифікації комп’ютерних систем та мереж.

Другий розділ посібника висвітлює питання теорії розв’язання задач багатокритеріальної оптимізації.

Третій розділ присвячено різним методам розв’язання багатокритеріальних задач на основі однокритеріальної оптимізації комп’ютерних систем та мереж. В даному розділі детально описано кожен метод, що дозволяє читачу зрозуміти різні підходи до розв’язання багатокритеріальних задач.

Четвертий розділ розглядає нормалізацію узагальненого критерію. Крім того, в даному розділі розкриті адитивний, мультиплікативний та максимінний (мінімаксний) критерії. У розділі описано також методи призначення вагових коефіцієнтів.

У п’ятому розділі описано сучасні підходи до розвязання задач багатокритеріальної оптимізації, зокрема з використанням множини Парето. В даному розділі авторами подано також приклади застосування такого підходу.

Шостий розділ присвячено основним поняттям моделювання та його параметрам, зокрема розглянуто види моделей, методи моделювання та рівні проектування в САПР.

У сьомому розділі описано загальні характеристики процесу математичного моделювання, а також класифікації, етапи та технології моделювання.

Восьмий розділ присвячений процесу побудови структурних моделей комп’ютерних систем та мереж на основі теорії мереж Петрі.

Дев’ятий розділ демонструє побудову структурних моделей комп’ютерних систем та мереж на основі теорії систем масового обслуговування. Крім того, в даному розділі авторами описані конкретні приклади розв’язання задач.

Десятий розділ присвячений авторським розробкам комп’ютерних систем та мереж та їх оптимізації.

Навчальний посібник «Дослідження і проектування комп’ютерних систем та мереж» буде корисним викладачам, студентам, аспірантам та науковцям галузі знань 12 «Інформаційні технології».

РОЗДІЛ 1. КОМП'ЮТЕРНІ СИСТЕМИ ТА МЕРЕЖІ

1.1 Класифікації сучасних комп'ютерних систем

На даний час існує дуже багато означень комп'ютерної системи. Наприклад, одним з них є наступне визначення. Комп'ютерна система (КС) – це будь-який пристрій або група взаємно поєднаних або пов'язаних пристроїв, один чи більше з яких, відповідно до певної програми, виконує автоматичну обробку даних [28].

Існує велика кількість ознак, за якими класифікують комп'ютерні системи: за цільовим призначенням та функціями, що вони виконують, типами та кількістю комп'ютерів або процесорів, архітектурою системи, режимами роботи, методами управління елементами системи, степенями роз'єднаності елементів комп'ютерної системи та ін.

За призначенням комп'ютерні системи поділяють на універсальні і спеціалізовані. Універсальні КС призначені для розв'язання широкого кола завдань, склад якого заздалегідь не визначений. Спеціалізовані КС орієнтовані на вирішення вузького класу задач.

За видом складових елементів комп'ютерні системи прийнято поділяти на багатокомп'ютерні і багатопроцесорні КС. Багатокомп'ютерна система (БКС) містить декілька комп'ютерів, кожен з яких має свою оперативну пам'ять і працює під керуванням своєї операційної системи.

Якщо в якості елементів КС використовуються процесори, то така КС належить до класу багатопроцесорних.

Залежно від ***типів комп'ютерів або процесорів***, з яких складається КС, розрізняють однорідні і неоднорідні системи. У складі однорідних систем – однотипні комп'ютери або процесори, у складі неоднорідних – різнотипні.

За ***характером просторового розподілу*** елементів КС діляться на системи зосередженого (локального) і розподіленого типів.

За ***методами управління елементами*** розрізняють КС централізовані, децентралізовані та зі змішаним керуванням.

За принципом закріплення обчислювальних функцій за окремими комп’ютерами або процесорами розрізняють системи з жорстким і плаваючим закріпленням функцій.

За режимом роботи розрізняють обчислювальні системи, що працюють в оперативному і неоперативному часових режимах.

При розробці компютерної систми метою є підвищення основних її характеристик:

– відношення вартість / продуктивність;
– надійність і відмовостійкість;
– масштабованість;
– сумісність і мобільність програмного забезпечення.

Для порівняння різних комп'ютерів між собою зазвичай використовуються стандартні методики вимірювання продуктивності. Ці методики дозволяють розробникам і користувачам використовувати отримані в результаті випробувань кількісні показники для оцінки тих чи інших технічних рішень, і саме продуктивність і вартість дають користувачеві раціональну основу для вирішення питання, який комп'ютер вибрати.

Найважливішою характеристикою обчислювальних систем є ***надійність***. Підвищення надійності базоване на принципі запобігання несправностей шляхом зниження інтенсивності відмов і збоїв за рахунок застосування електронних схем і компонентів з високим і надвисоким ступенем інтеграції, зниження рівня завад, полегшених режимів роботи схем, забезпечення теплових режимів їх роботи, а також за рахунок удосконалювання методів складання апаратури.

Відмовостійкість - це така властивість обчислювальної системи, що забезпечує їй, як логічній машині, можливість продовження дій, заданих програмою, після виникнення несправностей. Введення відмовостійкості

вимагає надлишкового апаратного та програмного забезпечення. Головною метою підвищення надійності систем є цілісність збережених у них даних.

Масштабованість являє собою можливість нарощування числа і потужності процесорів, обсягів оперативної і зовнішньої пам'яті та інших ресурсів обчислювальної системи. Масштабованість повинна забезпечуватися архітектурою і конструкцією комп'ютера, а також відповідними засобами програмного забезпечення.

Концепція програмної сумісності вперше в широких масштабах була застосована розробниками системи IBM/360. Основне завдання при проектуванні всього ряду моделей цієї системи полягала в створенні такої архітектури, яка була б однаковою з точки зору користувача для всіх моделей системи незалежно від ціни та продуктивності кожної з них. Перевагами такого підходу є збереження існуючого напрацювання програмного забезпечення при переході на нові моделі. З часом архітектура старіє і виникає потреба внесення радикальних змін в архітектуру і способи організації обчислювальних систем.

Одна з перших класифікацій комп'ютерних систем була запропонована Д. Шором на початку 70-х років. Вона цікава тим, що є спробою виділення типових способів компонування комп'ютерних систем на основі фіксованого числа базових блоків: пристрою керування, арифметико-логічного пристрою, пам'яті команд і пам'яті даних. Додатково передбачається, що вибірка з пам'яті даних може здійснюватися словами, тобто вибираються всі розряди одного слова, і/або бітовим шаром - по одному розряду з однієї і тієї ж позиції кожного слова (іноді ці два способи називають горизонтальною і вертикальною вибірками відповідно). Звичайно ж, при аналізі даної класифікації треба робити знижку на час і появи, оскільки передбачити велику різноманітність паралельних систем теперішнього часу тоді було у принципі неможливо. Отже, згідно з класифікацією Д. Шора, всі комп'ютери розбиваються на шість класів [29].

Одну з перших практично значимих класифікацій паралельних комп'ютерних систем подав у 1966 році співробітник фірми IBM Майкл Флін.

Його класифікація базується на оцінці потоку інформації, який поділено на потоки даних між основною пам'яттю та процесором, та потік команд, які виконує процесор. При цьому потік даних та команд може бути як одиничним, так і множинним. Згідно з М. Фліном, усі комп'ютерні системи поділяють так:

– ОКОД - комп'ютерні системи з одиничним потоком команд та одиничним потоком даних (SISD - Single Instruction Single Data stream);

– МКОД - комп'ютерні системи з множинним потоком команд та одиничним потоком даних (MISD - Multiply Instruction Single Data stream);

– ОКМД - комп'ютерні системи з одиничним потоком команд та множинним потоком даних (SIMD - Single Instruction Multiply Data stream);

– МКМД - комп'ютерні системи з множинним потоком команд та множинним потоком даних (MIMD - Multiply Instruction Multiply Data stream).

В 1978 році Д. Куком було запропоновано розширення класифікації Фліна. У своїй класифікації Д. Кук розділив потоки команд та даних на скалярні та векторні потоки. Комбінація цих потоків приводить в підсумку до 16 типів архітектури паралельних комп'ютерних систем.

Р. Хокні - відомий англійський фахівець в області паралельних обчислювальних систем, розробив свій підхід до класифікації, введеної ним для систематизації комп'ютерів, що потрапляють в клас MIMD по класифікації Фліна. Основна ідея класифікації полягає в наступному. Множинний потік команд може бути оброблений двома способами: або одним конвеєрним пристроєм обробки, що працює в режимі поділу часу для окремих потоків, або кожен потік обробляється своїм власним пристроєм. Перша можливість використовується в MIMD комп'ютерах, які називаються конвеєрними (наприклад, процесорні модулі в Denelcor HEP). Архітектури, використовують другу можливість, у свою чергу знову діляться на два класи:

– MIMD комп'ютери, в яких можливий прямий зв'язок кожного процесора з кожним, реалізований за допомогою перемикача;

– MIMD комп'ютери, в яких прямий зв'язок кожного процесора можливий тільки з найближчими сусідами по мережі, а взаємодія віддалених

процесорів підтримується спеціальною системою маршрутизації через процесори-посередники.

У 1989 році була зроблена чергова спроба розширити класифікацію Флінна і, тим самим, подолати її недоліки. Д. Скіллікорн розробив підхід, придатний для опису властивостей багатопроцесорних систем і деяких нетрадиційних архітектур, зокрема dataflow і reduction machine.

Пропонується розглядати архітектуру будь-якого комп'ютера, як абстрактну структуру, що складається з чотирьох компонентів:

– процесор команд (IP - Instruction Processor) - функціональний пристрій, що працює як інтерпретатор команд (в системі може бути відсутнім);

– процесор даних (DP - Data Processor) - функціональний пристрій, що працює як перетворювач даних, у відповідності з арифметичними операціями;

– ієрархія пам'яті (IM - Instruction Memory, DM - Data Memory) - запам'ятовуючий пристрій, в якому зберігаються дані та команди, що пересилаються між процесорами;

– перемикач - абстрактний пристрій, що забезпечує зв'язок між процесорами і пам'яттю.

Функції процесора команд багато в чому схожі з функціями пристроїв управління послідовних машин і, згідно Д. Скіллікорну, зводяться до наступних:

– на основі свого стану і отриманої від DP інформації IP визначає адресу команди, яка буде виконуватися наступна;

– здійснює доступ до IM для вибірки команди;

– отримує і декодує обрану команду;

– повідомляє DP команду, яку треба виконати;

– визначає адреси операндів і посилає їх в DP;

– отримує від DP інформацію про результат виконання команди.

У роботі Р. Дункан викладає свій погляд на проблему класифікації архітектур паралельних обчислювальних систем, причому відразу визначає той набір вимог, на який, з його точки зору, може спиратися шукана класифікація. З

класу паралельних машин повинні бути виключені ті, в яких паралелізм закладений лише на самому низькому рівні, включаючи [30]:

– конвеєризацію на етапі підготовки та виконання команди (instruction pipelining), тобто часткове перекриття таких етапів, як дешифрування команди, обчислення адрес операндів, вибірка операндів, виконання команди і збереження результату;

– наявність в архітектурі декількох функціональних пристроїв, що працюють незалежно, зокрема, можливість паралельного виконання логічних і арифметичних операцій;

– наявність окремих процесорів введення / виводу, що працюють незалежно і паралельно з основними процесорами.

Причини виключення перерахованих вище особливостей пояснюються наступним чином. Якщо розглядати комп'ютери, що використовують тільки паралелізм низького рівня, нарівні з усіма іншими, то, по-перше, практично всі існуючі системи будуть класифіковані як «паралельні» (що явно не буде позитивним фактором для класифікації), і, по-друге, такі машини будуть погано вписуватися в будь-яку модель чи концепцію, яка відображатиме паралелізм високого рівня.

Дункан дає неформальне визначення паралельної архітектури, причому саме неформальність дала йому можливість включити в даний клас комп'ютери, які раніше не вписувалися у систематику Флінна. Отже, паралельна архітектура - це такий спосіб організації обчислювальної системи, при якому допускається, щоб безліч процесорів (простих або складних) могло б працювати одночасно, взаємодіючи в міру потреби один з одним.

Сістолічні архітектури (їх найчастіше називають систолічним масивами) представляють собою безліч процесорів, об'єднаних регулярним чином (наприклад, система WARP). Звернення до пам'яті може здійснюватися тільки через певні процесори в межах масиву. Вибірка операндів з пам'яті і передача даних по масиву здійснюється в одному і тому ж темпі. Напрямок передачі

даних між процесорами фіксований. Кожен процесор за інтервал часу виконує невелику інваріантну послідовність дій.

Гібридні MIMD / SIMD архітектури, dataflow, reduction і wavefront обчислювальні системи здійснюють паралельну обробку інформації на основі асинхронного управління, як і MIMD системи. Але вони виділені в окрему групу, оскільки всі мають ряд специфічних особливостей, якими не володіють системи, традиційно пов'язані з MIMD [30].

MIMD / SIMD - типово гібридна архітектура. Вона припускає, що в MIMD системі можна виділити групу процесорів, що представляє собою підсистему, що працює в режимі SIMD (PASM, Non-Von). Такі системи відрізняються відносною гнучкістю, оскільки допускають реконфігурацію відповідно до особливостей розв'язуваної прикладної задачі.

Решта три види архітектур використовують нетрадиційні моделі обчислень. Dataflow використовують модель, в якій команда може виконаються відразу ж, як тільки обчислені необхідні операнди. Таким чином, послідовність виконання команд визначається залежністю за даними, яка може бути виражена, наприклад, у формі графа.

Модель обчислень, застосовувана в reduction машинах, інша і полягає в наступному: команда стає доступною для виконання тоді і тільки тоді, коли результат її роботи потрібно інший, доступній для виконання, команді як операнд.

Wavefront array архітектура поєднує в собі ідею систолічної обробки даних і модель обчислень, яка у dataflow. У цій архітектурі процесори об'єднуються в модулі і фіксуються зв'язки, по яких процесори можуть взаємодіяти один з одним. Проте, на противагу ритмічній роботі систолічних масивів, дана архітектура використовує асинхронний механізм зв'язку з підтвердженням (handshaking), через що «фронт хвилі» обчислень може міняти свою форму у міру проходження по всьому безлічі процесорів

1.2 Основні компоненти КС

Перелічимо основні компоненти:

– комп’ютери (ПК, ноутбуки, і таке інше);

– сервери;

– лінії зв'язку (мідні або оптичні кабелі, бездротовий зв’язок);

– конвертори й перетворювачі (оптичні, Wi-Fi тощо);

– комутаційне обладнання (хаби, свічі);

– програмне забезпечення (операційні системи, бази даних і таке інше) [31].

Більшість сучасних обчислювальних машин мають блочно-модульну конструкцію: апаратну конфігурацію, необхідну для виконання певних робіт, можна складати з готових вузлів та блоків.

Системний блок являє собою основний вузол, у якому зібрані найбільш важливі компоненти персонального комп'ютера. Основною компонентою є материнська плата, яка є своєрідним «фундаментом» для всіх комплектуючих комп'ютера. Саме в неї вставляються всі основні пристрої: відеокарта, оперативна пам'ять, процесор, жорсткі диски тощо. Інакше кажучи, це платформа, на якій будується вся конфігурація комп'ютера. На ній розміщені [32]:

– центральний процесор;

– шини — системи передачі даних та сигналів керування;

– оперативна пам'ять — набір мікросхем, призначених для зберігання даних під час їх безпосереднього опрацювання;

– постійна пам'ять — мікросхеми, призначені для постійного зберігання інформації, у тому числі і за вимкненого живлення.

У зв'язку з тим, що багато компонентів можуть бути інтегровані на материнській платі, то не всі вони можуть бути представлені як окремі

комплектуючі елементи, прикладом можуть бути плати із вмонтованими звуковою і відеокартами..

Центральний процесор (Central processing unit) — функціональна частина ЕОМ, що призначена для інтерпретації команд програми, керування пристроями комп’ютера та виконання арифметичних і логічних операцій над даними. Як правило, це компактний напівпровідниковий пристрій, що вставляється в гніздо на материнській платі.

За числом процесорів, що складають центральний процесор, розрізняють однопроцесорні й багатопроцесорні (мультипроцесорні) материнські плати.

Сервер - у комп'ютерній термінології термін може стосуватися окремого комп'ютера чи програми. Головною ознакою в обох випадках є здатність машини чи програми переважну кількість часу працювати автономно, без втручання людини, реагуючи на зовнішні події відповідно до встановленого програмного забезпечення. Втручання людини відбувається під час встановлення серверу і під час його сервісного обслуговування. Часто це роблять окремі адміністратори серверів з вищою кваліфікацією.

Сервер як комп'ютер - це комп'ютер у локальній чи глобальній мережі, який надає користувачам свої обчислювальні і дискові ресурси, а також доступ до встановлених сервісів; найчастіше працює цілодобово, чи у час роботи групи його користувачів [33].

Сервер як програма - програма, що надає деякі послуги іншим програмам (клієнтам). Зв'язок між клієнтом і сервером зазвичай здійснюється за допомогою передачі повідомлень, часто через мережу, і використовує певний протокол для кодування запитів клієнта і відповідей сервера. Серверні програми можуть бути встановлені як на серверному, так і на персональному комп'ютері, щоразу вони забезпечують виконання певних служб (наприклад, сервер баз даних чи веб-сервер).

Комп'ютер або програма, що установлена на цьому комп'ютері, здатні автоматично розподіляти інформацію чи файли під керуванням мережної ОС

або у відповідь на запити, надіслані у режимі on-line користувачами, і таким чином надавати послуги іншим комп'ютерам мережі (клієнтам).

Лінія зв'язку, **лінія передачі** — сукупність технічних пристроїв і фізичного середовища, що забезпечують передавання електричних сигналів одного, двох або багатьох каналів зв'язку на віддаль. Найпоширеніші електричні лінії передачі поділяють на дротові (кабельні лінії зв'язку, повітряні лінії зв'язку) та бездротові — радіотехнічні (наприклад, лінії радіорелейного зв'язку). Крім того, є лінії зв'язку звукові (гідроакустичний зв'язок) та оптичні. Для одночасного і незалежного передавання сигналів вдаються до ущільнення лінії зв'язку [34].

Перетворювач електричної енергії — електротехнічний пристрій, призначений для перетворення параметрів електричної енергії (напруги, частоти, числа фаз, форми сигналу). Для реалізації перетворювачів широко використовуються напівпровідникові прилади, тому що вони забезпечують високий ККД [35].

Комутаційний апарат — електричний апарат, який призначено для комутації електричного кола та проведення струму шляхом подачі або зняття напруги з електроустановки або її частини.

У загальному випадку комутаційні апарати поділяються на два типи:

– Контактний комутаційний апарат — апарат, який здійснює комутаційну операцію шляхом механічного взаємного переміщення контакт-деталей, внаслідок чого, відбувається комутація електричного струму. Контактні комунікаційні апарати, що працюють за значних (десятки ампер і більше) струмах та напругах (від сотень вольт і вище), та/або на індуктивне чи ємнісне навантаження, облаштовуються спеціальними системами гасіння електричної дуги.

– Безконтактний комутаційний апарат — апарат, який здійснює комутаційну операцію за рахунок зміни опору комутатора без механічного переміщення деталей. Безконтактні комутаційні апарати, що працюють на індуктивне або ємнісне навантаження облаштовуються спеціальними

системами захисту від електричного перевантаження, що виникає у момент комутації.

Програмне забезпечення (програмні засоби) - сукупність програм системи обробки інформації і програмних документів, необхідних для експлуатації цих програм.

Розрізняють системне програмне забезпечення (зокрема, операційна система, транслятори, редактори, графічний інтерфейс користувача); прикладне програмне забезпечення, що використовується для виконання конкретних завдань, наприклад, статистичне програмне забезпечення; інструментальне програмне забезпечення (комп'ютерні програми, призначені для проєктування, розробки, адміністрування і супроводження системного та прикладного програмного забезпечення) [36].

Виконання програмного забезпечення комп'ютером полягає у маніпулюванні інформацією та керуванні апаратними компонентами комп'ютера. Наприклад, типовим для персональних комп'ютерів є відтворення інформації на екран та отримання її з клавіатури.

Програмне забезпечення (*software*) та апаратне забезпечення (*hardware*) — це два комплементарні компоненти комп'ютера, причому межа між ними нечітка: деякі фрагменти програмного забезпечення на практиці реалізуються суто апаратурою мікросхем комп'ютера, а програмне забезпечення, в свою чергу, здатне виконувати (емулювати) функції електронної апаратури. По суті, призначення програмного забезпечення полягає в керуванні як самим комп'ютером так і іншими програмами та маніпулюванні інформацією.

Комплекс програм, які забезпечують управління компонентами комп'ютерної системи, такими як процесор, оперативна пам'ять, пристрої введення-виведення, мережеве обладнання, виступаючи як «міжшаровий інтерфейс», з одного боку якого — апаратура, а з іншого — додатки користувача. На відміну від прикладного програмного забезпечення, системне не вирішує конкретні практичні завдання, а лише забезпечує роботу інших

програм, надаючи їм сервісні функції, абстрагуючи деталі апаратної і мікропрограмної реалізації обчислювальної системи, керує апаратними ресурсами обчислювальної системи. Віднесення того чи іншого програмного забезпечення до системного є умовним, і залежить від угод, використовуваних у конкретному контексті. Як правило, до системного програмного забезпечення відносяться операційні системи, широкий клас сполучного програмного забезпечення.

Новим напрямком розвитку високопродуктивних обчислювальних систем є персональні суперкомп'ютери. Звичайно, вони не можуть зрівнятися за продуктивністю з традиційними суперкомп'ютерами, однак доступні користувачеві, дешеві у вартість виготовлення та експлуатації, мають невелику споживану потужність. Майже всі існуючі персональні суперкомп'ютери будуються на основі універсальних мікропроцесорів та спеціалізованих процесорних модулів, що дозволяє істотно підняти їх продуктивність при виконанні певних завдань.

Проблемою високопродуктивних систем на основі універсальних мікропроцесорів та реконфігуровних апаратних прискорювачів є їх недостатня "динамічність" – перед тим, як виконувати обчислювальний алгоритм на реконфігуровному прискорювачі, останній необхідно сконфігурувати, тобто синтезувати в ньому спеціалізований процесор, який реалізує даний алгоритм. Однак, враховуючи активну діяльність багатьох фірм в напрямку створення систем автоматизованого високорівневого синтезу програмованих логічних інтегральних схем (ПЛІС), засобів системного проектування ПЛІС з процедурних мов програмування високого рівня, очікується поява "самоконфігуровних" прискорювачів, а також засобів, які автоматично визначатимуть доцільність апаратної реалізації тих чи інших обчислювальних завдань чи їх частин в апаратному прискорювачі під час їх виконання [37].

1.3 Комп’ютерні мережі та їх класифікація

Комп’ютерна мережа – два або більше комп’ютерів, з’єднаних за допомогою каналів передачі даних (проводових ліній або радіозв’язку, ліній оптичного зв’язку) з метою об’єднання ресурсів та обміну інформацією.

Існування та функціонування мереж визначається ***протоколами і стандартами.*** Протокол – це сукупність правил (визначень, домовленостей), які регламентують формат і процедури обміну інформацією між двома або більшою кількістю незалежних пристроїв чи процесів. Іншими словами, протокол – це опис того, як програми, комп’ютери або інші пристрої мають функціонувати у процесі взаємодії між собою – від порядку передавання бітів до формату повідомлень електронної пошти.

З'єднання комп'ютерів в мережу забезпечує наступні основні можливості:

– об'єднання ресурсів – можливість резервувати обчислювальні потужності і засоби передачі даних на випадок виходу з ладу окремих з них з метою швидкого відновлення нормальної роботи мережі;

– розподіл ресурсів – можливість стабілізувати та підвищити рівень завантаження комп'ютерів та дорогого периферійного обладнання, управляти периферійними пристроями;

– поділ даних – можливість створювати розподілені бази даних, що розміщуються в пам'яті окремих комп'ютерів, і управляти ними з периферійних робочих місць;

– розподіл програмних засобів – можливість спільного використання програмних засобів;

– поділ обчислювальних ресурсів – можливість організувати паралельну обробку даних, використовуючи для обробки даних інші системи, що входять в мережу;

– режим для багатьох користувачів.

За розмірами розрізняють локальні та глобальні мережі. Локальна мережа, як правило, зв’язує не більш ніж сотню вузлів в одній локальній зоні (не більш ніж кілька кілометрів). Глобальна мережа може охоплювати територію регіону, держави чи кількох країн, з’єднувати як окремі комп’ютери, так і локальні мережі. Проміжним класом є міські (муніципальні) мережі, зорієнтовані на географічні області невеликих розмірів. Відмінність між названими класами мереж полягає не тільки в розмірах охоплюваних ними територій, а й у швидкості передавання даних – технології, які забезпечують більші швидкості, працюють на менших відстанях.

За способом використання каналу передавання даних розрізняють мережі з комутацією каналів і мережі з комутацією пакетів. Комутація каналів – це процес з’єднання двох або більшої кількості станцій з монопольним використанням каналу до його роз’єднання. У разі **комутації пакетів** повідомлення розбивається на частини – пакети, канал зайнятий тільки на час пересилання окремого пакета, після чого звільняється для передавання інших пакетів.

1.4 Компоненти комп’ютерних мереж

Для організації локальної комп’ютерної мережі необхідна наявність апаратного й програмного забезпечення [38]. До апаратного забезпечення відносяться:

1. ***Комп'ютери.*** Технічні характеристики комп'ютерів багато в чому визначають потенційні можливості утвореної з їхньою допомогою мережі. Спільне використання обчислювальних ресурсів привело до функціонального поділу комп'ютерів у мережі на комп'ютери, що надають ресурси (сервери), і комп'ютери, що споживають ресурси (робочі станції – клієнти). Сьогодні

практично кожна робоча станція в локальних мережах організацій надає свої, файлові ресурси у спільне користування.

Сервер мережі – це комп'ютер підключений до мережі, що надає користувачам мережі набір деяких послуг з використання й розподілу ресурсів мережі, наприклад, одночасний доступ користувачів до спільних даних, друку, приймання й обробка запитів до баз даних і т.д. ***Сервером*** також називають програмне забезпечення, що зберігає відповідну до свого ресурсу інформацію, що й відповідає на запити клієнтського програмного забезпечення.

Прикладами серверів можуть служити:

– сервер баз даних (SQL-сервер), що приймає запити по локальній мережі і повертає результати;

– сервер телекомунікацій, що забезпечує послуги зі зв'язку даної локальної мережі із зовнішнім світом;

– обчислювальний сервер для проведення обчислень, які неможливо виконати на робочих станціях;

– Web-сервер;

– Mail-сервер, поштовий сервер організації;

– файловий сервер, що підтримує спільне сховище файлів для всіх робочих станцій.

Робоча станція – це підключений до локальної мережі персональний комп’ютер, на якому користувач безпосередньо виконує свою роботу, використовує свою операційну систему й за допомогою якого має доступ до апаратних, програмних і інформаційних ресурсів мережі. Оскільки робочі станції в мережі виступають клієнтами, то *клієнтом* називається й програма, встановлена на комп'ютері користувача для створення запитів відповідному серверу і отримання відповіді. У загальному випадку *клієнтом* може бути й користувач.

2. ***Лінії зв'язку або канали передачі даних***. При побудові локальних КМ у якості ліній зв'язку використовуються різні типи фізичних середовищ передачі даних:

- *провідні*, побудовані на основі кабелю з електричним сигналом передачі даних - витої пари (скручена пара провідників, укладена в оболонку); з оптичним сигналом передачі даних – волоконно-оптичного;

- *бездротові* (з радіосигналом, мікрохвильовим сигналом передачі даних) – канали наземного і супутникового зв'язку.

Вибір тієї або іншої лінії зв'язку визначається необхідною швидкістю передачі даних, а також відстанями між окремими вузлами мережі.

Апаратними компонентами глобальних мереж (інтермереж) є, переважно, високопродуктивні комутатори та маршрутизатори.

Фізично комп'ютери можуть поєднуватися в мережу різними способами. Структура фізичного з'єднання комп'ютерів у мережу називається ***топологією***. Мережева топологія – це конфігурація графа, вершинам якого відповідають кінцеві вузли мережі (комп'ютери) і комунікаційне встаткування (концентратори, комутатори, маршрутизатори), а ребрам – фізичні або інформаційні зв'язки між вершинами [38].

3. ***Комутаційне встаткування***. Довжина мережі, відстань між робочими станціями в першу чергу визначаються фізичними характеристиками передавального середовища. При передачі даних у будь-якому середовищі відбувається згасання сигналу. Крім цього в локальних мережах можуть наступати колізії (передача декількома комп'ютерами даних по одній і тій же лінії зв'язку), що приводить до непрацездатності мережі. Щоб подолати ці й інші обмеження для зв'язку сегментів мережі використовується спеціальне ***комутаційне устаткування: комутатори, маршрутизатори.*** У даний час спостерігається тенденція до універсалізації комутаційного устаткування при збереженні базових назв.

Комутатори використовуються для логічної структуризації мереж з метою запобігання колізій, підвищення пропускної здатності мережі, агрегації каналів та базового захисту.

Маршрутизатор – спеціальний пристрій для забезпечення зв'язку між окремими мережами за допомогою мережних (IP) адрес. У даний час

спостерігається тенденція витиснення маршрутизаторів і заміна їх високопродуктивними комутаторами, що комбінують у собі як функції комутації так і маршрутизації.

4. ***З'єднуюче встаткування***. До з'єднуючого встаткування відносяться**: конектори –** рознімні й/або нероз'ємні з'єднувачі, що прикріплюються до кабелів**, різні кабельні адаптери** й **розгалуджувачі –** для стикування різних типів кабелів, підсилювачі й інші пристрої. Сполучне встаткування забезпечує можливість підключення різних мережних пристроїв до ліній зв'язку.

Програмні засоби КМ включають:

1. ***Мережна операційна система ОС*** **–** зв'язує всі комп'ютери й периферійні пристрої в мережі, координує функції всіх комп'ютерів і периферійних пристроїв у мережі, забезпечує захищений доступ до даних і периферійних пристроїв у мережі, надає власні ресурси й певні послуги в спільне користування.

При побудові складних мереж, як правило, один або два комп'ютери виділяють для виконання мережних функцій (***мережі з виділеним сервером***). До таких серверів пред'являються більш високі вимоги до їх технічного забезпечення, продуктивності, обсягу пам'яті, надійності захисту інформації. У мережах з виділеним сервером на сервері встановлюють серверну ОС, що володіє значними мережними можливостями. До переваг серверних ОС можна віднести забезпечення високої продуктивності мережі, наявність розвинених засобів керування й адміністрування в мережі. На серверах ведуться облікові записи користувачів мережі, які містять інформацію про всіх користувачів локальної мережі, підключених до даного сервера і їх правах доступу до ресурсів мережі, імена комп'ютерів, робочих груп і т.д.

2. ***Мережне програмне ПЗ*** – це прикладні програми (або мережеві служби), які розширюють можливості мережних ОС. Серед них можна виділити сервер баз даних, сервер інформаційного обміну і т.д. [39].

Реалізація мережних служб здійснюється програмним забезпеченням. Файлова служба й служба друку надаються операційними системами, а інші

служби забезпечуються мережними прикладними програмами. До традиційних мережних служб відносяться: Telnet, FTP, HTTP, SMTP, POP-3.

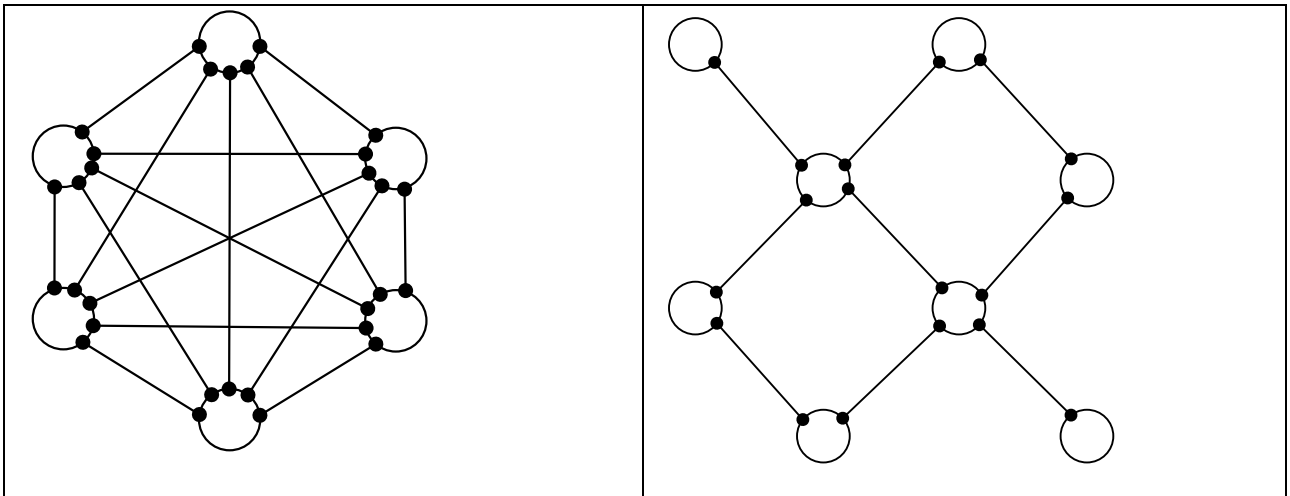
Повнозв'язна топологія – топологія комп'ютерної мережі, у якій кожен вузол підключений до всіх інших. Для кожної пари вузлів повинна бути виділена незалежна лінія, кожний комп'ютер повинен мати стільки комунікаційних портів скільки комп'ютерів у мережі. Повнозв'язна топологія має порівняно невеликі кінцеві розміри і найчастіше використовується в багатомашинних комплексах.

Недоліки: велика кількість фізичних зв'язків.

Інші варіанти топологій засновані на неповнозв'язних топологіях, коли для обміну даними між двома робочими станціями мережі можуть знадобитися проміжні вузли (пристрої) передачі даних.

Сітчаста мережа (анг. mesh) – це топологія в якій кожен вузол передає дані інших вузлів (рисунок 1.1). Оскільки всі вузли беруть участь у передачі, то це дозволяє змінити маршрут передачі даних в обхід розірваного шляху, вибравши альтернативний шлях до місця призначення. Висока відмовостійкість пов'язана зі складністю налаштування обладнання. Сітчаста топологія використовується в глобальних мережах (в т.ч. Інтернет) та безпровідних мережах.

Недоліки: складність побудови та налаштування, висока ціна.



Повнозв'язна топологія	Сітчаста топологія

Рисунок 1.1 – Повнозв'язна і сітчаста топологія

Топологія типу **шина**, являє собою загальний кабель (шина або магістраль), до якого приєднані всі робочі станції (рисунок 1.2). На кінцях кабелю містяться термінатори, для запобігання відбиття сигналу.

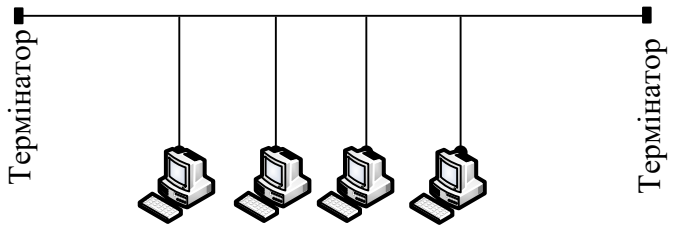


Рисунок 1.2 – Топологія шина

Повідомлення що відправляється робочою станцією поширюється на всі комп’ютери мережі. Кожна станція перевіряє – кому адресоване повідомлення і якщо їй, то обробляє його. Для того, щоб виключити одночасну посилку даних, застосовується або «несучий» сигнал, або один з комп'ютерів є головним і дає дозвіл іншим станціям.

В цій топології комп'ютери можуть передавати тільки по черзі, тому що лінія зв'язку спільна. В протилежному випадку передана інформація буде спотворюватися в результаті накладення (конфлікту, колізії). Таким чином, у шині реалізується режим напівдуплексного (half duplex) обміну (в обох напрямках, але по черзі, а не одночасно).

Переваги:

– відсутній центральний абонент, через який передається вся інформація, яка збільшує її надійність

– додавання нових абонентів у шину досить просте й звичайно можливе навіть під час роботи мережі.

У більшості випадків при використанні шини потрібно мінімальна кількість з’єднуючого кабелю в порівнянні з іншою топологією. Правда,

потрібно врахувати, що до кожного комп'ютера (крім двох крайніх) підходить два кабелі, що не завжди зручно.

– невеликий час установки мережі;

– дешевизна (потрібно менше кабелю й мережних пристроїв);

– простота настроювання;

Недоліки:

– будь-які неполадки в мережі, як обрив кабелю, вихід з ладу термінатора повністю припиняють роботу всієї мережі;

– складна локалізація несправностей;

– з додаванням нових робочих станцій падає продуктивність мережі.

Оскільки середовище передачі даних не проходить через вузли, підключені до мережі, втрата працездатності одного із пристроїв ніяк не позначається на інших пристроях. Хоча використання всього лише одного кабелю може розглядатися як перевага шинної топології, однак вона компенсується тим фактом, що кабель, використовуваний у цьому типі топології, може стати критичною точкою відмови. Інакше кажучи, якщо шина обривається, то жодне з підключених до неї пристроїв не зможе передавати сигнали.

Зірка – базова топологія комп’ютерної мережі, у якій усі комп’ютери мережі приєднані до центрального вузла (звичайно мережевий концентратор), утворюючи фізичний сегмент мережі. Подібний сегмент може функціонувати як окремо, так і в складі складної мережевої топології (як правило "дерево") [39, 40]. Як правило, саме на центральний вузол покладено функцію централізованого керуванню обміном (рисунок 1.3).

У функції концентратора входить забезпечення доступу станціям до середовища передачі даних. Для приєднання комп'ютера до концентратора використовується, як правило, вита пара. Головна перевага цієї топології перед загальною шиною - значне збільшення надійності. Будь-які неприємності з кабелем стосуються лише того комп'ютера, до якого цей кабель приєднаний, і тільки несправність концентратора може вивести з ладу всю мережу. До

недоліків топології типу зірка відноситься більш висока вартість мережного встаткування через необхідність придбання концентратора.

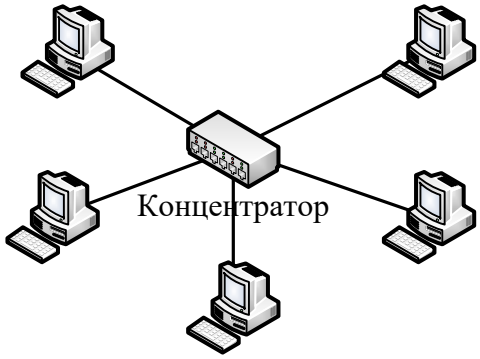


Рисунок 1.3 – Топологія зірка

Робоча станція відсилає дані на концентратор, а він визначає адресата й віддає інформацію йому. У певний момент часу тільки один вузол в мережі може пересилати дані. Якщо на концентратор одночасно приходять два пакети, обидві посилки виявляються не прийнятими й відправникам потрібно буде почекати випадковий проміжок часу, щоб відновити передачу даних. Цей недолік відсутній на мережному пристрої більш високого рівня - комутаторі, який, на відміну від концентратора, котрий подає пакет на всі порти, подає лише на певний порт - одержувачу. Одночасно може бути передано кілька пакетів.

Активна зірка. У центрі мережі міститься комп’ютер, що виступає у ролі сервера.

Пасивна зірка. У центрі мережі з даною топологією міститься не комп’ютер, а концентратор, або хаб (hub), що виконує ту ж функцію, що й повторювач (репітер). Він відновлює сигнали, які надходять, і пересилає їх в інші лінії зв'язку.

Переваги:

– вихід з ладу однієї робочої станції не відбивається на роботі всієї мережі в цілому;

– хороша масштабованість мережі;

– легкий пошук несправностей і обривів у мережі;

– висока продуктивність мережі (за умови правильного проектування);

– гнучкі можливості адміністрування.

– Недоліки:

– вихід з ладу центрального вузла (концентратора) приведе до непрацездатності мережі (або сегмента мережі) у цілому;

– для розгортання мережі як правило потрібно більше кабелю, ніж для більшості інших топологій;

– кінцеве число робочих станцій у мережі (або сегменті) обмежене кількістю комунікаційних портів у центральному вузлі (концентраторі).

Кільцева топологія передбачає передачу сигналів по кільцю від одного вузла до іншої, як правило, в одному напрямку. Якщо вузол не розпізнає пакет як “свій”, то він передає його наступному в кільці. При застосуванні даної топології необхідно вживати спеціальних заходів, щоб у випадку виходу з ладу або відключенні якої-небудь вузла не перервався канал зв'язку між іншими вузлами.

Кільце – це топологія, у якій кожний комп'ютер з'єднаний лініями зв'язку тільки із двома іншими: від одного він тільки одержує інформацію, а іншому тільки передає. На кожній лінії зв'язку, як і у випадку зірки, працює тільки один передавач і один приймач.

Важлива особливість кільця полягає в тому, що кожний вузол ретранслює (відновляє) сигнал, тобто виступає в ролі повторювача, тому згасання сигналу у всьому кільці не має ніякого значення, важливо тільки загасання між сусідніми вузлами кільця. Чітко виділеного центру в цьому випадку ні, усі комп'ютери можуть бути однаковими. Однак досить часто в кільці виділяється спеціальний абонент, який управляє обміном або контролює обмін. Зрозуміло, що наявність такого керуючого абонента знижує надійність мережі, тому що вихід його з ладу відразу ж паралізує весь обмін.

У методах управління право на наступну передачу (або право на захоплення мережі) переходить послідовно до наступного по кільцю комп'ютера. Підключення нових абонентів в «кільце» зазвичай не складне, хоча й вимагає обов'язкової зупинки роботи всієї мережі на час підключення. Кільцева топологія звичайно є самою стійкою до перевантажень, вона забезпечує впевнену роботу із самими великим потоками переданої по мережі інформації, тому що в ній, як правило, немає конфліктів (на відміну від шини), а також відсутній центральний абонент (на відміну від зірки).

Серед переваг виділимо можливість стійкої роботи без істотного падіння швидкості передачі даних при інтенсивному завантаженні мережі, оскільки використання маркера виключає можливість виникнення колізій.

До недоліків віднесемо те, що при виході з ладу однієї робочої станції, або інших неполадках (обрив кабелю позначиться на працездатності всієї мережі.

Найбільш широке застосування кільцева топологія одержала в оптоволоконних мережах: SONET (self-healing, bi directional rings.), FDDI, Token Ring.

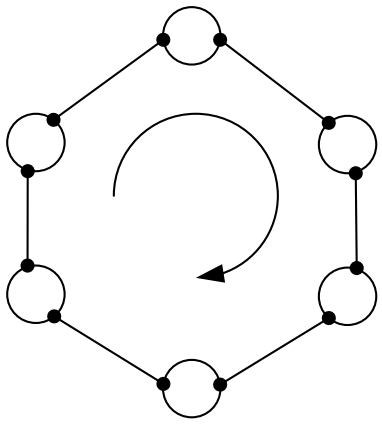


Рисунок 1.4 – Топологія кільце

Комутована топологія

У перших мережах Ethernet (10Base-2 і 10Base-5) використовувалася шинна топологія, коли кожен комп'ютер з'єднувався з іншими пристроями за

допомогою єдиного коаксіального кабелю, використовуваного в якості середовища передачі даних [39, 40]. Мережеве середовище було поділюваним і пристрої, перш ніж почати передавати пакети даних, повинні були переконатися, що вона вільна.

В наступному стандарті 10Base-T з топологією типу «зірка» кожен вузол уже підключався окремим кабелем до центрального пристрою - концентратора (hub). Концентратор працював на фізичному рівні моделі OSI і повторював сигнали, що надходили з одного з його портів на всі інші активні порти, попередньо відновлюючи їх. Використання концентраторів дозволило підвищити надійність мережі, оскільки обрив сегмента кабелю не спричиняв збій в роботі всієї мережі. Середовище передачі залишилося поділюваним (всі пристрої перебували в одному домені колізій). Крім цього загальна кількість концентраторів і з'єднуваних ними сегментів мережі було обмежено через часові затримки і інші причини.

Завдання сегментації мережі, тобто поділу користувачів на групи (сегменти) відповідно до їх фізичним розміщенням з метою зменшення кількості вузлів котрі змагаються за смугу пропускання була вирішена за допомогою пристрою, названого мостом (bridge) [39, 40].

Міст являв собою пристрій канального рівня моделі OSI (зазвичай двохпортовий), призначений для об'єднання сегментів мережі. На відміну від концентратора, міст не просто пересилав пакети даних з одного сегмента в інший, а й аналізував і передавав їх тільки в тому випадку, якщо така передача дійсно була необхідна, тобто адреса робочої станції отримувача належала іншому сегменту. Таким чином, міст ізолював трафік одного сегмента від трафіка іншого, зменшуючи домен колізій і підвищуючи загальну продуктивність мережі.

Збільшення кількості пристроїв і поява мультимедійного трафіку привела до появи комутаторів (switch). Комутатор являв собою багатопортовий міст і також функціонував на канальному рівні моделі OSI. Основна відмінність комутатора від мосту полягала в тому, що він міг установлювати одночасно

кілька з'єднань між різними парами портів. При передачі пакета через комутатор у ньому створювався окремий віртуальний (або реальний, залежно від архітектури) канал, по якому дані пересилалися «прямо» від порту-джерела до порту-одержувачеві з максимально можливою для використовуваної технології швидкістю. Такий принцип роботи одержав назву мікросегментація. Завдяки мікросегментації, комутатори одержали можливість функціонувати в режимі повного дуплекса (full duplex), що дозволяло кожній робочій станції одночасно передавати й приймати дані, використовуючи всю смугу пропускання в обох напрямках. Робочій станції не доводилося конкурувати за смугу пропускання з іншими пристроями, у результаті чого не відбувалися колізії, і підвищувалася продуктивність мережі.

У даний час комутатори є основним будівельним блоком для створення локальних мереж. Сучасні комутатори Ethernet перетворилися в інтелектуальні пристрої зі спеціалізованими процесорами для обробки й перенаправлення пакетів на високих швидкостях, і реалізації таких функцій, як організація резервування й підвищення відмовостійкості мережі, агрегування каналів, створення віртуальних локальних мереж (VLAN), маршрутизації, керування якістю обслуговування (Quality of Service), забезпечення безпеки й багатьох інших [41, 42].

З метою збільшення кількості портів, зручності керування й моніторингу виконується об'єднання декількох комутаторів в один логічний пристрій з допомогою фізичного стекування. Об'єднані в стек комутатори мають спільні таблиці комутації й маршрутизації (для комутаторів 3 рівня).

Маршрутизатори працюють на мережному рівні. Для маршрутизаторів мережа являє собою набір мережних адрес пристроїв і множина мережних шляхів. Маршрутизатори аналізують усі можливі шляхи між будь-якими двома вузлами мережі й вибирають самий короткий з них. При виборі можуть братися до уваги й інші фактори, наприклад, стан проміжних вузлів і ліній зв'язку, пропускна здатність ліній або вартість передачі даних.

Для того, щоб маршрутизатор міг виконувати покладені на нього функції йому повинна бути доступна більш розгорнута інформація про мережу, ніж та, яка доступна мосту. У заголовку пакета мережного рівня крім мережної адреси є дані, наприклад, про критерій, який повинен бути використаний при виборі маршруту, про час життя пакета в мережі, про те, якому протоколу верхнього рівня належить пакет. Завдяки використанню додаткової інформації, маршрутизатор може здійснювати більше операцій з пакетами, ніж комутатор.

1.5 Архітектура сучасної корпоративної мережі

При проектуванні корпоративної мережі потрібно дотримуватись таких принципів: ієрархія, модульність, відмовостійкість. Ієрархічний дизайн мережі розділяє мережу на більш дрібні і керовані мережі. Кожен рівень ієрархії орієнтований на певні ролі. Цей поділ забезпечує високий рівень гнучкості для оптимізації і вибору правильного мережевого обладнання, програмного забезпечення і функцій для виконання певних ролей для різних мережевих рівнів.

На рисунку 1.5 а) зображена трьохрівнева архітектура корпоративної мережі. Дана схема мережі повністю задовольняє вимогам відмовостійкості, модульності і ієрархічності [40, 41].

Рівень доступу (access layer) перший ярус або границя мережі кампусу (містечка). Це місце, де кінцеві пристрої (робочі станції, принтери, камери і т.п.) приєднуються до дротової частини мережі кампусу. Це також місце, де приєднуються пристрої, що розширюють мережу на один рівень - IP телефони і бездротові точки доступу. Шар доступу забезпечує інтелектуальне розмежування між мережевою інфраструктурою та обчислювальними пристроями, які використовують цю інфраструктуру. Він забезпечує безпеку, QoS. Це перший шар захисту в архітектурі мережевої безпеки і перша точка

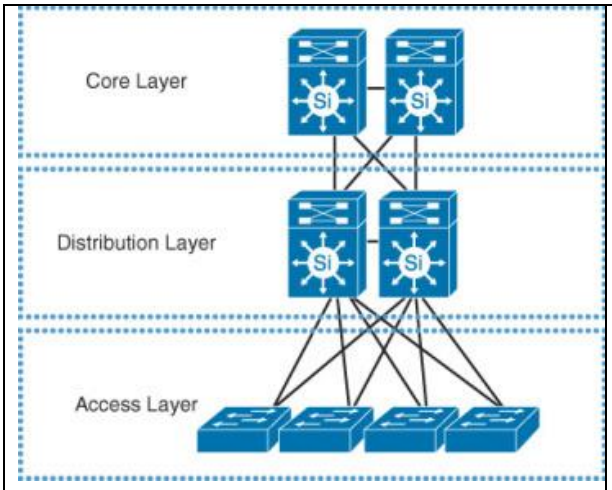
переговорів між кінцевими пристроями і мережевою інфраструктурою. Комутатор доступу (access switch) надає більшість перелічених сервісів рівня доступу і є ключовим елементом у забезпеченні багатьох сервісів корпоративної мережі.

Рівень розподілу (distribution layer) виступає в якості служби та границі контролю між доступом та ядром. Поєднує кілька цілей. Це агрегаційна точка для всіх комутаторів доступу і забезпечує послуги з з'єднання потоків трафіку в блоці розподілі-доступу. В ядрі мережі даний рівень бере участь у схемі базової маршрутизації. Рівень розподілу забезпечує агрегацію, контроль політики доступу та точки ізоляції між блоком розподілу та іншою частиною мережі.

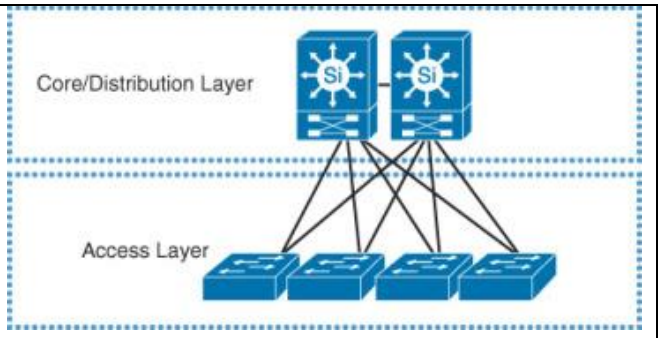
Ядро являє собою комплекс мережних пристроїв (маршрутизаторів і комутаторів), що забезпечують резервування каналів і високошвидкісну передачу даних між різними сегментами рівня розподілу. Забезпечує транспорт трафіку між сайтами і високопродуктивну маршрутизацію. Через критичну важливість цього рівня принципи проектування ядра повинні забезпечувати відповідний рівень відмовостійкості, який забезпечує можливість швидкого і плавного відновлення мережі після збою.

Ця схема мережі в основному призначена для великих організацій і операторів зв'язку де безперебійність надання послуг є основою роботи підприємства.

Трирівнева ієрархічна структура забезпечує максимальну продуктивність, доступність мережі і можливість масштабування структури мережі. Однак мережі багатьох невеликих підприємств незначно розширюються з плином часу. Тому дворівнева ієрархічна структура, в якій рівні ядра і розподілу об'єднані в один рівень, часто виявляється більш практичною. Ядро називається «виродженим», якщо функції рівнів ядра і розподілу реалізуються одним пристроєм [43]. Основним стимулом для використання проекту з виродженим ядром є скорочення вартості мережі при збереженні переваг трирівневої ієрархічної моделі. Дворівнева модель зображена на рисунку 1.5 б).



а) Трирівнева архітектура корпоративної мережі



б) Дворівнева архітектура корпоративної мережі

Рисунок 1.5 – Архітектури корпоративних мереж

Один з основних факторів впливу на вибір дворівневою і трирівневої мережевий архітектури - це тип мережі (віддалений офіс), що допомагає визначити характер мережі і її потенційне майбутнє зростання.

До переваг мережі з виродженим ядром належать [43]:

– менша вартість апаратних пристроїв через відмову від одного рівня архітектури;

– менша кількість з’єднань;

– забезпечення ієрархічної моделі;

– простота налаштування.

До недоліків можна віднести зменшення відмовостійкості і резервування.

Як видно з рисунку сучасна архітектура корпоративної мережі побудована на основі комбінації повнозв’язної топології на рівні ядра та сітчастої (mesh) топології на рівні розподілу.

1.6 Програмні засоби моделювання комп’ютерних систем та мереж

Для перевірки правильності функціонування КСМ існує фізичне моделювання або натурні експерименти. Зазвичай це відбувається на кінцевій стадії проектування КСМ. На початкові стадії проектування використовують

імітаційне моделювання. Одним з основних видів імітаційного моделювання є статистичне імітаційне моделювання. Статистична модель випадкового процесу - це алгоритм, за допомогою якого імітують роботу складної системи, що піддається випадковим впливам; імітують взаємодію елементів системи, які носять імовірнісний характер

Для побудови імітаційних моделей використовується спеціальні системи моделювання: MATLAB, спеціальні мови моделювання (загального застосування и проблемно-орієнтовані): GPSS/PC, GPSS/H, GPSS World, GASP, SIMSCRIPT , Object GPSS, Arena, SimProcess, Enterprise Dynamics, Auto-Mod, SIMPAS і універсальні мови моделювання SIMULA, PASCAL, CI і т.п.

Метод статистичного імітаційного моделювання - це спосіб вивчення складних процесів і систем, які піддаються випадковим впливам, за допомогою імітаційних моделей. Методика статистичного моделювання складається з таких етапів:

1. Моделювання на комп'ютері псевдовипадкових послідовностей з заданою кореляцією і законом розподілу ймовірностей (метод Монте-Карло), імітуючи випадкові значення параметрів при кожному дослідженні;

2. Перетворення одержаних числових послідовностей на імітаційних математичних моделях.

3. Статистична обробка результатів моделювання.

Імітаційне моделювання використовується в різних областях, зокрема в: комп'ютерних системах і мережах, бізнес процесах, військових діях, динаміці населення, IT- інфраструктурі, математичному моделюванні історичних процесів, логістиці, пішохідні динаміці, вуличному русі, виробництві, ринку і конкуренції, сервісних центрах, управління проектами, екосистемах тощо.

Розглянемо основні мови моделювання, які використовуються для імітації роботи КСМ.

Моделювання комп'ютерних систем в середовищі GPSS World. В основі GPSS World використовується оригінальна система комп'ютерного моделювання GPSS (General Purpose Simulation System). GPSS – загально

цільова система моделювання, яка була розроблена Джефрі Гордоном приблизно в 1960 році, яка призначена для моделювання процесів в системах масового обслуговування (СМО).

GPSS World – це прямий розвиток мови моделювання GPSS/PC для ОС Windows має розширені можливості, включає середовище користувача з інтегрованими функціями роботи з Інтернет.

GPSS World включає словник і граматику, за допомогою яких можуть бути розроблені моделі систем. Машина програма інтерпретує модель, написану на GPSS, надаючи розробнику моделі можливість проведення експериментів. Моделі систем на GPSS можуть бути представлені у вигляді блок-схем чи записані у вигляді послідовності операторів, еквівалентній блок-схемі. Блок-схема представляє собою набір фігур з характерним окресленням блоків, з'єднаних між собою стрілками. Розробнику моделей представляється набір більш ніж із 40 блоків. Вигляд кожного із блоків стандартний. Із допустимої множини блоків вибудовують послідовність, об'єднану зв'язками, які показують взаємодію блоків. Однакові блоки можуть зустрічатися в моделі довільну кількість раз, що визначається особливостями модельованих систем.

Різні події в реальних системах відбувається на протязі деякого періоду часу. Завдання приходять в систему, коли приходить їх черга, вони попадають на обслуговування в процесор, потім покидають систему. Якщо всі ці події представити в моделі, то їх проходження має бути організоване на фоні модельного часу. Коли починається моделювання, в інтерпретаторі планується прихід першого транзакту. Після цього таймер модельного часу встановлюється в значенні часу, який відповідає моменту появи першого транзакту в моделі. Цей транзакт входить в модель, а інтерпретатор GPSS просуває далі значення таймера до значення часу, коли відбувається наступна подія.

Особливості таймера GPSS такі: таймер реєструє тільки цілі значення, одиниця часу визначається розробником.

Особливістю системи GPSS є те, що тривалість моделювання визначається не тривалістю інтервалів часу між подіями, а числом подій, які виникають в системі.

При цьому модель утворюється із модельних блоків і модельних об'єктів. Модельні блоки в свою чергу виконують процеси імітації дій. Модельні об'єкти можуть бути фіксованими і динамічними. Динамічні об'єкти системи – це вимоги на обслуговування, які називаються транзанктами. Транзанкти створюються, переміщуються через модельні блоки, затримуються і знищуються (виводяться з моделі). Фіксовані об'єкти – це елементи СМО : черги, прилади, багатоканальні пристрої.

Блоки представлені в моделі сукупністю ключових слів і операндів. Для організації посилань блокам можна присвоювати символічні імена (алфавітно-цифрові символи, не більше п'яти). Операнди блоків задають інформацію, специфічну для даного блоку. Число операндів залежить від типу блоку. Значення одних операндів має бути вказані завжди, інші можна не вказувати, тоді вони задаються по замовчуванні.

Оператори GPSS записуються в вигляді стандартних 80 байтних записів:

1 Байт - ознака коментаря;

2-6 Байти - ім'я блоку;

8-18 Байти – операція;

19-71 Байти – операнди;

72 і далі - Байти - інтерпретатором не сприймаються.

В стрічковому редакторі GPSS PC переміщення по полям автоматизовано (при натисканні на клавішу пробіл курсор переміщується в першу позицію наступного поля блоку). Внесення транзактів в модель виконується блоком:

GENERATE A, B, C, D, E,

де A - середній інтервал часу;

B - половина поля допуску;

C - зміщення інтервалів;

D - обмежувач кількості транзактів;

E - рівень пріоритету.

Значення параметрів за замовчуванням: A - 0 , B - 0, C - відсутній, D - без обмеження, E - 0.

Всі можливі види розподілів інтервалів часу поступлення транзанктів в GPSS ділять на рівномірно розподілені і всі інші види розподілу. Блоком організується тільки рівномірний розподіл, інші види розподілів оформляються через визначення функцій.

Приведемо приклад генерації транзанктів:

GENERATE 5, 3, 10, 19, 16

Цей блок задає прихід першого транзакту (параметр C) в момент модельного часу 10, інші приходять у відповідності з рівномірним законом 5 □ 3 одиниці модельного часу. Через блок може ввійти в модель не більше 19 транзактів, пріоритет кожного - 16 (всього 128 рівнів пріоритетів 0 - 127).

Приклад основного меню GPSS World приведений на рисунку 1.6.

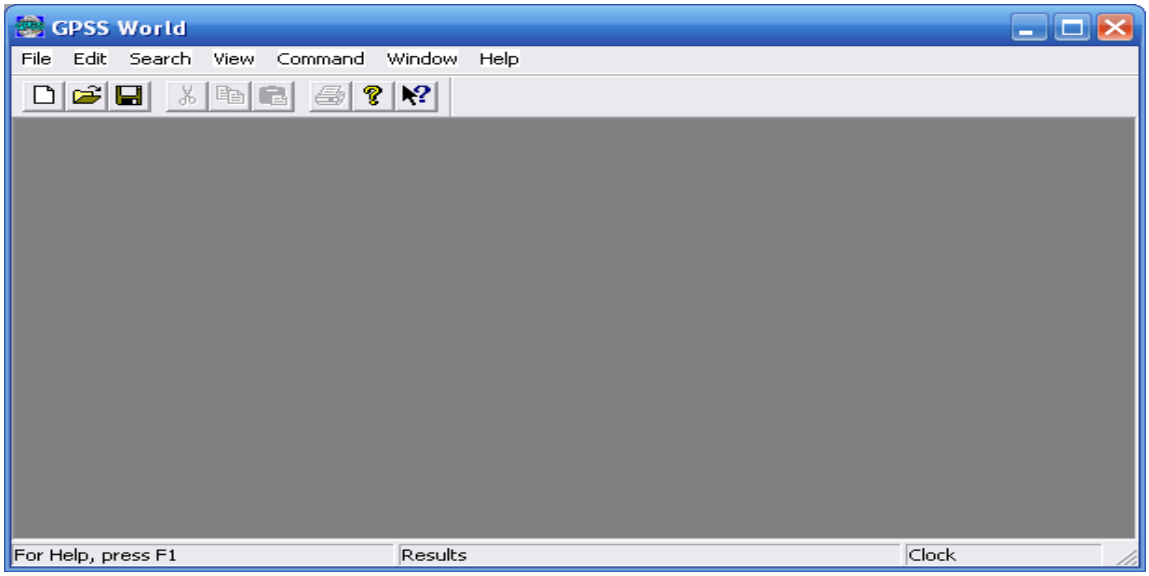


Рисунок 1.6 – основне меню GPSS World.

Основне вікно включає декілька компонент:

1. У верхній частині розміщена стрічка заголовка.

2. Нижче розміщено основне меню, а ще нижче – панель інструментів, за якою розміщена клієнтська область.

3. В самому низу головного вікна розміщена стрічка стану, яка розділена на три частини: ліва частина стрічки показує підказку з інформацією про використовуваний пункт меню; середина показує повідомлення про помилки; права частина стрічки призначена для відображення модельного часу в процесі виконання моделі.

Приклад вікна REPORT наведений на рисунку 1.7.

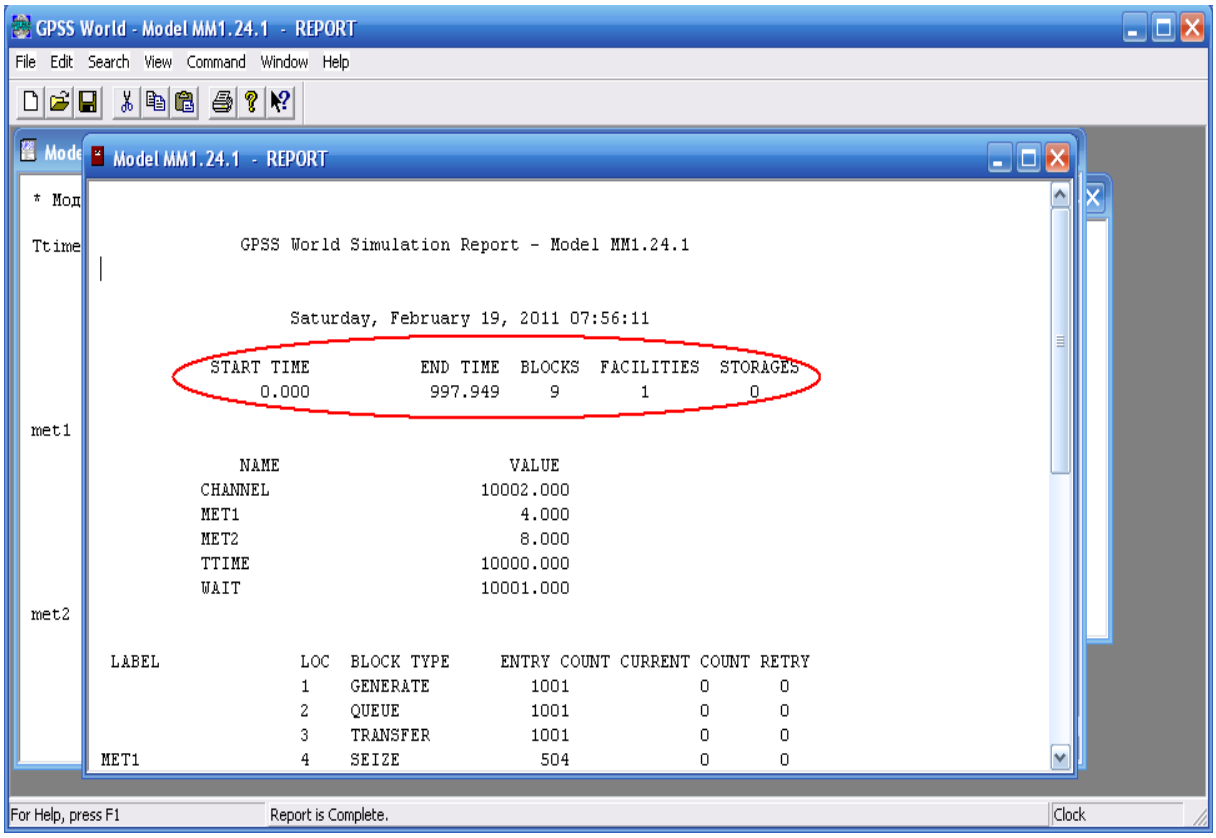


Рисунок 1.7 – вікно REPORT

Результати роботи моделі приведено на рисунку 1.8.

Загальна інформація про результати роботи моделі

START TIME	END TIME	BLOCKS	FACILITIES	STORAGES
0.000	1000.000	10	0	1

- START TIME – початковий час, тобто абсолютний модельний час у момент початку моделювання. Він дорівнює абсолютному модельному часові завдяки дії оператора RESET або CLEAR;
- END TIME – кінцевий час, тобто значення абсолютного модельного часу в момент, коли лічильник завершень набуває значення 0;
- BLOCKS – кількість блоків, використаних у моделі в момент завершення моделювання;
- FACILITIES – кількість одноканальних пристроїв, використаних у моделі в момент завершення моделювання;
- STORAGES – кількість багатоканальних пристроїв, використаних у моделі в момент завершення моделювання.

Рисунок 1.8 – Результат роботи моделі.

Для розв'язання задач аналізу комп'ютерних мереж використовують моделі на основі теорії мереж Петрі.

Розглянемо програмний засіб для моделювання КМ PETRI-NET.

На панелях інструментів (рисунок 1.9) розташовані кнопки, призначені для виконання функцій побудови графа мережі Петрі.

1 2 3 4 5 6 7 8 9 10 11 12

Рисунок 1.9 – Головна панель інструментів

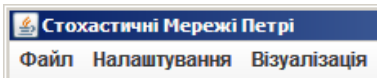
Функціональність кнопок робочої панелі є такою:

1. – Перевірка мережі Петрі та вихід з режиму редактування.
2. – Інструмент побудови вузлів.
3. – Інструмент побудови переходів.
4. – Інструмент побудови дуг.
5. – Додати токен.
6. – Інструмент редагування.

7. – Інструмент видалення елементів мережі.
8. – Збереження мережі Петрі.
9. – Збереження поточного стану мережі Петрі в оперативну пам’ять.
10. – Відтворення збереженого стану мережі Петрі з оперативної пам’яті.
11. – Запуск моделювання.
12. – Крок моделювання.

Робоче вікно програми призначене для побудови графічного представлення графу та симуляції роботи мережі Петрі. Робоче вікно містить дві панелі інструментів.

Зверху розташовані вкладки:



Вкладка ‘Файл’ містить наступні команди (рисунок 1.10):

- Нова мережа – створити новий файл (нова модель);
- Завантажити – відкрити раніше збережений файл;
- Зберегти – зберегти файл (швидке збереження).
- Зберегти як – зберегти файл під іншою назвою.
- Вийти – вихід з програми.

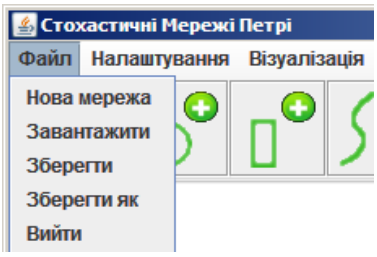


Рисунок 1.10 – Вкладка «Файл» підсистеми автоматизації аналізу мережі Петрі

Вкладка ‘Налаштування’ містить такі команди:

- Група Паралельно (одночасне спрацювання переходів).
- Ручний режим (перехід вибирає користувач).
- Випадковий режим (перехід обирається випадково).

- Група Послідовно (послідовне спрацювання переходів).
- Ручний режим (перехід вибирає користувач).
- Випадковий режим (перехід обирається випадково).

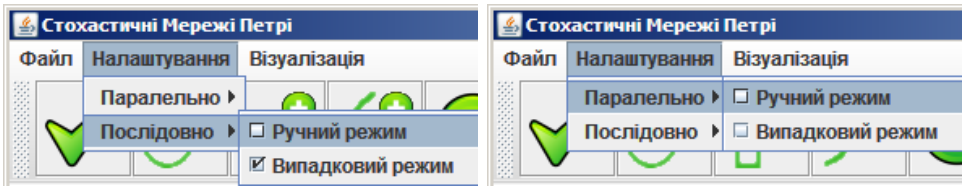


Рисунок 1.11 – Вкладка «Налаштування» ПАСМП

Вкладка ‘Візуалізація’ містить наступні команди:

- Опції – опції відтворення процесу симуляції: виділення кольором активного переходу та час затримки між спрацюваннями.

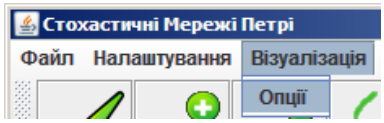


Рисунок 1.12 – Вкладка «Візуалізація» ПАСМП

Інструмент “Вузол”. Для створення вузла, необхідно натиснути на кнопку , після чого можна створювати елементи даного типу натискання лівої кнопки миші. Кожен вузол буде створено із властивостями по замовчуванню: порожнє поле імені, нульова кількість токенів та ймовірність спрацювання переходу 50% та графічним символом . Для виходу із режиму побудови вузлів, натискається кнопка . В цьому режимі також можна змінити положення вузла, навівши на нього курсор і натиснувши на ліву клавішу миші, перетягнути і відпустити клавішу.

Побудувати вузол можна на основі його параметрів, таких як: назва, кількість токенів. Ввійшовши в режим редактування та клацнувши по вузлу лівою клавішею миші викличемо меню налаштувань вузла (рисунок 1.13). Де можна вказати ім’я вузла можна в полі ‘**Ім’я** та кількість токенів в полі ‘**Токен**’. Після цього потрібно натиснути ‘**Зберегти**’ в меню параметрів для

підтвердження вибору або ‘**Відмінити**’ для відміни. Елемент вузол після внесення даних прийме вигляд  .

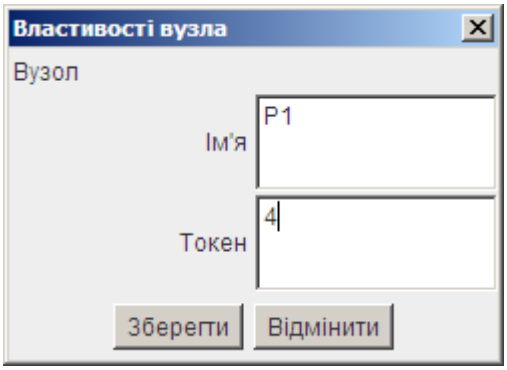


Рисунок 1.13 – Меню налаштувань властивостей вузла

Інструмент “Перехід”. Для створення переходу, потрібно клацнути на кнопку  . Далі клацаючи лівою клавішею миші можна розставити переходи у потрібних областях робочої області. Для виходу з режиму побудови переходу натискається кнопка  .

Кожному новоствореному переходу присвоюється ряд властивостей, а саме: ім’я, значення пріоритету, спосіб орієнтації, ймовірність спрацювання. Ввійшовши в режим редактування  та клацнувши по переходу лівою клавішею миші викличемо меню налаштувань переходу (рисунок 1.14). Де можна вказати ім’я переходу можна в полі ‘**Ім’я**’, значення пріоритету в полі ‘**Пріоритет**’, ймовірність спрацювання переходу в полі ‘**Ймовірність**’ та спосіб орієнтації графічного символу елемента переходу в полі ‘**Орієнтація**’. Після цього потрібно натиснути ‘**Зберегти**’ в меню параметрів для підтвердження вибору або ‘**Відмінити**’ для відміни. Графічний символ переходу буде відображено наступним чином  .

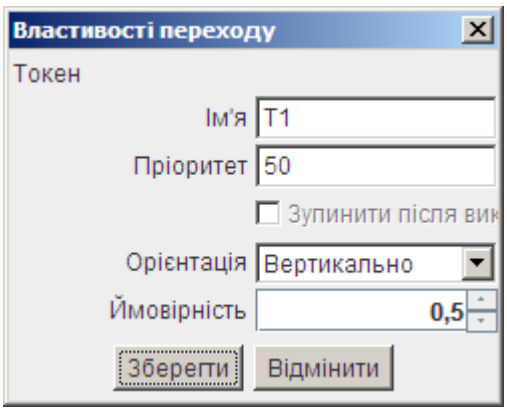
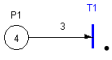


Рисунок 1.14 – Меню налаштувань властивостей переходу

Інструмент “Дуга”. Для створення зв’язку між об’єктами, клацніть на кнопку . Наведіть стрілку на вузол і клацніть по ньому лівою клавішею миші, потім наведіть вказівник на перехід і клацніть по ньому лівою клавішею миші. У результаті отримаємо дугу між вузлом і переходом.

Дуга з вузла в перехід і з переходу у вузол зображається суцільною лінією.

Якщо на шляху між об’єктами які необхідно зв’язати є інші об’єкти, то можна побудувати багатоточкову дугу в обхід цих об’єктів. Для цього спочатку потрібно клацнути лівою клавішею миші по вузлу, а потім клацати в порожніх місцях робочого вікна створюючи проміжки, зв’язані вузлами. Для завершення дуги необхідно клацнути по кінцевому об’єкту, у даному випадку переходу (вузлу).

Кожній новоствореній дузі присвоюється ряд властивостей, а саме: вага та прапорець інвертування. Ввійшовши в режим редактування та клацнувши по дузі лівою клавішею миші викличемо діалог налаштувань дуги (рисунок 1.15). Де можна вказати вагу дуги можна в полі ‘**Вага**’ та прапорець інвертування в полі ‘**Інвертувати**’. Після цього потрібно натиснути ‘**Зберегти**’ в меню параметрів для підтвердження вибору або ‘**Відмінити**’ для відміни. Графічний символ дуги буде відображено наступним чином 

.

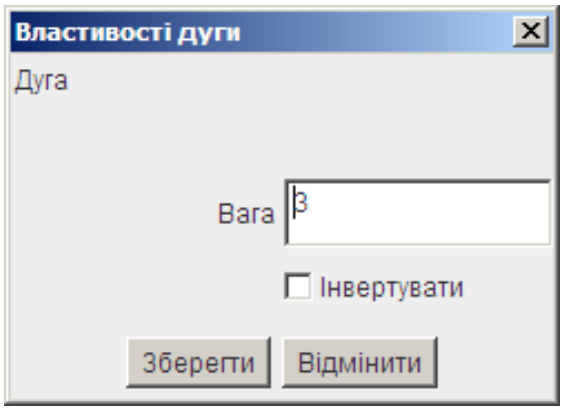


Рисунок 1.15 – Меню налаштувань властивостей дуги

Інструмент “Додати токен”. Для зручності конфігурації кількості токенів, що містяться у вузлах можна використати даний інструмент. Для активації інструменту потрібно натиснути на панелі інструментів кнопку . Після вибору вузлів клацаючи курсором по об’єктах їхня кількість токенів збільшується з кожним клацанням на одиницю. Під час видалення об’єкта стираються всі його вхідні і вихідні зв’язки з іншими об’єктами. Для виходу натискається кнопка .

Інструмент “Видалення”. Для видалення об’єктів з графа потрібно натиснути на панелі інструментів кнопку . Після вибору клацаючи курсором по об’єктах вони видаляються. Під час видалення об’єкта стираються всі його вхідні і вихідні зв’язки з іншими об’єктами. Для виходу натискається кнопка .

Процес симуляції роботи мережі Петрі можна запустити в автоматичному або покроковому режимі. Приклад мережі Петрі зображено на рисунку 1.16. Отримані результати дають змогу стверджувати, що розроблена підсистема працює правильно та коректно.

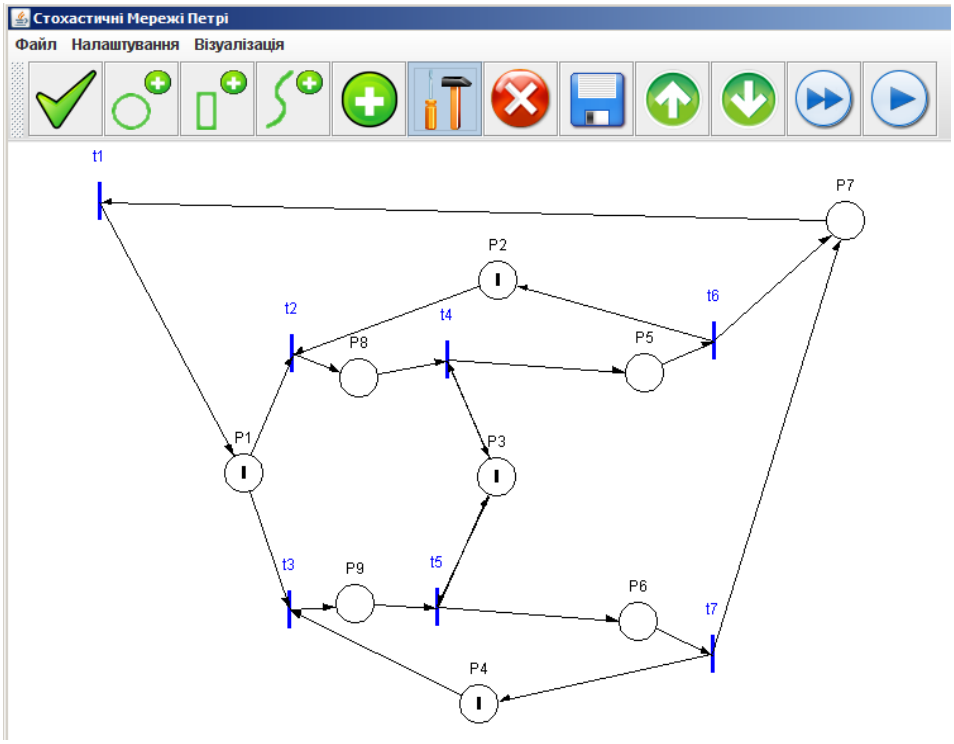


Рисунок 1.16 – Тестова мережа Петрі

При автоматичному режимі, якщо граф коректний, моделювання продовжуватиметься доти, поки не закінчаться токени у всіх вузлах, або до примусової зупинки, якщо граф циклічний. Зупинити процес моделювання можна натиснувши повторно кнопку . Після того, як моделювання зупинено, його можна продовжити або у автоматичному режимі, або у покроковому.

Для нормального продовження потрібно натиснути . Для покрокового режиму потрібно натиснути . Кожне натиснення здійснює один крок моделювання. У покроковий режим можна перейти без припинення, якщо відразу натискувати кнопку . В процесі симуляції графічні символи вузлів змінюють кількість доступних токенів.

Моделювання комп'ютерних комп'ютерних мереж в середовищі Packet Tracer.

Процес імітаційного моделювання комп'ютерних мереж складається з декількох етапів (рисунок 1.17): розробки моделі мережі, власне імітаційного моделювання (симуляції, simulation), збору і аналізу статистичних даних.

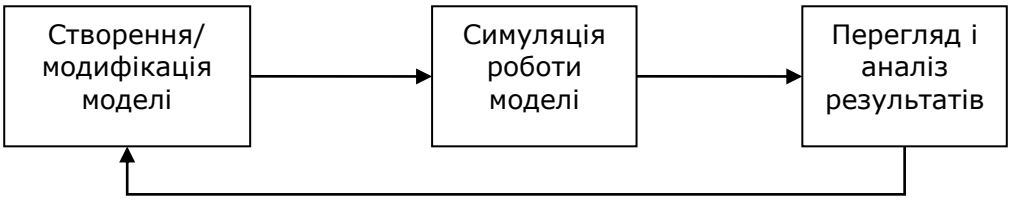


Рисунок 1.17 – Послідовність моделювання

Packet Tracer – програма імітаційного моделювання (симулятор) мережі передачі даних, що випускається фірмою Cisco Systems. Програма дозволяє створювати працездатні моделі мережі, налаштовувати (командами Cisco IOS) маршрутизатори та комутатори, перевіряти на працездатність топології. Програма дозволяє використовувати тільки обладнання Cisco Systems, зокрема, серії маршрутизаторів Cisco 1800, 2600, 2800 і комутаторів 2950, 2960, 3650.

Cisco Packet Tracer має два типи представлення схеми мережі: логічний і фізичний. Логічне представлення дозволяє користувачам будувати логічну топологію мережі, розміщувати, підключати, групувати мережеві пристрої. Фізичне представлення (простір) дозволяє графічно представити логічну мережу, даючи поняття про масштаби та налаштування обладнання.

Cisco Packet Tracer передбачає роботу візуалізацію поведінки мережі в двох режимах роботи: у реальному режимі часу та режимі імітації. Усі операції з мережею відбуваються в режимі реального часу. В режимі імітації користувач може бачити та контролювати часові інтервали, деталізацію передачі даних, розповсюдження даних через мережу.

Cisco Packet Tracer підтримує роботу, зокрема, наступних протоколів:

-на прикладному (Application) рівні: FTP, SMTP, POP3, HTTP, FTP, Telnet, SSH, DNS, DHCP, NTP, SNMP;

-на транспортному (Transport) рівні: TCP and UDP, IP Fragmentation, RTP;

-на мережевому (Network) рівні: BGP, IPv4, ICMP, ARP, IPv6, IPSec, RIPv1, L3 QoS, NAT;

- рівні доступу/інтерфейсу (Network Access/Interface): Ethernet (802.3), 802.11, PPPoE, STP, RSTP, WPA, EAP.

Cisco Packet Tracer дозволяє створювати мережу із маршрутизаторів, комутаторів, концентраторів, бездротових пристроїв, кінцевих пристроїв (робочих станцій, серверів, телефонів та ін.). Для генерації трафіку використовуються сервери із встановленим програмним забезпеченням. Маршрутизатори використовуються для пошуку оптимального маршруту передачі даних на підставі алгоритмів маршрутизації, наприклад, вибір маршруту (шляху) з найменшою кількістю транзитних вузлів. Комутатори призначені для об'єднання кількох вузлів у межах одного або кількох сегментів мережі, в т. ч. підтримки режиму віртуальних локальних мереж. Концентратори просто повторюють пакет, прийнятий одним портом усім іншим портам. Бездротові пристрої включають точки доступу, бездротові маршрутизатори, мобільні пристрої (див. рисунки нижче). Крім того Packet Tracer містить сервери DHCP, HTTP, TFTP, FTP, різні модулі до комп'ютерів і маршрутизаторів, різні типи кабелів.

Інтерфейс Cisco Packet Tracer. Головне вікно зображено на рисунку 1.18. Інтерфейс програми, розділений на області.

1. головне меню програми;

2. панель інструментів, частина яких просто дублює пункти меню;

3. перемикач між логічним і фізичним представленням;

4. панель інструментів, містить інструменти виділення, видалення, переміщення, масштабування об'єктів, а так само формування довільних пакетів;

5. перемикач між реальним режимом (real-time) і режимом симуляції;

6. панель з групами кінцевих пристроїв і ліній зв'язку;

7. панель мережевих пристроїв;

8. панель створення призначених користувацьких сценаріїв;

9. робочий простір.

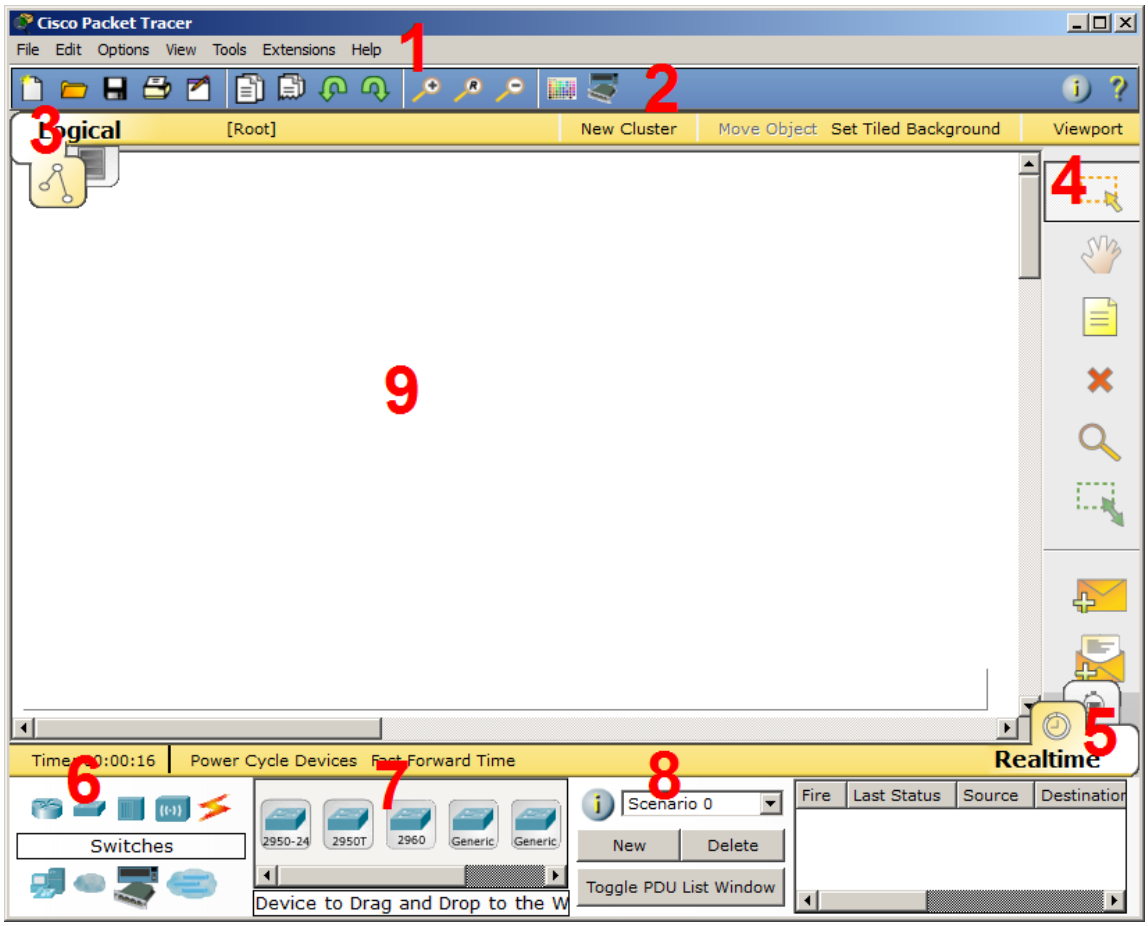


Рисунок 1.18 - Інтерфейс програми Cisco Packet Tracer

Головне меню програми має такі пункти:

• Файл - містить операції відкриття / збереження документів;

• Виправлення - стандартні операції «копіювати/вирізати, скасувати/повторити»;

• Настройки - говорить сама за себе;

• Вид - масштаб робочої області і панелі інструментів;

• Інструменти - колірна палітра і кастомізація кінцевих пристроїв;

• Розширення - майстер проектів, розрахований на багатокористувацький режим і плагіни;

• Допомога.

На рисунку 1.19 показано інтерфейс та послідовність вибору окремих пристроїв мережі.

Режим симуляції. Cisco Packet Tracer містить інструмент для симуляції роботи мережі, в якому можна імітувати і симулювати стан роботи мережі і практично будь-які мережеві події. Наприклад можна простежити, як реагуватиме мережа в разі збоїв або наприклад що станеться, якщо від'єднати будь-якої кабель або вимкнути живлення одного з мережевих пристроїв.

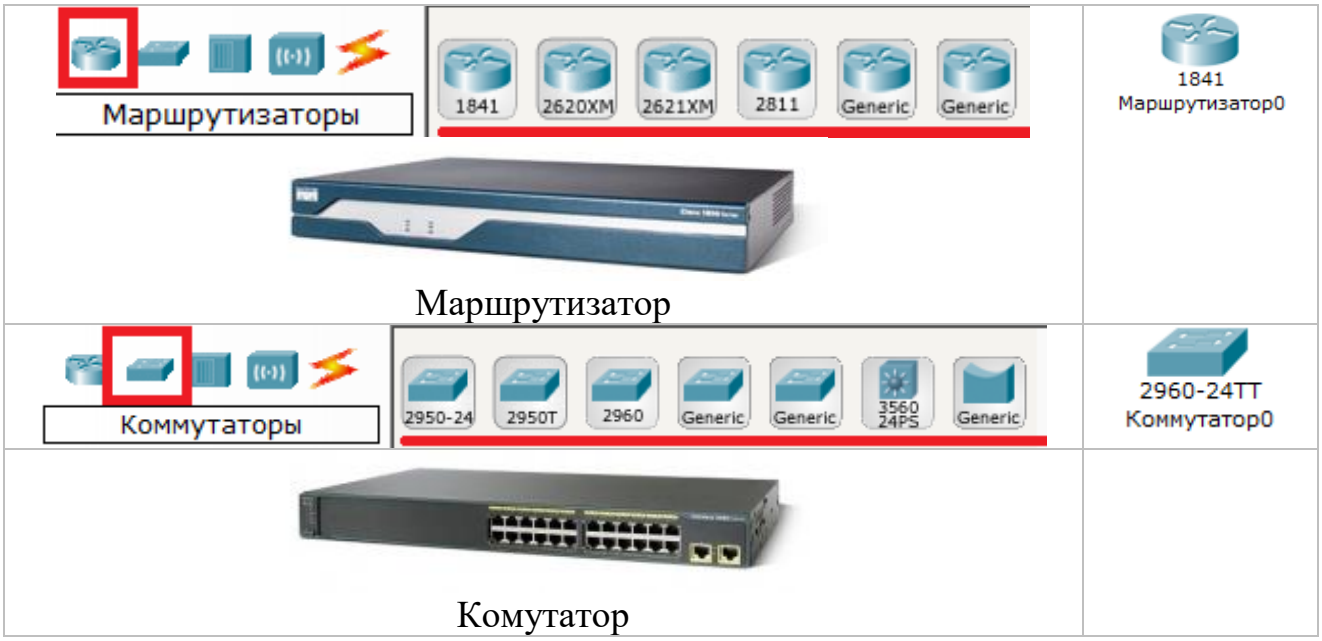


Рисунок 1.19 – Вибір Маршрутизаторів та комутаторів

Режим симуляції дозволяє простежити структуру пакета і переглянути, з якими параметрами пакет проходить за рівнями моделі OSI. Щоб перейти в режим симуляції потрібно клікнувши на іконку симуляції в правому нижньому кутку робочого простору (натиснути Shift + S). На рисунку 1.20 бачимо вікно подій, кнопка скидання (очищає список подій), управління відтворенням і фільтр протоколів. Наприклад при використанні ICMP протоколу програмою ping для перевірки зв’язку вікно симуляції буде виглядати наступним чином. Клік на пакеті покаже нам докладну інформацію. При цьому ми побачимо модель OSI. Відразу видно, що на 3-му рівні (мережевий) виник пакет на вихідному напрямі, який піде на другий рівень, потім до першого, на фізичну середу і передасться на наступний вузол (рисунок 1.21). А на іншій вкладці можна подивитися структуру пакета (рисунок 1.22).

З допомогою Cisco Packet Tracer можна реалізувати і базову політику мережної безпеки малого підприємства, зокрема:

- захист від несанкціонованого підключення до портів комутатора кожної підмережі (реалізується за допомогою організації парольної автентифікації, жорсткою прив'язкою MAC-адрес у таблиці MAC-адрес та конфігурування VLAN);

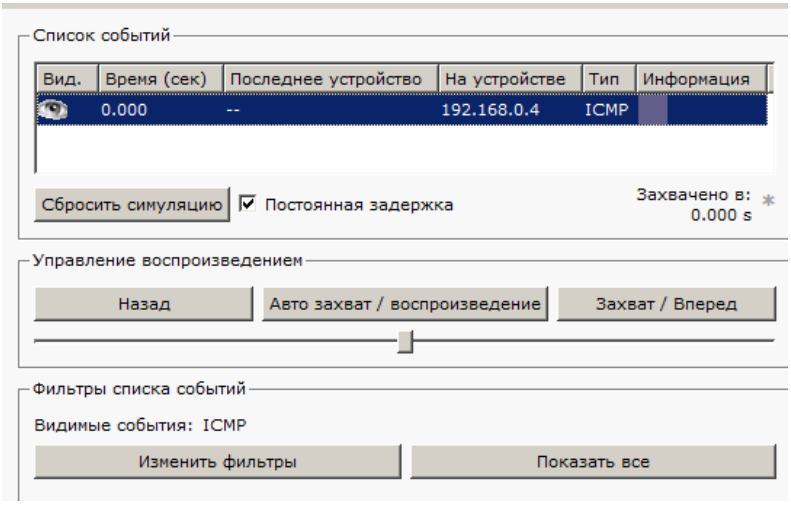


Рисунок 1.20 – Моніторинг роботи протоколів

- захист від несанкціонованого підключення до портів маршрутизатора чи серверів (реалізується за допомогою організації парольної автентифікації);

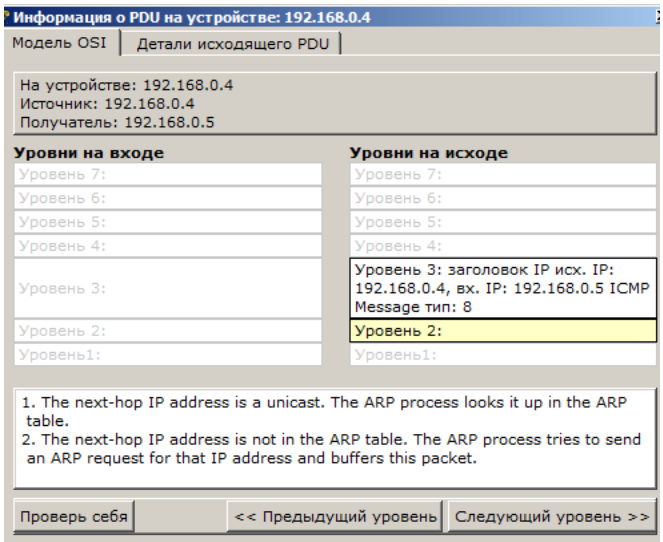


Рисунок 1.21 – Рівні моделі OSI

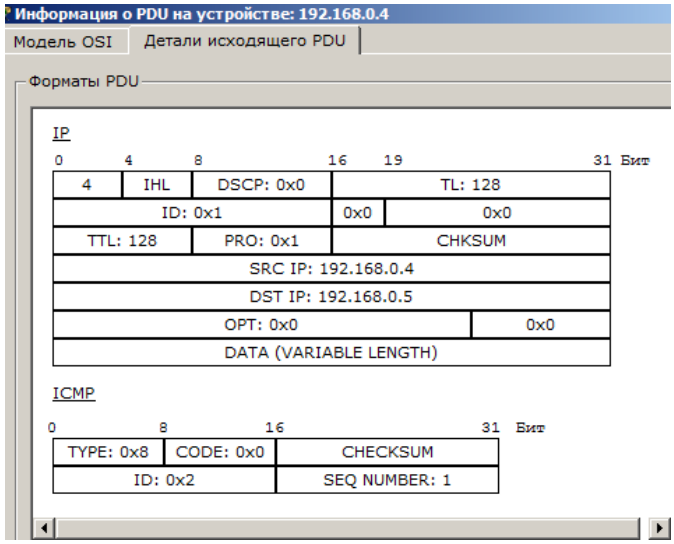


Рисунок 1.22 – Структура пакета

Cisco Packet Tracer також містить пристрій що виконує функції апаратного міжмережевого екрану - програмно-апаратний міжмережевий екран Cisco ASA 5505. Пристрій дозволяє розділити сегменти, мережі в яких обробляється конфіденційна інформація, персональні дані працівників, а також

відкрита інформація. Міжмережевий екран Cisco Systems серії ASA (Adaptive Security Appliances) - вископродуктивний пристрій для фільтрації трафіку, захисту від вірусів, черв'яків та різних видів інтернет-атак, організації шифрованих каналів через інтернет для зв'язку офісу та філій, а також для віддаленого підключення співробітників та організації extranet-доступу. Пристрій ASA 5505 реалізує механізм інспекції пакетів з урахуванням стану протоколу (stateful inspection firewall) на рівнях 2-7 OSI.

Контрольні питання до розділу 1:

1. Що таке комп’ютерна система?
2. Які основні компоненти комп’ютерної системи?
3. Назвіть основні класифікації комп’ютерних систем.
4. Дайте визначення поняття протокол. Наведіть приклади проколів.
5. Дайте визначення принципів комутації каналів та комутації пакетів.
6. Опишіть апаратне забезпечення комп’ютерної мережі
7. Опишіть типи серверів комп’ютерної мережі
8. Дайте визначення призначення і функцій комутаторів та маршрутизаторів.
9. Опишіть програмні засоби комп’ютерної мережі
10. Дайте визначення повнозв’язної топології комп’ютерної мережі. Наведіть переваги і недоліки.
11. Дайте визначення сітчастої (mesh) топології комп’ютерної мережі. Наведіть переваги і недоліки.
12. Дайте визначення топології комп’ютерної мережі типу шина. Наведіть переваги і недоліки.
13. Дайте визначення топології комп’ютерної мережі типу зірка. Наведіть переваги і недоліки.
14. Дайте визначення кільцевої топології комп’ютерної мережі. Наведіть переваги і недоліки.

15. Дайте визначення комутованої топології комп’ютерної мережі. Наведіть переваги і недоліки.

16. Дайте визначення трьохрівневої архітектури корпоративної мережі.

17. Назвіть основні програмні засоби моделювання комп’ютерних систем і мереж та їх основні можливості.

РОЗДІЛ 2. ОСНОВНІ ПОНЯТТЯ ТА ВИЗНАЧЕННЯ ТЕОРІЇ РОЗВЯЗАННЯ ЗАДАЧ БАГАТОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ

2.1. Основні елементи теорії оптимізації

Під терміном “оптимізація”, як правило, в техніці розуміють процес, який дає можливість отримати найкраще рішення в заданих умовах. Слід зазначити що “найкраще рішення” деякої задачі з точки зору певної особи, яка визначає, що це є саме “найкраще рішення”, чи по іншому його називають оптимальне. З точки зору іншої особи, це рішення може бути вже не “найкраще”, а близьке до оптимального. Тому у визначенні “найкращого рішення” присутній певний суб’єктивізм, який, зрозуміло, що по можливості в процесі розв’язання оптимізаційних задач бажано усунути.

Відповідно, в процесі розв’язання складних технічних оптимізаційних задач, оцінку “найкращого рішення” має приймати проектувальник (інженер), яка має значний досвід в цій області чи колектив досвідчених науковців.

Наступним моментом вище наведеного визначення є те, що це отримане оптимальне рішення “в заданих умовах”. В інших умовах отриманий результат вже може і не бути найкращим, оптимальним.

Загалом існує багато визначень терміну “оптимізації”, але усі вони містять вищезазначені елементи, і можуть відрізнятися врахуванням лише специфіки та особливості області застосування.

Зокрема, деякі з них:

Оптимізáція — 1. процес вибору найкращого варіанта з можливих. 2. Процес приведення системи до найкращого (оптимального)стану [44].

Оптимізацією (від латів. optimum - найкраще) називається процес знаходження екстремуму (максимуму або мінімуму) певної функції або вибору найкращого (оптимального) варіанту з множини можливих.

Оптимізáція (від латів. optimus - найкращий) — процес надання будь-чому найвигідніших характеристик, співвідношень (наприклад, оптимізація

виробничих процесів і виробництва) [45].

З досвіду відомо що, в більшості випадків в процесі розв'язання практичних оптимізаційних задач вдається лише покращити вже існуюче рішення, а не знайти "найкраще". Відповідно, необхідно ставитися до цього з розумінням, і це пов'язано з тим, що процес оптимізації є надзвичайно складним на який впливають різні за природою параметри і кількість яких є надзвичайно велика. Більше того, досить часто, ми не знаємо чи існує те "найкраще рішення" взагалі в даній області зміни проектних параметрів (задані умови) і де його шукати.

Перш ніж приступити до обговорення питань оптимізації, введемо ряд визначень.

Проектні параметри (шукані змінні). Цим терміном позначають незалежні змінні параметри, які повністю і однозначно визначають вирішувану задачу проектування.

Проектні параметри - невідомі величини, значення яких обчислюються в процесі оптимізації. Як проектні параметри можуть бути будь-які основні або похідні величини, які призначені для кількісного опису об'єкта проектування. Так, це можуть бути невідомі значення об'єму оперативної пам'яті, тактова частота мікропроцесора, довжини мікродавача, маса чутливого елемента мікродавача, температура, тощо. Число проектних параметрів характеризує ступінь складності даного завдання проектування деякого технічного процесу чи об'єкту. Звичайне число проектних параметрів позначають через n, а самі проектні параметри через x з відповідними індексами. Таким чином n проектних параметрів певного завдання позначатимемо через

$$x_1, x_2, x_3,\ldots, x_n .$$

Додаткову інформацію про класифікація та особливості проектних параметрів можна знайти в роботах [46, 47].

В процесі вивчення та формалізації оптимізаційних задач оперують такими складовими, як критерій оптимізації (КО) та цільова функція (ЦФ).

Критерій оптимізації (КО) – це є певна властивість об'єкта

проектування [47]. **Критерій оптимальності** – показник чи система показників якості роботи деякої системи, значення якої має бути мінімізовано (максимізовано) [48]. Для прикладу: вартість виготовлення мікродавача, максимальна швидкість автомобіля, максимальна потужність підсилювача низької частоти, швидкодія мікроконтролера та ін. Визначає критерій оптимізації, як правило, інженер чи науковець, який володіє знаннями в певній галузі та має досвід. В кожній оптимізаційній задачі необхідно знайти найбільше чи найменше значення КО.

Для того, щоб оперувати критерієм оптимізації в процесі розв'язання оптимізаційної задачі, необхідно мати справу з його кількісною оцінкою, а не з якісною. Тому можна сказати так, що формалізують КО, тобто записують в математичній формі залежність КО від проектних параметрів. Відповідно, даний запис називається **цільовою функцією.**

Цільова функція, **функція цілі –** функція, найбільше чи найменше значення якої шукається в задачах математичного програмування з врахуванням наявних обмежень [48].

В процесі розв'язання оптимізаційної задачі цільова функція дає змогу кількісно порівняти два чи більше альтернативних рішення. З математичної точки зору цільова функція описує деяку $(n+1)$ - вимірну поверхню. Її значення визначається проектними параметрами

$$f = f(x_1, x_2, ..., x_n).$$

Більше інформації про критерії оптимальності та цільові функції можна почерпнути з робіт [39 - 43].

Прикладами цільових функцій, які часто зустрічаються в інженерній практиці, можуть бути математичний вираз для визначення вартості виробу, його ваги, міцності, ККД та інші. У випадку, коли цільова функція містить тільки один проектний параметр, то цільову функцію можна представити з допомогою кривої на площині. Якщо проектних параметрів два, то цільова функція зображуватиметься поверхнею в просторі трьох змінних. При трьох і більше проектних параметрах поверхні, що задаються цільовою функцією,

називаються **гіперповерхнями** і не піддаються зображенню звичайними засобами. Топологічні властивості поверхні цільової функції відіграють велику роль в процесі оптимізації, оскільки від них залежить вибір найбільш ефективного методу пошуку оптимальних рішень.

Цільова функція в ряді випадків може приймати найнесподіваніші форми. Наприклад, її не завжди вдається виразити в математичній формі, в інших випадках вона може бути кусково-гладкою функцією. Для задання цільової функції іноді може знадобитися таблиця технічних даних (наприклад, таблиця стану водяної пари) або може знадобитися провести експеримент. В ряді випадків проектні параметри приймають тільки цілі значення. Прикладом може бути комірок пам’яті персонального комп’ютера чи кількість мікропроцесорів. Іноді проектні параметри мають тільки два значення - так чи ні. Якісні параметри, такі як задоволення, яке випробовує покупець, що придбав виріб, надійність, естетичність, теж можливо враховувати в процесі оптимізації, хоча їх складно охарактеризувати кількісно. Проте в якому б вигляді не була представлена цільова функція, вона має бути однозначною функцією проектних параметрів.

В ряді оптимізаційних задач потрібне введення більше однієї цільової функції. Іноді одна з них може виявитися несумісною з іншою. Прикладом служить проектування ноутбуків, коли одночасно потрібно забезпечити максимальну потужність, мінімальну вагу та мінімальну вартість. В таких випадках розробник має ввести систему пріоритетів і поставити у відповідність кожній цільовій функції деякий безрозмірний множник. В результаті з'являється так звана “функція компромісу”, що дає змогу в процесі оптимізації користуватися однією складною цільовою функцією (в даному випадку розробник має справу з задачею багатокритеріальної оптимізації оскільки присутні декілька критеріїв оптимізації).

Якщо порівняти між собою КО і ЦФ, то можна стверджувати, що критерій оптимальності визначає якісну оцінку якоїсь властивості об’єкта проектування, або його властивість чи властивості, а цільова функція кількісну

оцінку критерію чи групи критеріїв оптимізації.

Пошук мінімуму і максимуму. Одні алгоритми оптимізації пристосовані для пошуку максимуму, інші - для пошуку мінімуму. Проте незалежно від типу розв’язуваної задачі на екстремум можна користуватися одним і тим же алгоритмом, оскільки задачу мінімізації можна легко перетворити на задачу пошуку максимуму, помінявши знак цільової функції на зворотний. Чим, на практиці, досить часто користуються проектувальники в процесі побудови технічних пристроїв.

Множина допустимих рішень. Так називається область, визначена всіма *n* проектними параметрами. Простір рішення не такий великий, як може здаватися на перший погляд, оскільки він зазвичай обмежений рядом умов, пов'язаних з фізичною суттю завдання. Обмеження можуть бути такими сильними, що задача не матиме жодного задовільного розв’язання. Слід зазначити, що дуже часто у зв'язку з обмеженнями оптимальне значення цільової функції досягається на одній з меж області множини допустимих розв’язань задачі.

Локальний оптимум. Так називається точка простору рішень, в якій цільова функція має найбільше значення в порівнянні з її значеннями в усіх інших точках її околу.

Досить часто простір проектування містить багато локальних оптимумів і слід дотримуватися обережності, щоб не прийняти перший з них за оптимальне розв’язання задачі.

Глобальний оптимум. Глобальний оптимум - це оптимальне рішення для всієї множини допустимих рішень. Воно краще за всі інші рішення, відповідні локальному оптимуму, і саме його шукає розробник. Можливий випадок декількох рівних глобальних оптимумів, розташованих в різних частинах простору проектування.

2.2. Особливості об’єкта оптимізації

Для більш повного уявлення про оптимізаційні задачі, зупинимося докладніше на характеристиках об'єкта оптимізації і сукупності даних, необхідних для оптимізації об'єкта.

Об'єкти оптимізації можна класифікувати по ряду ознак. До таких ознак відносяться:

- число оптимізованих параметрів об'єкта;
- число екстремумів характеристики об'єкта, використаної як показник якості;
- обсяг апріорної інформації про об'єкт;
- спосіб математичного опису об'єкта.

По числу змінних параметрів розрізняють одно- і багатопараметричні об'єкти. В залежності від кількості екстремумів об'єкти поділяються на одноекстремальні і багатоекстремальні, причому в останньому випадку оптимізаційна задача зводиться до пошуку глобального екстремуму, тобто мінімального мінімуму і максимального максимуму.

В залежності від обсягу апріорної інформації, можуть бути екстремальні об'єкти, для яких існує математичний опис, і залежність показника якості Q від параметрів оптимізації X відома. Для таких об'єктів є достатній обсяг апріорної інформації. Існує також великий клас об'єктів, для яких немає ніякого математичного опису. Малий обсяг апріорної інформації про подібні об'єкти послужив приводом називати їх об'єктами типу "чорний ящик".

Сукупність параметрів, які оптимізуються, утворює вектор параметрів об'єкта оптимізації і характеризує вид оптимізаційної задачі. Якщо число параметрів, які оптимізуються, більше одиниці ($n>1$), то задача відноситься до багатопараметричних, а при $n=1$ вона переходить в одно параметричну оптимізаційну задачу.

Сукупність показників якості утворить вектор показників якості об'єкта $F = \{f_1, ..., f_m\}$.

При необхідності характеризувати об'єкт групою показників якості, задача класифікується як багатокритеріальна чи векторна, якщо ж для оптимізації обраний лише один показник якості, то задача переходить в однокритеріальну чи скалярну.

2.3. Постановка оптимізаційної задачі

В процесі побудови оптимізаційної задачі (ОЗ) необхідно інженерові чи науковцеві вирішити такі завдання:

1) Вибір та визначення критерію оптимізації;

2) Визначити проектні параметри;

3) Побудувати цільову функцію (модель залежності між КО та проектними параметрами);

4) Визначити обмеження;

5) Вибір методу розв’язання оптимізаційної задачі.

Надалі будемо вважати, що ОЗ сформульована, якщо вона містить усі вище перераховані елементи. В ряді випадків у процесі автоматизованого проектування деякого технічного об’єкта КО можна отримати з технічного завдання, де записані вимоги до приладу чи технологічного процесу його виготовлення. Проблеми виникають в тих випадках, коли КО є багато і необхідно вибрати з них один, або декілька.

Не менш важливим питанням є визначення переліку множини проектних параметрів. Кількість проектних параметрів є необмеженою. В даному випадку йде мова про параметри, які мають суттєвий вплив на критерій оптимальності (тобто якусь властивість об’єкта проектування). Тобто, на практиці необхідно

відкинути ті проектні параметри, вплив яких виражається значеннями на рівні похибки вихідного параметра.

Дане питання детально розглядається в теорії побудови математичних моделей об’єкта проектування. Лишень з тою відмінністю, що замість вихідних параметрів ми оперуємо критерієм оптимальності [47].

Дещо інша ситуація з етапом побудови цільової функції. Все дуже красиво виглядає на лекціях, коли в якості прикладу береться стандартна задача і без проблем будується ЦФ. На практиці під час проектування технічного пристрою ще невідомо остаточно його структуру, параметри елементів структури, тощо. Тому побудувати ЦФ надзвичайно складно. Тим більше побудувати адекватну модель залежності вихідного параметра від вхідних, а на основі моделі – ЦФ. Інколи взагалі така залежність невідома, або побудувати математичну залежність КО від проектних параметрів неможливо. Тому, необхідно володіти певним досвідом у цій галузі і ставитись до цього етапу надзвичайно серйозно. Оскільки від адекватності розробленої математичної моделі залежності між критерієм оптимальності та проектними параметрами залежить точність отриманих результатів оптимізації.

Сукупність обмежень відіграє дуже важливу роль при постановці і розв’язання оптимізаційної задачі. Найбільш часто зустрічаються обмеження виду рівності ($x_i = x_{i0}$) чи нерівності ($x_{imin} \square x_i \square x_{imax}$). Обмеження накладаються на змінні проектні параметри, а також на показники якості. Зміст цього етапу полягає в тому, що часто якість системи характеризується не одним, а групою показників якості, тому якщо система оптимізується по одному показнику якості, то інші можуть досягти такої величини. Отже, якщо обраний якийсь параметр системи як критерій якості, то на інші показники якості і змінні параметри накладаються обмеження. Якщо в задачі багатокритеріальної (векторної) оптимізації перевести частину показників якості в розряд обмежень, то можна її звести до однокритеріальної (скалярної) задачі.

Четвертий етап, в більшості випадків не представляє складності, оскільки обмеження визначаються вимогами технічного завдання на проектування.

Завершальними етапами розв’язання оптимізаційної задачі є вибір методу оптимізації і сама оптимізація, тобто знаходження оптимальних значень проектних параметрів (вхідних змінних), при яких цільова функція (якості) досягає мінімуму чи максимуму.

Вибір методу оптимізації залежить, в основному, від виду цільової функції, що у свою чергу визначається особливостями об'єкта оптимізації: його складністю, структурою, математичним описом об'єкта, наявністю апріорних даних про об'єкт проектування.

Найбільш повно розроблені методи оптимізації, які отримали назву "методи математичного програмування". До них відносяться методи лінійного, геометричного, випуклого, нелінійного, стохастичного програмування.

Для застосування того чи іншого методу необхідно, щоб цільова функція відповідала певним вимогам, а саме: її аналітичне вираз повинен бути певного виду.

Варто мати на увазі, що структура вираження цільової функції цілком залежить від особливостей математичного опису об'єкта оптимізації, тому що функція якості знаходиться із математичного опису об'єкта (його математичної моделі).

Як раніше вказувалося, опис об'єкта може бути представлено або у вигляді аналітичних співвідношень, або у вигляді алгоритмів. Остання форма опису практикується при описі складних об'єктів, коли доцільне застосування методів машинного моделювання. Тому не завжди вдається отримати цільову функцію у формі явної аналітичної залежності.

Усе це ускладнює в багатьох випадках застосування добре розроблених методів математичного програмування.

Для розв’язання багатопараметричних оптимізаційних задач при алгоритмічному методі опису об'єктів проектування можуть бути використані пошукові методи оптимізації.

Отже, в якості прикладу наведемо формулювання однокритеріальної оптимізаційної задачі.

Необхідно знайти такі значення проектних параметрів x_1, x_2, x_3, x_4 та x_5 об'єкта дослідження, для яких цільова функція досягає максимального значення

$$\max f(x_1, x_2, x_3, x_4, x_5),$$

при цьому виконуються такі обмеження нерівності та рівності:

$g_1(x_1, x_2, x_3, x_4, x_5) \geq 10,$

$g_2(x_1, x_2, x_3, x_4, x_5) \leq 8,$

$h_1(x_1, x_2, x_3, x_4, x_5) = 12,$

$h_2(x_1, x_2, x_3, x_4, x_5) = 3,$

$x_1, x_2, x_3, x_4, x_5 \geq 0.$

В наведеному прикладі кількість проектних параметрів рівна п'яти. При чому їхні значення можуть бути більшими, або рівними нулю. Для обмеження у формі рівностей і два обмеження у формі нерівностей.

2.4. Причини багатокритеріальності в задачах оптимального проектування

Майже будь-яке складне технічне завдання прийняття рішення є багатокритерійним, оскільки при виборі найкращого варіанту доводиться враховувати багато різних вимог, і серед цих критеріїв зустрічаються такі, що суперечать один одному. Проте, майже всі математичні методи оптимізації призначені для знаходження екстремуму однієї функції – тобто для знаходження оптимального розв'язання за одним критерієм. Тому найчастіше намагаються звести багатоцільове завдання до одноцільового. Ця процедура в більшості випадків призводить до серйозного спотворення сутності проблеми і, отже, до невиправданої заміни одного завдання іншим.

Якщо при розв’язанні одноцільових задач методологічних проблем не виникає, а можливі тільки обчислювальні труднощі, то інакше йде справа з багатокритерійними задачамиТут основні нюанси пов'язані з наступною проблемою: що слід вважати якнайкращою альтернативою в завданні з декількома цільовими функціями, які суперечливі між собою, і досягають максимуму в різних точках множини альтернатив?

Цілі, багатокритеріальної задачі, можуть знаходитися одна з одною в наступних відносинах:

1. Цілі взаємно нейтральні. В такій ситуації система може стосовно окремих цілей характеризуватися і розглядатися незалежно.

2. Цілі кооперуються. Тут, як правило, систему вдається розглянути стосовно однієї мети, а інші досягаються одночасно.

3. Цілі конкурують. В цьому випадку одну з цілей можна досягти лише за рахунок іншої.

Якщо цілі частково нейтральні, частково кооперовані і частково конкурують між собою, то завдання формулюється таким чином, що потрібно брати до уваги тільки конкуруючі цілі. Розгляд нейтральних або кооперативних цілей не представляє особливих труднощів, так що проблеми, орієнтовані на множину цілей, перш за все повинні бути розглянуті в частині конкуруючих цілей, якщо всі вони разом не можуть бути виражені одновимірним параметром.

Найбільш загальною математичною моделлю прийняття оптимального рішення є завдання багатопараметричної оптимізації. Наведемо декілька причин, які приводять до багатокритеріальних завдань.

1. Однією з причин, що приводить до багатакритеріальності, є **множина технічних вимог**, які пред'являються до характеристик проектованого пристрою.

2. Наступною причиною багатакритеріальності є **необхідність забезпечення оптимальності проектованого пристрою за різних умов його функціонування**, тобто забезпечення екстремальних значень критерію

оптимальності при невизначеності умов, в яких доводиться працювати пристрою.

3. В тих випадках, коли проектований пристрій складається з декількох взаємозв'язаних вузлів і блоків, оптимальність всього пристрою визначається ефективністю і якістю його окремих частин, кожна з яких може бути охарактеризована, принаймні, хоч би одним власним критерієм оптимальності $f_i(\vec{x})$.

В цьому випадку функціонування всього пристрою можна вважати якнайкращим, якщо за рахунок вибору керованих параметрів $\vec{x}$ забезпечуються екстремальні значення всіх приватних критеріїв оптимальності як основних підцілей однієї загальної мети проектування.

2.5. Постановка задачі багатокритеріальної оптимізації

В загальному випадку задачами багатокритеріальної оптимізації прийнято вважати такі задачі, де існує декілька критеріїв оптимізації.

Одне з можливих визначень наведено наступне [49, 50]:

Багатокритеріальна оптимізація або програмування (англ. Multi-objective optimization) — це процес одночасної оптимізації двох або більше конфліктуючих цільових функцій в заданій області визначення.

Багатокритеріальності проблема – задача вибору рішення при наявності декількох функцій цілі [48].

Задача багатокритеріальної оптимізації зустрічаються в багатьох галузях науки та техніки.

Постановка багатокритеріальної оптимізації вже оперує з такими елементами, як цільові функції, а не критерії.

В загальному випадку задачу багатокритеріальної оптимізації можна сформулювати наступним чином [51, 52].

Знайти такі значення проектних параметрів, для яких забезпечуються екстремальні значення цільових функцій:

$$\min(\max) f_i(\vec{x}),\ (i=1,n),\ \ j=1,m, \tag{2.1}$$

при виконанні наступної системи обмежень

$$g_k(\overline{X}) \le \alpha_k,\ (k=1,l), \tag{2.2}$$

$$h_m(\overline{X}) = \beta_m,\ (m=1,j),$$

$$b_1 \le x_j \le b_m .$$

де вектор розв'язків $\vec{x} = (x_1, x_2, ..., x_n)^T$ належать до не порожньої області визначення D.

Отже задача багатокритеріальної оптимізації полягає у пошуку вектора цільових змінних, який задовільняє накладеним обмеженням та оптимізує векторну функцію, елементи якої відповідають цільовим функціям. Ці функції утворюють математичне описання критерію задовільності та, зазвичай, взаємно конфліктують. Звідси, «оптимізувати» означає знайти такий розв'язок, за якого значення цільових функцій були б прийнятними для постановника задачі [52].

Контрольні запитання до розділу 2

1. Що таке критерій оптимізації?
2. Що Ви розумієте під цільовою функцією?
3. Яка задача називається задачею багатокритеріальної оптимізації?
4. Які види критеріїв оптимізації Ви знаєте?
5. Який критерій оптимізації називається векторним?
6. Який критерій оптимізації називається скалярним?
7. Що таке проектні параметри?
8. Що таке локальний оптимум?
9. Що таке глобальний оптимум?

10. Які причини багатокритеріальності Ви знаєте?

11. Наведіть приклад багатокритеріальної оптимізаційної задачі?

12. Які методи розв’язання задач багатокритеріальної оптимізації Ви знаєте?

13. На основі якої інформації визначаються проектні параметри?

14. На основі якої інформації визначаються критерії оптимізації?

15. Яка різниця між критерієм оптимізації та цільовою функцією?

16. Які основні кроки включає процес розв’язання оптимізаційних задач?

17. Які види обмежень Ви знаєте?

18. Наведіть приклад багатокритеріальної оптимізаційної задачі?

19. Наведіть приклад взаємно нейтральних критеріїв оптимальності?

20. Наведіть приклад критеріїв оптимальності, які кооперуються в процесі рішення деякої задачі?

21. Наведіть приклад критеріїв оптимальності, які взаємнонейтральні в процесі рішення задачі проектування?

РОЗДІЛ 3. МЕТОДИ РОЗВ’ЯЗАННЯ БАГАТОКРИТЕРІАЛЬНИХ ЗАДАЧ НА ОСНОВІ ОДНОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ

В другому розділі цієї роботи було зазначено, що **задачі багатокритеріальної оптимізації** (ЗБО) - це такі задачі оптимізації, в яких використовується не один, а кілька критеріїв оптимальності. На практиці такі задачі виникають, коли проектований об'єкт (чи вихідний параметр об’єкта розроблення) не може бути описаний однокритеріальною залежністю, або в тій ситуації, коли об'єднати окремі критерії в єдиний узагальнений критерій не представляється можливим. В ряді випадків таке об'єднання критеріїв у єдиний узагальнений критерій застосовується і воно буде розглянуто в третьому розділі даної роботи. Але це об'єднання, як правило, буває формальним, або по іншому його називають штучним, яке не несе ніякого фізичного навантаження.

З математичної точки зору не існує ідеального методу або способу розв’язання таких задач. Кожний з них має свої певні переваги та недоліки і область застосування. В цьому розділі розглянемо методи, які можна звести до одного критерія та методи, які дають змогу попередньо розв’язати задачі оптимізації як однокритеріальні з наступними перетвореннями, що дають змогу вважати отримані результати, як результати розв’язання ЗБО. Отже розглянемо більш детально такі методи розв’язання багатокритеріальних задач оптимізації, а саме:

- метод головної компоненти (МГК);
- метод комплексного критерію;
- лексикографічний метод (ЛМ);
- метод поступок (МП);
- метод ідеальної точки (МІТ);
- метод умовного центра мас (МУЦМ).

3.1. Метод головної компоненти

Ідея методу головної компоненти полягає в тому, що узагальнений критерій (часом його називають критерій якості) зв’язується з одним із критеріїв, що вибраний в ролі основного (головного), а на основі інших критеріїв будують обмеження. В такому випадку отримують задачу однокритеріальної оптимізації для розв’язання якої використовують відповідні методи на основі виду та особливостей цільової функції обмеження. Блок-схема алгоритму роботи цього методу наведена на рисунок 3.1. Наведена блок схема включає такі основні кроки, як вибір основного критерію оптимальності, побудова обмежень на основі інших критеріїв та, відповідно, розв’язання отриманої однокритеріальної задачі.

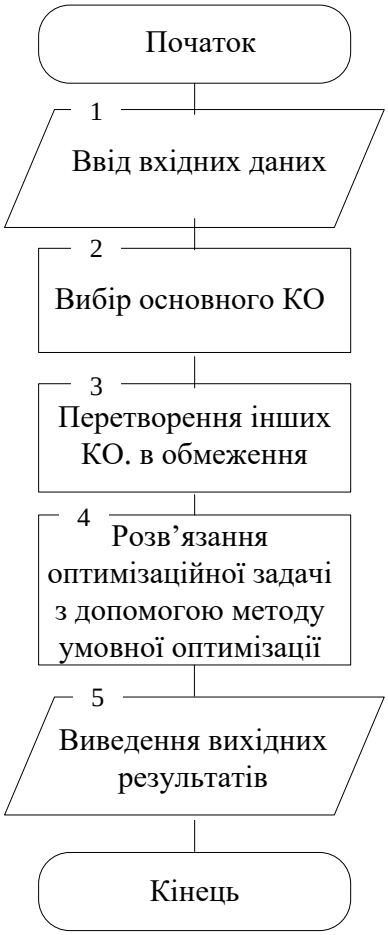


Рисунок 3.1 – Блок-схема алгоритму застосування методу головної компоненти

В загальному випадку припустимо, що нам необхідно в процесі автоматизованого проектування деякого складного об'єкта розв'язати ЗБО, яка має n критеріїв оптимальності:

$$F = \{\max f_1(\overline{X}), \max f_2(\overline{X}), ..., \min f_{n-1}(\overline{X}), \min f_n(\overline{X})\}, \quad (3.1)$$

де $f_1(\overline{X})$ - перша цільова функція, яка описує, для прикладу, продуктивність деякого об'єкту, $f_2(\overline{X})$ - друга цільова функція, яка описує надійність цього технічного пристрою, … , $f_n(\overline{X})$ - n-а цільова функція, яка описує його собівартість.

З обмеженнями наступного виду

$$g_k(\overline{X}) \le \alpha_k, \ (k = 1, l), \quad (3.2)$$

$$h_m(\overline{X}) = \beta_m, \ (m = 1, j),$$

де $g_k(\overline{X}) \le \alpha_k$ - k-те обмеження нерівності, $h_m(\overline{X}) = \beta_m$ - m-те обмеження рівність, α_k .та β_m - деякі константи, l - кількість обмежень нерівностей, j - кількість обмежень рівностей.

Припустимо, що на основі деякого методу розробник визначив, що головним критерієм оптимальності є собівартість його виготовлення ($f_n(\overline{X})$). Відповідно, задача багатокритеріальної оптимізації, яку необхідно розв'язати з допомогою методу головної компоненти, буде сформульована наступним чином:

знайти

$$\min f_n(\overline{X}), \quad (3.3)$$

при виконанні таких обмежень:

$$f_1(\overline{X}) \ge A_1, \ f_2(\overline{X}) \ge A_2, \ \dots, \ f_{n-1}(\overline{X}) \le A_{n-1}, \quad (3.4)$$

$$g_k(\overline{X}) \le \alpha_k, \ (k = 1, l),$$

$$h_m(\overline{X}) = \beta_m, \ (m = 1, j),$$

$$A_{<i>} = < A_1, A_2, ..., A_n >,$$

де $A_i (i = 1, 2, ..., n)$ - граничні значення критеріїв оптимальності (константи).

Отже метод головної компоненти полягає в довільному виборі одного із критеріїв в якості головного, по якому проводиться оптимізація і вибирається рішення. При цьому інші компоненти переводяться в розряд обмежень.

Цей метод простий, наочний і часто застосовується при проектуванні, в машинобудівній практиці, однак принциповим його недоліком є довільність у виборі головного критерію. Можна навести багато прикладів із історії науки і техніки, коли довільний і невірний вибір цього критерію призводив до трагічних наслідків чи, щонайменше, до малоефективних результатів.

В літературі, присвяченій питанню оптимізації програм, в якості головного показника вибирають, наприклад, їх собівартість чи термін експлуатації. При оптимізації металоконструкцій - металоємність. Коли вирішуються технологічні задачі прийнято використовувати в якості головного критерію продуктивність праці, продуктивність обладнання.

Для кращого розуміння особливостей застосування цього методу, наведемо приклад ЗБО, яка включає три цільові функції та три обмеження. Отже необхідно знайти такі значення проектних параметрів x_1, x_2, x_3, x_4 та x_5, для яких цільові функції приймають максимально можливі значення:

$$\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4, \tag{3.5}$$

$$\max f_2(\bar{x}) = 3x_1 - x_2,$$

$$\max f_3(\bar{x}) = x_1 - 4x_2,$$

при виконанні таких умов:

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$$x_1,\ x_2,\ x_3,\ x_4,\ x_5 \geq 0.$$

Припустимо, що головним критерієм є перший, а другий та третій, відповідно, необхідно ввести в список обмежень. Оскільки, в другому та третьому КО шукаємо максимум, то цільові функції мають бути не менше

деякої уявної величини (що в процесі проектування може визначатися технічним завданням на виріб).

Для прикладу, в процесі проектування автомобілів, це може бути його максимальна швидкість (тобто більшою може бути, але не меншою деякого значення) у випадку проектування підсилювача звукової частоти – вихідна потужність у випадку розроблення комп'ютера – об'єм пам'яті , та ін.

Повертаючись до нашого прикладу приймемо, що значення другого КО, має бути більше-рівне чотирьох, а третього – шести. Тоді два додаткові обмеження можна записати у наступній формі:

$$3x_1 - x_2 \geq 4,$$

$$x_1 + 4x_2 \geq 6.$$

Задача, яку в кінцевому випадку необхідно розв'язати, буде включати цільову функцію $\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4$ та обмеження з задачі (3.5) плюс обмеження.

Неозброєним оком видно, що ця задача є задачею лінійного програмування і для розв'язання якої використаємо метод великих чисел.

В процесі застосування М-методу, модифікуємо обмеження, тобто перетворимо 4-е та 5-е обмеження нерівності в рівності, при цьому віднімаючи змінні x_6 та x_7:

$$3x_1 - x_2 - x_6 = 4, \tag{3.6}$$

$$x_1 + 4x_2 - x_7 = 6.$$

До цих обмежень додамо штрафи R_1 та R_2.

$$3x_1 - x_2 - x_6 + R_1 = 4, \tag{3.7}$$

$$x_1 + 4x_2 - x_7 + R_2 = 6.$$

Після врахування штрафів у цільовій функції, отримаємо таку оптимізаційну задачу:

$$\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4 - MR_1 - MR_2,$$

$$x_1 + 2x_2 + x_3 = 6, \tag{3.8}$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

і додаткові обмеження з введеними штрафами

$3x_1 - x_2 - x_6 + R_1 = 4,$

$x_1 + 4x_2 - x_7 + R_2 = 6,$

$x_1, x_2, x_3, x_4, x_5, x_6, x_7, R_1, R_2 \geq 0.$

Слід зауважити, що штрафи перемножені на велике число і введені у ЦФ зі знаком мінус, бо маємо задачу максимізації.

Надалі модифікуємо ЦФ з (3.8) шляхом введення замість штрафів R_1 та R_2, вирази, які визначимо з виразів (3.7). Тоді отримаємо:

$$R_1 = 4 - 3x_1 + x_2 + x_6, \qquad (3.8)$$

$$R_2 = 6 - x_1 - 4x_2 + x_7.$$

Модифікуємо ЦФ з (2.5) і будемо мати такий вираз:

$$\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4 - M(4 - 3x_1 + x_2 + x_6) - M(6 - x_1 - 4x_2 + x_7) =$$

$$= 2x_1 + 3Mx_1 + Mx_1 + 3x_2 - Mx_2 + 4Mx_2 + x_3 + 2x_4 - Mx_6 - Mx_7 - 4M - 6M =$$

$$= (2 + 4M)x_1 + (3 + 3M)x_2 + x_3 + 2x_4 - Mx_6 - Mx_7 - 10M.$$

Приймемо, що $M = 1000$, тоді ЦФ прийме такий вид:

$$\max f_1(\bar{x}) = (2 + 4M)x_1 + (3 + 3M)x_2 + x_3 + 2x_4 - Mx_6 - Mx_7 - 10M =$$

$$= 4002x_1 + 3003x_2 + x_3 + 2x_4 - 1000x_6 - 1000x_7 - 10000.$$

Надалі необхідно розв’язати симплекс-методом задачу ЛП, де необхідно замінити ЦФ на модифіковану.

В кінцевому випадку отримуємо такі значення змінних та ЦФ:

$x_1 = 1{,}69, \quad x_2 = 1{,}08, \quad x_3 = 2{,}15, \quad x_4 = 3{,}54, \quad x_5 = 4{,}6,$

$f_1(\bar{x}) = -6{,}15$

$$f_1(\bar{x}) = 1{,}69 * 2 + 1{,}08 * 3 + 2{,}15 + 2 * 3{,}54 = 3{,}38 + 3{,}24 + 2{,}15 + 7{,}08 = 15{,}85$$

Отже, метод головного критерію застосовується в таких задачах, як мінімізація затрат при умові виконання плану по виробництву різних видів продукції, максимізація випуску комплектних наборів при обмеженні на використовувані ресурси та ряд інших.

До переваг МГК можна віднести такі характеристики як простоту та наглядність, а його недоліком є певна присутність суб'єктивності в процесі вибору головного критерія (свавілля).

3.2. Лексикографічний метод

Ідея методу ґрунтується на тому, що спершу часткові критерії оптимізації ранжують по їх відносній важливості і поступово розв'язують задачі однокритеріальної оптимізації, починаючи з найважливішого критерія оптимізації.

В загальному випадку алгоритм (рисунок 3.2) розв'язання ЗБО з використанням лексикографічного методу включає такі кроки:

Крок 0. На попередньому кроці розв'язання ЗБО ранжуємо часткові критерії в порядку спадання їхньої важливості. Пронумерувавши після цього критерії, можна вважати, що перший критерій – найважливіший.

Крок 1. Розв'язати задачу багатокритеріальної оптимізації (знайти такі значення проектних параметрів $\bar{x}$, для яких):

$$\max f_1(\bar{x}),$$

$$g_i(\bar{x}) \geq B_i,$$
$$h_j(\bar{x}) = A_j.$$

і знаходиться максимальне значення критерію $f_1^*(\bar{x}_1^*)$.

Крок 2. Розв’язати задачу багатокритеріальної оптимізації

$$\max f_2(\bar{x})$$

$$g_i(\bar{x}) \geq B_i,$$
$$h_j(\bar{x}) = A_j.$$

$$f_1(\bar{x}) = f_1^*(\bar{x}_1^*)$$

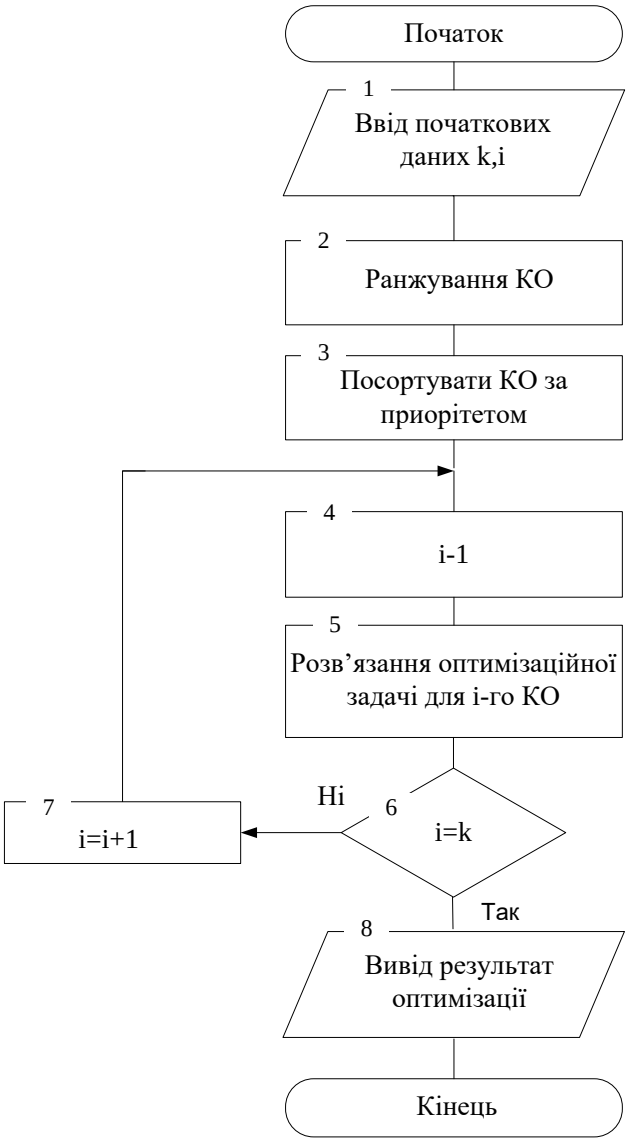


Рисунок 3.2 – Блок-схема алгоритму розв’язання ЗБО з використанням лексикографічного методу

Знайдене розв'язання $\bar{x}_2^*$ максимізує другий критерій, який задовольняє при цьому додатковому обмеженню, при виконанні якого досягається максимум по першому критерію.

Крок *k*. Розв'язати задачу багатокритеріальної умовної оптимізації

$$\max f_k(\bar{x})$$

$$g_i(\bar{x}) \geq B_i,$$
$$h_j(\bar{x}) = A_j,$$

$$f_1(\bar{x}) = f_1^*(\bar{x}_1^*),\ f_2(\bar{x}) = f_2^*(\bar{x}_2^*),\ \ldots,\ f_{k-1}(\bar{x}) = f_{k-1}^*(\bar{x}_{k-1}^*).$$

Розв'язання $\bar{x}_k^*$ максимізує k - й критерій оптимальності, одночасно задовольняючи обмеження на значеннях попередніх критеріїв оптимальності.

Ця процедура продовжується до тих пір, поки не буде максимізоване значення останнього із часткових критеріїв, після чого процедура завершується.

В процесі застосування вищенаведеного алгоритму після 1-го, 2-го, …, $k-1$-го кроків необхідно цільову функцію перевести в список обмежень, що наведено нижче:

$$f_1(\bar{x}) \geq f_1^* \text{ – після 1-го кроку,} \qquad (3.9)$$

$$f_2(\bar{x}) \geq f_2^* \text{ – після 2-го кроку,}$$

$$\ldots$$

$$f_{k-1}(\bar{x}) \geq f_{k-1}^* \text{ – після } k-1\text{-го кроку,}$$

де $f_1^*,\ f_2^*,\ \ldots,\ f_{k-1}^*$ - отримане значення цільової функції після розв'язання задачі на першому, другому, …, $k-1$-му кроці.

Звернемо увагу на те, що лексикографічний метод не призводить до нескінченної ітераційної процедури, яка зупиняється, коли одержаний результат задовольняє розробника. Навпаки, описана процедура має наперед відоме обмежене число кроків, яке не більше числа кількості часткових критеріїв.

Слід зауважити, що в лексикографічному методі часто можливості вибору не залишається вже після оптимізації по першому критерію, так що процес зразу ж зупиняється. В цьому випадку, задача багатокритеріальної оптимізації виявляється зведеною до однокритеріальної задачі з найбільш важливим критерієм, причому, значеннями інших критеріїв практично нехтують. Якщо ж після оптимізації першого критерію і залишається якась свобода дій, то її може виявитися недостатньо для одержання задовільних значень інших критеріїв оптимальності.

Необхідно додати, що задача ранжування критеріїв оптимальності, за важливістю зовсім не проста. Окрім того, лексикографічна процедура, загалом, некоректна з обчислювальної точки зору, так як малі збурення параметрів початкової задачі можуть призвести до серйозної помилки в результатах. Хоча і були розроблені стійкі обчислювальні методи розв'язку цієї задачі, використання лексографічної процедури у випадку нестійкості до початкових даних вимагає особливої уваги інженера чи розробника. Теоретичною основою лексографічного методу є лексографічний порядок, який задається на множині векторів. Будучи повним, він тим не менше, не володіє властивістю неперервності, що може привести до несподіваних наслідків.

Для кращого розуміння особливостей та ідеї методу наведемо приклад, а саме: необхідно розв'язати таку задачу багатокритеріальної оптимізації (задача лінійного програмування) з трьома цільовими функціями, трьома обмеженнями у вигляді рівностей і п'ятьма змінними (Див.формули підрозділу 3.1):

Після проведення операції ранжування, для прикладу, маємо таку ситуацію:

1. $\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4$ – найважливіший КО;
2. $\max f_2(\bar{x}) = 3x_1 - x_2$ – менш важливий КО;
3. $\max f_3(\bar{x}) = x_1 - 4x_2$ – найменш важливий КО.

Згідно з вищенаведеним алгоритмом на першому кроці необхідно розв'язати таку задачу однокритеріальної оптимізації:

$$\max f_1(\bar{x}) = 2x_1 + 3x_2 + x_3 + 2x_4, \tag{3.10}$$

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Застосувавши симплекс – метод до розв'язання задачі (3.10) отримаємо такі результати:

$$f_1^{\max} = f_1^* = 22, \tag{3.11}$$

$$x_1 = 0,\ x_2 = 0,\ x_3 = 6,\ x_4 = 8,\ x_5 = 4.$$

Наступний крок алгоритму передбачає введення першого критерія оптимізації в список обмежень (справу ми маємо з цільовою функцією, яка описує цей критерій і дає змогу отримати числове значення оцінки цього критерію) і розв'язання однокритеріальної оптимізаційної задачі з другим критерієм. Тобто, оптимізаційну задачу наступного виду:

$$\max f_2(\bar{x}) = 3x_1 - x_2, \tag{3.12}$$

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$2x_1 + 3x_2 + x_3 + 2x_4 \geq 22$ – додаткове обмеження на основі $f_1(\bar{x})$,

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

В результаті розв'язання задачі (3.12) отримаємо такі значення:

$f_2^* = 0$, $f_1^* = 22$,

$x_1 = 0,\ x_2 = 0,\ x_3 = 6,\ x_4 = 8,\ x_5 = 4.$

Останній етап розв'язання задачі (3.1) з використанням лексикографічного методу передбачає введення другої цільової функції в

список обмежень і розв'язання нижченаведеної оптимізаційної однокритеріальної задачі лінійного програмування:

$$\max f_3(\bar{x}) = x_1 - 4x_2, \tag{3.13}$$

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$2x_1 + 3x_2 + x_3 + 2x_4 \geq 22$ – 1-ше додаткове обмеження на основі $f_1(\bar{x})$,

$3x_1 - x_2 \geq 0$ – 2-ге додаткове обмеження на основі $f_2(\bar{x})$,

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Отримані кінцеві результати розв'язання задачі (3.1) наступні:

$$f_3^* = 0,\ f_2^* = 0,\ f_1^* = 22, \tag{3.14}$$

$$x_1 = 0,\ x_2 = 0,\ x_3 = 6,\ x_4 = 8,\ x_5 = 4.$$

Необхідно зауважити, що (підтверджено наведеним прикладом) в лексикографічному методі часто можливості вибору не залишається вже після оптимізації по першому критерію, так що процес зразу же зупиняється, як наслідок використовується цей метод, на практиці, дуже рідко.

Розвитком лексографічного методу є метод поступок.

3.3. Метод поступок

Для задач, у яких критерії не рівнозначні, застосовують інший метод розв'язання, а саме: **метод поступок**. Перш ніж розв'язувати поставлену задачу за методом поступок, необхідно:

Крок 1. Розмістити критерії оптимальності за їхньою значимістю (найважливіший необхідно розмістити першим, потім менш важливий і т.д.).

Крок 2. Відшукати оптимальне значення f_1^* для цільової функції f_1.

Крок 3. Зробити поступку по першому показнику ефективності, тобто погіршити величину f_1^* до значення $f_1^{**} = k_1 f_1^*$.

Крок 4. Ввести в задачу додаткове обмеження $f_{1i}\, f_1^{**}$.

Крок 5. Відшукати оптимальне значення f_2^* цільової функції f_2.

Крок 6. Зробити поступку по другому показнику ефективності, тобто погіршити величину f_2^* до значення $f_2^{**} = k_2 f_2^*$.

Крок 7. Ввести в оптимізаційну задачу додаткове обмеження $f_{2i} = f_2^{**}$.

Крок 8. Нову задачу з двома додатковими обмеженнями розв'язати по третьому показнику ефективності і т.д.

Крок 9. Процес розв'язання задачі закінчується, коли рішення буде отримано по всім показникам. Кінцевий результат і буде найбільш раціональним - отримано оптимальне значення найменш важливого критерію при умові гарантованих значень попередніх показників ефективності.

Приклад блок-схеми алгоритму застосування методу поступок до розв'язання ЗБО наведено на рисунок 3.3. Особливістю якого є те, що важливість критеріїв оптимальності також визначається на основі знань та досвіду інженера, що також є суб'єктивним фактором.

Необхідно додати основні математичні співвідношення для процесу введення додаткових обмежень після виконання поступки.

$f_1(\bar{x}) \geq f_1^* - \Delta_1 = f_1^{**}$ – додаткове обмеження після 1-ї поступки;

$f_2(\bar{x}) \geq f_2^* - \Delta_2 = f_2^{**}$ – додаткове обмеження після 2-ї поступки;

. . .

$f_{n-1}(\bar{x}) \geq f_{n-1}^* - \Delta_{n-1} = f_{n-1}^{**}$ – додаткове обмеження після $n-1$-ї поступки.

Застосуємо метод поступок до розв’язання задачі багатокритеріальної оптимізації (3.1). Будемо вважати, що критерії оптимізації уже розміщені по їх важливості. Найбільш важливий розміщений першим, тобто перший крок алгоритму уже виконали. Розв’язання однокритеріальної оптимізаційної задачі ми також вже виконали з першим КО (Крок 2 алгоритму) і розв’язання наведено вище (Див. вирази 3.11). Ці два кроки ми вже виконали в процесі застосування лексикографічного методу до розв’язання ЗБО. Третій крок передбачає операцію виконання поступки. Для розглядуваної задачі зробимо поступку в розмірі 33%, а саме ($k_1 = 0{,}33$):

$$f_1^{**} = f_1^* - \Delta_1 = f_1^* - 0{,}33 * f_1^* = 22 - 0{,}33 * 22 = 22 - 7{,}27 = 14{,}7 . \quad (2.15)$$

Рисунок 3.3 – Блок-схема алгоритму застосування методу поступок

Введемо в задачу відповідно наступне додаткове обмеження (Крок 4), а саме будемо мати таку форму:

$2x_1 + 3x_2 + x_3 + 2x_4 \geq 14{,}7$ – 1-ше додаткове обмеження на основі $f_1(\bar{x})$.

В даному випадку ми погіршили значення першого критерію на 33%, зменшили значення цільової функції тобто з 22 до 14,7.

П’ятий крок алгоритму передбачає розв’язання однокритеріальної оптимізаційної задачі з другим КО та модифікованою системою обмежень, що наведено нижче:

$$\max f_2(\bar{x}) = 3x_1 - x_2, \qquad (3.15)$$

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$2x_1 + 3x_2 + x_3 + 2x_4 \geq 14{,}7$ – додаткове обмеження на основі $f_1(\bar{x})$,

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Слід зауважити, що задача (3.15) аналогічна до (3.12) лишень з відмінністю в додатковому обмеженні, де змінено максимальне значення цільової функції 22 на 14,7.

Результати розв’язання задачі (3.15) наведено нижче:

$$f_3^* = 2{,}42,\ f_2^* = 7{,}26,\ f_1^* = 14{,}7, \qquad (3.16)$$

$$x_1 = 2{,}42,\ x_2 = 0,\ x_3 = 3{,}58,\ x_4 = 3{,}16,\ x_5 = 6{,}42.$$

Надалі, знову необхідно зробити поступку, але вже для другого критерію оптимальності. Для прикладу, виберемо величину поступки для другого критерію оптимальності в 40 %:

$$f_2^{**} = f_2^* - \Delta_2 = f_2^* - 0{,}4 * f_2^* = 7{,}26 - 0{,}4 * 7{,}26 = 7{,}26 - 2{,}9 = 4{,}36. \qquad (3.17)$$

Тоді додаткове обмеження набуде наступного вигляду:

$3x_1 - x_2 \geq 4{,}36$ – 2-ше додаткове обмеження на основі $f_2(\bar{x})$.

Наступний крок алгоритму передбачає розв’язання вже такої задачі (з третім КО і двома додатковими обмеженнями):

$$\max f_3(\bar{x}) = x_1 - 4x_2, \qquad (3.18)$$

$$x_1 + 2x_2 + x_3 = 6,$$

$$2x_1 + x_2 + x_4 = 8,$$

$$-x_1 + x_2 + x_5 = 4,$$

$2x_1 + 3x_2 + x_3 + 2x_4 \geq 14{,}7$ – 1-ше додаткове обмеження на основі $f_1(\bar{x})$,

$3x_1 - x_2 \geq 4{,}36$ – 2-ге додаткове обмеження на основі $f_2(\bar{x})$,

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Отримані кінцеві результати розв’язання задачі (3.18) подані нижче :

$$f_1^* = 14{,}7,\ f_2^* = 4{,}36,\ f_3^* = 7{,}74, \qquad (3.19)$$

$$x_1 = 1{,}94,\ x_2 = 1{,}45,\ x_3 = 1{,}16,\ x_4 = 2{,}68,\ x_5 = 4{,}48.$$

Отже, метод послідовних поступок доцільно застосовувати для розв’язання тих багатокритеріальних задач, в яких усі часткові критерії природнім чином впорядковані за ступенем важливості, причому кожний критерій настільки істотно більш важливий, ніж наступний, що можна обмежитися врахуванням тільки попарного зв'язку критеріїв і вибирати допустиме зниження чергового критерію з врахуванням поведінки тільки одного наступного критерію.

Особливо зручним є випадок, коли вже в результаті попереднього аналізу багатокритеріальної задачі виявляється, що можна допустити поступки тільки в межах “інженерної” точності (5-10% від найбільшої величини критерію).

3.4. Методи цільового програмування

Основна ідея методів цієї групи полягає в припущенні, що існує певна точка, де значення усіх критеріїв $f_i(\bar{x})$, $i=1,n$ досягають найоптимальнішого (цільового, найбільшого $f_i^{\max}(\bar{x})$) значення (тобто досягається певна ціль). Відповідно, в самому загальному випадку задача цільового програмування (ЦП) може бути сформульована як мінімум сум відхилень цільових функцій (критеріїв) від цільових значень (оптимальних) з нормованими коефіцієнтами ($\alpha_1, \alpha_2, ..., \alpha_n$):

$$\Delta\left(F(\bar{x}), F^{\max}\right) = \left[\sum_{i=1}^{n} \alpha_i \left| f_i(\bar{x}) - f_i^{\max} \right|^p \right]^{\frac{1}{p}} \to \min, \qquad (3.20)$$

де $F^{\max} = \left(f_1^{\max}, f_2^{\max}, ..., f_n^{\max}\right)$ - вектор цільових значень КО; $\Delta\left(F(\bar{x}), F^{\max}\right)$ - відхилення між $f(\bar{x})$ та $f^{\max}$, $1 \le p < \infty$.

Досить часто в літературі точку $F^{\max}$ називають ідеальною чи утопічною.

Наведемо приклад розв'язання ЗБО з використанням виразу (3.20).

$$f_1 = (x_1 + 2x_2)\exp(-x_2) \to \max, \qquad (3.21)$$

$$f_2 = (3x_1 + 2x_2)\exp(-(3x_1 + x_2)) \to \max,$$

$$f_3 = x_1 + x_2,$$

$$2x_1 + x_2 \le 2,$$

$$x_2 - x_1 \le 3,$$

$$x_1, x_2 \ge 0.$$

Приймемо, що $\alpha_1 = 0{,}5$, $\alpha_2 = 0{,}3$ та $\alpha_3 = 0{,}2$.

Визначити

$$F(\bar{x}) = \{f_1, f_2, f_3\} \to \max.$$

В процесі розв'язання задачі отримали наступні значення ЦФ

$f_1^{\max} = 1{,}0748$,

$f_2^{\max} = 0{,}7357$,

$f_3^{\max} = 2$.

При визначенні $\Delta\left(F(\bar{x}), F^{\max}\right)$ отримали наступне значення:

$$\Delta\left(F(\bar{x}), F^{\max}\right) = 0{,}32534.$$

Значення компонент вектора, відповідно:

$f_1^{\max}$	$f_1(\bar{x})$	$f_2^{\max}$	$f_2(\bar{x})$	$f_3^{\max}$	$f_3(\bar{x})$
1,0748	0,7815	0,7358	0,3609	2	1,16784

Метод умовного центра мас

Метод умовного центра мас знайшов досить широке розповсюдження в процесі автоматизованого проектування. На основі цього методу розв'язані задачі забезпечення ефективності лісогосподарських машин для рубки, догляду, лісомеліоративних агрегатів, гідроманіпуляторів трельовочних машин.

Нехай послідовно знайдені значення екстремумів для кожної цільової функції $f_j(\bar{x})$, що відповідає точкам у просторі проектних параметрів з координатами $\left\{x_1^{i*}, x_2^{i*}, ..., x_n^{i*}\right\}$.

Введемо поняття "умовної маси" точки [54]:

$$m_i = \frac{\sum_i f_i(x_1^{i*}, x_2^{i*}, ..., x_n^{i*})}{f_i(x_1^{i*}, x_2^{i*}, ..., x_n^{i*})},$$

де $f_i(x_1^{i*}, x_2^{i*}, ..., x_n^{i*})$ - значення i-ї цільової функції при сукупності керованих параметрів, що забезпечують екстремальне його значення. Будемо

вважати, що компромісному розв'язанню буде задовольняти набір параметрів, які відповідають точці з координатами "умовного центра мас":

$$x_j^{**} = \frac{\sum m_i x_j^{i*}}{\sum m_i}.$$

Знайдені з використанням цього методу середньозважені значення параметрів x_j^{**} враховують не тільки інтереси всіх показників якості, але і чутливість кожного по відношенню до даного проектного параметра.

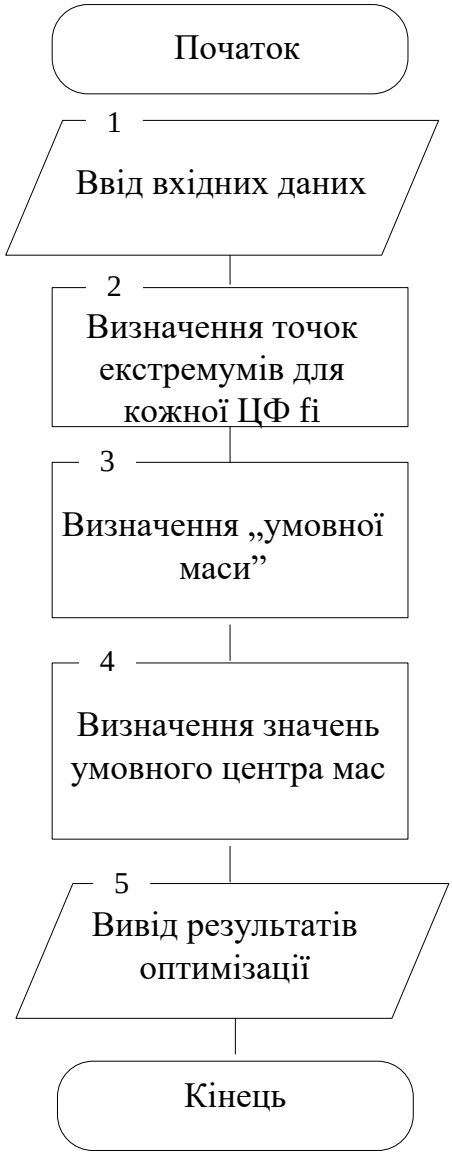


Рисунок 3.4 – Блок-схема алгоритму застосування методу умовного центра мас

Наведемо приклад розв'язання ЗБО з використанням методу умовного центра мас. Отже, в якості прикладу, візьмемо задачу, де є три цільові функції, три обмеження та п'ять змінних і якщо формулюється таким чином, що необхідно знайти такі значення проектних параметрів, щоб забезпечити максимум ЦФ- й.

$$\max f_1(\bar{x}) = 2x_1 + x_2 + 3x_3 + x_4 + 4x_5, \quad (3.22а)$$

$$\max f_2(\bar{x}) = x_1 + 3x_2, \quad (3.22б)$$

$$\max f_3(\bar{x}) = 2x_1 + 2x_2, \quad (3.22в)$$

і виконувалися такі обмеження:

$$2x_1 + x_2 + x_3 = 8, \quad (3.23)$$

$$x_1 + 3x_2 + x_4 = 3,$$

$$x_1 + x_2 + x_5 = 5,$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Розв'яжемо послідовно три оптимізаційні задачі з одним критерієм оптимальності. Перша задача включає ЦФ (3.22а) та обмеження (3.23), дуга – (3.22б) і (3.23), а третя (3.22в) і (3.23).

В результаті застосування симплекс-методу отримали наступні розв'язання задач багатокритеріальної оптимізації:

для першої:

$$f_1^* = 47, \quad (3.24)$$

$$x_1 = 0,\ x_2 = 0,\ x_3 = 8,\ x_4 = 3,\ x_5 = 5.$$

для другої:

$$f_2^* = 3, \quad (3.25)$$

$$x_1 = 0,\ x_2 = 1,\ x_3 = 7,\ x_4 = 0,\ x_5 = 3.$$

для третьої:

$$f_3^* = 6, \quad (3.26)$$

$$x_1 = 3,\ x_2 = 0,\ x_3 = 2,\ x_4 = 0,\ x_5 = 2.$$

Визначимо параметри “умовного центра мас”, зокрема для нашого випадку:

$$m_i = \frac{\sum_i f_i\left(x_1^{i*}, x_2^{i*}, x_3^{i*}, x_4^{i*}, x_5^{i*}\right)}{f_i\left(x_1^{i*}, x_2^{i*}, x_3^{i*}, x_4^{i*}, x_5^{i*}\right)},$$

$$m_1 = \frac{56}{47} \approx 1.19,\ m_2 = \frac{56}{3} \approx 18.67,\ m_3 = \frac{56}{6} \approx 9.33. \qquad (3.27)$$

Значення проектних параметрів “умовного центра мас”:

$$x_1^{**} = \frac{0*1{,}19 + 0*18{,}67 + 9{,}33*3}{1{,}19 + 18{,}67 + 9{,}33} = \frac{27{,}99}{29{,}19} \approx 0{,}96, \qquad (3.28)$$

$$x_2^{**} = \frac{0*1{,}19 + 1*18{,}67 + 0*9{,}33}{1{,}19 + 18{,}67 + 9{,}33} = \frac{18{,}67}{29{,}19} \approx 0{,}64,$$

$$x_3^{**} = \frac{8*1{,}19 + 7*18{,}67 + 2*9{,}33}{1{,}19 + 18{,}67 + 9{,}33} = \frac{158{,}87}{29{,}19} \approx 5{,}44,$$

$$x_4^{**} = \frac{3*1{,}19 + 0*18{,}67 + 0*9{,}33}{1{,}19 + 18{,}67 + 9{,}33} = \frac{3{,}57}{29{,}19} \approx 0{,}12,$$

$$x_5^{**} = \frac{5*1{,}19 + 4*18{,}67 + 2*9{,}33}{1{,}19 + 18{,}67 + 9{,}33} = \frac{99{,}29}{29{,}19} \approx 3{,}4.$$

Відповідно значення цільових функцій:

$$f_1^* = 2*0{,}96 + 1*0{,}64 + 3*5{,}44 + 1*0{,}12 + 4*3{,}4 = \qquad (3.29)$$

$$= 1{,}92 + 1{,}28 + 16{,}32 + 0{,}12 + 13{,}6 = 33{,}24,$$

$$f_2^* = 1*0{,}96 + 3*0{,}64 = 0{,}96 + 1{,}92 = 2{,}88,$$

$$f_3^* = 2*0{,}96 + 2*0{,}64 = 1{,}92 + 1{,}28 = 3{,}2.$$

Метод умовного центра мас широко використовується в процесі розв’язання задач багатокритеріальної оптимізації, в процесі автоматизованого проектування складних об’єктів і систем.

Метод ідеальної точки

Ідея методу “ідеальної точки” ґрунтується на тому, що постулюється існування “ідеальної точки” для розв’язку задачі, у якій досягається екстремум всіх критеріїв (принцип Джофріона). Оскільки ідеальна точка, в абсолютній більшості випадків не знаходиться серед припустимих, виникає проблема знаходження точки, що „найближча” до ідеальної, і належить до множини припустимих рішень. Все було б добре, якщо б існувало єдине об’єктивне поняття “віддалі”, однак це не так – якщо на площині ми можемо з тим чи іншим обґрунтуванням застосовувати Евклідову метрику, то, наприклад, на поверхні кулі найкоротшою віддаллю буде дуга, а не пряма.

Таким чином, для розв’язання задачі багатокритеріальної оптимізації за допомогою методу “ідеальної точки” необхідно насамперед визначити її координати, і надалі визначити метрику, за допомогою якої можна було б виміряти віддаль до оптимальної точки. Для визначення координат “ідеальної точки” розв’язуємо n однокритеріальних задач за кожним з критеріїв оптимізації $\max f_1(a,x)$ $x \in X$, $i = \overline{1,n}$. Сукупність оптимальних значень критеріїв кожної з однокритерійних задач $f_i = \max f_i(a,x)$, $x \in X$ і визначаємо координати ідеальної точки $f = (f_1,...,f_n)$ в просторі критеріїв. Якщо “ідеальна точка” належить до множини припустимих (що зустрічається вкрай рідко), то розв’язок отриманий.

В іншому випадку визначаємо “віддаль” до ідеальної точки, вводячи метрику, і розв’язуємо однокритерійну задачу знаходження точки з числа припустимих, яка найменш віддалена від ідеальної. Таким чином задача матиме вигляд $f(x) = \rho(f(a,x) - f^*) \Rightarrow \min$, $x \in X$. Якщо обрана Евклідова метрика, то критерій буде мати вигляд $f(x) = \sqrt{\sum_{i=1}^{n} \left(f_i(x) - f_i^*\right)^2} \Rightarrow \min$.

Застосуємо метод ідеальної точки до розв’язання ЗБО. Результати розв’язання однокритеріальних задач наведено на рисунку 3.5 та на рисунку 3.6.

В завершення можна додати ти, що метод ідеальної точки теж широко використовується в процесі розв’язання ЗБО.

Початок роботи програми

f_1=47	f_2=3	f_3=6
x_1=0	x_1=0	x_1=3
x_2=0	x_2=1	x_2=0
x_3=8	x_3=7	x_3=2
x_4=3	x_4=0	x_4=0
x_5=5	x_5=4	x_5=2

Рисунок 3.5 – Результати розв’язання однокритеріальних задач

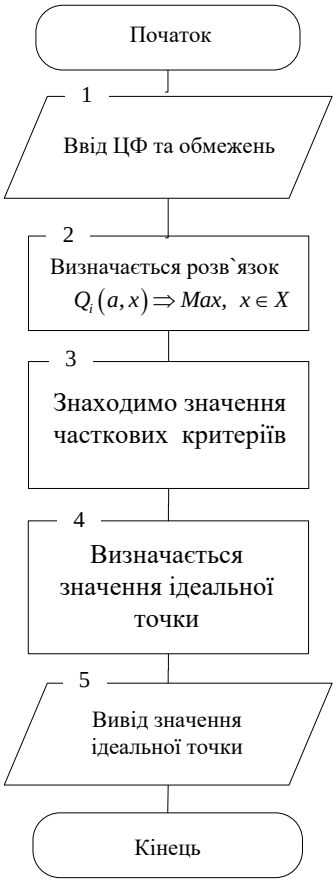


Рисунок 3.6 – Алгоритм розв’язку задачі багатокритеріальної оптимізації за допомогою методу ідеальної точки

Розрахунок проведено

x_1=1

x_2=0,333333333333333

x_3=5,66666666666667

x_4=1

x_5=3,66666666666667

f_1=35

f_2=2

$f3$=2,66666666666667

Рисунок 3.7 – Результати розв’язання ЗБО з використанням методу ідеальної точки

3.5 Метод комплексного критерію

Метод комплексного критерію, на відміну від методу головної компоненти, застосовується досить рідко. Він полягає в переході від векторного критерію до скалярного шляхом утворення сумарного монопоказника. При цьому основна ідея методу полягає в складанні однієї функції, аргументами якої служать компоненти вектора корисного ефекту. Особливо частим випадком є представлення такої функції у вигляді дробу, де в чисельнику стоять всі величини, збільшення яких бажано (наприклад, ефект), а в знаменнику ті, які хотілося б зменшити.

Контрольні запитання до розділу 3

1. Яка ідея методу головної компоненти?
2. Яка ідея лексикографічного методу?

3. Яка ідея методу поступок?

4. Яка ідея методу ідеальної точки?

5. Яка ідея методу умовного центра мас?

6. Які переваги та недоліки МГК?

7. Які переваги та недоліки лексикографічного методу?

8. Які переваги та недоліки методу поступок?

9. Які переваги та недоліки методу ідеальної точки?

10. Які переваги та недоліки методу умовного центра мас?

11. Які основні кроки включає алгоритм застосування МГК при розв’язанні ЗБО?

12. Які основні кроки включає алгоритм застосування лексикографічного методу при розв’язанні ЗБО?

13. Які основні кроки включає алгоритм застосування методу поступок при розв’язанні ЗБО?

14. Які основні кроки включає алгоритм застосування методу поступок при розв’язанні ЗБО?

15. Які основні кроки включає алгоритм застосування методу умовного центра мас при розв’язанні ЗБО?

16. Яка ідея методу комплексного критерію?

17. Які переваги та недоліки методу комплексного критерію?

РОЗДІЛ 4. МЕТОДИ РОЗВ'ЯЗАННЯ ЗАДАЧ БАГАТОКРИТЕРІАЛЬНОЇ ОПТИМІЗАЦІЇ З ВИКОРИСТАННЯМ УЗАГАЛЬНЕНОГО (ІНТЕГРАЛЬНОГО) КРИТЕРІЮ ОПТИМАЛЬНОСТІ

Суть цього методу полягає в тому, що власні критерії $f_i(\vec{x}), i = \overline{1,n}$ будь-яким чином поєднуються в один інтегральний критерій $F(x) = \Phi(f_1(x), f_2(x), ..., f_n(x))$, а потім знаходиться максимум чи мінімум цієї цільової функції.

Якщо об'єднання власних критеріїв будується, виходячи з об'єктного взаємозв'язку власних критеріїв і критерію узагальненого, то тоді оптимальне рішення оптимізаційної задачі буде коректне. Але таке об'єднання здійснити вкрай складно чи неможливо, тому, як правило, узагальнений критерій є результатом чисто формального об'єднання власних (часткових) критеріїв оптимальності.

В залежності від того, яким чином власні критерії поєднуються в узагальнений критерій оптимальності розрізняють наступні види узагальнених критеріїв:

1. Адитивний критерій;
2. Мультиплікативний критерій;
3. Максимінний (мінімаксний) критерій.

Маючи декілька рішень, кожне з яких отримано на основі врахування багатьох критеріїв оптимізації, необхідно їх порівняти між собою, або надати перевагу одному з них. Для розв'язання цієї задачі будується функція корисності, яка дає змогу визначити показник ефективності рішення і процес надання переваги зводиться до порівняння чисел-значень.

При цьому особа, що приймає рішення (ОПР), враховує, що один набір значень локальних критеріїв володіє перевагою над іншими, якщо йому відповідає більше значення функції переваги.

4.1 Нормалізація узагальненого критерію

В процесі порівняння значень критеріїв оптимальності їх необхідно звести до однієї розмірності, або до безрозмірного виду.

В якості стандарту вибрано перетворення в шкалу зі значеннями в діапазоні від 0 до 1.

1. Якщо відомо деяке еталонне значення ($f^{еталон}$), то

$$f_{H,i} = \frac{f_i^0}{f_i^{еталон}}, \qquad \text{де } i = 1,2,...,m. \tag{4.1}$$

2. Якщо відомо максимально можливе значення $f^{\max}$, то

$$f_{H,i} = \frac{f_i^0}{f_i^{\max}}, \qquad \text{де } i = 1,2,...,m. \tag{4.2}$$

3. Якщо відомо діапазон зміни від $f^{\min}$ до $f^{\max}$, то

$$f_{H,i} = \frac{f_i^0}{f_i^{\max} - f_i^{\min}}, \text{ де } i = 1,2,...,m, \tag{4.3}$$

$$\text{чи } f_{H,i} = \frac{f_i^0 - f_i^{\min}}{f_i^{\max} - f_i^{\min}} \qquad \text{де } i = 1,2,...,m. \tag{4.4}$$

Вирази для зведення безрозмірного виду наведені на (4.1 - 4.4) найчастіше зустрічаються в задачах проектування, а більшість еталонних значень, максимально можливих та діапазон змін беруться з технічного завдання на виріб.

4.2 Адитивний критерій

Цільову функцію будують шляхом додавання нормованих значень власних критеріїв. В загальному випадку узагальнена цільова функція має

наступний вид:

$$F(\vec{x}) = \sum_{i=1}^{n} \alpha_i \frac{f_i(\bar{x})}{f^o i(\bar{x})} = \sum_{i=1}^{n} \alpha_i f_{H,i}(\bar{x}) \rightarrow \max(\min), \qquad (4.5)$$

де n – кількість обєднаних часткових критеріїв; C_i – ваговий коефіцієнт i – го часткового критерію; $f_i(\bar{x})$ – числове значення i – го часткового критерію; $f^o i(\bar{x})$ – i – й нормований дільник; $f_{H,i}(\bar{x})$ – нормоване значення i – го часткового критерію.

Власні критерії мають різну фізичну природу і тому різну розмірність. А значить просто підсумовувати їх некоректно. У зв'язку з цим у попередній формулі числові значення власних критеріїв поділяються на деякі дільники, що нормують і призначається в такий спосіб:

1. Як нормуючі дільники приймаються директивні значення чи параметри критеріїв, які задаються замовником (що можна отримати з технічного завдання). Вважається, що значення проектних параметрів, закладені в технічному завданні, є оптимальними чи найкращими.

2. Як нормуючі дільники приймаються максимальні (мінімальні) значення критеріїв, що досягаються в області припустимих рішень.

Розмірності власних критеріїв оптимальності та відповідних дільників, що нормують, однакові, тому в підсумку узагальнений адитивний критерій виходить безрозмірною величиною. Алгоритм розв’язання задачі багатокритеріальної оптимізації зображено на рисунку 4.1.

Наведемо приклад, який призначений для визначення оптимального варіанта машини з використанням узагальненого (інтегрального) адитивного критерію. Частковими критеріями, з допомогою яких оцінені варіанти машини, являються її продуктивність та надійність. Два критерії “працюють” на максимум, тобто найкращими варіантами машини являються ті з них, які забезпечують найбільшу її продуктивність та надійність. Початкові дані для розв’язання задачі наведені в таблиці 4.1.

Цільова функція на основі узагальненого адитивного критерію запишеться в такий спосіб:

$$F(x) = \alpha_1 \frac{f_1(\bar{x})}{f_1^{(0)}(\bar{x})} + \alpha_2 \frac{f_2(\bar{x})}{f_2^{(0)}(\bar{x})} \to \max. \qquad (4.6)$$

Як нормуючі дільники в цій задачі приймемо найкращі (максимальні) значення власних критеріїв: $f_1^{(0)}(\bar{x}) = 4000\,шт/год$, $f_2^{(0)}(\bar{x}) = 1500\,год$.

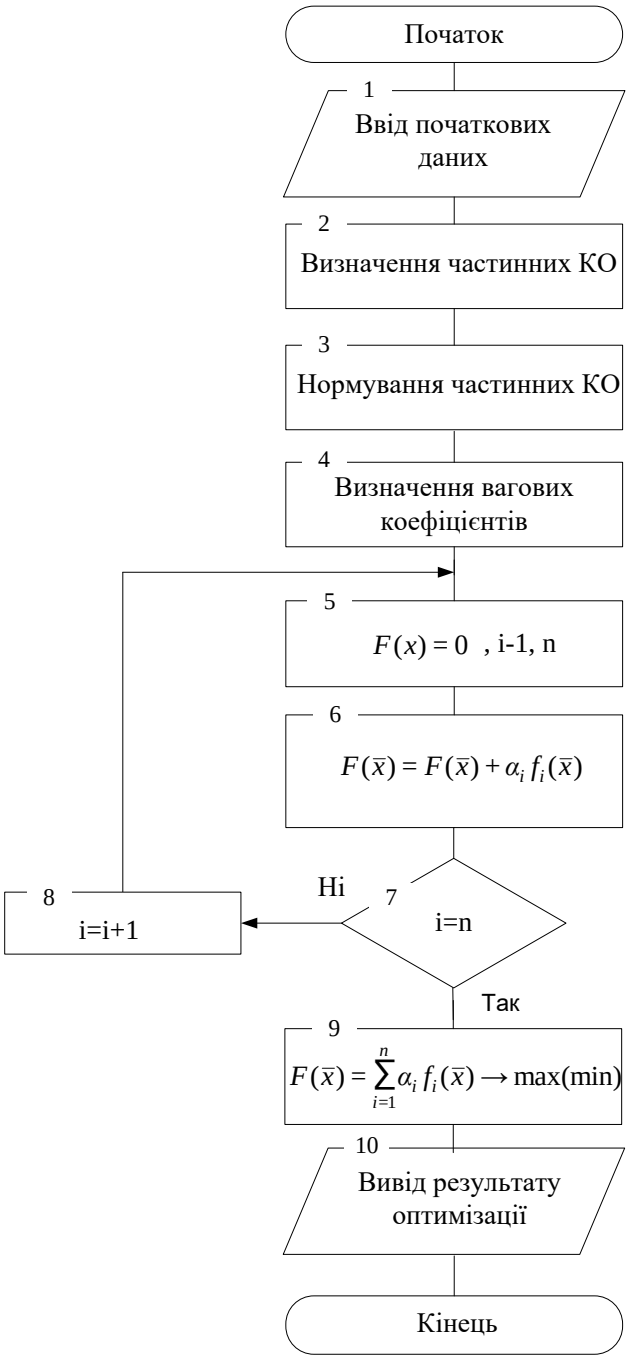


Рисунок 4.1 – Блок-схема алгоритму розв’язання ЗБО з використанням узагальненого адитивного критерію

Таблиця 4.1 – Початкові дані для визначення оптимального варіанту виконання машини

Критерій	Ваговий коефіцієнт	Значення критеріїв для варіантів виконання машини		
		Варіант 1	Варіант 2	Варіант 3
Продуктивність, шт.год	0,6	1000	2000	4000
Надійність, год	0,4	1500	1000	500

Значення узагальненого адитивного критерію розраховуються для кожного варіанта машини:

$$\text{Варіант 1. } F(\bar{x}) = 0{,}6(1000/4000) + 0{,}4(1500/1500) = 0{,}55, \quad (4.7)$$

$$\text{Варіант 2. } F(\bar{x}) = 0{,}6(2000/4000) + 0{,}4(1000/1500) = 0{,}558,$$

$$\text{Варіант 3. } F(\bar{x}) = 0{,}6(4000/4000) + 0{,}4(500/1500) = 0{,}732.$$

Оптимальним є 3-й варіант машини, тому що йому відповідає максимальне значення узагальненого адитивного критерію.

Один з недоліків цього методу полягає в тому, що вагові коефіцієнти призначає, в більшості випадків, сам проектувальник. Різні проектувальники можуть призначати різні вагові коефіцієнти. Розглянемо дещо іншу ситуацію, для прикладу, $\alpha_1 = 0{,}4$; $\alpha_2 = 0{,}6$. Визначимо тепер значення адитивних критеріїв для варіантів машини:

$$\text{Варіант 1. } F(\bar{x}) = 0{,}4 * 0{,}25 + 0{,}6 * 1 = 0{,}7, \quad (4.8)$$

$$\text{Варіант 2. } F(\bar{x}) = 0{,}4 * 0{,}5 + 0{,}6 * 0{,}67 = 0{,}602,$$

$$\text{Варіант 3. } F(\bar{x}) = 0{,}4 * 1 + 0{,}6 * 0{,}33 = 0{,}598.$$

Тобто, при такій зміні значень вагових коефіцієнтів оптимальним уже буде 1 варіант машини.

Перевага даного методу полягає в тому, що, як правило, завжди вдається визначити єдиний оптимальний варіант розв’язання оптимізаційної задачі.

До недоліків цього методу можна віднести таке:

1. Труднощі (суб'єктивізм) у визначенні вагових коефіцієнтів.

2. Адитивний критерій не випливає з об'єктної ролі власних критеріїв і тому виступає як формальний математичний прийом.

3. В адитивному критерії відбувається взаємна компенсація власних критеріїв, тобто зменшення одного з них може бути компенсовано збільшенням іншого критерію.

4.3 Мультиплікативний критерій

Цільова функція з використанням мультиплікативного критерію записується наступним чином:

$$F(\bar{x}) = \prod_{i=1}^{n} \alpha_i f_i(\bar{x}) \rightarrow \max(\min), \qquad (4.9)$$

де $\prod$ – знак добутку; α_i - ваговий коефіцієнт i -го часткового критерію; $f_i(\bar{x})$ - числове значення i -го часткового критерію.

На рисунку 4.2 зображено блок-схема алгоритму розв’язання ЗБО з використанням узагальненого мультиплікативного критерію.

Переваги мультиплікативного критерію є наступні:

1. Не потрібно нормування власних критеріїв.

2. Практично завжди визначається одне оптимальне рішення.

До недоліків можна віднести наступне:

1. Труднощі (суб'єктивізм) у визначенні вагових коефіцієнтів.

2. Перемножування різних розмірностей.

3. Взаємна компенсація значень власних критеріїв.

Застосовуємо цей метод до розв’язання задачі (Див. табл.4.1)

Цільова функція на основі мультиплікативного критерію запишеться в такий спосіб:

$$F(\bar{x}) = \alpha_1 \frac{f_1(\bar{x})}{f_1^{(0)}(\bar{x})} * \alpha_2 \frac{f_2(\bar{x})}{f_2^{(0)}(\bar{x})} \rightarrow \max. \quad (4.10)$$

Як нормуючі дільники в цій задачі приймемо найкращі (максимальні) значення власних критеріїв:

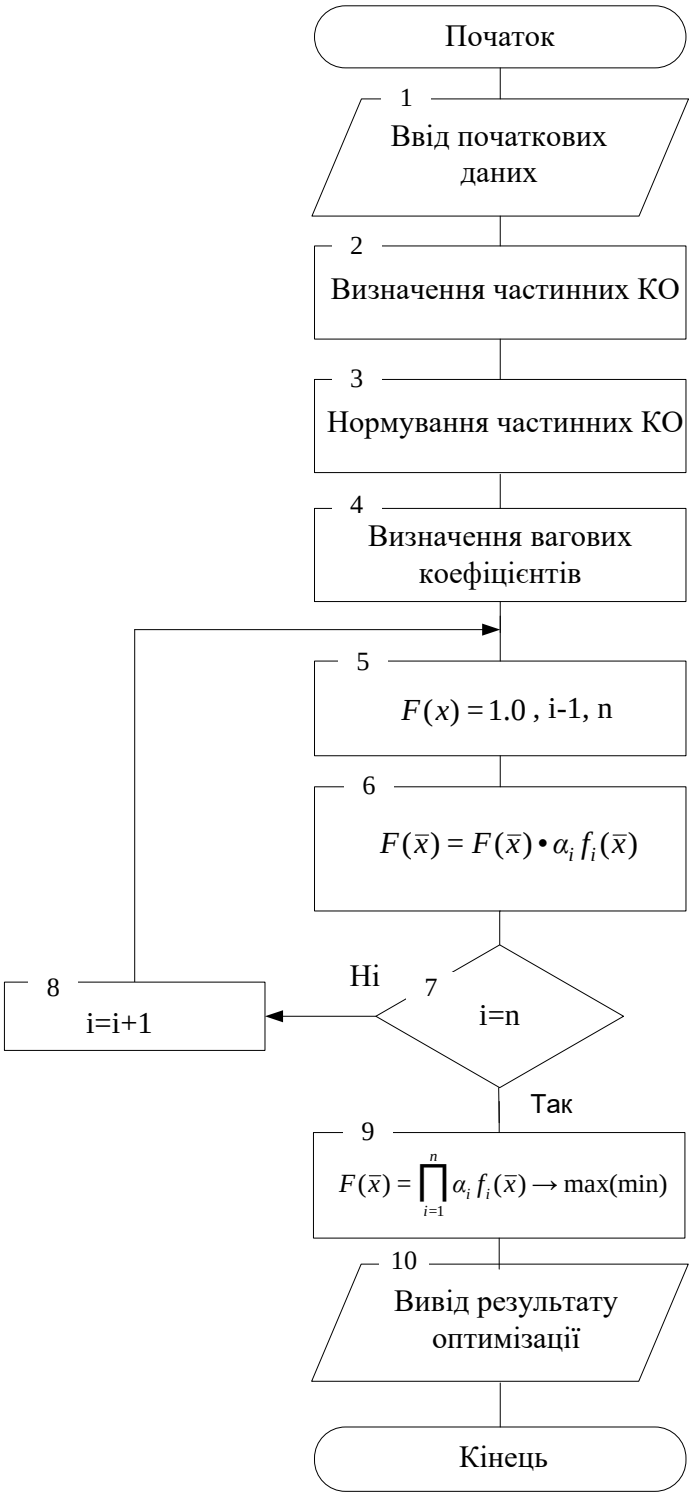


Рисунок 4.2 – Блок-схема алгоритму розв’язання ЗБО з використанням узагальненого мультиплікативного критерію

$f_1^{(0)}(\bar{x}) = 4000\,шт/год$, $f_2^{(0)}(\bar{x}) = 1500\,год$.

Значення узагальненого мультиплікативного критерію розраховуються для кожного варіанта машини і отримаємо:

Варіант 1.

$$F(\bar{x}) = \{0,6(1000/4000)\} * \{0,4(1500/1500)\} = 0,15 * 0,4 \approx 0,06.$$

Варіант 2.

$$F(\bar{x}) = \{0,6(2000/4000)\} * \{0,4(1000/1500)\} = 0,3 * 0,267 \approx 0,08.$$

Варіант 3.

$$F(\bar{x}) = \{0,6(4000/4000)\} * \{0,4(500/1500)\} = 0,6 * 0,13 \approx 0,08.$$

Оптимальними є 2-й варіант і 3-й варіанти машини.

Один з недоліків цього методу полягає також в тому, що вагові коефіцієнти призначає проектувальник. Різні проектувальники можуть призначати різні вагові коефіцієнти. Нехай, для прикладу, $\alpha_1 = 0,4$; $\alpha_2 = 0,6$. Визначимо тепер значення мультиплікативних критеріїв для варіантів машини:

Варіант 1. $F(\bar{x}) = (0,4 * 0,25) * (0,6 * 1) \approx 0,06$.

Варіант 2. $F(\bar{x}) = (0,4 * 0,5) * (0,6 * 0,67) \approx 0,08$.

Варіант 3. $F(\bar{x}) = (0,4 * 1) * (0,6 * 0,33) \approx 0,08$.

Тобто, при такій зміні значень вагових коефіцієнтів не змінилося.

Розглянемо іншу ситуацію, для прикладу $\alpha_1 = 0,1$; $\alpha_2 = 0,9$. Визначимо тепер значення адитивних та мультиплікативних критеріїв для варіантів машини:

Варіант 1. $F(\bar{x}) = 0,1 * 0,25 + 0,9 * 1 = 0,925$.

Варіант 2. $F(\bar{x}) = 0,1 * 0,5 + 0,9 * 0,67 = 0,653$.

Варіант 3. $F(\bar{x}) = 0,1 * 1 + 0,9 * 0,33 = 0,397$.

Тобто, при такій зміні значень вагових коефіцієнтів оптимальним буде 1 варіант машини.

Варіант 1. $F(\bar{x}) = (0,1 * 0,25) * (0,9 * 1) \approx 0,0225$.

Варіант 2. $F(\bar{x}) = (0{,}1 * 0{,}5) * (0{,}9 * 0{,}67) \approx 0{,}03.$

Варіант 3. $F(\bar{x}) = (0{,}1 * 1) * (0{,}9 * 0{,}33) \approx 0{,}03.$

Оптимальними є 2-й варіант і 3-й варіанти машин.

Отже, з даного прикладу можна зробити висновок, що мультиплікативний критерій менш чутливий до неточності визначення вагових коефіцієнтів.

Розглянемо іншу ситуацію, для прикладу $\alpha_1 = 0{,}0$; $\alpha_2 = 1{,}0$. Визначимо тепер значення адитивних та мультиплікативних критеріїв для варіантів машини (адитивний):

Варіант 1. $F(\bar{x}) = 0{,}0 * 0{,}25 + 1{,}0 * 1 = 1{,}0.$

Варіант 2. $F(\bar{x}) = 0{,}0 * 0{,}5 + 1{,}0 * 0{,}67 = 0{,}67.$

Варіант 3. $F(\bar{x}) = 0{,}0 * 1 + 1{,}0 * 0{,}33 = 0{,}33.$

Тобто, при такій зміні значень вагових коефіцієнтів оптимальним буде 1 варіант машини (мультиплікативний).

Варіант 1. $F(\bar{x}) = (0{,}0 * 0{,}25) * (1{,}0 * 1) = 0{,}0.$

Варіант 2. $F(\bar{x}) = (0{,}0 * 0{,}5) * (1{,}0 * 0{,}67) = 0{,}0.$

Варіант 3. $F(\bar{x}) = (0{,}0 * 1) * (1{,}0 * 0{,}33) = 0{,}0.$

Оптимальні альтернативи відсутні.

Отже, з даного прикладу можна зробити висновок, що якщо в мультиплікативному критерії рівний нулю (або близький до нуля) хоча б один частковий критерій, то результуючий критерій рівний нулю також.

Наведемо інший приклад застосування адитивної та мультиплікативної згорток в процесі розв’язання задачі багатокритеріальної оптимізації.

Задача векторної оптимізації (знайти максимум)

$$f_1(x_1, x_2) = 2x_1 + 5x_2 + 4x_3, \qquad f_2(x_1, x_2) = 3x_1 + 4x_2 + 4x_3,$$

$$x_1 + x_2 + x_3 \le 45, \qquad x_1 + 2x_2 + x_3 \le 40,$$

$$2x_1 + 3x_3 \le 75, \qquad 2x_1 + 4x_3 \le 90,$$

$$3x_1 + 4x_2 \le 50. \qquad 3x_1 + 2x_2 \le 25$$

$$x_1, x_2 \geq 0.$$

Відомо, що $\alpha_1 = 0{,}2$, а $\alpha_2 = 0{,}8$.

(Розв'язання: при $\alpha_1 = 0{,}2$, а $\alpha_2 = 0{,}8$, $F_{\max}^{(1)} = 126{,}75$.

$\alpha_1 = 0{,}7$, а $\alpha_2 = 0{,}3$, $F_{\max}^{(1)} = 131{,}125$)

Використаємо адитивну згортку

$$\max F(x_1, x_2) = \alpha_1 f_1(x_1, x_2) + \alpha_2 f_2(x_1, x_2),$$

$$x_1 + x_2 + x_3 \leq 45,$$

$$2x_1 + 3x_3 \leq 75,$$

$$3x_1 + 4x_2 \leq 50$$

$$x_1 + 2x_2 + x_3 \leq 40,$$

$$2x_1 + 4x_3 \leq 90,$$

$$3x_1 + 2x_2 \leq 25,$$

$$x_1, x_2 \geq 0.$$

Можемо використати графічний метод розв'язання задачі лінійного програмування чи симплекс метод.

$$\max F(x_1, x_2) = 0{,}2(2x_1 + 5x_2 + 4x_3) + 0{,}8(3x_1 + 4x_2 + 4x_3) =$$

$$= 0{,}4x_1 + x_2 + 0{,}8x_3 + 2{,}4x_1 + 3{,}2x_2 + 3{,}2x_3 = 2{,}8x_1 + 4{,}2x_2 + 4x_3$$

$$x_1 + x_2 + x_3 + x_4 = 45,$$

$$2x_1 + 3x_3 + x_5 = 75,$$

$$3x_1 + 4x_2 + x_6 = 50$$

$$x_1 + 2x_2 + x_3 + x_7 = 40,$$

$$2x_1 + 4x_3 + x_8 = 90,$$

$$3x_1 + 2x_2 + x_9 = 25$$

$$\alpha_1 = 0{,}4, \quad \alpha_2 = 0{,}6,$$

$$\max F(x_1, x_2) = 0{,}4(2x_1 + 5x_2 + 4x_3) + 0{,}6(3x_1 + 4x_2 + 4x_3) =$$
$$= 0{,}8x_1 + 2x_2 + 1{,}6x_3 + 1{,}8x_1 + 2{,}4x_2 + 2{,}4x_3 = 2{,}6x_1 + 4{,}4x_2 + 4x_3,$$

$$\alpha_1 = 0{,}6, \quad \alpha_2 = 0{,}4,$$
$$\max F(x_1, x_2) = 0{,}6(2x_1 + 5x_2 + 4x_3) + 0{,}4(3x_1 + 4x_2 + 4x_3) =$$
$$= 1{,}2x_1 + 3x_2 + 2{,}4x_3 + 1{,}2x_1 + 1{,}6x_2 + 1{,}6x_3 = 2{,}4x_1 + 4{,}6x_2 + 4x_3,$$

$$\alpha_1 = 0{,}7, \quad \alpha_2 = 0{,}3,$$
$$\max F(x_1, x_2) = 0{,}7(2x_1 + 5x_2 + 4x_3) + 0{,}3(3x_1 + 4x_2 + 4x_3) =$$
$$= 1{,}4x_1 + 3{,}5x_2 + 2{,}8x_3 + 0{,}9x_1 + 1{,}2x_2 + 1{,}2x_3 = 2{,}3x_1 + 4{,}7x_2 + 4x_3,$$

Згрупуємо отримані результати оптимізації:

$\alpha_1 = 0{,}2$, а $\alpha_2 = 0{,}8$, тоді $F_{\max}^{(1)} = 126{,}75$,

$\alpha_1 = 0{,}4$, а $\alpha_2 = 0{,}6$, тоді $F_{\max}^{(1)} = 128{,}5$,

$\alpha_1 = 0{,}6$, а $\alpha_2 = 0{,}4$, тоді $F_{\max}^{(1)} = 130{,}25$,

$\alpha_1 = 0{,}7$, а $\alpha_2 = 0{,}3$, тоді $F_{\max}^{(1)} = 131{,}125$,

$\alpha_1 = 1{,}0$, а $\alpha_2 = 0{,}0$, тоді $F_{\max}^{(1)} = 133{,}75$.

До недоліків цих методів можна віднести те, що рішення, які оптимізують узагальнену функцію, може виявитися незадовільним по одному чи декількох часткових критеріїв.

Це можна пояснити тим, що при досягненні максимуму узагальненої функції надзвичайно малі значення деяких показників $f_i(\bar{x})$ компенсуються великими значеннями інших.

4.4 Максімінний (мінімаксний) критерій

Максімінний (мінімаксний) критерії працюють за принципом компромісу, який ґрунтується на ідеї рівномірності. Сутність принципу максіміна полягає в наступному. При проектуванні складних технічних систем і наявності великого числа власних критеріїв встановити між ними аналітичний взаємозв'язок дуже складно, а в більшості випадків і неможливо. Тому намагаються знайти такі значення проектних (параметрів) $\bar{x} = \{x_1, x_2, ..., x_m\}$, при яких нормовані значення всіх власних критеріїв рівні між собою:

$$\alpha_i f_i(X) = K, \tag{4.11}$$

де $f_i(X)$ – нормоване значення i -го часткового критерію; K – константа.

При великій кількості власних критеріїв через складні взаємозв'язки домогтися виконання зазначеного вище співвідношення дуже складно. Тому, на практиці, так варіюють значеннями проектних змінних проектування x_1, x_2,…, x_m, при яких послідовно "підтягуються" ті нормовані критерії, чисельні значення яких у вихідному рішенні виявилися найменшими. Оскільки ця операція виробляється в області компромісу, підтягування “відстаючого” критерію неминуче призводить до зниження значень частини інших критеріїв. Але при проведенні ряду кроків можна домогтися визначеного ступеня зрівноважування суперечливих власних критеріїв, що і є метою принципу максиміна.

Формально принцип максиміна формулюється в такий спосіб: вибрати такий набір змінних $X^{(0)} \in X$, при якому реалізується максимум з мінімальних нормованих значень власних критеріїв, тобто $F(X^{(0)}) = \max \min f_i(X)$.

Такий принцип вибору $X^{(0)}$ іноді називають гарантований результат. Він запозичений з теорії ігор, де є основним принципом. Блок-схема алгоритму розв’язання ЗБО з використанням максимінного підходу (рисунок 4.3).

Якщо часткові критерії необхідно мінімізувати, то найбільш відстаючим критерієм є той, який приймає максимальне значення. В цьому випадку застосовують принцип мінімакса: $F(X^{(0)}) = \max \min f_i(X)$.

Наведемо приклад застосування максимінного підходу. Отже в таблиці 4.2 наведено результати вхідних даних.

Таблиця 4.2

$K_1^* = 8$, $K_1^* = 9$, $K_1^* = 6$, $K_1^* = 10$.

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**
K1	3	6	7	4	8	5	9	3	2
K2	3	4	4	9	4	8	7	7	5
K3	3	7	9	2	4	7	6	6	9
K4	9	5	3	7	9	3	7	5	3

В таблиці 4.2 S_i - *i*-й варіант рішення задачі багатокритеріальної оптимізації, K_i - *i*-й критерій оптимізації, K_i^* - номінальне значення *i*-го КО. В кожній комірці таблиці розміщено значення частинного критерія оптимальності для *i*-го варіант рішення частинного критерія. На наступному кроці необхідно звести усі значення критеріїв до безрозмірного виду. Для цього розділимо кожне значення частинного критерія оптимальності на номінальне значення даного КО. Отримані результати зображено в таблиці 4.3 (від другої до п’ятої стрічки таблиці). Далі для кожної альтернативи вибирається мінімальне значення безрозмірного значення критерія (шоста стрічка таблиці 4.3). Наступний крок алгоритму передбачає вибір максимального значення з шостої стрічки таблиці. В даному випадку вибраним варіантом є друга альтернатива.

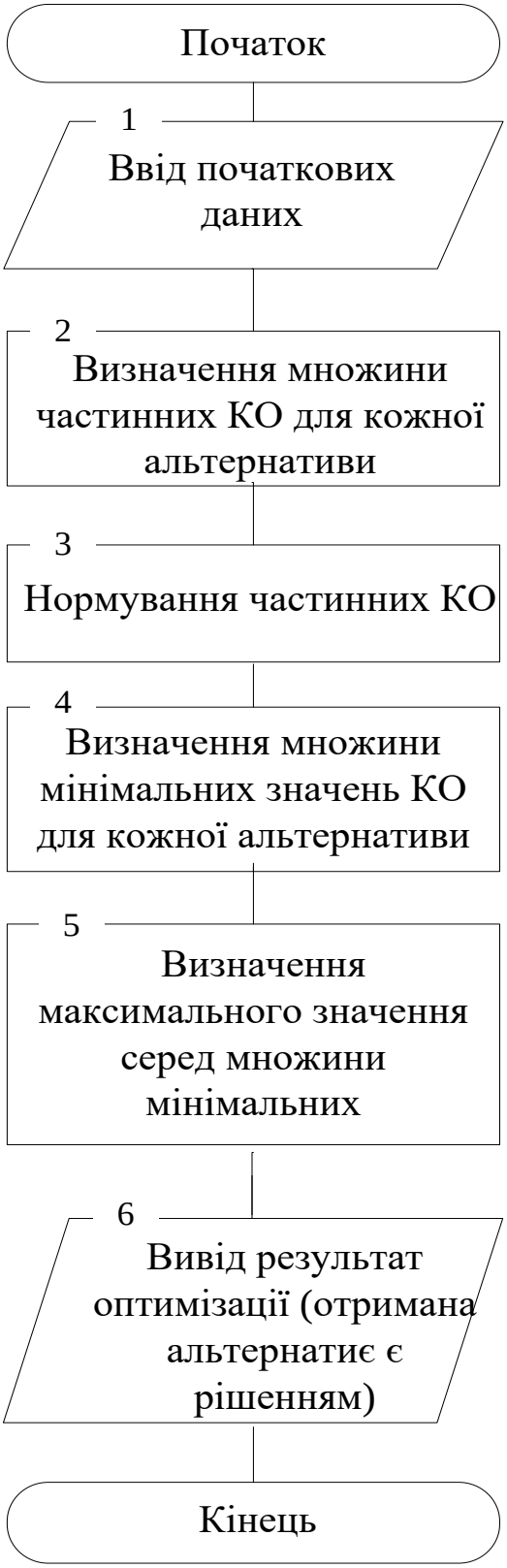


Рисунок 4.3 – Блок-схема алгоритму розв’язання ЗБО з використанням максимінного підходу

Таблиця 4.3 – Результати розв’язання максимінної задачі

	S_1	**S_2**	**S_3**	**S_4**	**S_5**	**S_6**	**S_7**	**S_8**	**S_9**
K_1	3/8	6/8	7/8	4/8	8/8	5/8	9/8	2/8	2/8
K_2	3/9	4/9	4/9	9/9	3/9	8/9	7/9	7/9	5/9
K_3	3/6	7/6	9/6	2/6	4/6	7/6	6/6	6/6	9/6
K_4	9/10	5/10	3/10	7/10	9/10	3/10	4/10	5/10	3/10
Min	3/9	4/9	3/10	2/6	3/9	3/10	4/10	2/8	2/8
	0,33	0,44	0,3	0,33	0,33	0,3	0,4	0,25	0,25
		Max							

Інший приклад, де наведемо особливості застосування мінімаксного підходу. Вхідні дані розміщені в тій же таблиці 4.2, а результати в таблиці 4.4.

Отже, перші кроки застосування мінімаксного підходу подібні до застосування максимінного підходу. Тобто, необхідно отримати значення часткових критеріїв оптимізації для кожної альтернативи і звести до безрозмірного виду шляхом ділення на номінальне значення кожного критерія оптимальності (див. від другої до п’ятої стрічки таблиці 4.4). Далі для кожної альтернативи вибирається максимальне значення безрозмірного значення критерія (шоста стрічка таблиці 4.4). Наступний крок алгоритму передбачає вибір мінімальне значення з шостої стрічки таблиці. В даному випадку вибраним варіантом є перша альтернатива (див. рисунок 4.4).

Таблиця 4.4 – Результати розв’язання мінімаксної задачі

	S_1	**S_2**	**S_3**	**S_4**	**S_5**	**S_6**	**S_7**	**S_8**	**S_9**
K_1	3/8	6/8	7/8	4/8	8/8	5/8	9/8	2/8	2/8
K_2	3/9	4/9	4/9	9/9	3/9	8/9	7/9	7/9	5/9
K_3	3/6	7/6	9/6	2/6	4/6	7/6	6/6	6/6	9/6
K_4	9/10	5/10	3/10	7/10	9/10	3/10	4/10	5/10	3/10
Max	9/10	7/6	9/6	9/9	8/8	7/6	9/8	6/6	9/6
	0,9	1,17	1,5	1,0	1,0	1,17	1,125	1,0	1,5
	Min								

Рисунок 4.4 – Блок-схема алгоритму розв’язання ЗБО з використанням мінімаксного підходу

4.5 Методи визначення значень вагових коефіцієнтів

У випадку рівноцінних критеріїв оптимальності w_i вибираються однаковими

$$w_i = \frac{1}{n},\ i = 1,2,...,n, \tag{4.12}$$

де n - кількість критеріїв оптимальності.

Для нерівноцінних критеріїв, тобто критеріїв, для яких особа, що приймає рішення може встановити пріоритет по важливості. В цьому випадку значення вагових коефіцієнтів вибираються відповідно до важливості критерію.

$$w_i > 0,\ \sum_{i=1}^{n} w_i = 1,\ i = 1,2,...,n. \tag{4.13}$$

Найбільш відомими, які широко використовуються проектуванні є метод ранжування та метод приписування балів.

Метод ранжування

Маємо: l - експертів та n - часткових критеріїв $f_i(\bar{x})$, де $i = \overline{1,n}$.

Цифрою 1 – позначаємо найбільш важливий КО, 2 – наступний за важливістю і цифрою n - критерій з найменшим значенням важливості.

Потім присвоюємо ранг кожному КО. Ранг 1 має оцінку n і є найбільш важливим з точки зору експерта, а ранг 2 – оцінка $(n-1)$ і так далі до рангу n, якому присвоюємо оцінку 1.

Позначимо ранг $i-го$ критерію (всіх є n) k-им експертом (який є l) $r_i^{(k)}$ вагові коефіцієнти w_i, $i = \overline{1,n}$ визначається з допомогою наступної формули:

$$C_i = \sum_{k=1}^{l} r_i^{(k)} \Big/ \sum_{i=1}^{n}\sum_{k=1}^{l} r_i^{(k)},\ i = \overline{1,n}. \tag{4.14}$$

Наведемо приклад застосування методу ранжування. Візьмемо п’ять критеріїв $F_i(\bar{x})$ і трьох експертів. Припустимо, що поставлені оцінки експертами наступні:

Таблиця 4.5

експерт / критерій	експерт 1	експерт 2	експерт 3
f_1	1 (5)	3 (3)	2 (4)
f_2	4 (2)	2 (4)	3 (3)
f_3	3 (3)	1 (5)	1 (5)
f_4	5 (1)	4 (2)	4 (2)
f_5	2 (4)	5 (1)	5 (1)

Результати розв’язання задачі:

$$\sum_{i=1}^{n}\sum_{k=1}^{l} r_i^{(k)},\ i=\overline{1,n}=3*15=45,$$

$$i=1,\ C_1=\sum_{k=1}^{l} r_1^{(k)} \Big/ 45 = (5+3+4)\Big/45 \approx 0{,}27, \qquad (3.15)$$

$$i=2,\ C_2=\sum_{k=1}^{l} r_2^{(k)} \Big/ 45 = (2+4+3)\Big/45 \approx 0{,}2,$$

$$i=3,\ C_3=\sum_{k=1}^{l} r_3^{(k)} \Big/ 45 = (3+5+5)\Big/45 \approx 0{,}29,$$

$$i=4,\ C_4=\sum_{k=1}^{l} r_4^{(k)} \Big/ 45 = (1+2+2)\Big/45 \approx 0{,}11,$$

$$i=5,\ C_5=\sum_{k=1}^{l} r_5^{(k)} \Big/ 45 = (4+1+1)\Big/45 \approx 0{,}13.$$

Метод приписування балів

Використаємо в якості вхідних даних з попереднього прикладу. Визначимо параметри ваги $i-го$ критерію на основі $k-го$ експерта

Таблиця 4.6

критерій \ експерт	експерт 1	експерт 2	експерт 3
f_1	5/15=0,33	3/15=0,20	4/15=0,27
f_2	2/15=0,13	4/15=0,27	3,15=0,20
f_3	3/15=0,20	5/15=0,33	5/15=0,33
f_4	1/15=0,07	2/15=0,13	2/15=0,13
f_5	4/15=0,27	1/15=0,07	1/15=0,07

Експерти оцінюють важливість КО по шкалі від 0 до 10 балів. Причому, можуть бути оцінки з дробовим значеннями та оцінки з однаковими значеннями для різних КО.

Знаючи бали $h_i^{(k)}$ $i-\tilde{a}\hat{i}$ критерію на основі $k-\tilde{a}\hat{i}$ експерта

$$H_i^{(k)} = h_i^{(k)} \Big/ \sum_{i=0}^{n} h_i^k . \tag{4.16}$$

Вага визначення для $i-\tilde{a}\hat{i}$ критерію $F_i(\bar{x})$ на основі оцінок $k-\tilde{a}\hat{i}$ експерта. Потім використовують формулу

$$C_i = \sum_{k=1}^{l} H_i^{(k)} \Big/ \sum_{i=1}^{n}\sum_{k=1}^{l} H_i^{(k)}, \; i=\overline{1,n},$$

$$i=1,\; C_1 = \sum_{k=1}^{l} H_1^{(k)} \Big/ 3 = (0{,}33+0{,}2+0{,}27)/3 \approx 0{,}27, \tag{4.17}$$

$$i=2,\; C_2 = \sum_{k=1}^{l} H_2^{(k)} \Big/ 3 = (0{,}13+0{,}27+0{,}2)/3 \approx 0{,}2,$$

$$i=3,\ C_3=\sum_{k=1}^{l}H_3^{(k)}\Big/3=(0{,}2+0{,}33+0{,}33)\Big/3\approx 0{,}29,$$

$$i=4,\ C_4=\sum_{k=1}^{l}H_4^{(k)}\Big/3=(0{,}07+0{,}13+0{,}13)\Big/3\approx 0{,}11,$$

$$i=5,\ C_5=\sum_{k=1}^{l}H_5^{(k)}\Big/3=(0{,}27+0{,}07+0{,}07)\Big/3\approx 0{,}13.$$

Отже, наведемо переваги та недоліки використання методів на основі згорток для розв'язання задач багатокритеріальної оптимізації.

Переваги:

1. Широко використовується в процесі проектування.

2. Дає змогу максимізувати запас по працездатності технічного вибору.

3. Шляхом зміни значень вагових коефіцієнтів є можливість досліджувати область слабо ефективних рішень (оптимальних за Слейтером).

Недоліки:

В процесі розв'язання задачі приходиться вирішувати багато однокритеріальних задач, які є складними та нелінійними і інколи зробити це неможливо.

Контрольні запитання до розділу 3

1. Що Ви розумієте під узагальненим критерієм?
2. Як будується адитивний узагальнений критерій?
3. Які переваги та недоліки адитивного критерію?
4. Як будується мультиплікативний узагальнений критерій?
5. Які переваги та недоліки мультиплікативного узагальненого критерія
6. Який алгоритм розв'язання ЗБО з використанням адитивної згортки?

7. Який алгоритм розв’язання ЗБО з використанням мультиплікативної згортки?

8. З якою метою виконують нормування часткових КО?

9. Який алгоритм застосування максимінного(мінімаксного) узагальненого критерія?

10. Які переваги та недоліки максимінного(мінімаксного) узагальненого критерія?

11. Які методи визначення вагових коефіцієнтів Ви знаєте?

12. Наведіль приклади зведення критеріїв оптимізації до безрозмірного виду.

13. Який алгоритм розв’язання ЗБО з використанням максимінної згортки?

14. Який алгоритм розв’язання ЗБО з використанням мінімаксної згортки?

РОЗДІЛ 5. РОЗВ'ЯЗАННЯ ЗБО З ВИКОРИСТАННЯМ ПОБУДОВИ МНОЖИНИ ПАРЕТО

Розглянуті в попередніх розділах методи розв'язання ЗБО зручно використовувати, коли кількість альтернативних рішень не є великою. Для випадків, коли маємо велику кількість альтернативних рішень, зручніше використати поєднання методів розв'язання ЗБО, яке полягає в тому, що на першому кроці необхідно визначити альтернативи, що належать до множини Парето, а на другому, для прикладу, згортки (адитивна чи мультиплікативна) чи на основі досвіду проектувальника аналізувати вибір альтернатив.

5.1. Основні поняття та визначення

Отже, процес розроблення КСМ визначається технічним завданням, де визначено основні вимоги до об'єкту синтезу. Кожна вимога – це окремий критерій, який залежить від проектних параметрів. Відповідно, розроблення КСМ передбачає задоволення усіх вимог ТЗ. Математична формалізація цієї задачі визначає розв'язання задачі багатокритеріальної оптимізації.

В процесі пошуку рішень ЗБО кількість варіантів, які задовольняють ТЗ може досягати значної кількості. Відповідно, необхідно вибрати найоптимальніше. Разом з тим, слід додати, що буде існувати ще більша кількість рішень, які не будуть задовольняти ТЗ на розроблення КСМ, які також треба відкинути.

Пошук оптимальних розв'язків ЗБО з використанням побудови множини Парето передбачає використання ряду термінів, підходів та означень.

ЗБО формально описується множиною критеріїв

$$F(\overline{X}) = (f_1(\overline{X}), f_2(\overline{X}),..., f_k(\overline{X})),$$

$$\overline{X} = (x_1, x_2, ..., x_l), \quad k = \overline{1,n}, \qquad l = \overline{1,j} \tag{5.1}$$

де k - кількість критеріїв оптимальності, j - кількість проектних змінних.

Кількість рішень такої ЗБО (5.1) може бути безліч, або жодного (в даному випадку йде мова про розв'язки, які задовольняють ТЗ).

В процесі розв'язання ЗБО необхідно вибирати кращі розв'язки. В цьому випадку доведеться порівнювати два рішення ЗБО між собою.

Позначимо перше розв'язання ЗБО через $\overline{X}^1$, а друге – $\overline{X}^2$ (зрозуміло, що $\overline{X}^1$ та $\overline{X}^2$ належать до області визначення, яку позначимо через D, тобто, $\overline{X}^1, \overline{X}^2 \in D$. Стверджують, що $\overline{X}^1$ краще (володіє перевагою) у порівнянні з $\overline{X}^2$, якщо усі значення цільових функцій $f_k(\overline{X}^1) \geq f_k(\overline{X}^2)\ \forall k = \overline{1,n}$. Разом з тим, існує хоча б один з критеріїв ($k_{\max}$) для якого $f_{k\max}(\overline{X}^1) > f_{k\max}(\overline{X}^2)$. Тобто, розв'язання ЗБО $\overline{X}^1$ не гірше по цих усіх часткових критеріях і серед них існує одне, яке має перевагу над одним з часткових критеріїв розв'язання ЗБО $\overline{X}^2$.

Розв'язання $\overline{X}^* \in D$ задачі (5.1) називається **ефективним розв'язанням** ЗБО, якщо для нього не існує розвязань з більшою перевагою. Інакше можна сказати, що ефективним розв'язанням називається таке розв'язання $\overline{X}^*$, яке не можна покращити за якимось з часткових критеріїв, не погіршивши, при цьому, значення інших критеріїв.

Множина ефективних розв'язань ЗБО називається множиною Парето і позначається *Pareto(D).*

Вектор значень критеріїв, обчислених для ефективного розв'язання $F_{eff}(\overline{X}^*)$, називається **ефективною оцінкою**. Сукупність всіх ефективних оцінок, тобто образ множини Парето в просторі критеріїв, називається **множиною ефективних оцінок** і, як правило, позначається як $F_{eff}(Pareto)$.

Розв'язання $\overline{X}^1 \in D$ називається **слабоефективним розв'язанням задачі** (2.10), якщо для нього не існує розв'язання $\overline{X}^2$ такого, що $\forall i = 1,k$ $f(\overline{X}^1) > f(\overline{X}^2)$, іншими словами, слабоефективне розв'язання – розв'язання, яке не може бути покращено одночасно за всіма критеріями.

На практиці, в більшості випадків, ми маємо справу з слабо ефективним

розв’язанням ЗБО.

5.2 Правило визначення рішень множини Парето

Принцип Парето дає змогу звузити клас можливих претендентів на остаточне розв’язання і виключити з розгляду свідомо неконкурентоздатні варіанти. А остаточний вибір здійснюється на основі додаткової інформації про переваги особи, яка приймає рішення.

Правило. Множина Парето – ефективних оцінок $Pareto(Y_D)$ являє собою «північно – східну» границю множини Y_D без тих його частин, які паралельні одній з координатних осей чи розміщені в «глибоких» провалах.

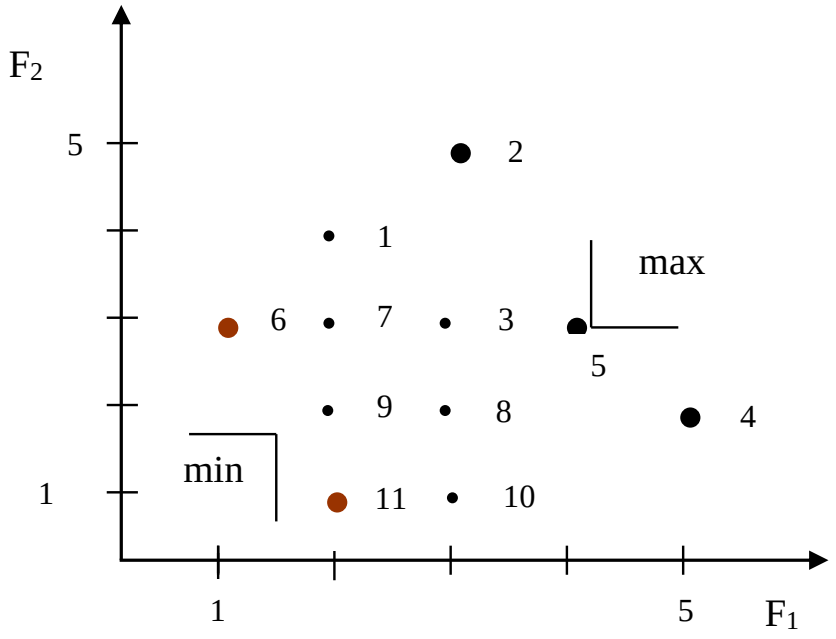


Рисунок 5.1 – Ілюстрація правила для дискретних значень оцінок часткових критеріїв

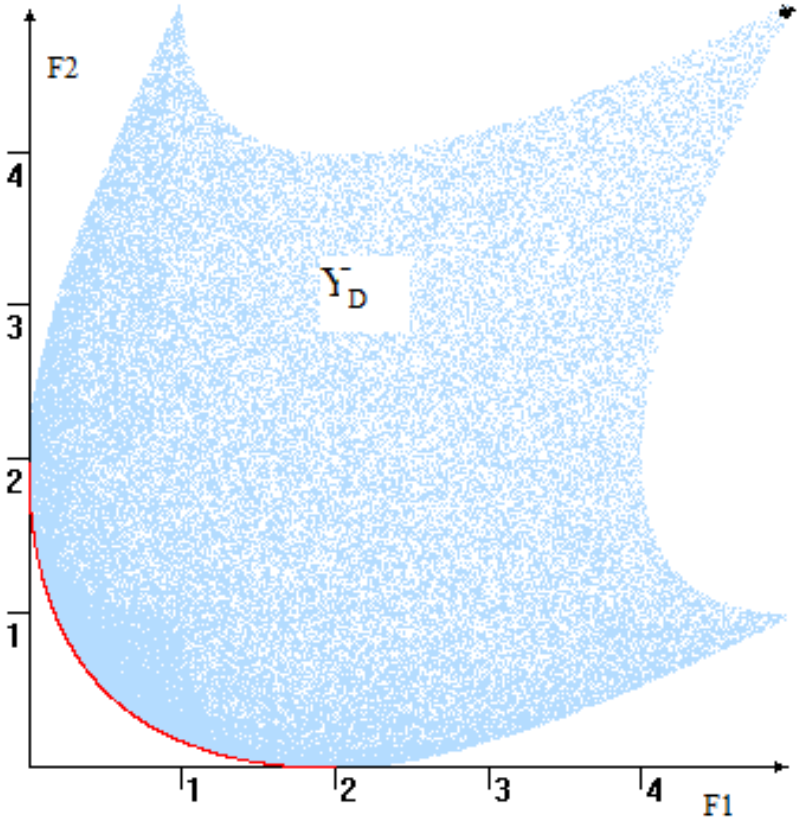


Рисунок 5.2 – Ілюстрація правил для. Неперервної області розв'язання ЗБО

5.3 Приклади розв'язання ЗБО з використанням побудови множини Парето

Наведемо приклад і визначимо рішення, які відносяться до множини Парето. Існує 10 варіантів металоріжучих станків, серед яких для проектованої дільниці необхідно вибрати найкращий. Станки оцінені експертами по двох показниках (критеріях): продуктивності та надійності. Оцінка проводилась по 11 - бальній шкалі від 0 до 10. Результати оцінки станків наведені в таблиці 5.1.

В даному випадку, чим більше значення оцінки, тим кращий параметр станка.

Таблиця 5.1 –Експертні оцінки станків по критеріях продуктивності та надійності

Критерії	Оцінки експертів (бали) для станків									
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
Продуктивність (П)	6	4	10	3	10	0	2	4	6	7
Надійність (Н)	6	2	1	7	4	4	10	4	8	2

Застосуємо графічний підхід до розв'язання даної ЗБО. Отже, позначимо горизонтальну вісь як надійність, а вертикальну – продуктивність. Кожне з рішень, яке включає дві оцінки експертів, позначаємо точкою. Коли відобразимо всі рішення на системі координат надійність-продуктивність, то отримаємо розміщення точок, яке зображено на рисунку 5.3.

Згідно з принципом Парето – ефективними рішеннями є станки C_5, C_7 і C_9. При цьому, слід зауважити, що станки C_1 і C_3, не входять до цієї множини, оскільки C_3 має однакове значення оцінки з C_5 за критерієм «продуктивність», але володіє гіршою оцінкою за критерієм «надійність». Аналогічна ситуація з станком C_1.

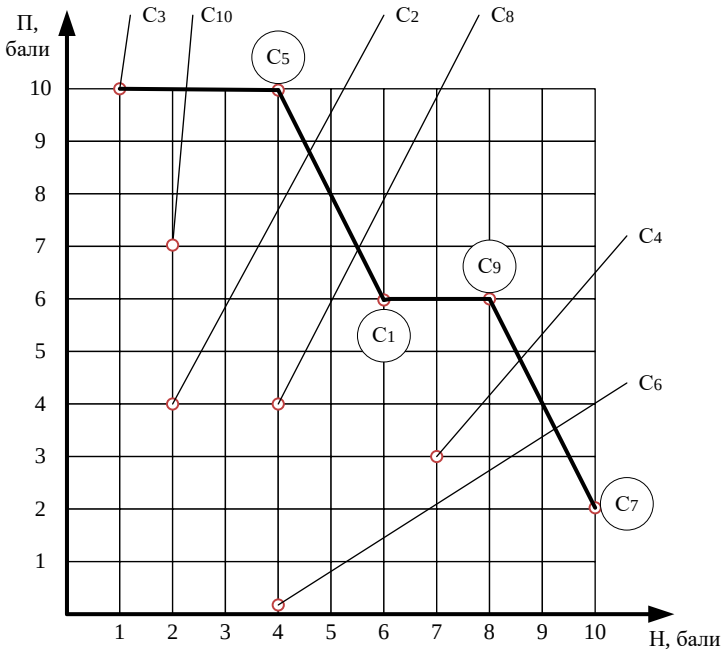


Рисунок 5.3 – Парето – ефективними рішеннями є варіанти станків С5, С7 та С9.

Виберемо інші два критерії оптимальності для даної задачі, а саме: вартість та затрати на їхнє обслуговування. В даному випадку чим менші значення критеріїв оптимальності, тим краще. Наведемо значення цих критеріїв у таблиці 5.2.

Таблиця 5.2 – Значення параметрів станків

Критерії	Параметри станків									
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
Ціна. тис.грн	60	40	100	30	100	30	20	40	60	70
Затрати на обслуго-вування, тис.грн	6	2	1	7	4	4	10	4	8	2

Аналогічно застосуємо графічний підхід до визначення ефективних рішень множини Парето. Відповідно, отримаємо наступні результати, а саме: ефективними рішеннями є станки C_3, C_2 C_6 та C_7 (див. рисунок 5.4).

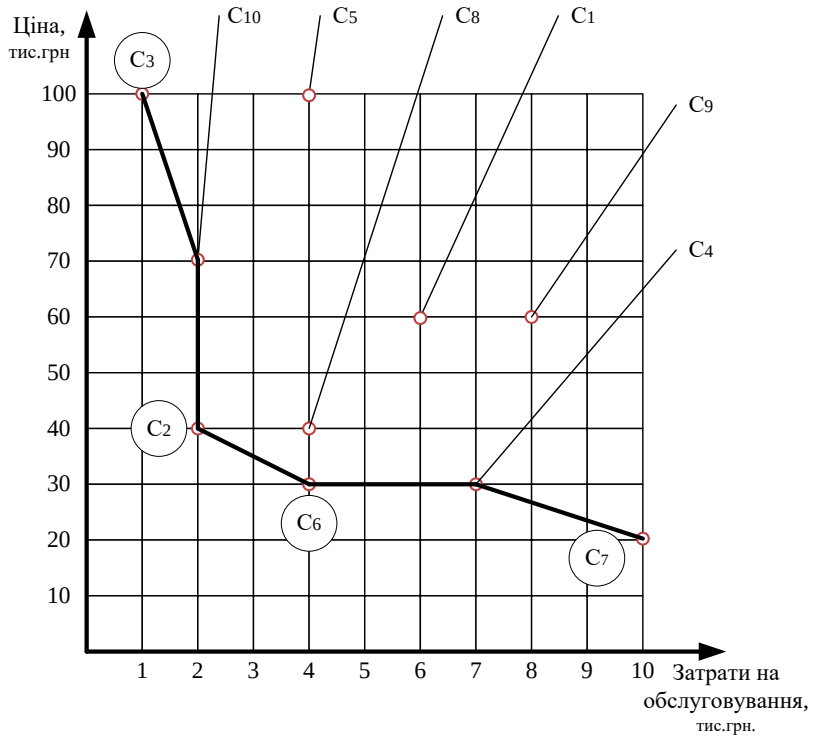


Рисунок 5.4 – Ефективними рішеннями є станки C_3, C_2 , C_6 та C_7

Наведемо наступний підхід знаходження оптимальних рішень, які відносяться до множини Парето для випадку, коли цільові функції та обмеження задані у вигляді виразів.

Припустимо, що необхідно знайти максимум ЗБО з двома критеріями оптимальності. В даному випадку маємо дві цільові функції.

$$\begin{aligned} f_1(x) &= -x_1 + 2x_2 + 2 \to \max, \\ f_2(x) &= x_1 + x_2 + 4 \to \max \end{aligned} \tag{5.2}$$

Причому, забезпечити виконання таких умов:

$$\begin{aligned} & x_1 + x_2 \le 15, \\ & 5x_1 + x_2 \ge 1, \\ & -x_1 + x_2 \le 5, \\ & x_2 \le 20. \end{aligned} \tag{5.3}$$

Використаємо для цього розроблену програмну систему. Приклад меню для введення параметрів ЗБО (5.2 – 5.3) зображено на рисунках 5.5 та 5.6.

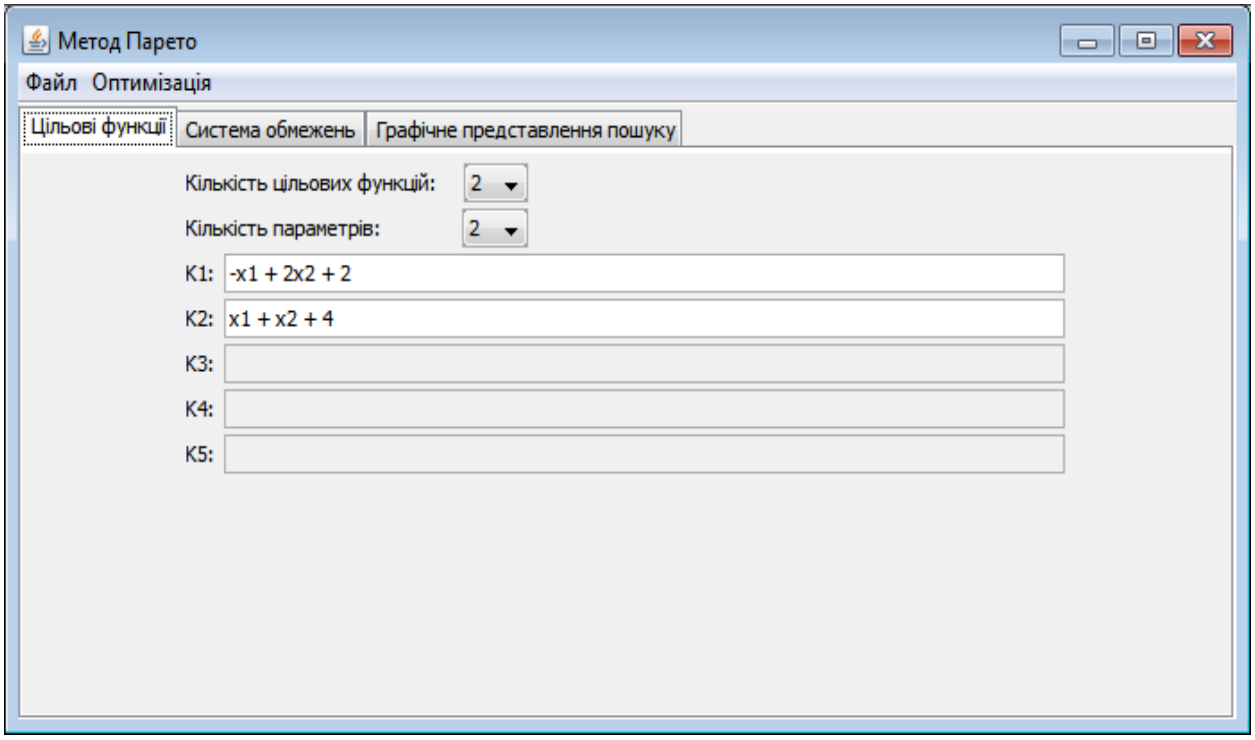


Рисунок 5.5 – Закладка «Цільові функції»

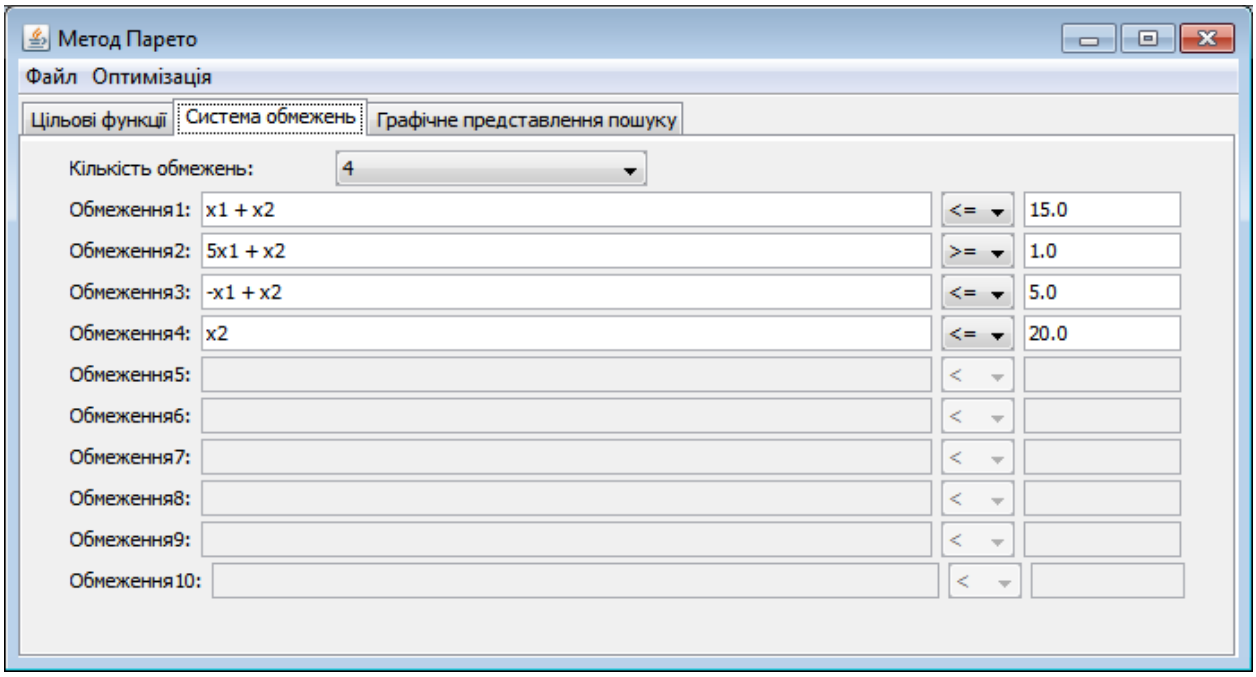


Рисунок 5.6 – Закладка «Система обмежень»

Результати розв’язання двохкритеріальної ЗБО зображено на рисунку 5.7.

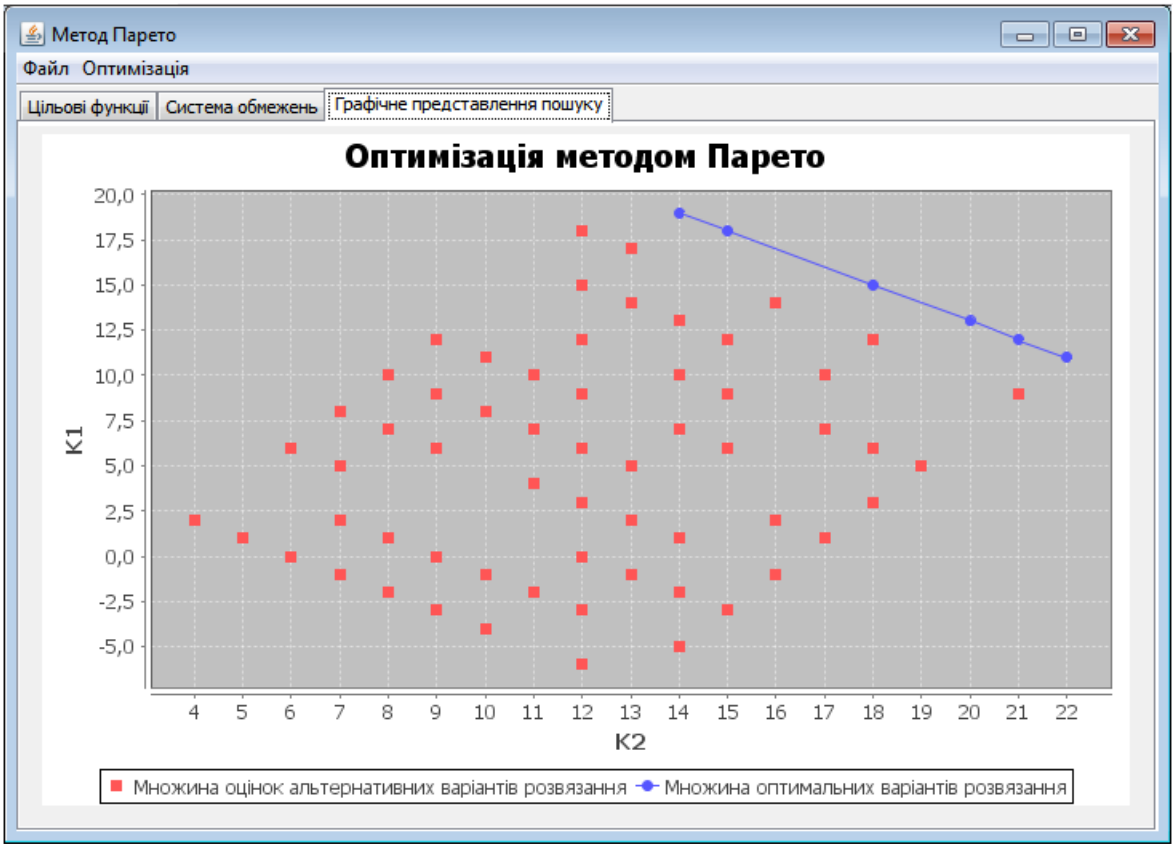


Рисунок 5.7 – Закладка «Графічне представлення оптимізації»

З отриманого графіка бачимо, що до множини оптимальних рішень належать рішення, які розміщені на лінії.

Контрольні запитання до розділу 5

1. Які основні кроки включає алгоритм застосування методу на основі побудови множини Парето?
2. Назвіть принцип Парето?
3. Які розв’язання ЗБО включає множина ефективних оцінок?
4. Яке розв’язання ЗБО називається слабоефективним?
5. Навести приклад рішень ЗБО з використанням побудови множини Парето при пошуку максимуму?
6. Навести приклад рішень ЗБО з використанням побудови множини Парето при пошуку мінімуму?
7. Яке розв’язання ЗБО називається ефективним ?

РОЗДІЛ 6. ЗАГАЛЬНА ХАРАКТЕРИСТИКА МОДЕЛЕЙ ТА ПРОЦЕСУ МОДЕЛЮВАННЯ. РОЛЬ ТА РІВНІ МОДЕЛЮВАННЯ В АВТОМАТИЗОВАНОМУ ПРОЕКТУВАННІ

Система автоматизованого проектування (САПР) включає сім видів забезпечень, зокрема: програмне, інформаційне, математичне, технічне, лінгвістичне, організаційне та методичне [45]. Ядром будь-якої САПР є математичне забезпечення.

Математичне забезпечення систем автоматизованого проектування (МЗ САПР) [45 – 48] включає моделі, методи та алгоритми, які використовуються під час автоматизованого проектування (моделювання).

6.1 Поняття про об’єкт моделювання (проектування) та його основні параметри

Проектування [45, 46] – це комплекс робіт, метою яких є отримання опису ще неіснуючого технічного об’єкта, який достатній для реалізації та виготовлення об’єкта в заданих умовах. **Об’єктом проектування** будемо називати виріб (для прикладу, автомобіль, інтегральна схема, персональний комп’ютер тощо) чи процес (технологічний процес виготовлення двигуна автомобіля, інтегральної схеми, літака та ін.).

Будь-який об’єкт проектування характеризується множиною параметрів. Під **параметром** будемо розуміти величину, яка виражає властивість об’єкта проектування (системи), або його частини чи впливу зовнішнього середовища на об’єкт розроблення (систему).

В процесі автоматизованого проектування окрім терміну “параметр” об’єкту моделювання досить часто використовують термін “**фазова змінна**” – величина, яка характеризує енергетичне чи інформаційне наповнення елемента або підсистеми. Сукупність значень фазових змінних зафіксованих в певний момент часу процесу функціонування називається **станом об’єкта проектування** (моделювання).

Під **поведінкою** (динамікою) будемо розуміти зміну стану об’єкта (системи) в процесі функціонування.

Усі параметри об’єкта моделювання (проектування) можна розділити на чотири основні групи: вхідні параметри ($X_{вх.}$); зовнішні ($X_{зов.}$); внутрішні ($X_{вн.}$); вихідні параметри ($X_{вих.}$).

Отже, **вхідними параметрами** об’єкта проектування будемо називати такі параметри, якими ми можемо керувати або з допомогою яких керуємо об’єктом. В залежності від складності об’єкта проектування, кількість вхідних параметрів може бути різна. Для складних об’єктів кількість може доходити до тисяч, навіть десятків тисяч. Позначимо всі вхідні параметри вектором $\overline{X_{вх.}}$, а при необхідності виділення конкретного j-го вхідного параметра $x_{вх.j}$. В загальному випадку $\overline{X_{вх.}} = (x_{вх.1}, x_{вх.2}, ..., x_{вх.n})$, n – загальна кількість вхідних параметрів.

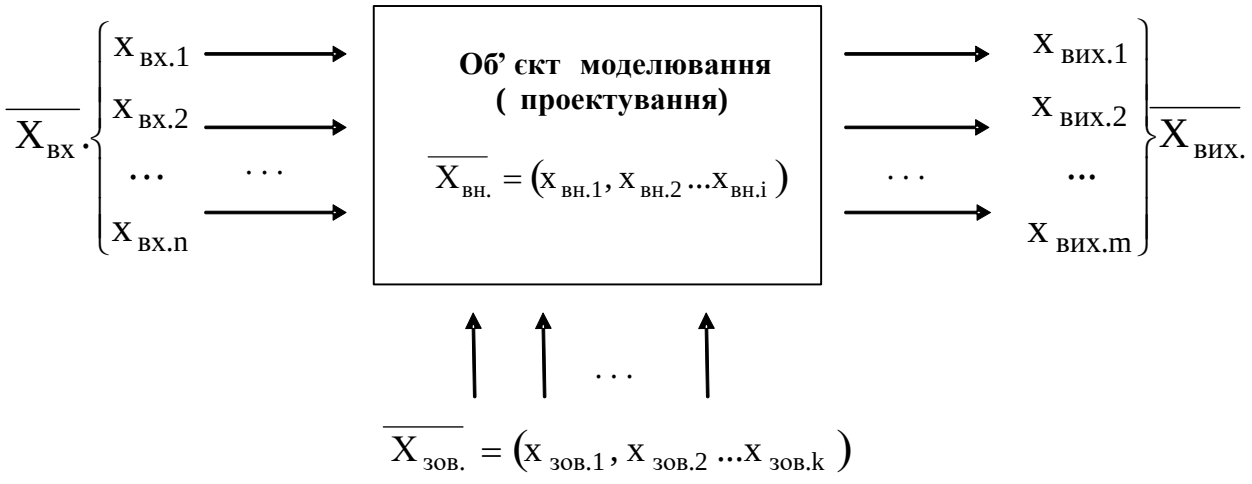


Рисунок 6.1 – Групи параметрів об’єкта моделювання (проектування)

Для прикладу, якщо в якості об’єкта проектування розглядати електронну схему підсилювача низької частоти, то вхідними параметрами є амплітуда струму вхідного сигналу, частота вхідної напруги, напруга та струм живлення, тощо.

До **параметрів зовнішнього середовища** відносяться такі параметри: температура оточуючого середовища, вологість, тиск, вібрації та інші. Відповідний вектор для параметрів зовнішнього середовища запишемо як $\overline{X_{зов.}} = (x_{зов.1}, x_{зов.2}, ..., x_{зов.k})$, а у випадку, зазначення конкретного параметра – $x_{зов.j}$. Як правило, при проектуванні будь-якого об’єкта в технічному завданні відведено розділ, в якому задаються граничні значення параметрів оточуючого середовища (кліматичні умови) в яких має функціонувати об’єкт проектування (моделювання).

Внутрішні параметри об’єкта проектування дають змогу характеризувати його внутрішню структуру та описати функціонування окремих складових даного об’єкта. До них відносяться, у випадку розгляду тієї ж електричної схеми, опори резисторів, ємності конденсаторів, індуктивності котушок, товщини провідників та інше. При описі параметрів цієї групи будемо використовувати наступне позначення $\overline{X_{вн.}} = (x_{вн.1}, x_{вн.2}, ..., x_{вн.i})$.

Вихідні параметри об’єкта проектування залежать від вхідних, зовнішніх та внутрішніх параметрів, що можна записати вигляді наступної функціональної залежності $\overline{X_{вих.}} = \varphi(\overline{X_{вх}}, \overline{X_{вн.}}, \overline{X_{зов.}})$, де φ – деяка функція, яка дає змогу встановити зв’язок між вектором вихідних параметрів та векторами вхідними, внутрішніх і зовнішніх параметрів.

У випадку розгляду тієї ж електричної схеми підсилювача, то вихідними параметрами даного об’єкта проектування є вихідна напруга, вихідний струм, коефіцієнт підсилення, вихідна потужність, його маса тощо.

6.2 Поняття моделі та моделювання

Зміст понять “модель”, ”моделювання” в різних сферах науки та техніки можуть дещо відрізнятися. Але незважаючи на це можна виокремити одну визначальну спільну властивість: модель завжди в тій чи іншій мірі імітує або заміняє оригінал. Цей факт дає нам можливість стверджувати, що певний об'єкт **A** (тут і надалі термін об’єкт трактується в як найширшому значенні: як було зазначено вище об’єктами можуть бути не тільки матеріальні тіла, а й різноманітні явища, процеси, абстрактні поняття і т.д.) є моделлю об’єкта **B** відносно деякої системи характеристик (властивостей), якщо **A** будується (вибирається) для імітації (заміни) **B** за цими характеристиками. З цієї точки зору **моделювання** можна розглядати як процес побудови моделей, точніше процес представлення об'єкта-оригінала адекватною моделлю та проведення дослідження з використанням цієї моделі з метою отримання певної інформації про оригінал (об’єкт проектування). З філософської точки зору під моделюванням розуміють процес опосередкованого пізнання реальності. Слід відзначити, що в процесі моделювання модель виступає одночасно і як засіб, і як об'єкт досліджень, який знаходиться у певному відношенні подібності до модельованого об'єкту. **Подібність** – це взаємно-однозначна відповідність між двома об'єктами, коли відомі функції переходу від параметрів одного об'єкту до параметрів іншого, а математичні описи можуть бути зроблені тотожними. Цей своєрідний дуалізм моделі є характерним для різних систем автоматизованого проектування, оскільки під час автоматизованого проектування розробник оперує не самими об'єктами, а їхніми моделями, тобто моделювання виступає і як предмет, і як засіб створення проекту складної системи.

Отже, **модель** – це спеціальний об’єкт, який в деякому сенсі та з певною ступінню подібності заміняє оригінал.

Дещо інше визначення **моделі** – це об’єкт чи система об’єктів, які в певній мірі подібні до об’єкта моделювання (проектування), дають змогу

відобразити основні властивості об’єкта розгляду, процесу чи явища і описують вихідні параметри моделювання (проектування).

З вищенаведених визначень слідує, що з принципової точки зору, не існує моделі, яка б була повним еквівалентом оригіналу, тобто описувала всі параметри об’єкта моделювання. Будь-яка модель відображає лише деякі сторони оригіналу (параметри, які цікавлять розробника оригіналу), завдяки чому модель набуває певної ідеалізованої форми. Тому, досить часто для всестороннього вивчення оригіналу доводиться будувати і досліджувати цілу низку моделей. Складність моделювання як процесу полягає у відповідному виборі такої сукупності моделей, які заміняють реальний об’єкт у потрібному відношенні (відображають лише певні характеристики об’єкта моделювання).

В основі процесу моделювання лежать інформаційні процеси, оскільки в процесі реалізації моделі отримують інформацію про об’єкт-оригінал, проведення експерименту з моделями супроводжується керуючою інформацією і на останньому етапі відбувається обробка (інтерпретація) отриманих результатів.

Слід додати, що процес моделювання, згідно вищенаведеного визначення, передбачає розв’язання двох задач, а саме: побудову моделі оригіналу (в англомовній літературі – modeling) та процес дослідження параметрів оригіналу з використанням побудованої моделі (simulation).

6.3 Види моделей

Класифікація моделей може здійснюватися за різними критеріями і носить умовний характер. Більшість дослідників поділяють моделі на два великих класи: **предметні** (експериментальні) та **абстрактні** (теоретичні).

Предметні моделі утворюються з сукупності матеріальних об’єктів і представляють собою реально існуючі пристрої двох основних типів.

Предметні моделі першого типу (їх часто називають **фізичними**) відтворюють оригінал у спрощеному вигляді, причому природа матеріальних елементів (складових частин) цих моделей може відрізнятися від природи елементів з яких складається об’єкт моделювання. Прикладом предметної фізичної моделі може бути **макет**. Фізичні моделі створюються на основі теорії подібності, причому подібність здійснюється за тими параметрами, які є суттєвими для дослідника. Так, наприклад, для вивчення опору руху корабля потрібна модель, зовнішні форми якої подібні до зовнішніх форм оригіналу, а для дослідження міцності цього ж корабля – модель, що імітує його силовий каркас. **Предметні моделі другого типу** – **моделі на основі методу аналогій** – ґрунтуються на тому факті, що різні фізичні явища описуються за допомогою однакового математичного апарату. Наприклад, коливальні процеси в механічних та електричних системах описуються однаковими звичайними диференціальними рівняннями другого порядку. Це означає, що замість проведення досить складного і дорогого експерименту з механічною системою можна провести простіший експеримент з електричною системою, яка в цьому випадку і буде виступати моделлю.

Абстрактні моделі описують об’єкт дослідження за допомогою певної мови. Абстрактність цих моделей проявляється в тому, що компонентами моделі є поняття (креслення, схеми, графіки, рівняння, алгоритми), а не фізичні елементи. За характером мови опису абстрактні моделі можна поділити на **фізичні**, **математичні**, **економічні** тощо. **Абстрактна фізична** модель описує оригінал на основі абстрактних уявлень мовою фізики. Їх отримують в результаті абстрагування від несуттєвих, з точки зору дослідника, властивостей та зв’язків об’єкта дослідження. Прикладами теоретичних фізичних моделей можуть бути такі поняття, як матеріальна точка, абсолютно тверде тіло, пружне середовище, абсолютно чорне тіло тощо. В механіці та фізиці дуже часто побудова абстрактних фізичних моделей виступає проміжним, підготовчим етапом процесу побудови математичних моделей. Зокрема, загальне визначення

м**атематичної моделі** (ММ) – це абстрактна модель представлена мовою математичних співвідношень.

Ці моделі мають форму функціональних залежностей між групами параметрів досліджуваного об'єкту. За ступенем абстрагування математичні моделі знаходяться на найвищому рівні ієрархії. Власне побудові та дослідженню цих моделей і буде присвячена найбільша увага в цьому конспекті лекцій. Серед абстрактних моделей розрізняють гносеологічні, інформаційні, сенсуальні та концептуальні моделі. **Гносеологічні моделі** *призначені для дослідження об'єктивних законів природи* (наприклад, моделі сонячної системи, біосфери, світового океану тощо). **Інформаційні моделі** *описують поведінку оригіналу, а не імітують його*. **Сенсуальні моделі** – *це моделі, що впливають на почуття людини* (музика, мистецтво тощо). **Концептуальні моделі** *мають за мету виявлення причинно-наслідкових зв'язків, які характерні для об'єкту дослідження і які є суттєвими в рамках цього дослідження.* Один і той же об'єкт може бути представлений різними концептуальними моделями, наприклад одна концептуальна модель може відображати часові аспекти функціонування системи, а інша – вплив відмов на працездатність системи.

6.4 Методи моделювання

Під **методом** будемо розуміти спосіб розв'язання деякої складної задачі. Досить часто під **методом** розуміють об'єднання моделей та алгоритмів для розв'язання деякої складної задачі, що акцентує увагу на складових і підпорядкованості моделей та алгоритмів терміну метод. В цьому випадку під **алгоритмом** будемо розуміти набір кроків для досягнення поставленої мети.

Під час моделювання та проектування складних систем застосовують такі методи **аналітичного**, **чисельного**, **імітаційного**, **натурного** і **напівнатурного** моделювання.

Аналітичні методи призначені для отримання функціональних залежностей шляхом послідовного застосування математичних формул та правил. Аналітичні методи [32] застосовують у випадку, коли математична модель записана у вигляді рівнянь, наприклад диференціальних або інтегральних. При використанні аналітичних методів часто виникають труднощі пов'язані з неможливістю отримання розв'язку в аналітичній формі, що значно обмежує сферу їх застосування. Але незважаючи на ці труднощі аналітичного підходу, результати отримані в аналітичній формі є універсальними і мають дуже велику цінність, оскільки вони дають змогу перевірити точність інших підходів. Розроблені спеціальні мови опису аналітичних перетворень для комп'ютерів (наприклад, "Аналітик"), цілі системи програмування ("Авто-Аналітик") та пакети символьних математичних перетворень ("Mathematica").

Чисельні методи ґрунтуються на побудові скінченої послідовності дій над числами [32 – 40], яка призводить до бажаного результату. Дослідження математичних моделей за допомогою чисельних методів полягає в заміні "неперервних" математичних операцій та відношень на відповідні дискретні аналоги: інтегралів на суми, похідних на їх аналоги у формі різницевих співвідношень, нескінченні суми на скінченні і т.д. Серед недоліків чисельних методів слід відзначити те, що вони завжди дають наближений розв'язок, який містить певну похибку та можуть давати нестійкий розв'язок, який сильно залежить від деяких вхідних даних. До переваг чисельних методів слід віднести значно ширшу, у порівнянні з аналітичними, сферу застосування (універсальність) та можливість багаторазового застосування розроблених алгоритмів до розв'язання різних задачах. Результатом чисельних методів завжди є набори чисел (для прикладу розподіли напружень та зміщень у конструкції об'єкта проектування тощо), які потім можна представити у вигляді таблиць і графіків.

Аналітичні та чисельні методи відносяться до "чисто" математичних засобів дослідження моделей, застосування яких значно ускладнюється у

випадках, коли необхідно відтворити в моделі динаміку складних просторово-часових відношень між компонентами системи, усю різноманітну гамму існуючих зв'язків, законів керування, адаптивних властивостей тощо. Дослідження таких систем доцільніше та ефективніше проводити за допомогою методів імітаційного моделювання. Ці методи використовують, коли моделі являють собою змістовний опис об'єктів дослідження у формі алгоритмів. Моделі такого типу називаються імітаційними або алгоритмічними. Вони адекватно відображають як структуру систем, що досягається ототожнюванням елементів системи з відповідними елементами алгоритмів, так і процеси функціонування системи, зображені в логіко-математичнй формі. Особливістю цього підходу до моделювання є те, що для опису моделі використовуються спеціальні алгоритмічні мови, які є більш гнучким засобом опису складних систем, ніж мова математичних функціональних відносин. Завдяки цьому в імітаційних моделях часто знаходять відображення багато деталей структури та функцій складних систем, які вимушено втрачаються або нехтуються в математично строгих моделях. Для складних систем характерним є статистичний підхід, у зв'язку з чим для аналізу імітаційних моделей часто використовується метод Монте-Карло [41, 42]. Сам процес побудови та аналізу імітаційних моделей за допомогою цього методу отримав назву статистичне моделювання. Суть **статистичного моделювання** полягає в отриманні за допомогою комп'ютера статистичних даних про процеси, які відбуваються в модельованій системі, з наступною обробкою їх методами математичної статистики. До переваг статистичного моделювання слід віднести принципову можливість проведення аналізу систем довільної складності з довільним ступенем деталізації. Негативна сторона – трудомісткість процесу моделювання та частковий характер отриманих результатів, на основі яких важко виявити загальні закономірності.

Методи **натурного моделювання** [43] включають проведення експерименту з реальним об'єктом та обробку результатів на основі теорії подібності [44 - 47]. Вони грунтуються на вимірюванні характеристик процесів,

що відбуваються в реальних системах. Перевага натурного моделювання полягає в тому, що експериментальні дослідження є, в більшості випадків, джерелом найбільш достовірних даних, а недолік – результати носять частковий характер.

Напівнатурне моделювання складних об’єктів здійснюють за допомогою комбінованих моделей. В структуру таких моделей включають математичні співвідношення, які описують функціонування елементів об’єкту розроблення, а також деякі реальні матеріальні елементи. Завдяки цьому часто вдається досягнути оптимальної взаємодії між обчислювальним та натурним експериментами. Найбільш ефективними методи напівнатурного моделювання є при проектуванні автоматизованих систем управління, які часто складаються з елементів різної фізичної природи.

Досить часто під час моделювання та проектування використовують так звані “**еврістичні**” методи (підходи, моделі). В цьому випадку мається на увазі такий метод дослідження, який грунтується на неформальних, інтуітивних міркуваннях і припущеннях дослідника, що спираються на його досвід при розв’язанні подібних задач, задач з інших галузей науки та техніки тощо. В більшості випадків еврістичні методи дають змогу значно скоротити кількість варіантів, які необхідно переглянути під час пошуку розв’язання задачі чи вирішення проблеми. Разом з тим необхідно зауважити, що цей метод (прийом) не гарантує отримання найкращого розв’язання задачі.

6.5 Рівні проектування (моделювання) в САПР

Будь-який об’єкт проектування з позицій системного аналізу можна розглядати як систему. Відповідно під **системою** [30] будемо розуміти множину елементів, які знаходяться у відношеннях та зв’язках між собою, а

елементом – таку частину системи, представлення про яку не доцільно піддавати при проектуванні подальшому поділу (декомпозиції).

Характерними ознаками складних систем є великі розміри як за кількістю складових частин, так і за кількістю параметрів, що характеризують процес їх функціонування. Складність поведінки системи, як єдиного цілого, обумовлена різноманітністю взаємоперетинаючих зв'язків між її елементами, а також цілеспрямованістю та багатофакторністю функціонування. Отже складна система характеризується наступними властивостями: цілісності, цілеспрямованості, можливістю поділу на складові, ієрархічності, багатоаспектністю. Тому такі системи принципово неможливо точно описати і передбачити їх поведінку. Єдиний метод, який дає змогу суттєво спростити процес проектування, а часто і експлуатацію такої системи є моделювання, і в першу чергу математичне моделювання.

Під час проектування складних систем часто використовується **системний підхід**, згідно з яким система розглядається з позицій переходу від загального до часткового. Системний підхід дає змогу на всіх етапах дослідження системи та побудови її моделі врахувати всі фактори пропорційно до їх важливості. Системний підхід означає, що будь-яка система є інтегрованим цілим навіть тоді, коли вона складається з окремих підсистем, причому проектування системи починається з формулювання мети її функціонування.

Практичне застосування системного підходу до проектування технічних пристроїв та процесів різного функціонального призначення втілено в **блочно-ієрархічному підході** [28-31], який передбачає використання принципу ієрархічності для структурування представлень про об'єкти за ступенем деталізації описів та принцип декомпозиції (блочності, модульності) для розбиття представлень кожного рівня на складові (довершені блоки) з можливістю їх поблокового аналізу та синтезу.

На верхньому рівні, для прикладу проектування МЕМС, позначимо як S^1_{MEMS}, де одиниця означає перший рівень деталізації. Оскільки МЕМС є

складною системою і її можна розбити на блоки нижчого рівня для зручності розв'язання задач синтезу та аналізу, то введемо рівень 2, який міститиме n блоків. Відповідно, кожний блок другого рівня позначимо через $S_{MEMS}^{2,j}$, де j – номер блока другого рівня розбиття ($j=1,2,..n$). У цьому випадку МЕМС можна описати, як $S_{MEMS}^{1} = \bigcup_{j=1}^{n} S_{MEMS}^{2,j}$.

Оскільки блоки другого рівня також є складними об'єктами і їх можна розглядати як системи відносно до блоків третього рівня та доцільно з технічного погляду, розбити на простіші блоки, то кожний блок (система відносно блоків третього рівня) другого рівня можна описати як об'єднання блоків третього рівня $S_{MEMS}^{2,j} = \bigcup_{l=1}^{K_j} S_{MEMS}^{3,l}$, де K_j – кількість блоків третього рівня в j-му блоці (системі) другого рівня, l – номер блока третього рівня розбиття ($l=1,2,...,K_j$).

При технічній доцільності наявності блоків четвертого рівня кожний блок третього рівня можна описати так $S_{MEMS}^{3,l} = \bigcup_{z=1}^{Z_j} S_{MEMS}^{4,z}$, де Z_l – кількість блоків четвертого рівня в l-му блоці (системі) третього рівня, z – номер блока четвертого рівня розбиття ($z=1,2,...,Z_l$).

Так процес продовжується доти, поки блоки m-го рівня вже недоцільно з певних міркувань піддавати декомпозиції на простіші. Блоки найнижчого рівня, як правило, називають базовими елементами.

Припустивши, що процес проектування потребує чотири рівні ієрархії (деталізації), її можна описати за допомогою наступного виразу

$$S_{MEMS}^{1} = \bigcup_{j=1}^{n} S_{MEMS}^{2,j} \bigcup_{l=1}^{K_j} S_{MEMS}^{3,l} \bigcup_{z=1}^{Z_j} S_{MEMS}^{4,z} .$$

Поділ на блоки виконується, як правило, за функціональною ознакою та технічною доцільністю.

Отже, згідно з блочно-ієрархічним підходом в процесі автоматизованого проектування складних систем, проектування здійснюється в декілька етапів або, як прийнято казати, на різних рівнях. Кожний рівень проектування визначається ступенем деталізації опису системи, причому методика проектування безпосередньо залежить від рівня. На кожному рівні проектування поняття системи, елементів системи, закону функціонування елементів і т.д., можуть мати різний зміст. Те, що трактувалося як система на одному рівні, може розглядатися як елемент системи на іншому рівні. На кожному з рівнів проектування розв'язують множину задач моделювання, при цьому, використовується специфічний математичний апарат. Тобто задачі проектування розв'язують через моделювання. Відповідно, вживається вираз: задача моделювання для системного рівня, рівень компонентного моделювання тощо.

Для прикладу під час проектування інтегральних схем (ІС), розрізняють наступні рівні:

1) **рівень структурного (системного)** проектування (моделювання) з використанням імітаційних моделей та застосуванням спеціалізованих мов моделювання, теорій множин, алгоритмів, графів, систем масового обслуговування, теорії мереж Петрі тощо;

2) **рівень функціонально-логічного** проектування (моделювання) з використанням моделей у вигляді логічних рівнянь та застосування апарату багатозначної алгебри логіки;

3) **схемотехнічний рівень,** на цьому рівні при побудові моделей використовують звичайні диференціальні рівняння та їх системи, VHDL-AMS – мову для опису цифрових, аналогових та гетерогенних елементів системи;

4) **компонентний рівень**, на якому моделі систем та їх елементів мають вигляд систем алгебраїчних, диференціальних, диференціальних рівнянь в частинних похідних та інтегро-диференціальних рівнянь, які досліджуються за допомогою методів функціонального аналізу, теорії диференціальних рівнянь, аналітичних та чисельних методів.

Для випадку застосування блочно-ієрархічного підходу до проектування механічних систем, найчастіше використовують такі назви рівнів: рівень проектування комплектів, агрегатів, зібраних одиниць та деталей.

Контрольні запитання до розділу 6

1. Які основні складові математичного забезпечення САПР Ви знаєте?
2. Які групи параметрів об’єкта проектування Ви знаєте?
3. Що таке модель та математична модель?
5. Що таке метод?
6. Що таке алгоритм?
7. Які види моделей Ви знаєте?
8. В чому основний зміст блочно-ієрархічного підходу?
9. Які рівні включає автоматизоване апроектування інтегральних схем?
10. Що називається системою?
11. Що Ви розумієте під елементом системи?
12. Які методи моделювання Ви знаєте?
13. Які переваги аналітичних методів?
14. Які переваги чисельних методів?
15. Які види моделей Ви знаєте?
16. Що таке фазова змінна?
17. Що Ви розумієте під станом системи?
18. Що ви розумієте під динамікою системи?
19. Який математичний аппарат використовується на системному рівні?

20. Який математичний аппарат використовується на схемотехнічному рівні?

21. Який математичний аппарат використовується на компонентному рівні?

22. Яка особливість методів натурного моделювання?

23. Що Ви розумієте під інформаційною моделлю?

24. Що таке структурна схема?

25. Які дві основні складові включає моделювання?

РОЗДІЛ 7. ЗАГАЛЬНА ХАРАКТЕРИСТИКА ПРОЦЕСУ МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

7.1. Види опису математичних моделей

В загальному випадку під математичною моделлю (ММ) розуміють будь-який математичний опис, що відображає з потрібною точністю структуру та/або процес функціонування деякої реальної системи в реальних умовах. Визначення **математичної моделі**, що враховує специфіку та особливості автоматизованого проектування є наступне: **математична модель** це набір математичних знаків для встановлення зв'язку між вихідними та вхідними і внутрішніми параметрами об'єкта проектування в заданих умовах. Отже, математичний опис полягає у встановленні зв'язків між параметрами процесу та виявлені його граничних і початкових умов, а також у формалізації цього процесу у вигляді системи математичних співвідношень.

У найпростішому випадку математичний опис (математична модель) об'єкта проектування має вигляд явної функції, що виражає змінну величину (вихідні параметри) через інші змінні, які називаються аргументами (вхідні, внутрішні та параметри зовнішнього середовища), і в загальному випадку може бути записана наступним чином:

$$y = f(x_1, x_2, ..., x_n), \tag{7.1}$$

де y – вихідний параметр моделі, f – функція перетворення, $x_1, x_2, ...x_n$ – множина вхідних, внутрішніх та параметрів зовнішнього середовища.

Прикладом такої математичної моделі може бути вираз для визначення величини електростатичної сили F_{el}, що виникає між пластинами плоского конденсатора:

$$F_{el} = C\frac{U^2}{2d},\ C = \varepsilon_o \varepsilon \frac{S}{d}, \tag{7.2}$$

де C – електрична ємність; S – площа пластин; U – прикладена напруга; d – відстань між пластинами конденсатора; ε, ε_o – діелектрична

проникність середовища між пластинами конденсатора та діелектрична проникність у вакуумі.

В більш складному випадку ММ має вигляд рівняння такого виду:

$$F(y, x_1, x_2, ...x_n) = 0, \quad (7.3)$$

яке виражає залежність (7.1) в неявній формі. Наприклад це моделі, що описують залежності між її параметрами з допомогою трансцендентних рівнянь, систем лінійних алгебричних рівнянь (СЛАР) тощо.

У складнішому випадку співвідношення математичної моделі можуть бути записані у формі звичайних диференціальних рівнянь

$$F(y, y', y'', ...y^{(n)}, x_i, x_i', ...x_i^{(m)}, t) = 0, \quad (7.4)$$

які зв'язують незалежну змінну t, відомі функції $x_i = x_i(t)$, невідому функцію $y = y(t)$ та похідні функцій x_i, y. Прикладом такої ММ може бути диференціальне рівняння другого порядку, що описує зміщення пластини електричного конденсатора під дією зовнішньої сили з врахуванням електростатичної сили (7.2) між його пластинами

$$m\frac{d^2l}{dt^2} + b\frac{dl}{dt} + kl = F, \; F = F_a + F_{el},$$

де l – зміщення; F_a – механічна сила; m – маса пластини; F – сумарна прикладена сила; k – коефіцієнт пружності пружини; b – коефіцієнт демпферування.

В загальному випадку до цієї групи входять моделі, що описують процеси з використанням систем звичайних диференціальних рівнянь, диференціальних рівнянь n-го порядку тощо.

І, нарешті, математична модель може включати диференціальні рівняння в частинних похідних

$$F(y, x_i, t_1, t_2, ...t_k, D_k^n) = 0, \quad (7.5)$$

в яких, на відміну від звичайних диференційних рівнянь, шукана функція y залежить від декількох незалежних змінних, наприклад, температура може

залежати від просторових координат та часу. Тут через D_k^n позначено часткову похідну n-ого порядку від змінної t_k.

Прикладом таких моделей є ММ, що описує переміщення тонкої пластини, при цьому припускається, що розподіл сили по пластині є рівномірний, а її краї - жорстко защемлені:

$$\frac{\partial^4 w(x,y)}{\partial x^4} + 2\frac{\partial^4 w(x,y)}{\partial x^2 \partial y^2} + \frac{\partial^4 w(x,y)}{\partial y^4} = \frac{P}{D},\ D = \frac{Eh^3}{12(1-\nu^2)},$$

де $w(x,y)$ – прогини пластини (вертикальні переміщення пластини); x,y – осі, які формують координатну площину xOy, що рівновіддалена від основ пластини; h – товщина пластини; E – модуль пружності матеріалу пластини; ν – коефіцієнт Пуассона матеріалу пружного елемента; P – інтенсивність розподіленого на поверхні пластини навантаження.

Необхідно зауважити, що коректне формулювання вищенаведеної моделі потребує визначення ще крайових умов.

Співвідношення (7.1 – 7.5) можна узагальнити використовуючи поняття оператора. Розглянемо деяку систему стан якої, в довільний момент часу $t \in [0,T]$, описується певним набором Y- величин, які називаються **характеристиками стану**. Характеристики стану залежать як від власних параметрів системи, так і різних зовнішніх впливів з боку оточуючого середовища. Опис цієї залежності – це і є суть математичної моделі цієї системи. В загальному випадку, характеристики стану, власні параметри та зовнішні фактори є функціями одного і того ж або різних аргументів. Правила перетворення однієї функції в іншу називають **оператором**. Тоді математична модель, в найбільш загальному випадку, має наступний операторний вигляд:

$$Y(t) = L\{X(s),t\}, \tag{7.6}$$

де через L позначено оператор, а під $X(s)$ розуміється набір величин, які тим чи іншим способом впливають на характеристики стану. Частковим випадком такого оператора є **функціонал**, який задає правила перетворення функції в скалярну величину.

Наведене вище тлумачення оператора як правила перетворення функцій можна узагальнити наступним чином. Нехай задано дві множини функцій U та F і нехай функція u є елементом множини U, функція f - множини F, тобто $u \in U$, $f \in F$. Тоді в операторному рівнянні

$$Lu = f, \tag{7.7}$$

оператор L задає відповідність між елементами множин U і F, а розв'язати операторне рівняння (7.7) означає знайти $u \in U$ при заданих $f \in F$ та вигляд оператора L.

Для побудови математичних моделей можуть використовуватися як універсальні фундаментальні закони природи (закони збереження маси, енергії, другий закон Ньютона), так і феноменологічні закони (закон Гука, закон Фур'є), тобто достатньо добре емпірично обґрунтовані закони з обмеженою областю дії (яка також встановлена емпірично). Математичний опис завжди є відображенням фізичної сутності деякого реального об'єкту з його характерними особливостями та обмеженнями. Математична модель концентрує у вигляді математичних співвідношень сукупність наших знань, уявлень та гіпотез про відповідний об'єкт дослідження. Вона (математична модель) завжди описує поведінку реальної системи лише наближено, оскільки наші знання не є абсолютними, а гіпотези та припущення не враховують усі фактори. Тому, математичні моделі відносяться до класу **гомоморфних** моделей (**макромодель**), під якими розуміють моделі, що відображають лише основні властивості об'єкту дослідження, причому між гомоморфною моделлю та оригіналом відсутнє повне, поелементне відображення. На відміну від гомоморфних, в **ізоморфних** моделях спостерігається повна поелементна відповідність між моделлю та оригіналом (**повна модель**).

7.2. Класифікація математичних моделей

В залежності від специфіки зв'язку між характеристиками стану та вхідними даними розрізняють **детерміновані** та **стохастичні** математичні

моделі. В **детермінованих моделях** у кожний фіксований момент часу характеристики стану системи (об’єкта моделювання чи проектування) однозначно визначаються через вхідні дані. Детерміновані моделі будують на основі фундаментальних теоретичних законів, таких як закони збереження енергії, маси, закони термодинаміки, кінетики тощо. Усі величини, які входять до складу детермінованих моделей задають у вигляді конкретних чисел, векторів та функцій. Якщо хоча б один параметр системи (об’єкта моделювання чи проектування) приймає випадкові значення, то таку побудовану математичну модель будемо називати **стохастичною**. У цьому випадку під однозначністю визначення характеристик стану системи розуміють однозначність визначення ймовірністного розподілу цих характеристик за заданими розподілами ймовірностей вхідних даних. В принципі будь-якій реальній системі притаманні в тій чи іншій мірі випадкові флуктуації. Тому, якщо під час моделювання такої системи впливом цих випадкових факторів не можна знехтувати, то доводиться будувати і досліджувати стохастичну модель. У протилежному випадку достатньо обмежитися детермінованою моделлю. Виокремлення детермінованих моделей в окремий клас можна пояснити широким використанням та різноманітністю математичних методів їх дослідження.

В більш загальному випадку можемо стверджувати, що в **детермінованих** математичних моделях інженер чи науковець оперує лише математичними очікуваннями параметрів об’єкта моделювання, а в **стохастичних** – випадковими розподілами цих параметрів (чи хоча б випадковим розподілом одного з цих параметрів ММ).

Розрізняють також **лінійні** та **нелінійні**, розподілені та зосереджені за просторовими координатами, **неперервні** та **дискретні**, **стаціонарні** та **нестаціонарні** математичні моделі.

Математична модель називається **лінійною**, якщо для оператора моделі виконується принцип суперпозиції. У протилежному випадку модель називається **нелінійною**. Нелінійні моделі, в свою чергу, можна поділити на

алгебричні та **трансцендентні**. У рівняннях **нелінійної алгебраїчної моделі** над змінними проводяться лише звичайні арифметичні операції та операції піднесення до степеня з раціональним показником, а в рівняння **нелінійної трансцендентної моделі** можуть входити інші функції над змінними (тригонометричні, логарифмічні і т.д.). Більшість реальних систем є нелінійними, хоча на практиці, найчастіше використовуються лінійні моделі, які з задовільною точністю описують поведінку таких систем і є більш доступними для дослідження.

В **нестаціонарних** моделях вихідні параметри (рішення задачі) змінюються з часом, в іншому випадку – вихідні параметрі не залежать від змінної часу. **Неперевні** моделі дають змогу визначити зміну вихідних параметрів в будь-який момент часу (для прикладу, модель, яка дає змогу визначити вихідні параметри аналогового електричний сигналу). **Дискретні** моделі оперують дискретною інформацією, тобто даними, отриманими в певні відлікі часу (наприклад, моделі, що оперують з цифровою формою представлення електричного сигналу).

Важливий клас математичних моделей складають так звані **подібні** моделі, які забезпечують перехід до оригіналу на основі теорії подібності.

Основні ознаки класифікації та типи математичних моделей, які використовуються в системах автоматизованого проектування (САПР) наведено в таблиці 7.1.

Таблиця 7.1

Ознака класифікації	**Математичні моделі**
Характер відображених властивостей об’єкта	Структурні, функціональні
Приналежність до ієрархічного рівня	Мікрорівня, макрорівня, метарівня
Степінь деталізації опису в середині кожного рівня	Повні, макромоделі
Спосіб представлення властивостей об’єкта	Аналітичні, алгоритмічні, схемні, імітаційні
Спосіб отримання моделі	Теоретичні, емпіричні

За характером відображених властивостей об'єкта ММ поділяються на **структурні** та **функціональні**.

Структурні ММ призначені для відображення структурних властивостей об'єкта. Структурні математичні моделі поділяються на **топологічні** й **геометричні**.

Функціональними називають такі математичні моделі, які дають змогу визначити вихідні параметри об’єкта моделювання на основі вхідних та зовнішніх.

Топологічні ММ відображають склад і взаємозв’язки між елементами об'єкта. Найчастіше їх застосовують для опису об'єктів, які складаються з великого числа елементів, при розв’язанні задач прив'язки конструктивних елементів до певних просторових позицій (наприклад, задачі компонування обладнання, розміщення деталей, трасування з'єднань) чи до відносних моментів часу (наприклад, при розробці розкладів руху громадського транспорту, технологічних процесів). Топологічні моделі можуть мати форму графів, таблиць (матриць), списків тощо.

Геометричні ММ відображають геометричні властивості об'єктів, у них додатково до інформації про взаємне розташування елементів містяться дані про форму деталей. Геометричні ММ можуть виражатися сукупністю рівнянь ліній і поверхонь; алгебричних співвідношень, які описують області, що складають тіло об'єкта; графами і списками, що відображають конструкцію з типових конструктивних елементів, тощо. Геометричні ММ застосовують при розв’язанні задач конструювання в машинобудуванні, приладобудуванні, радіоелектроніці, для оформлення конструкторської документації, при задані вихідних даних на розробку технологічних процесів виготовлення деталей.

Використання принципів блочно-ієрархічного підходу до проектування приводить до появи ієрархії математичних моделей проектованих об'єктів. Кількість ієрархічних рівнів, при моделюванні, визначається складністю проектованих об'єктів, технологіями, які використовують в процесі проектування і можливістю засобів проектування. Однак для більшості

прикладних областей можна віднести наявні ієрархічні рівні до одного з трьох узагальнених рівнів, названих надалі **мікро-**, **макро-** і **метарівнями**.

В залежності від місця в ієрархії математичні моделі поділяються на ММ, які відносяться до мікро-, макро- і метарівнів.

Особливістю **ММ на мікрорівні** є відображення фізичних процесів, що протікають в об'єкті та неперервні в просторі і часі. Типові ММ на мікрорівні включають диференціальні рівняння в часткових похідних (ДРЧП). В них незалежними змінними є просторові координати та час. З допомогою цих рівнянь розраховують поля механічних напруг і деформацій, електричних потенціалів, тисків, температур тощо. Можливості застосування ММ у виді ДРЧП обмежені окремими деталями, спроби аналізувати з їхньою допомогою процеси в багатокомпонентних середовищах, складальних одиницях, електронних схемах не можуть бути успішними через надмірний ріст витрат машинного часу і пам'яті персонального комп’ютера.

На **макрорівні** використовують дещо грубішу дискретизацію простору по функціональному признаку, що приводить до представлення ММ на цьому рівні у виді систем звичайних диференціальних рівнянь (ЗДР). В цих рівняннях незалежною змінною є час t, а вектор залежних змінних складає фазові змінні, які характеризують стан елементів дискретизованного простору. Такими змінними є сили і швидкості в механічних системах, напруги і сили струму в електричних системах, тиски та витрати в гідравлічних і пневматичних системах тощо. Системи ЗДР є універсальними моделями на макрорівні, придатними для аналізу як динамічних, так і статичних режимів роботи об'єктів моделювання. Моделі для стаціонарних режимів можна також представити у виді систем алгебричних рівнянь. Порядок системи рівнянь залежить від числа виділених елементів об'єкта. Якщо порядок системи наближається до 100, то оперувати такою моделлю стає важко і тому необхідно переходити до представлення моделі на метарівні.

Метарівень характеризується великою розмаїтістю типів ММ. Для багатьох об'єктів ММ на метарівні, як і раніше, представляють системами ЗДР.

Однак, тому що в моделях не описуються внутрішні для елементів фазові змінні, а фігурують лише фазові змінні, які відносяться до взаємних зв'язків між елементами. Укрупнення елементів на метарівні означає одержання ММ прийнятної розмірності для значно більш складних об'єктів, ніж на макрорівні.

В ряді предметних областей вдається використовувати специфічні особливості функціонування об'єктів для спрощення ММ. Прикладом є електронні пристрої цифрової автоматики, в яких можливо застосовувати дискретне представлення таких фазових змінних, як напруги та струми. В результаті ММ стає системою логічних рівнянь, що описують процеси перетворення сигналів. Такі логічні моделі істотно більш економічні, ніж моделі електричні, що описують зміни напруг і сил струмів як функцій часу. Важливий клас ММ на метарівні складають моделі на основі систем масового обслуговування, що використовуються для опису процесів функціонування інформаційних і обчислювальних систем, виробничих ділянок, ліній і цехів.

За ступенем деталізації опису об’єктів моделювання в межах кожного ієрархічного рівня виділяють **повні ММ** і **макромоделі**.

Повна ММ – модель, в якій фігурують фазові змінні, що характеризують стани всіх наявних між елементних зв'язків (тобто стану всіх елементів проектованого об'єкта).

Макромодель – ММ, у якій відображаються стани значно меншого числа міжелементних зв'язків, що відповідає опису об'єкта при укрупненому виділенні елементів.

За формою представлення математичні моделі поділяють на **інваріантні**, **аналітичні** й **алгоритмічні** та **схемні** [30].

Інваріантна форма запису відношень моделі передбачає використання традиційної мови математики не залежно від методу розв’язання задачі.

Аналітичні ММ являють собою явні вирази вихідних параметрів як функцій вхідних, внутрішніх та зовнішніх параметрів об'єкта моделювання. Такі ММ характеризуються високою економічністю, однак одержання такої форми вдається лише в окремих часткових випадках, як правило, при прийнятті

певних спрощень і обмежень, що знижує точність і звужує область адекватності моделі. Приклад таких моделей зображено у формі виразів (7.1 та 7.2).

Схемна форма представлення ММ передбачає використанням деякої графічної мови, а саме: графів, еквівалентних схем, діаграм тощо. Прикладами таких моделей можуть бути еквівалентна схема транзистора чи електромеханічного актюатора, модель у формі мережі Петрі електромеханічної системи, граф фрагмента електричної схеми та ін. Відповідні моделі, як правило, використовують на системному та функціонально-логічному рівнях автоматизованого проектування і дещо рідше на схемотехнічному.

Алгоритмічні ММ виражають зв'язки вихідних параметрів з вхідними, внутрішніми та зовнішніми параметрами у формі алгоритму. Типовою алгоритмічною математичною моделлю є система рівнянь доповнена алгоритмом чисельного методу розв’язання й алгоритмом обчислення вектора вихідних параметрів як функціоналів розв’язання системи рівнянь. Прикладами таких моделей можуть бути вирази (7.4) і (7.5), якщо додати початкові та краєві умови з алгоритмами розв’язання сформульованих задач. В більшості випадків, алгоритмічні моделі використовують на схемотехнічному та компонентному рівнях автоматизованого проектування, хоча можливе їх використання і на вищих рівнях абстракції.

Особливим підвидом алгоритмічних математичних моделей **є імітаційні**. Імітаційна – алгоритмічна модель, яка відображає поведінку досліджуваного об'єкта в часі при задані зовнішніх впливів на об'єкт. Прикладами імітаційних ММ можуть бути моделі динамічних об'єктів у виді систем ЗДР і моделі на основі систем масового обслуговування, задані в алгоритмічній формі.

При одержанні ММ використовують неформальні і формальні методи. Неформальні методи застосовують на різних ієрархічних рівнях для одержання ММ елементів. Ці методи включають вивчення закономірностей процесів і явищ, зв'язаних з об’єктом моделювання, виділення істотних факторів, прийняття різного роду допущень і їхнє обґрунтування, математичну

інтерпретацію наявних процесів і т.п. Для виконання цих операцій, в загальному випадку, відсутні формальні методи, в той же час, від результату цих операцій істотно залежать показники ефективності ММ - ступінь універсальності, точність та економічність. Тому, побудова ММ елементів, як правило, здійснюється кваліфікованими фахівцями, які одержали підготовку як у відповідній предметній області, так і в питаннях математичного моделювання на електронно-обчислювальній машині.

Застосування неформальних методів можливо для синтезу теоретичних і емпіричних ММ. Теоретичні ММ створюються в результаті дослідження процесів і їхніх закономірностей, властивих розглянутому класу об'єктів і явищ. Емпіричні ММ - в результаті вивчення зовнішніх проявів властивостей об'єкта за допомогою вимірів фазових змінних на зовнішніх входах і виходах та обробки результатів вимірів.

Рішення задач моделювання елементів полегшується завдяки тому, що для побудови більшості технічних об'єктів використовуються типові елементи (кількість типів порівняно невелика). Тому розробка ММ елементів виконується порівняно рідко. Одноразово побудовані ММ елементів надалі багаторазово застосовують при розробці різноманітних систем з цих елементів.

Формальні методи застосовують для одержання ММ систем при відомих математичних моделях елементів.

Таким чином, у програмах автоматизованого синтезу та аналізу, які використовують в САПР, одержання ММ проектованого об'єкта забезпечується реалізацією ММ елементів і методів формування ММ систем.

7.3 Вимоги до математичних моделей

Найважливішою вимогою до математичної моделі є вимога її **адекватності** (відповідності) об’єкту-оригіналу відносно вибраної системи

його характеристик. Під цим, як правило розуміють: 1) правильний якісний опис об'єкта за вибраними характеристиками; 2) правильний кількісний опис об'єкта за вибраними характеристиками з необхідною точністю в розумних межах. Отже, під **адекватністю ММ** будемо розуміти можливість відображення заданих властивостей моделі з заданою точністю. **Точність** визначається як степінь співпадіння вихідних параметрів моделі та об'єкту-оригінала.

Оберненою величиною до точності моделі є її похибка. Для прикладу, відносна похибка моделі i-го вихідного вихідного параметра визначається з виразу [50]

$$\varepsilon_i = \left| \frac{\left(y_{model,i} - y_{opuginal,i} \right)}{y_{opuginal,i}} \right| * 100\%,$$

де ε_i – похибка моделі i-го вихідного вихідного параметра, $y_{model,i}$ – значення i-го вихідного вихідного параметра порахованого з використанням побудованої моделі, а $y_{opuginal,i}$ – значення i-го вихідного вихідного параметра порахованого з використанням об'єкту-оригіналу.

При визначенні похибки математичної моделі ε_{model} використовується одна з норм вектора $\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_n)$, а саме: $\varepsilon_{model} = \max|\varepsilon_i|$, де $i = \overline{1,n}$, або $\varepsilon_{model} = \sqrt{\varepsilon_1^2 + \varepsilon_2^2 + ... + \varepsilon_n^2}$.

Розглянемо, для прикладу, побудовану математичну модель, яка дає змогу визначити чотири вихідних параметрів деякого об'єкта проектування. Значення порахованих вихідних параметрів з використанням розробленої ММ наступні (в безрозмірному вигляді): 10.5, 23.4, 100.1 та 56.7, а значення отримані з допомогою експерименту: 9.3, 25, 112.4 і 49.2.

Визначимо величини відносних похибок для кожного вихідного параметра і загальну похибку побудованої ММ:

$$\varepsilon_1 = \left|\frac{(10.5-9.3)}{9.3}\right| * 100\% = 12.9\%,\ \varepsilon_2 = \left|\frac{(23.4-25)}{25}\right| * 100\% = 6.4\%,$$

$$\varepsilon_3 = \left|\frac{(100.1-112.4)}{112.4}\right| * 100\% = 10.9\%,$$

$$\varepsilon_4 = \left|\frac{(56.7-49.2)}{49.2}\right| * 100\% = 15.2\%,$$

$$\varepsilon_{model} = \sqrt{12.9^2 + 6.4^2 + 10.9^2 + 15.2^2} = \sqrt{166.41 + 40.96 + 118.81 + 231.04} = \sqrt{557.22} = 23.61\%.$$

Наведений приклад показує, що похибка моделі не може бути меншою за найбільше значення похибки одного з вихідних параметрів математичної моделі.

Необхідно зауважити, що точність моделі залежить від умов функціонування об'єкту-оригіналу в просторі зміни вхідних параметрів. В цьому випадку зручніше оперувати поняттям **області адекватності** моделі, що визначає область в просторі зміни вихідних параметрів, де виконується умова $\varepsilon_{model} < \varepsilon_{граничне}$ ($\varepsilon_{граничне}$– наперед задане граничне значення похибки ММ).

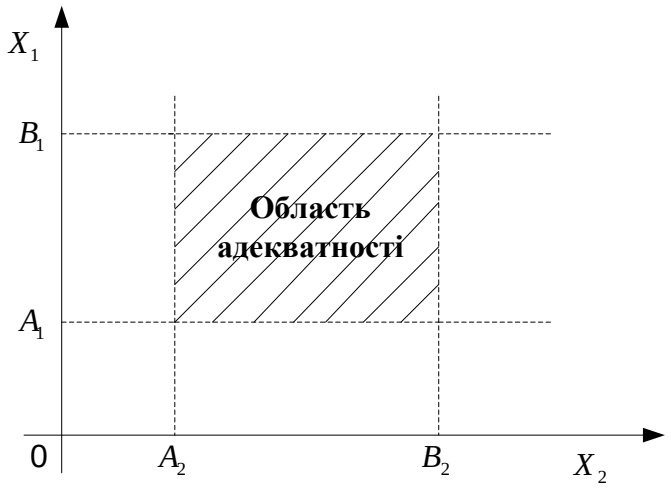


Рисунок 7.2 – Приклад області адекватності моделі

Слід підкреслити відносний характер наведеного поняття адекватності, тобто прив'язку адекватності моделі до характеристик об'єкту дослідження, які прийнято за основні в рамках цього дослідження. Так, наприклад, якщо

вивчається реакція об’єкта на зовнішні збурення того чи іншого класу, то модель, адекватна відносно одного класу збурень, може виявитися неадекватною відносно іншого класу збурень. Іншим характерним прикладом може бути класичне рівняння теплопровідності, отримане на основі закону Фур’є. Це рівняння добре описує еволюцію температури, тобто є адекватним у кількісному відношенні. Крім того, з нього випливає ряд наслідків якісного характеру, які також правильно описують реальний процес поширення тепла: збереження кількості тепла, неможливість концентрації та осциляції температури і т.д. Але з іншої сторони, з цього рівняння випливає також фізично абсурдний висновок про нескінченність швидкості поширення тепла. Таким чином, якщо суттєвою характеристикою процесу вважати швидкість поширення тепла (хоча потреба в цьому виникає досить рідко), то класичне рівняння теплопровідності, як модель реального процесу поширення тепла, стає неадекватним не тільки у кількісному, а й у якісному відношеннях. Для того, щоб отримати модель адекватну і за швидкістю поширення тепла, необхідно уточнити закон Фур’є шляхом врахування інерційності молекул.

Упущення того факту, що адекватність математичної моделі завжди відносна і має свої рамки застосування може привести (і, на жаль, не раз приводило) до спроб нав’язати реальному об’єкту властивості його моделі, наприклад, серйозному сприйнятті твердження, що швидкість поширення тепла є дійсно нескінченною. На жаль, у більш складних випадках, неадекватність моделі виявити набагато складніше, тому застосування неадекватної моделі може привести до того, що можна або не виявити, або занадто спотворити те що насправді є, натомість вивчати те, що нам не потрібно і навіть те, чого взагалі не існує. Як правило у таких випадках перевірка адекватності здійснюється на раціональному рівні, іншими словами “на хлопський розум”, що, безумовно, вступає у конфлікт з чисто дедуктивним (логічним) рівнем і є предметом гострої критики так званих “чистих” математиків. Перевірка адекватності моделі може слугувати також раціональним обґрунтуванням законності застосування прийнятих гіпотез та припущень.

До певної міри протилежною до вимоги адекватності математичної моделі є вимога її **достатньої простоти** відносно вибраної системи характеристик. Модель є достатньо простою, якщо сучасні засоби дослідження (фізичні, математичні, і зокрема, обчислювальні) дають можливість провести економно, в сенсі затрат праці, але з задовільною точністю кількісний або якісний аналіз вибраних характеристик. Встановлення компромісу між вимогами адекватності та простоти моделі знаходиться на рубежі науки та мистецтва, і в значній мірі залежить від досвіду дослідника. Загальна тенденція полягає в тому, що чим більш адекватною є модель, тим менш простою вона є, хоча досить часто ускладнення моделі може погіршити її адекватність. Іншими словами, спрощення моделі, як правило, знижує її адекватність, хоча є й приклади, коли при спрощенні моделі її адекватність зростає. Можливі також випадки, коли при однаковому ступені простоти вдається побудувати більш адекватну модель, і навпаки, при тій же адекватності – більш просту модель.

Суттєвий вплив на адекватність, і відповідно, на простоту моделі має проблема правильного вибору системи незалежних величин, які достатньо повно характеризують стан об’єкту, який моделюється. Такі величини називаються **характеристиками стану,** або **визначальними параметрами**. Логічно припустити, що збільшення кількості визначальних параметрів покращує адекватність моделі. Однак при занадто великій кількості цих параметрів модель може виявитися занадто складною і її дослідження буде дуже ускладнено. З іншого боку, зменшення визначальних параметрів може привести до втрати адекватності моделі.

Важливими параметрами математичних моделей окрім адекватності є **універсальність** та **економічність**. **Універсальність** визначається кількістю та складом врахованих в моделі зовнішніх і вихідних параметрів, а **економічність** характеризується затратами вичислювальних ресурсів для її реалізації. Як правило, на практиці, економічність визначається затратами машинного часу та об’ємом необхідної пам’яті (оперативної та пам’яті на зовнішніх носіях).

Досить часто, в процесі моделювання, оперують поняттям **достовірності** результатів моделювання, аналізу тощо. Отже **достовірність** – форма існування (встановлення) істини, яка обгрунтована кількісним способом (наприклад, експериментом, логічним доведенням тощо) для суб'єкта, який пізнає. Процедура оцінки достовірності результатів моделювання називається аналізом адекватності (відповідності) моделі до модельованого об'єкта, системи чи процесу.

7.4 Основні параметри методів та алгоритмів

Основними параметрами методів є **похибка** (точність), **економічність**, **універсальність**, **надійність** та ін.

При переході від математичної моделі до чисельного методу виникають похибки, які називають **похибками методу.** Вони пов'язані з тим, що будь-який чисельний метод відтворює математичну модель наближено. Найбільш типовими похибками метода є **похибка дискретизації і похибки заокруглення**.

Як правило застосування чисельного методу для заданої математичної моделі розбивається на два етапи: формулювання дискретної задачі; розробка вичислювального алгоритму, який дасть змогу знайти рішення дискретної задачі.

Наприклад, якщо початкова математична задача сформульована в вигляді системи диференціальних рівнянь, то для чисельного рішення необхідно замінити її системою досить великої кількості лінійних алгебричних рівнянь. Це відбувається в результаті заміни неперервної області незалежної змінної кінцевою множиною дискретних точок, в яких знаходять рішення. В цьому випадку говорять, що проведена дискретизація початкової математичної задачі. Найпростішим прикладом дискретизації є побудова різницевої схеми шляхом

заміни диференціальних виразів скінчено-різницевими аналогами. Зрозуміло, що рішення дискретизованої задачі відрізняється від рішення початкової задачі. Різниця відповідних рішень і називається похибкою дискретизації.

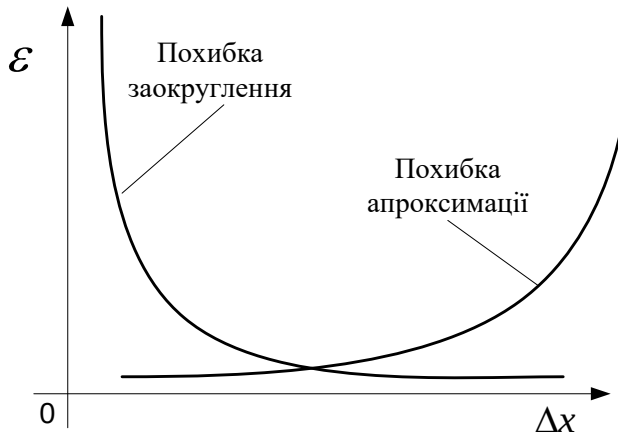


Рисунок 7.3 – Залежності похибок методу від величини кроку дискретизації

Дискретизована модель являє собою систему великої розмірності алгебраїчних рівнянь. Неможливо знайти рішення такої системи точно в явному вигляді. Тому доводиться використовувати той чи інший чисельний алгоритм рішення системи лінійних алгебраїчних рівнянь. Вхідні дані цієї системи, а саме, коефіцієнти і праві частини задаються в ПК не точно, а з заокругленням. В процесі роботи алгоритму похибки заокруглення, як правило, накопичується, і в результаті рішення, отримане на ПК, буде відрізнятися від точного рішення дискретизованої задачі. Кінцева похибка називається похибкою заокруглення (чи вичислювальною похибкою).

Таким чином, слід розрізняти похибки моделі і методу, реалізуючого дану математичну модель, і при виборі методу (алгоритму) враховувати питання похибки.

Наступним важливим параметром методів є їх **економічність**. По аналогії з параметром економічності моделей – це є затрати ресурсу персонального компютера (оперативна память та час центрального процесора).

Універсальність методів (алгоритмів) характеризується областю їх застосування. Тобто, чим більша множина задач, які можна розв'язувати з використанням даного методу, тим метод є більш універсальним.

Досить часто важко сформулювати обмеження на застосування того чи іншого методу. Відповідно, можливі ситуації, коли попередньо визначені вимоги застосування методу виконуються, однак задовільне рішення отримати неможливо. Тому, ймовірність вдалого застосування методу менша за одиницю. Отже під **надійністю** методу (алгоритму) будемо розуміни ймовірність P правильного отримання результату розв'язання задачі.

В САПР необхідно використовувати надійні методи та алгоритми. Для підвищення надійності методів, досить часто, використовують поєднання різних методів, різного роду настройку методів тощо. Все це робиться для того, щоб добитися значення параметра надійності рівного одиницю чи, хоча б, близького до неї.

Одній і тій же математичній задачі можна поставити у відповідність множину різних дискретних моделей, однак не всі із них придатні для практичної реалізації. Вичислювальні алгоритми, призначені для ПК, повинні задовольняти ряду вимог. Можна виділити дві групи вимог. Перша група пов'язана з адекватністю дискретної моделі вихідної математичної задачі, друга - з реалізацією чисельного методу на ПК.

До першої групи відносяться такі вимоги, як сходимість чисельного методу, виконання дискретних аналогів законів збереження, кількісно правильна поведінка рішення початкової задачі. Пояснимо сказане. Припустимо, що дискретною моделлю задачі є різницева схема, і при заміні диференціальних виразів скінченними різницями отримують велику кількість алгебричних рівнянь. Чим точніше ми хочемо отримати рішення, тим потрібно обрати менший крок сітки, чи параметр дискретизації (x), і тим більшу кількість рівнянь доведеться вирішувати. Говорять, що чисельний метод сходиться, якщо при необмеженому збільшенні кількості рівнянь рішення дискретизованої задачі прагне до точного рішення початкової задачі. Далі, відомо, що

диференціальні рівняння математичної фізики є наслідком законів збереження. Тому природно вимагати, щоб для різницевої схеми виконувались аналоги таких законів збереження. Різницеві схеми, які задовольняють цьому методу, називаються консервативними.

Друга група вимог до чисельних методів пов'язана з об'ємом оперативної пам'яті ПК, з можливістю отримати рішення відповідної системи рівнянь за прийнятний час, з стійкістю алгоритму. Алгоритм називається **стійким**, якщо в процесі рішення ріст вичислювальних похибок обмежений зверху, і **нестійким**, якщо похибки зростають необмежено. Існують такі алгоритми, які стійкі при виконанні граничних умов для параметрів дискретної моделі. Такі алгоритми називаються **умовно-стійкими**. Ясно, що використовувати слід стійкі чи умовно стійкі алгоритми.

7.5 Основні етапи математичного моделювання

Враховуючи вище сказане, **математичне моделювання**, у широкому значенні цього терміну, можна трактувати як процес побудови та дослідження математичної моделі з метою фіксації та вивчення певних властивостей та відношень оригіналу, а також перенесення отриманих результатів на оригінал за допомогою математичних методів. У літературі досить часто термін "математичне моделювання" використовується також і у вузькому значенні. У цьому випадку під математичним моделюванням розуміють лише або метод побудови моделі, або метод її дослідження.

В загальному випадку в процесі підготовки та проведення математичного моделювання виділяють такі етапи:

1. абстрагування: зосередження на властивостях, які є загальними для багатьох об'єктів та явищ матеріального світу та нехтування відмінностями існуючими між ними;

2. формулювання: вибір деякої множини засобів (символів, графічних образів тощо) для зображення абстрактних понять;

3. маніпуляція: правила перетворення символьного представлення як засіб передбачення результату аналогічних маніпуляцій в реальному світі;

4. аналіз: реальна інтерпритація отриманих математичних результатів з метою їх імплементації;

5. аксіоматизація: строге формулювання тих властивостей, які були виведені з реального світу і які є загальними при маніпуляціях як в матеріальному світі, так і над абстрактними символами, що представляють реальний світ.

Перший та частково другий етапи відносяться до процесу формалізації задачі, другий та третій – власне до процесу моделювання задачі, тобто до формування моделей, їх перетворення та аналізу, п’ятий – до обґрунтування методів моделювання і підготовки основи для їх подальшого розвитку. На останніх етапах робиться спроба сконцентрувати основні фактичні відомості про об’єкти, які охоплюються цими абстрактними поняттями, в декількох коротких, але потужних аксіомах і потім строго довести, що висновки, отримані в результаті маніпуляцій з цими абстрактними представленнями є справедливими і для прообразів реального світу. Ці етапи тісно взаємопов’язані між собою, і тому їх виділення є до певної мірі відносним. Так, математична модель зазвичай будується з орієнтацією на певний метод дослідження цієї моделі. З іншого боку, в процесі проведення математичного дослідження або інтерпретації розв’язку може виникнути потреба уточнити або навіть суттєво змінити математичну модель.

Процесу математичного моделювання дуже часто передує етап формалізації задачі, під яким розуміють процес переходу від опису об’єкта в термінах його конкретних властивостей до опису в термінах абстрактних змінних стану та незалежних змінних при визначеній меті дослідження і переліку обмежень. Можна виділити такі етапи процесу формалізації задачі:

1. вибір характеристик стану об’єкта, які будуть досліджуватися;

2. кожній характеристиці стану ставиться у відповідність абстрактний об’єкт, який називається змінною;

3. визначення всієї множини значень змінних;

4. введення припущень та гіпотез на основі яких визначається кількість незалежних змінних у ролі яких виступають просторові координати та/або час;

5. визначення відношень, обмежень та зв’язків між змінними.

Кінцевою метою процесу формалізації задачі є підготовка умов для формального запису математичної моделі, а сам запис моделі, який полягає у використанні засобів математичної мови для позначення виявлених на останньому етапі відношень, обмежень та зв’язків, відноситься вже до першого етапу математичного моделювання. Звичайно, часто ці два процеси - формалізація та моделювання можуть тісно переплітатися.

7.6 Поняття про обчислювальний експеримент

На сучасному етапі розвитку науки і техніки роль математичного моделювання значно зросла у зв’язку з інтенсивним застосуванням комп’ютерної техніки. Сьогодні важко уявити собі проведення фундаментальних чи прикладних наукових досліджень без поєднання математичного моделювання та комп’ютера. В науковій літературі для підкреслення важливості та ефективності такого поєднання введено навіть окремий термін – **обчислювальний експеримент**, під яким розуміють дослідження властивостей об’єкту або явища шляхом розв’язування за допомогою комп’ютерної техніки задачі, яка являє собою ММ цього явища чи об’єкта. Багаторазове проведення розрахунку моделі для різних наборів вхідних даних дає змогу дослідити роль та вплив різних факторів на протікання того чи іншого процесу або поведінку об’єкту. Обчислювальний експеримент дає змогу

правильно планувати та проводити натурний (фізичний) експеримент, а правильне використання його результатів дає змогу суттєво скоротити терміни проектно-конструкторських робіт, знизити затрати матеріалів та енергоресурсів, а також виявити нові теоретичні та технічні якості досліджуваного процесу. Успішна реалізація обчислювального експерименту обумовлена тісним взаємозв'язком трьох факторів: потужної комп'ютерної техніки, на якій здійснюється чисельний експеримент, адекватних математичних моделей досліджуваних процесів та чисельних методів їх аналізу. Наявність сучасних програмних систем автоматизації проведення математичних розрахунків таких, як MATHCAD, MATLAB, які частково або повністю реалізують багато чисельних методів, і, що найголовніше, містять програмні засоби реалізації нових та нестандартних схем чисельних методів, ще більше підвищують роль та важливість обчислювального експерименту в проведенні фундаментальних наукових досліджень та в системах автоматизації проектувальних робіт, а також значно спрощують його реалізацію. Серед чисельних методів комп'ютерної реалізації обчислювального експерименту найефективнішими є метод скінчених різниць, метод скінчених елементів та метод граничних елементів [54-60].

7.7 Алгоритм побудови математичної моделі

В загальному випадку процес побудови математичної моделі включає такі кроки:

1. **Вибір властивостей, які необхідно відобразити в моделі.** Цей вибір базується на аналізі можливих застосувань моделі й визначає степінь універсальності моделі.

2. **Збір початкової інформації про вибрані властивості об'єкта.** Джерелами відомостей можуть бути досвід та знання інженера, який розробляє

модель, науково-технічна література, перш за все довідкова, опис прототипів – відомих ММ для елементів, близьких за властивостями до дослідного об’єкта, результати дослідного вимірювання параметрів тощо.

3. **Синтез структури математичної моделі.** Структура ММ – загальний вид математичних співвідношень моделі без конкретизації числових значень параметрів. Структура ММ також може бути представлена в графічній формі, для прикладу в формі еквівалентної схеми чи графа. Синтез структури ММ – найбільш відповідальна, трудомістка і найважче формалізовувана операція.

4. **Розрахунок числових значень параметрів ММ**. Ця задача ставиться як задача оптимізації похибки моделі даної структури.

5. **Оцінка точності та адекватності ММ**. При отриманні математичної моделі кроки 2 – 5 методики можуть виконуватися багаторазово в процесі послідовних наближень до бажаного результату.

7.8 Поняття методології та технології моделювання (проектування)

Вирішення складних проблем моделювання чи проектування потребує використання цілої групи прийомів, підходів, алгоритмів і методів. Більше того, йде мова про використання групи вже встановлених методів, їх послідовність застосування до розв’язання певної задачі чи вирішення проблеми і оперують, в цьому випадку, терміном методологія. Отже, методологія – [68] встановлене коло використовуваних методів (на цей час вже опробованих при розв’язанні певної задачі чи проблеми) для виконання яких-небудь складних дій. Досить часто це слово використовують в складі інших термінів таких як “методологія проектування на системному рівні”, “методологія моделювання”, “методологія програмування”, “загальна методологія автоматизованого проектування”, яка,

згідно роботи [49], включає в себе всі аспекти процесу автоматизованого проектування від формулювання проблеми до документування результатів.

У випадку дослідження та пошуку методів, прийомів і алгоритмів та визначення послідовності їх застосування до розв'язання нової задачі чи проблеми, то говорять про стратегію моделювання (проектування). Основною характеристикою цього етапу є те, що наперед невідомо результат застосування підходів, алгоритмів і методів. Він може бути як позитивний для розробника (тобто, отримано розв'язання задачі з задаю точністю тощо), так і – негативний (не використовувати цей метод до розв'язання цієї задачі).

Під технологією моделювання (проектування) [49] будемо розуміти опробовану послідовність дій чи операцій, що дає змогу технічно виконати моделювання (проектування) заданого об'єкта. Необхідно зауважити, що технологія є опробованою стратегією, яка позбавлена елементів пошуку та невизначеності на ключових етапах процесу моделювання (проектування).

Контрольні запитання до розділу 7

1. Які види опису ММ Ви знаєте?
2. Які ММ є детермінованими?
3. Які ММ є стохастичними?
4. Які ММ є нелінійними?
5. Які ММ є дискретними?
6. Які ММ є нестаціонарними?
7. Яка ММ називається функціональною, а яка - структурною?
8. Яка різниця між мікро- та макромоделями?
9. Які форми представлення ММ Ви знаєте?
10. Які основні вимоги до ММ Ви знаєте?

11. За якою формулою визначається похибка ММ?

12. Що Ви розумієте під адекватністю ММ?

13. Які основні вимоги до методів та алгоритмів Ви знаєте?

14. Яку має залежність похибка заокруглення від величини кроку дискретизації?

15. Що Ви розумієте під універсальністю методу?

16. Що Ви розумієте під надійністю методу?

17. Що Ви розумієте під терміном “вичислювальний експеримент”?

18. Які основні кроки побудови математичної моделі Ви знаєте?

19. Яка особливість алгоритмічної форми представлення математичних моделей?

20. Яка особливість схемної форми представлення математичних моделей?

РОЗДІЛ 8. ПОБУДОВА СТРУКТУРНИХ МОДЕЛЕЙ НА ОСНОВІ ТЕОРІЇ МЕРЕЖ ПЕТРІ

8. 1 Теоретичні основи простих мереж Петрі

В процесі автоматизованого проектування КСМ, як правило, використовується блочно-ієрархічний підхід [58], який передбачає такі рівні проектування: системний; функціонально-логічний; схемотехнічний; компонентний.

На системному рівні проектування розв'язують задачі синтезу та аналізу. Для розв'язання задач аналізу, як правило, використовують моделі на основі теорії мереж Петрі та систем масового обслуговування.

Основна мета представлення КСМ у вигляді мереж Петрі та подальшого їх аналізу полягає в отриманні важливої інформації про структуру, динамічну поведінку модельованих систем та вихідних параметрів системи і її складових .

В загальному випадку мережа Петрі включає множину позицій, множину переходів та множину вхідних та вихідних дуг.

Модель на основі простої мережі Петрі описується з допомогою наступних співвідношень

$$N_{petry} = \{S, T, F, M_0\},$$

де $S = \{S_1, S_2, ..., S_g\}$ - множина позицій (стани); $T = \{t_1, t_2, ..., t_v\}$ - множина переходів; F - множина дуг, яка включає дві підмножини вхідних та вихідних дуг по відношенню до переходу; M_0 - множина, яка задає початкове маркування мережі Петрі.

Прості мережі Петрі використані для аналізу динаміки роботи КСМ.

В найпростішому випадку структура КСМ наведена на рисунку 8.1. Для аналізу динаміки даної структури застосовуємо теорію мереж Петрі. На рисунку 8.2 наведена відповідна проста мережа Петрі.

Множина позиції включає сім елементів $S = \{s_1, s_2, ..., s_7\}$, множина переходів - $T = \{t_1, t_2, ..., t_5\}$, а множина дуг 14 (7 вхідних та 7 вихідних), що можна записати в наступному вигляді:

$$F = \{< 0, t_1 s_1 >, < s_1 + s_3, t_2, s_2 >, < s_2 + s_7, t_3, s_3 + s_4 >, < s_4 + s_6, t_4, s_5 + s_7 >, < s_5, t_5, s_6 >\},$$

де: $< s_1 + s_3, t_2, s_2 >$ - означає, що перехід t_2 має дві вхідні дуги від позицій s_1 та s_3 і одну вихідну, яка направлена до позиції s_2.

Початкову розмітку наведеної мережі запишемо у наступній формі $M_0 = S_3 + S_6 + S_7$, що означає наявність по одному маркеру в позиціях S_3, S_6 та S_7. Призначення кожної позиції та переходів наведено в таблицях 8.1 та 8.2.

Рисунок 8.1 – Базова структура мікроконтролерної системи

Таблиця 8.1 – Позиції мережі Петрі

Позиція	Призначення
S_1	Очікування обробки сигналу від давача
S_2	Обробка даних МП
S_3	МП в стані очікування
S_4	Дані оброблені
S_5	Стан роботи актюатора
S_6	Актюатор готовий до роботи
S_7	Позиція дозволу вибору оброблених МП даних

Робота наведеної на рисунку 8.2 мережі Петрі починається після спрацювання переходу t_1, який означає наявність на виході давача сигналу, який має обробити МП. Дана подія асоціюється з появою маркера в S_1. У випадку, коли мікропроцесор вільний (наявність маркера в S_3), то спрацьовує

перехід t_2, що означає початок обробки даних МП. Після завершення обробки МП сигналу та дозволу виводу даних (позиції S_2 і S_7 містять маркери), то виконується подія “завершення обробки даних МП”. Цей перехід генерує маркер в позицію S_6, що означає готовність актюатора до обробки нового сигналу відбувається спрацювання переходу t_4, тобто початок обробки актюатором сигналу від МП. Перехід t_4 переводить актюатор в стан виконання і дає змогу МП вивести оброблені дані, якщо такі є (маркер в позиції S_7). Після завершення роботи актюатором відбувається перехід t_3, який генерує маркер в позицію S_6, сигналізуючи цим, що актюатор готовий до обробки наступного сигналу.

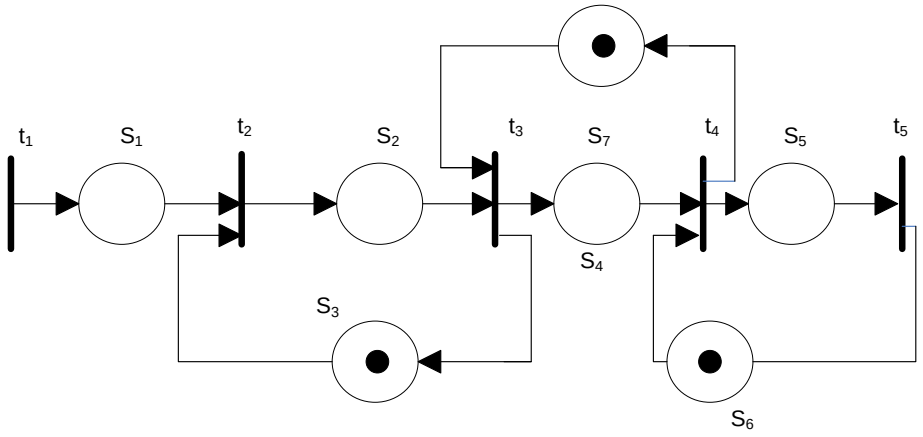


Рисунок 8.2 – Мережа Петрі для найпростішої структури мікроконтролерної системи

Таблиця 8.2 – Переходи мережі Петрі

Перехід	Призначення переходу
t_1	Активізація давача
t_2	Початок обробки даних МП
t_3	Завершення обробки даних
t_4	Початок роботи актюатора
t_5	Завершення роботи актюатора

8.2 Розширення мереж Петрі

Досить часто, в процесі моделювання, використовують розширення мереж Петрі, а саме [64 -72]:

- часові мережі Петрі;
- мережі Петрі з пріоритетами;
- інгібіторні МП;
- стохастичні МП;
- кольорові МП та інші.

Модель об'єкта для системного рівня проектування на базі часової мережі Петрі використовує певне ускладнення простої мережі і пов'язана з додаванням до кожного з переходів інформації про нижню та верхню часові межі. З допомогою даної мережі можна визначити швидкодію проектованого пристрою. В даному випадку маємо часову мережу, яку математично можна описати з допомогою наступного виразу

$$N_{time} = \{S, T, F, Eft, Lft, M_0\},$$

де Eft, Lft - функції, що ставляться у відповідність до кожного з переходів і визначають нижню (Eft) та (Lft) часові межі, які задовольняють наступні умови $Eft(t) \le Lft(t)$.

Слід зазначити, що в часовій мережі Петрі зміна одного стану на інший відбувається при завершенні деякого часу або при виконанні певного переходу мережі [81-85].

Модифікуємо мережу на рисунку 8.2 в часову мережу Петрі, додавши до кожного з переходів значення нижньої та верхньої часових меж, які позначаються d_1 та d_2 (рисунок 8.3).

Отже, якщо перехід можливий в часовій мережі в деякий момент часу τ, тобто вхідні місця переходу містять хоча б по одній фішці, то даний перехід відбудеться в деякий момент часу з інтервалу $\lfloor \tau + d_1, \tau + d_2 \rfloor$, при виконання

умови про незмінність вхідних позицій. У випадку, якщо перехід залишався без змін протягом цього інтервалу і не спрацював, формується спрацювання переходу в момент часу $\tau + d_2$.

Для оцінки мінімального часу циклу, у випадку однієї вхідної та вихідної дуг можна використати наступну формулу:

$$C = \max\left\{ \frac{T_k}{N_k} : k = 1,2,...,g \right\},$$

де $T_k = \sum_{t_i \in L_k} \tau_i$ - сума затримок переходів в сумі k; $T_k = \sum_{p_i \in L_k} M(p_i)$ - загальна кількість маркерів у позиціях в циклі k; g – кількість циклів в мережі Петрі.

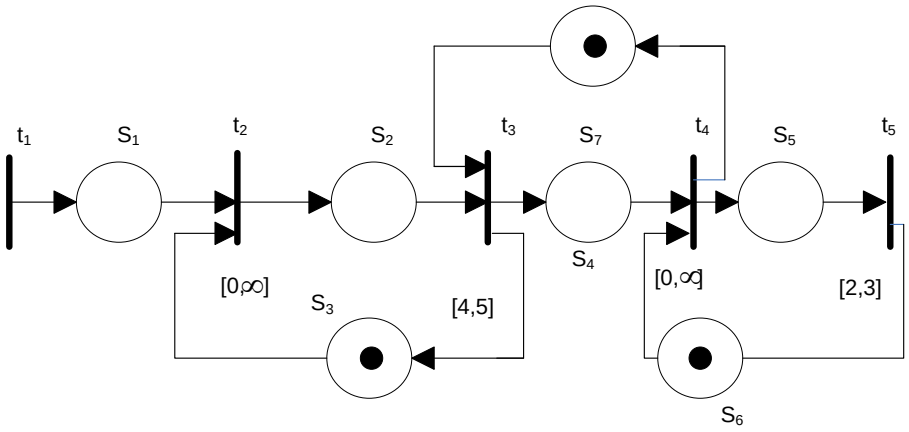


Рисунок 8.3 – Приклад часової мережі Петрі

Значно ускладнюється мережа Петрі для структур мікроконтролерних систем, які включають по декілька давачів і актюаторів та один мікропроцесор. Приклад структур з двома давачами та двома актюаторами наведено на рисунку 8.4, а відповідна мережа Петрі на рисунку 8.5.

Наведена мережа має суттєвий недолік, який полягає в тому, що при наявності маркерів в позиціях S_1, S_2 і S_3 - може спрацювати як перехід t_3, так і перехід t_4.

Рисунок 8.4 – Мікроконтролерна система з двома давачами та двома актюаторами

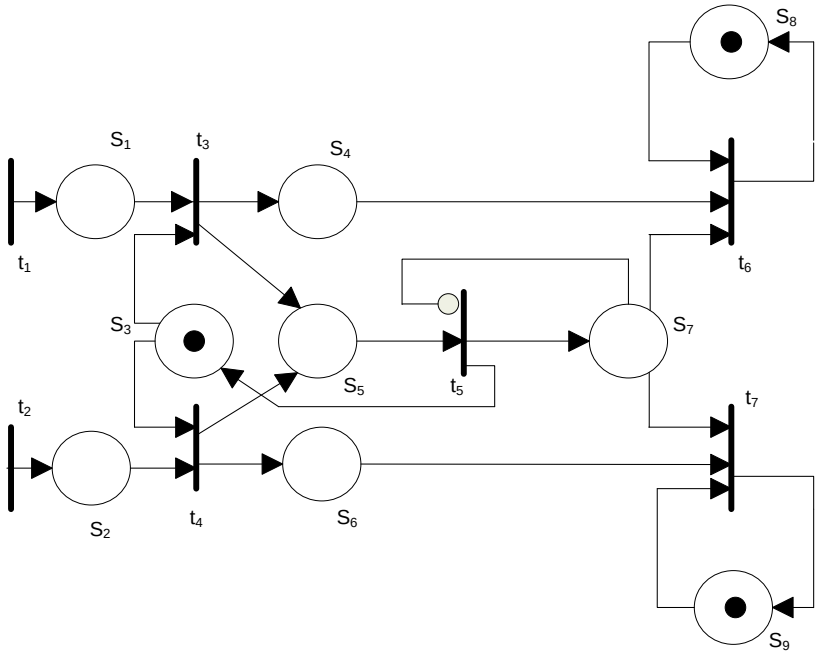


Рисунок 8.5 – Приклад мережі Петрі з двома давачами, одним МП та двома актюаторами

В реальних системах дана невизначеність, в більшості випадків, неприпустима [86, 87]. Тому дещо вдосконалимо структуру мережі шляхом додавання пріоритетів при обробці сигналів від давачів. В мережах з пріоритетами кожному з переходів надається свій пріоритет. Правило спрацювання модифікується наступним чином: якщо пріоритет даного переходу не менше будь-якого іншого можливого переходу і на його вході є хоча б по одній фішці, то він спрацьовує. Відсутність пріоритету відповідає найнижчому значенню пріоритету. Приклад відповідної мережі з пріоритетами наведено на рисунку 8.6.

Модель, яка враховує пріоритет включає множину пріоритетів для кожного з переходів, має наступний вигляд:

$$N_{priorutet} = \{S,T,F,PR,M_0\},$$

де $PR = \{\mathrm{Pr}_1, \mathrm{Pr}_2, ..., \mathrm{Pr}_v\}$ - множина пріоритетів, а Pr_1 - величина пріоритету для першого переходу.

В даному випадку найвищий пріоритет має перехід t_3, а усі інші – 0. Це означає, що обробка сигналу від давача 1 буде мати вищий пріоритет і при наявності маркерів у позиціях S_1, S_2 і S_3 - спрацює перехід t_3, а не t_4.

Модель, яка враховує час та пріоритети має наступний вигляд:

$$N_{priorutet\_time} = \{S,T,F,Eft,Lft,PR,M_0\}.$$

Приклад відповідної мережі наведено на рисунку 8.7.

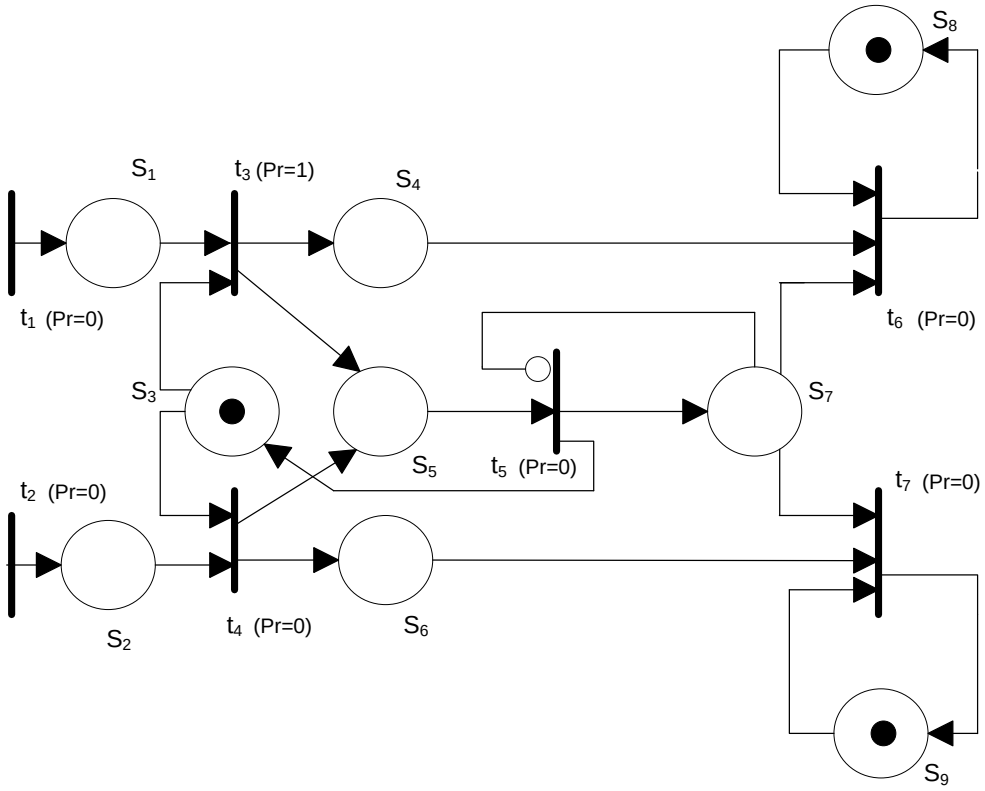


Рисунок 8.6 – Мережа Петрі з пріоритетами

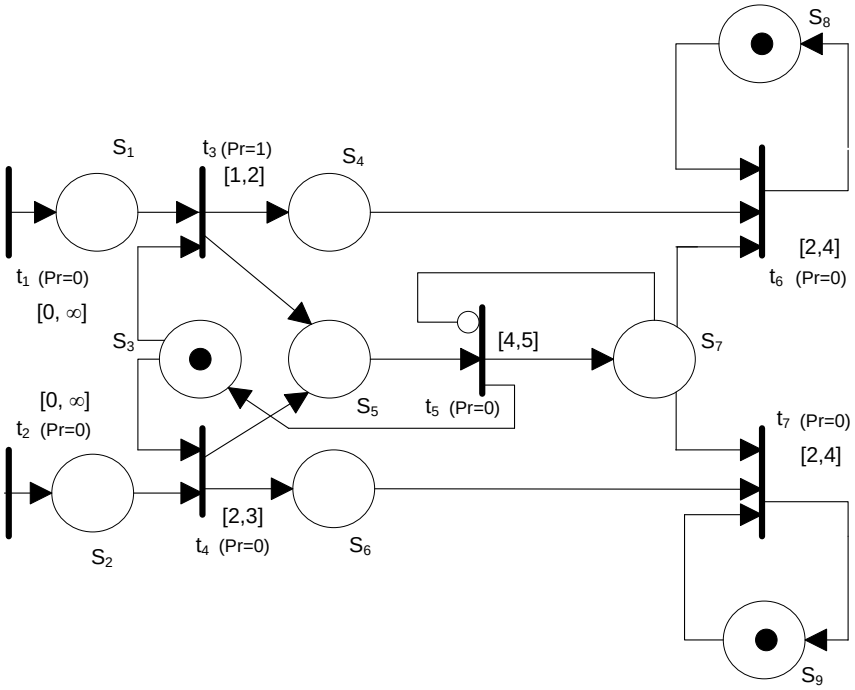


Рисунок 8.7 – Часова мережа Петрі з пріоритетами

Для розширення можливостей моделювання запропоновано використовувати моделі на основі кольорових мереж Петрі, які передбачають врахування змінних різного типу та умов спрацювання переходів. Математична модель на основі кольорової мережі Петрі має наступний вигляд [90]

$$N_{colour} = \{S, T, F, M_0, Type, Type\_S, Type\_F, Condition\},$$

де Type- множина типів; Type_S - множина, яка відображає доступну множину типів у позиціях мережі; Type_F - множина типів маркерів, що збуджують перехід та типи маркерів, які будуть згенеровані переходом; Condition - множина умов збудження переходів.

Вдосконалено структурну модель на основі інгібіторних мереж Петрі, яка грунтується на простій мережі Петрі й описується за допомогою таких співвідношень:

$$N_{petry} = \{P, T, F, M_0\}, \quad (8.1)$$

де $P = \{P_1, P_2, ..., P_g\}$ – множина позицій (станів); $T = \{t_1, t_2, ..., t_v\}$ – множина переходів; $F = \{F_{in}, F_{out}, F_{not}\}$ – множина дуг, яка включає три підмножини; $F_{in} = \{F_{in,1}, F_{in,2}, ..., F_{in,l}\}$ – вхідних; $F_{out} = \{F_{out,1}, F_{out,2}, ..., F_{out,m}\}$ –

вихідних; $F_{not} = \{F_{not,1}, F_{not,2}, ..., F_{not,k}\}$ – інгібіторних дуг по відношенню до кожного переходу; M_0 – множина, яка задає початкове маркування мережі Петрі; g,v – кількість позицій та переходів; $l + m + k = n$ – сумарна кількість дуг [91-93].

Приклад структури МЕМС та інгібіторна мережа Петрі наведені на рисунку 8.8 та риснку 8.9.

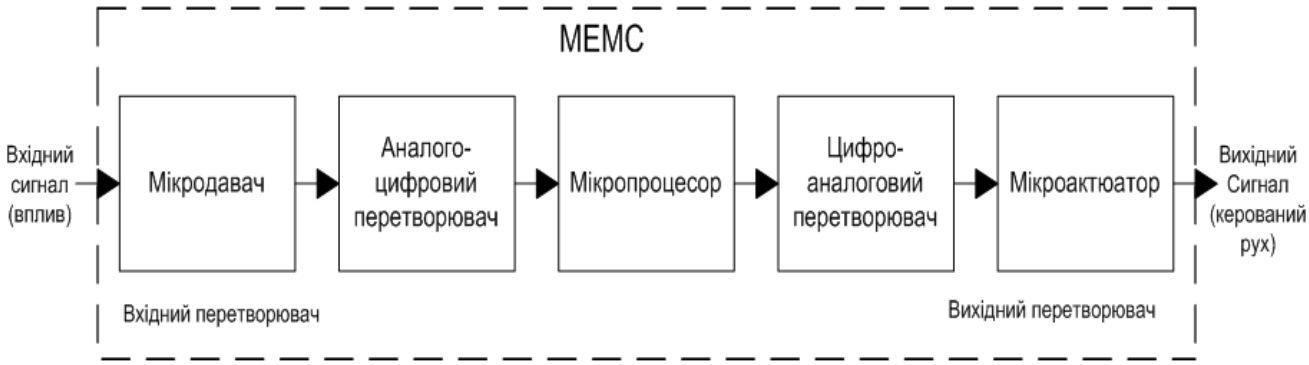


Рисунок 8.8 – Структура МЕМС

Проведені дослідження показали, що модель на основі інгібіторних мереж Петрі дає змогу підвищити параметр економічності вдосконаленої моделі шляхом зменшення об’єму обчислень на 10 – 15%.

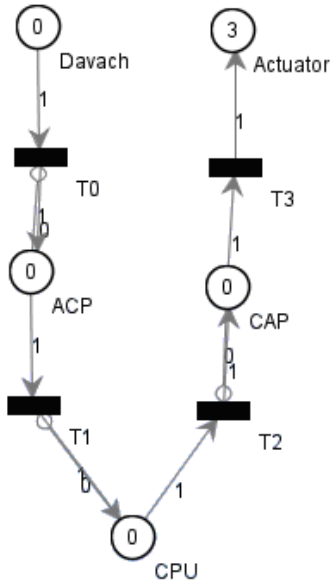


Рисунок 8.9 – Приклад інгібіторної мережі Петрі

Модель, яка ґрунтується на теорії стохастичних мереж Петрі, що є розширенням простих мереж, і запишемо її з використанням наступного виразу:

$$N_{petry\_stoxastic} = \{P,T,F,M_0,Sto\}, \quad (8.2)$$

де $Sto = \{St_1,Sto_2,...,Sto_v\}$ – множина ймовірностей спрацьовування переходів; St_1 – значення ймовірності спрацьовування першого переходу; F – множина дуг, яка включає дві підмножини вхідних і вихідних дуг (на відміну від інгібіторних мереж Петрі) по відношенню до кожного переходу.

Така модель дає можливість визначити вихідні параметри мікроелектромеханічних систем та динаміку роботи системи з врахуванням стохастичної природи параметрів елементів структури інтегральних мікросистем.

Особливістю стохастичних мереж Петрі є те, що в процесі їх роботи необхідно використовувати генератор випадкових чисел. Стандартні програми генераторів типу rand та інші не задовольняють поставленим до них вимогам. Тому, необхідно розробляти принципово нові генератори рівномірного закону розподілу випадкової величини (ГРЗРВВ) чи модифікувати існуючі [95-102].

Модель, яка описана виразом (8.2) може бути вдосконалена шляхом введення параметра пріоритету спрацювання переходу. Таку модель опишемо використовуючи наступні вирази:

$$N_{petry\_stoxastic\_prior} = \{P,T,F,M_0,Sto,PR\}, \quad (8.3)$$

де $PR = \{Pr_1,Pr_2,...,Pr_v\}$ – множина пріоритетів переходів; Pr_1 – значення пріоритету переходу.

Така модель стохастичної мережі Петрі з пріоритетами дає користувачу засіб встановлення пріоритетності виконання процесів в мережі.

Модель (8.3) дає змогу врахувати часові параметри. Для цього в модель (8.3) введемо підмножину, яка включає інформацію про мінімально і максимально можливий час виконання того чи іншого процесу. Тоді модель для врахування часових параметрів буде такою:

$$N_{petry\_stoxastic\_prior\_time} = \{P,T,F,M_0,Sto,PR,Time\}, \quad (8.4)$$

де $Time = \{Time\_1, Time\_2, ..., Time\_v\}$ – множина часових параметрів переходів; $Time\_1 = \{Time\_1_{\min}, Time\_1_{\max}\}$ – підмножина часових параметрів переходу, яка включає два елементи мінімальний і максимальний час виконання першої операції.

8.3. Досліджувані параметри системи з використанням моделей на основі мереж Петрі

Моделі на основі мереж Петрі використані для аналізу **динаміки** роботи технічного пристрою, що проводиться на підставі побудованого **графа досяжності** в автоматизованому режимі.

В загальному випадку граф досяжності призначений для відображення можливих станів системи і переходів між цими станами. Відповідно інформаційна модель графа досяжності представлена в такому виді:

$$G_{досяжності} = (S, L),$$

де S – множина станів системи; L – множина зв'язків (ребер) між станами системи (елементів).

Припустивши, що кількість станів системи є кінцеве число і рівне n, то $S = \{S_1, S_2, ..., S_n\}$, де S_i – i - ий стан системи.

Процес дослідження моделей на основі мереж Петрі ґрунтується на дослідженні таких властивостей, як обмеженість, безпечність, збереженість, досяжність та живучість.

Обмеженість (чи K-обмеженість) має місце, якщо число міток в будь-якій позиції мережі не може перебільшити значення K. При проектуванні автоматизованих систем визначення K дає можливість обґрунтовано вибирати

ємності накопичувачів, тощо. Наприклад, можливість необмеженого росту числа міток свідчить про небезпеку необмеженого росту довжини черг.

Безпечність — частковий випадок обмеженості, а саме це 1-обмеженість. Якщо для деякої позиції встановлено, що вона безпечна, то її можна представляти одним тригером.

Збереженість характеризується постійністю загрузки ресурсів, тобто

$$\sum A_i N_i = const$$

де N_i — число маркерів в i-й позиції, A_i — ваговий коефіцієнт.

Досяжність $M_k \to M_j$ характеризується можливістю досягнення маркування M_j з стану мережі, який характеризується маркуванням M_k.

Живучість мережі Петрі визначається можливістю спрацьовування будь-якого переходу при функціонуванні моделюючого об'єкта. Відсутність живучості означає або про надлишок апаратури в проектованій системі, або свідчить про можливість виникнення зациклень, тупиків, блокувань.

В основі досліджень перерахованих властивостей мереж Петрі лежить аналіз досяжності. Один з методів аналізу досяжності будь-якого маркування з стану M_0 — побудова графу досяжності. Початкова вершина графу відображає маркування (стан) M_0, а інші вершини відповідним маркуванням. Дуга з M_i в M_j означає подію $M_i \to M_j$ і відповідає спрацьовуванню переходу t. В складних мережах граф може містити надто велике число вершин і дуг. Однак при побудові графу можна не відображати всі вершини, так як багато з них є дублями (дійсно, від маркування M_k завжди породжується один і той же підграф поза залежністю від того, з якого стану система пришла в M_k). Тупики виявляються за відсутністю дозволених переходів з будь-якої вершини, тобто за наявністю гілок (листків) — термінальних вершин. Необмежений ріст числа маркерів в якійсь позиції свідчить про порушення обмеження [101-110].

Наведемо приклади аналізу досяжності. Мережа Петрі і граф досяжності розміток представлені на рисунку 8.10. На рисунку вершини графа зображені в вигляді маркувань, дуги відзначені спрацьовуючими переходами. Мережа є

необмеженою і живою, так як мітки можуть накопичуватися в позиції p_5, спрацьовують усі переходи, а тупики відсутні.

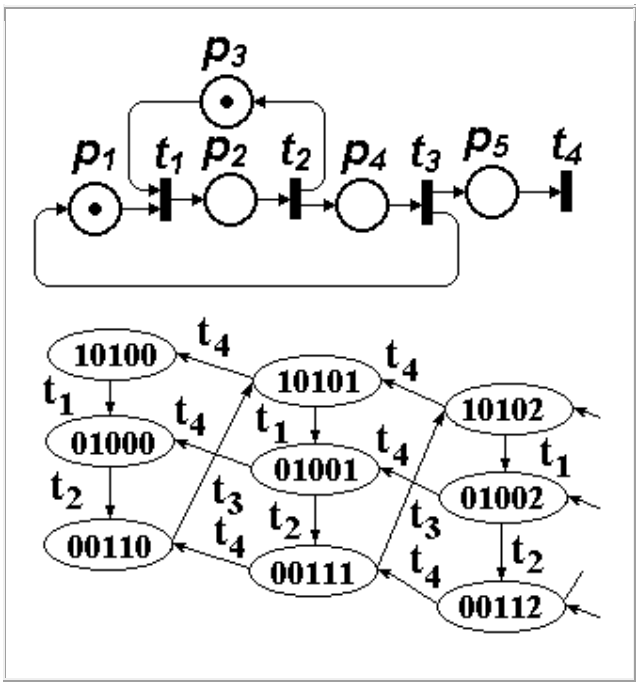


Рисунок 8.10 – Мережа Петрі та її граф досяжності

Мережа Петрі та граф досяжності розміток представлені на рис. 11. Мережа, моделююча двохпроцесорну обчислювальну систему з загальної пам’яттю, є безпечною, живою, всі розмітки досяжні.

Контрольні запитання

1. Що таке мережа Петрі?
2. На якому рівні автоматизованого проектування використовуються моделі на основі мереж Петрі?
3. Які види мереж Петрі Ви знаєте?
4. Які особливості функціонування простої мережі Петрі?
5. Які особливості функціонування стохастичної мережі Петрі?
6. Які особливості функціонування часової мережі Петрі ?
7. Які особливості функціонування кольорової мережі Петрі?

8. Які особливості функціонування мережі Петрі з пріоритетами?

9. Що таке стан системи?

10. Що таке граф досяжності?

11. Які основні вихідні параметри аналізу мереж Петрі Ви заєте?

12. Що Ви розумієте під маркуванням мережі Петрі?

13. Що Ви розумієте під обмеженістю МП?

14. Що Ви розумієте під безпечністю МП?

15. Що Ви розумієте під збереженістю МП?

16. Що Ви розумієте під досяжністю МП?

17. Що Ви розумієте під живучістю МП?

18. Які вихідні параметри об’єкта проектування на системному рівні дають змогу визначити моделі на основі мереж Петрі?

19. Які розширення мереж Петрі Ви знаєте?

РОЗДІЛ 9. ПОБУДОВА СТРУКТУРНИХ МОДЕЛЕЙ КСМ НА ОСНОВІ ТЕОРІЇ СИСТЕМ МАСОВОГО ОБСЛУГОВУВАННЯ

9.1 Основи систем масового обслуговування

Розв’язання задач аналізу КСМ потребує побудови математичних моделей. Досить широко для цього використовується апарат теорії систем масового обслуговування.

Основи теорії систем масового обслуговування (СМО) були закладені в працях датського математика, співробітника Копенгагенської телефонної компанії Агнера Крарупа Ерланга і отримали широкий розвиток у подальших дослідженнях [111-115].

З використанням математичного апарату систем масового обслуговування можна розв’язувати як задачі аналізу, так і задачі синтезу.

В загальному випадку, в задачах аналізу визначають оцінку ефективності функціонування систем масового обслуговування при незмінних, наперед заданих вхідних характеристиках системи; структури системи; дисципліни обслуговування; потоках вимог та законів розподілу часу їх обслуговування та ін.

Використання СМО в задачах синтезу пов’язано з пошуком оптимальних параметрів проектованої чи досліджуваної системи.

Для прикладу, з використанням моделей на основі СМО можна розв’язати такі задачі, а саме:

1. Задачу визначення кількості магістральних ліній зв’язку на кожній місцевій автоматичній телефонній станції.

2. Задачу по визначенню кількості каналів опрацювання даних при реалізації системи інтелектуального будинку.

3. Задачу по визначенню кількості мікропроцесорів для забезпечення заданої обчислювальної потужності пристрою.

4. Задачу визначення кількості касових апаратів у кожному з продовольчих магазинів, що належать фірмі, яка має розгалужену торгову мережу.

5. Задачу визначення кількості бензоколонок і чисельності обслуговуючого персоналу на кожній бензозаправній станції.

6. Задача пов’язана з визначенням кількості педіатрів у дитячій лікарні та інші.

З вищенаведених задач можна зробити висновок, що, в більшості випадків, це є задачі, де присутній пристрій з обмеженим ресурсом.

Математична модель системи масового обслуговування включає наступні основні елементи: потік вимог, що надходять на вхід системи (вхідний потік); чергу, що складається з вимог, які очікують на обслуговування; систему обслуговування; вихідний потік обслужених вимог, характеристики якості системи; механізм (дисципліну) обслуговування.

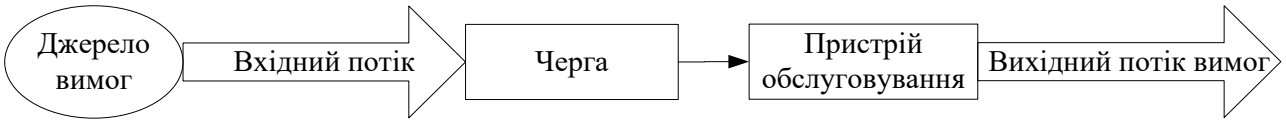


Рисунок 9.1 – Основні складові системи масового обслуговування

В процесі використання СМО використовують поняття **Марківського процесу**. Процес називається Марківським, якщо на вхід надходить найпростіший потік вимог, а час обслуговування замовлень розподіляється за показниковим законом. У такому разі в системі протікає Марківський процес, тобто ймовірність характеристик в майбутньому залежить від стану системи у даний момент часу і не залежить від того, коли і як система опинилася у цьому стані.

Під поняттям найпростішого потоку будемо розуміти такий потік подій, який має наступні властивості: **стаціонарності**, **відсутність наслідків** та **ординарності**.

В даному випадку, потік подій **називається стаціонарним**, якщо ймовірність надходження певної кількості вимог за проміжок часу $(0, \tau)$ дорівнює ймовірності надходження цієї кількості вимог за будь-який інший проміжок часу $(t, t+\tau)$, тобто ймовірність надходження вимог залежить лише від тривалості проміжку надходження вимог і не залежить від того, в який саме момент ми починаємо розглядати надходження вимог.

Потік подій будемо **називати потоком без наслідків**, якщо ймовірність надходження певної кількості вимог до системи у довільний момент часу не залежить від того, коли і скільки вимог надійшло до цього моменту часу.

Відповідно, потік подій **називається ординарним**, якщо ймовірності надходження двох і більше вимог до системи за проміжок часу такі, що ними нехтуємо у порівнянні з ймовірністю надходження однієї вимоги за цей проміжок часу. Умова ординарності виражає практичну вірогідність надходження подій поодинці, а не кількох одразу.

Потік подій із властивостями стаціонарності, відсутності наслідків та ординарності називається найпростішим або стаціонарним Пуассонівським потоком [112-116].

При дослідженні СМО звичайно вважають, що вхідний потік вимог підпорядковується закону Пуассона, за яким розглядають відносно рідкі події. За законом Пуассона, ймовірність появи точно K подій із n за проміжок часу t визначається виразом.

$$P_n(k) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}\Bigg|_{\substack{k=1 \\ t=\Delta t}} = \frac{(\lambda \Delta t)^1 e^{-\lambda \Delta t}}{1!} \approx (\lambda \Delta t)^1 e^0 = \lambda \Delta t\,,$$

де $\lambda = 1/t_1$ - середнє число вимог (об'єктів), що надходять до СМО за одиницю часу (секунду, хвилину, годину, тиждень); t_1 - середнє значення інтервалів часу між появами вимог, подій; k - кількість одночасних вимог.

Ймовірність появи однієї вимоги в інтервалі від t до $t+\Delta t$, дорівнює $\lambda * \Delta t$ та не залежить від t, а ймовірність появи у цьому інтервалі більше однієї вимоги дуже мала (дорівнює нулю).

Ще одним важливим поняттям в теорії СМО є **поняття графа станів системи**. Основним поняттям при аналізі процесу системи масового обслуговування є стан системи. Знаючи стан системи можна передбачити у імовірнісному сенсі її поведінку. Простіший потік – це стаціонарний Пуассонівський потік. Якщо всі потоки подій, що переводять систему із одного стану в інший являються Пуассонівськими, то для цих систем ймовірність стану описується за допомогою систем звичайних диференційних рівнянь. Отже, СМО може перебувати в певній кількості станів, які позначимо наступним чином через $S_0, S_1, S_2, ..., S_n, S_{n+1}$.

Для прикладу, на рисунку 9.2 зображено граф станів даної системи масового обслуговування. На графі станів стрілками зображають інтенсивність того потоку подій, який переводить систему з одного стану в інший. Граф станів з позначенням біля стрілок інтенсивності – називається розміченим. В процесі дослідження функціонування системи необхідно визначити ймовірності станів системи, тобто:

$$P_0(t) \to S_0, \quad P_1(t) \to S_1 \quad \dots \quad P_n(t) \to S_n.$$

Якщо складений розмічений граф стану, то для побудови математичної моделі, тобто для складання системи звичайних диференційних рівнянь рекомендується використовувати наступні правила: похідна $\frac{\lambda P_u(t)}{dt}$ ймовірності перебування системі у стані n дорівнює алгебраїчній сумі наступних величин: число величин цієї суми дорівнює числу стрілок на графі стану системи, що з'єднує стан n з іншими станами. Якщо стрілка направлена до стану n, то відповідна величина береться зі знаком "+" . Якщо стрілка направлена від стану n – то зі знаком "-". Кожна величина суми дорівнює добутку ймовірностей того стану, з якого направлена стрілка на інтенсивність потоку подій, що переводять систему по даній стрілці.

У відповідності з розміченим графом стану, використовуючи даний стан, запишемо систему звичайних диференційних рівнянь ймовірностей стану таким чином:

$$\frac{dP_0}{dt} = -P_0(t)\lambda + P_1(t)\mu,$$

$$\frac{dP_n}{dt} = P_{n-1}(t)\lambda - (\lambda + \mu)P_n(t) + P_{n+1}(t)\mu.$$

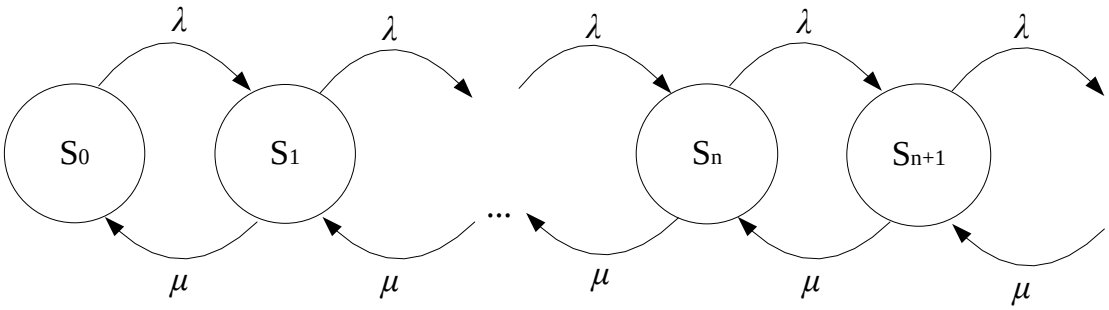


Рисунок 9.2 – Приклад схеми графа станів досліджуваної системи

Систему диференціальних рівнянь складають по графу станів відносно невідомих ймовірностей P_j існування станів S_j. При цьому, використовують такі правила:

1. Число рівнянь дорівнює $(n+1)$ - числу станів S_j в графі.

2. В лівій частині рівняння стоїть похідна ймовірності по часу, а в правій частині – члени, пов’язані дугами, які виходять та входять у вершину даного стану S_j.

3. Кожний член правої частини береться зі знаком «+», якщо входить у вершину S_j, і зі знаком «-», якщо виходить з неї. Кожний член правої частини дорівнює добутку інтенсивності потоку інформації на ймовірність того стану, звідки виходить дуга.

4. Одне з отриманих рівнянь є зайвим і замінюється рівнянням суми ймовірностей повної групи для взаємно несумісних подій

З графу отримуємо рівняння.

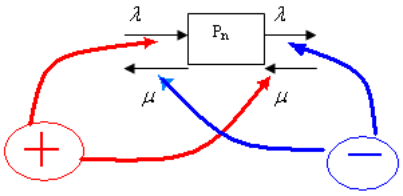


Рисунок 9.3 – Схема побудови диференціальних рівнянь

$$\frac{P_0(t)}{dt} = -\lambda P_0(t) + \mu P_1(t)$$

$$\frac{P_1(t)}{dt} = \lambda P_0(t) - (\lambda + \mu) P_1(t) + \mu P_2(t)$$

...

$$\frac{P_n(t)}{dt} = \lambda P_{n-1}(t) - (\lambda + \mu) P_n(t) + \mu P_{n+1}(t)$$

9.2 Модель розімкнутої системи масового обслуговування

Загалом, існує багато різних варіантів СМО. Одним з найбільш широко використовуваних є одноканальні. Одноканальні СМО бувають розімкнуті та замкнуті. В загальному випадку прикладом розімкнутої СМО може бути система, яка включає процесор та пристрої вводу/виводу. Така система, в спрощеному вигляді, зображена на рисунку 9.4.

Модель такої системи з використанням апарату теорії СМО передбачає виконання таких умов: стаціонарності, ординарності та відсутності післядії. Відповідно, такий вхідний потік є простим.

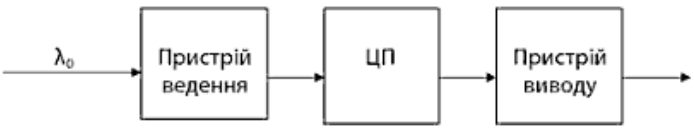


Рисунок 9.4 – Приклад розімкнутої СМО

Відомі також інтенсивність λ надходження потоків вимог (середнє число обслуговування за одиницю часу - $1/\Delta t_{обсл}$). В нашій задачі потік вимог простіший.

Існує визначений математичний прийом, що значно полегшує вивід диференційного рівняння для імовірнісного стану. Спочатку будується розмічений граф стану з показом можливих переходів. Це полегшує дослідження та робить його більш наочним. Граф стану, на якому проставлені не тільки стрілки переходів, але й інтенсивність відповідних потоків подій називають розміченим [111-113].

Зобразимо розмічений граф стану одноканальної розімкнутої системи масового обслуговування з очікуванням (див.рисунок 9.5):

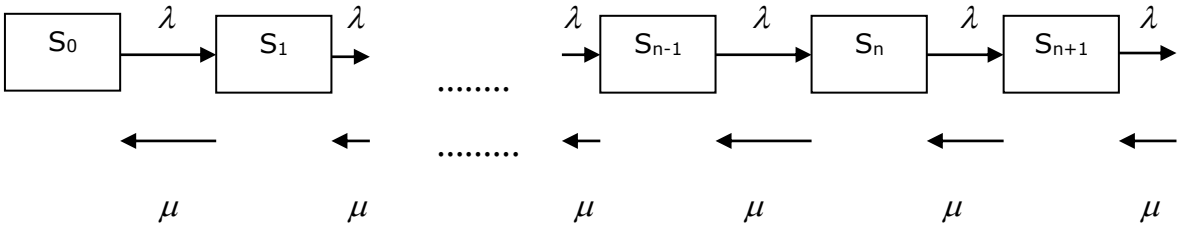


Рисунок 9.5 – Граф стану одноканальної розімкнутої СМО

Обмежимося дослідженням режиму роботи, який встановився, замкнутої одноканальної системи. Тоді:

$$\frac{dP_0}{dt} = 0 \ (n=0,1,...).$$

Дійсно, замість системи диференційних рівнянь отримуємо систему алгебраїчних рівнянь:

$$-P_0\lambda + P_1 = 0,\ P_0\lambda - (\lambda+\mu)P_1 + P_2\mu = 0,\ \ldots,\ P_{n-1}\lambda - (\lambda+\mu)P_n + P_{n-1}\mu = 0.$$

Використовуючи отриману систему алгебраїчних рівнянь, легко виразити ймовірності стану системи у вигляді квадратної рекурентної формули. З

першого рівняння визначається ймовірність присутності однієї вимоги в системі.

$$P_1 = P_0 \frac{\lambda}{\mu} = \psi P_0 .$$

Аналогічним чином визначаються інші значення ймовірностей.

Отже, з використанням моделей можемо отримати такі параметри: P – ймовірність простою каналу обслуговування; P_n - ймовірність того, що в системі знаходяться n-вимог; $N_{сист}$ - середнє число вимог, що знаходяться в системі; $N_{чер}$ - середнє число вимог, що знаходяться в черзі; $T_{сист}$ - середній час очікування вимог в системі.

9.3 Модель на основі замкнутої СМО

Іншим широко використовуваним варіантом є замкнута одноканальна СМО (див.рисунок 9.6).

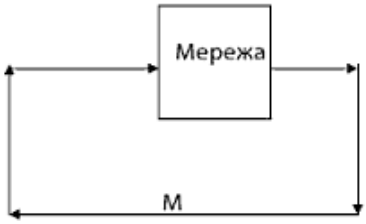


Рисунок 9.6 – Приклад замкнутої одноканальної СМО

Нехай досліджується деяка система масового обслуговування з обмеженою кількістю вимог в системі, тобто вимоги, що обслуговуються, знову повертаються в систему обслуговування. Інтенсивність надходження однієї вимоги в систему відома і дорівнює λ. Інтенсивність обслуговування також відома та дорівнює μ. Число вимог, що потребують обслуговування, дорівнює m.

Стан системи будемо пов’язувати з числом вимог, що знаходяться в системі. При цьому можливі два стани: число вимог, що поступили в систему, дорівнює нулю $(n=0)$, тобто канали обслуговування простоюють; число вимог, що поступили в систему $(0=n\le m)$.

Зобразимо розмічений граф стану одноканальної замкнутої системи масового обслуговування з очікуванням:

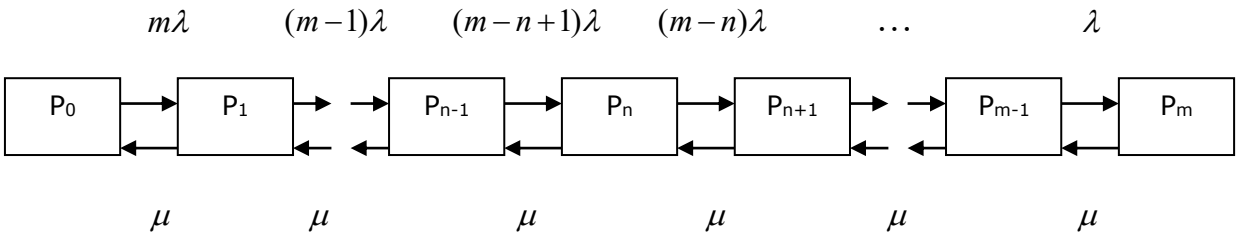


Рисунок 9.7 – Розмічений граф стану одноканальної замкнутої СМО

У відповідності до розміченого графа стану та використовуючи правило Колмогорова, запишемо систему диференційних рівнянь для ймовірності стану:

$$\frac{dP_0}{dt}=-m\lambda P_0(t)+\mu P_1(t);$$

$$\frac{dP_n}{dt}=-\left[(m-n)\lambda+\mu\right]P_n(t)+(m-n+1)\lambda P_{n-1}(t)+\mu P_{n+1}(t),$$

$$\frac{dP_m(t)}{dt}=-\mu P_m(t)+\lambda P_{m-1}(t).$$

Обмежимося дослідженням режиму роботи системи, що встановився. Тоді:

$$\frac{dP_n(t)}{dt}=0,\quad n=(0,1,...,m)$$

і замість системи звичайних диференційних рівнянь ми отримуємо систему алгебраїчних рівнянь:

$$m\lambda P_0-\mu P_1=0,$$

$$[(m-n)\lambda+\mu]P_n-(m-n+1)\lambda P_{n-1}-\mu P_{n+1}=0.$$

З використанням вищенаведеної моделі можна визначити основні характеристики досліджуваної системи, а саме: ймовірність, що в системі є n вимог; ймовірність простою каналу обслуговування; середнє число вимог, що знаходяться в черзі; середнє число вимог, що знаходяться в системі; середній час очікування в черзі; середній час очікування вимоги в системі та ін.

9.4 Задача аналізу багатоканальної розімкнутої системи з очікуванням

Актуальним до практики є варіант багатоканальної СМО. Для прикладу, це може бути багатопроцесорна система (див.рисунок 9.8).

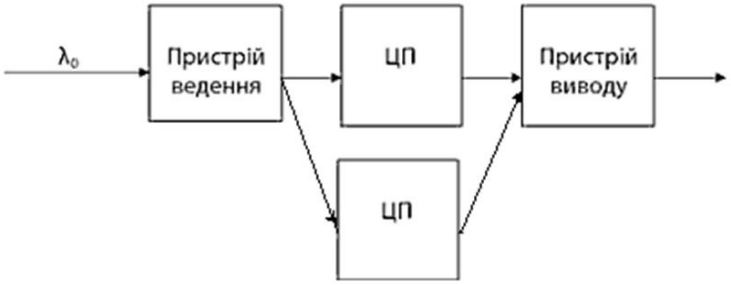


Рисунок 9.8 – Приклад спрощеної схеми багатопроцесорної системи

Отже, побудуємо модель. Нехай відомі інтенсивність λ надходження потоку вимог в систему, та інтенсивність μ обслуговування цих вимог. Число каналів обслуговування N_k.

В цій задачі можливі два випадки: в системі n змінюється $0 \le n < N$; число вимог $n \ge N_k$ - числу каналів.

В першому випадку всі вимоги, що знаходяться в системі, одночасно обслуговуються, і не всі канали зайняті. Загальна інтенсивність обслуговування: $\mu * n$.

Зобразимо розмічений граф стану багатоканальної розімкнутої системи масового обслуговування:

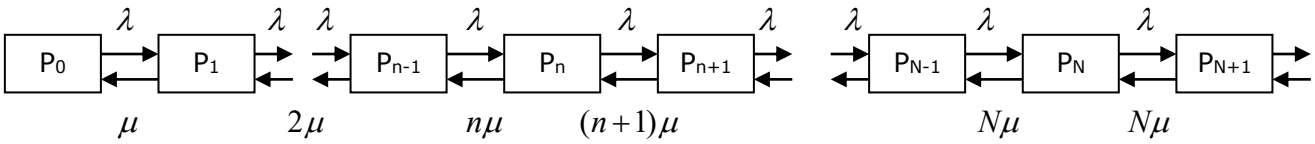


Рисунок 9.9 – Розмічений граф стану багатоканальної розімкнутої СМО

У відповідності з розміченим графом стану і правилом Колмогорова запишемо систему звичайних диференційних рівнянь для стану системи.

$$n = 0,$$

$$\frac{dP_0(t)}{dt} = -P_0(t)\lambda + P_1(t)\mu,$$

.....

$$1 \le n < N,$$

$$\frac{dP_n(t)}{dt} = P_{n-1}(t)\lambda - (\lambda + n\mu)P_n + P_{n+1}(n+1)\mu = 0,$$

.....

$$n \ge N,$$

$$\frac{dP_n(t)}{dt} = P_{n-1}(t)\lambda - (\lambda + N\mu)P_n(t) + N\mu P_{n+1}(t) = 0.$$

Обмежимося дослідженням режиму роботи системи, що встановився, коли λ - *const*, μ - *const*

$$t \to \infty,\ P_n(t) \to const,$$

$$\frac{dP_n(t)}{dt} = 0 \quad (n = 0,1,...),$$

і тоді замість системи звичайних диференційних рівнянь отримуємо систему алгебраїчних рівнянь:

$$n = 0,$$

$$-\lambda P_0 + \mu P = 0,$$

...

$$1 \le n < N,$$

$$\lambda P_{n-1} - (\lambda + n\mu)P_n + P_{n+1}(n+1)\mu = 0,$$

...

$$n \ge N,$$

$$\lambda P_{n+1} - (\lambda + N\mu)P_n + P_{n+1}N\mu = 0.$$

Використовуючи отримані алгебраїчні рівняння можна визначити ймовірність того, що в системі знаходяться n вимог $P_n(t)$, ймовірність простою каналів обслуговування $P_0(t)$, середнє число вимог, що знаходяться в черзі, середній час очікування, середнє число вільних каналів обслуговування та інші параметри досліджуваного об’єкта.

9.5 Основи мови GPSS

Мова GPSS (General Purpose Simulation System) призначена для моделювання складних технічних об'єктів, представлених моделями на основі систем масового обслуговування.

В математичних моделях (ММ) складних об'єктів, представлених у вигляді систем масового обслуговування (СМО), фігурують засоби обслуговування - обслуговуючі апарати (ОА), і заявки, що обслуговуються - транзанкти. Так, в моделі комп’ютерної системи ОА відображають процесори, а транзанкти – програми, що поступають на виконання [110-120].

Стан СМО характеризується станами ОА, транзанктів і черг до ОА. Стан ОА описується двійковою змінною, яка може приймати значення "зайнятий" або "вільний". Змінна, що характеризує стан транзанкта, може мати значення "обслуговування" або "очікування". Стан черги характеризується кількістю транзанктів, що знаходяться в ній.

В загальному випадку, імітаційна модель СМО являє собою алгоритм, що відображає поведінку СМО, тобто який відображає зміни стану СМО у часі при заданих потоках заявок, що поступають на входи системи. Параметри вхідних потоків заявок - зовнішні параметри СМО. Вихідними параметрами є величини, що характеризують властивості системи - якість її функціонування. Приклади вихідних параметрів: продуктивність СМО - середнє число заявок, що обслуговуються за одиницю часу; коефіцієнти завантаження обладнання - відношення часів обслуговування до загального часу в кожному ОА; середній час обслуговування однієї заявки. Основна властивість ОА, що враховується в моделі СМО - це витрати часу на обслуговування, тому внутрішніми параметрами в моделі СМО є величини, що характеризують цю властивість ОА. Звичайно час обслуговування розглядається як випадкова величина і як внутрішні параметри фігурують параметри законів розподілу цієї величини.

Імітаційне моделювання дає змогу дослідити СМО при різних типах вхідних потоків і інтенсивності надходження заявок на входи, при варіаціях параметрів ОА, при різних дисциплінах обслуговування заявок. Дисципліна обслуговування - правило, по якому заявки надходять з черг на обслуговування. Величина, яка характеризує право на першочергове обслуговування, називається пріоритетом. У моделях СМО заявки, що приходять на вхід зайнятого ОА, утворять черги, окремі для заявок кожного пріоритету. При звільненні ОА на обслуговування приймається заявка з непустої черги з найбільш високим пріоритетом.

Основний тип ОА - пристрої, саме в них відбувається обробка транзанктів з витратами часу. До ОА відносяться також накопичувачі (пам'яті), що відображають засоби зберігання деталей, що обробляються в виробничих лініях або даних, що обробляються в обчислювальних системах. Накопичувачі характеризуються не часом обслуговування заявок, а місткістю - максимально можливою кількістю заявок, що одночасно знаходяться в накопичувачі.

До елементів імітаційних моделей СМО крім ОА відносять також вузли і джерела заявок. Зв'язки ОА між собою реалізовують вузли, тобто характеризують правила, по яких заявки прямують до того чи іншого ОА.

Об'єкти мови GPSS поділяють на категорії і типи.

Найменування категорій: операційна, апаратна, динамічна, обчислювальна, статистична, запам'ятовуюча, згруповуюча.

Найменування типів об'єкта: блоки, повідомлення, пристрої пам'яті, логічні ключі, арифметичні і булеві змінні, функції, черги, таблиці, комірки, матриці комірок, групи, списки.

Блоки. З об'єктами пов'язані певні сукупності блоків, які описують функціонування самої системи, що моделюється, або які містять додаткову інформацію про порядок моделювання. Моделювання полягає в просуванні повідомлень (транзанк-тів) від блоку до блоку. Це просування створює блок GENERATE. Кожне просування повідомлення є подією в моделі. Комплекс програм, який планує виконання подій, реалізує функціонування блоків моделі, реєструє статистичну інформацію про проходження повідомлень, називається симулятором. Симулятор реєструє час настання кожної з відомих на даних момент подій і виконує їх з наростаючою часовою послідовністю. Затримку повідомлень у часі, згідно із заданим законом розподілу, може здійснити блок ADVANCE. Реалізація подій в моделі може бути заблокована при невиконанні деяких умов; ці події будуть виконані при сприятливій зміні блокуючих умов.

Симулятор забезпечує відлік модельного часу в прийнятих одиницях, які називають абсолютним умовним часом. З кожним повідомленням пов'язаний відносний умовний час, відлік якого починається при вході повідомлення в систему, яка моделюється і закінчується при виході повідомлення з системи. Виведення з системи здійснює блок TERMINATE.

Операційна категорія. Блоки і повідомлення - два основних типи об'єктів мови GPSS. Практично всі зміни стану моделі відбуваються внаслідок

входу повідомлень в блоки і виконання ними своїх функцій. З блоками пов'язані карти, які керують процесом моделювання.

Карта SIMULATE вказує на необхідність проведення моделювання. При її відсутності проводиться тільки трансляція програми.

Карта START вказує на отримання початкових даних і початок моделювання. Закінчення моделювання проводиться при обнуленні лічильника кількості повідомлень, що виводяться, який задається в полі A. Поле C визначає інтервал видачі проміжної статистики.

Набори керуючих карт дозволяють стирати накопичені дані, повторювати виконання програми та змінювати параметри блоків.

Апаратна категорія. Мова GPSS оперує трьома групами обладнання: пристроями, пам’яттю і логічними ключами. До групи пристроїв відносяться блоки SEIZE, RELEASE, PREEMPT, RETURN. Введення в моделюючу програму опису пристрою дає змогу автоматично реєструвати статистичну інформацію.

Групу пам’ятей утворюють блоки ENTER, LEAVE і карта опису пам'яті STORAGE. Введення в моделюючу програму пам'яті дає змогу автоматично реєструвати статистичну інформацію.

Для управління ключами використовується оператор LOGIG. Передбачено три режими зміни стану ключа: скидання в “0”; встановлення в “1”; інвертування зміни стану ключа на протилежне.

Динамічна категорія. Динамічні об'єкти - це повідомлення (транзанкти). У процесі моделювання вони створюються, породжують інші повідомлення, збираються і знищуються. Кожному повідомленню відповідає набір параметрів, кількість яких може бути встановлена до 100. Якщо кількість параметрів не називається, то воно приймається рівним 12. Повідомленням можна присвоювати пріоритет від 0 до 127; якщо пріоритет не названий, то він приймається рівним 0. З динамічною категорією об'єктів пов'язані блоки, основні з яких можна поділити на п'ять груп.

Група затримки складається з єдиного блоку ADVANCE, група створення і знищення повідомлень з блоків GENERATE, TERMINATE, SPLIT, ASSEMBLE; група зміни маршрутів повідомлень - з блоків TRANSFER, LOOP, GATE, TEST. Блок TRANSFER має шість основних режимів використання.

Блок GATE в пристроях використовує наступні умови переходу: U - використовується; NU - не використовується; I - зайнято перериваючим повідомленням; NI - не зайнято перериваючим повідомленням.

Блок GATE використовує наступні умови переходу: SF - заповнена; SE - пуста; SNF - не заповнена; SNE - не пуста.

Група синхронізації повідомлень включає в себе блоки MATCH і GATHER. Зв'язані блоки MATCH не допускають просування повідомлення, що поступило першим, поки не поступило друге повідомлення. Блок GATHER затримує повідомлення доти, поки не збереться вказана кількість повідомлень.

Блоки ASSIGN, INDEX, MARK, PRIORITY складають групу зміни атрибутів повідомлень.

Обчислювальна категорія. У обчислювальній категорії використовуються об'єкти трьох видів: арифметичні змінні, логічні (булеві) змінні і функції. Арифметичні змінні описуються блоком VARIABLE в режимі цілих чисел і FVARIABLE в режимі з плаваючою точкою. Назву карти описують арифметичні дії над стандартними числовими атрибутами (СЧА).

Аргументи і результати розглядаються як цілі числа. При обчисленні використовується п'ять алгебраїчних операцій: “+” (додавання); “-“ (віднімання); “*” (множення); “/” (ділення з відкиданням залишку); ділення на нуль не вважається помилкою і дає результат, рівний нулю; “@” (ділення по модулю, при якому часка відкидається і зберігається залишок, який вважається додатнім).

Допускається використання не більш п'яти дужок.

Як будь-яка мова вона включає словник і граматику, за допомогою яких можуть бути розроблені моделі систем. Машинна програма інтерпретує

модель, написану на GPSS, надаючи розробнику моделі можливість проведення експериментів. Моделі систем на GPSS можуть бути представлені у вигляді блок-схем чи записані у вигляді послідовності операторів, еквівалентні блок-схемі. Блок-схема представляє собою набір елементів з характерним окресленням блоків, з'єднаних між собою стрілками. Розробнику моделей представляється набір більш ніж із 40 блоків. Вигляд кожного із блоків стандартний. Із допустимої множини блоків вибудовують послідовність, об'єднану зв'язками, які показують взаємодію блоків. Однакові блоки можуть зустрічатися в моделі довільну кількість раз, що визначається особливостями модельованих систем.

Різні події в реальних системах відбувається на протязі деякого періоду часу. Завдання приходять в систему, коли приходить їх черга, вони попадають на обслуговування в процесор, потім покидають систему. Якщо всі ці події представити в моделі, то їх проходження має бути організоване на фоні модельного часу.

Коли починається моделювання, в інтерпретаторі планується прихід першого транзанкту. Після цього таймер модельного часу встановлюється в значенні часу, який відповідає моменту появи першого транзанкту в моделі. Цей транзанкт входить в модель, а інтерпретатор GPSS просуває далі значення таймера до значення часу, коли відбувається наступна подія.

Особливості таймера GPSS: таймер реєструє тільки цілі значення; одиниця часу визначається розробником.

Блоки представлені в моделі сукупністю ключових слів і операндів. Для організації посилань блокам можна присвоювати символічні імена (алфавітно-цифрові символи, не більше п'яти).

Операнди блоків задають інформацію, специфічну для даного блоку. Число операндів залежить від типу блоку. Значення одних операндів має бути вказані завжди, інші можна не вказувати, тоді вони задаються по замовчуванні.

Оператори GPSS записуються в вигляді стандартних 80 Байтних записів.

1 Байт - ознака коментаря,

2-6 Байти - ім'я блоку,

8-18 Байти - операція,

19-71 Байти - операнди,

72 і далі - Байти - інтерпретатором не сприймаються.

В стрічковому редакторі GPSS PC переміщення по полях автоматизоване (при натисканні на клавішу пробіл курсор переміщується в першу позицію наступного поля блоку).

Внесення транзанктів в модель виконується блоком

GENERATE A, B, C, D, E,

де A - середній інтервал часу, B - половина поля допуску, C - зміщення інтервалів, D - обмежувач кількості транзанктів, E - рівень пріоритету.

Значення параметрів за замовчуванням: A - 0 , B - 0, C - відсутній, D - без обмеження, E - 0.

Усі можливі види розподілів інтервалів часу поступлення транзанктів в GPSS ділять на рівномірно розподілені та всі інші види розподілу. Блоком організується тільки рівномірний розподіл, інші види розподілів оформляються через визначення функцій.

GENERATE 5, 3, 10, 19, 16

Цей блок задає прихід першого транзанкту (параметр C) в момент модельного часу 10, інші приходять у відповідності з рівномірним законом 5±3 одиниці модельного часу. Через блок може ввійти в модель не більше 19 транзанктів, пріоритет кожного - 16 (всього 128 рівнів пріоритетів 0 - 127).

Транзанкти видаляються із моделі блоком TERMINATE A (завершити). Параметр A - покажчик зменшення лічильника завершення моделювання, він задає величину, яка повинна відніматися із спеціального лічильника кожен раз, коли транзанкт входить в блок TERMINATE. За замовчуванням значення операнду A=0. Вхід транзанкту в такий блок TERMINATE не викликає зменшення лічильника завершення. В моделі може бути будь-яка кількість блоків TERMINATE, але лічильник завершення тільки один.

Займання вільного приладу моделюється при вході транзанкту в блок SEIZE A який має наступні властивості:

1. Якщо прилад з іменем A занятий, транзанкт не може ввійти в блок, він має очікувати в черзі.

2. Якщо прилад вільний, транзанкт може ввійти в блок, при цьому відбувається зміна статусу приладу із "незайнято" в "зайнято".

A - символічне чи числове ім'я приладу, який займають.

Звільнення зайнятого приладу здійснюється при вході транзанкту в блок RELEASE A. Призначення цього блоку - зміна стану раніше занятого приладу з "зайнято" на "незайнято".

Цей блок ніколи не забороняє вхід транзанкту.

Реалізація затримки у часі здійснюється у блоці

ADVANCE A, B

Можливі варіанти розподілу часу затримки поділяються на дві категорії: рівномірний розподіл та інші розподіли. Блоком задається рівномірний розподіл, інші вимагають використання функцій. A - середній час затримки, B - половина поля допуску.

Блок ADVANCE ніколи не перешкоджає входу транзанктів. Будь-яке число транзанктів може знаходитись в цьому блоці одночасно. Новоприбулий транзанкт ніяк не вплине на вже ті, що вже знаходяться в блоці.

Значення за замовчуванням A - нуль, B - нуль. Класичний спосіб використання блоків при моделюванні приладу з іменем PRO із затримкою 6 ±3

```
SEIZE  PRO
ADVANCE   6, 3
RELEASE  PRO
```

Оператор START має чотири параметри A, B, C, D.

```
START   A,  B,  C,  D,
```

де: A - початкове значення лічильника завершення моделювання, B – ознака подавления друку, C - початкове значення лічильника проміжного друку, D - ознака роздруку ланцюгів.

Окрім лічильника завершення моделювання в GPSS існує лічильник проміжного друку (ЛПД). В процесі моделювання кожний раз при відніманні із лічильника завершення деякої величини така ж величина віднімається із ЛПД. Коли ЛПД приймає нульове значення, видається стандартний друк результатів, потім ЛПД автоматично приймає значення операнду С і моделювання продовжується [120].

Перехід транзанктів в блок відмінний від наступного здійснюється оператором TRANSFER (передати).

Розрізняють три режими роботи оператора:

1). Безумовний режим.

Оператор TRANSFER, <ім'я> - визначає безумовний перехід до блоку який має <ім'я>.

Наприклад: TRASNSFER, BLOC1

2). Статистичний режим.

Оператор TRANSFER .<N>, < n1>, < n2> визначає передачу транзанктів в N% випадків блоку < n2>.

Наприклад: TRANSFER .25,BLOC1,BLOC2

В 25 % випадків блок передає транзанкт в BLOC2, в 75% випадків в BLOC1.

TRANSFER .30,,CPU

В 30% випадків передача відбувається в блок CPU, в 70%-- в наступний.

3). Умовний режим.

Оператор TRANSFER BOTH, < ім'я>, < ім'я>

визначає передачу транзанктів будь-якому із двох блоків, готових його прийняти.

Наприклад: TRANSFER BOTH,BLOC1,BLOC2

Блок проводить спробу передати транзанкт в BLOC1, якщо той зайнятий, то в блок BLOC2, якщо BLOC2 зайнятий, то транзанкт очікує в даному блоці.

TRANSFER BOTH,,BLOC2

При цьому мається на увазі, що спочатку проводиться спроба передати транзанкт в наступний блок, якщо він не може прийняти, то виконується спроба передати транзанкт в блок з іменем BLOC2, якщо BLOC2 не може прийняти транзанкт, то транзанкт очікує в даному блоці.

9.6 Основи роботи з системою GPSS Word та приклади розв'язання задач

Для запуску на виконання системи GPSS Word необхідно двічі клікнути на виконуючому модулі GPSS World Student Version.exe. Після цього на екрані монітора з'явиться основне меню системи (див. рисунок 9.10).

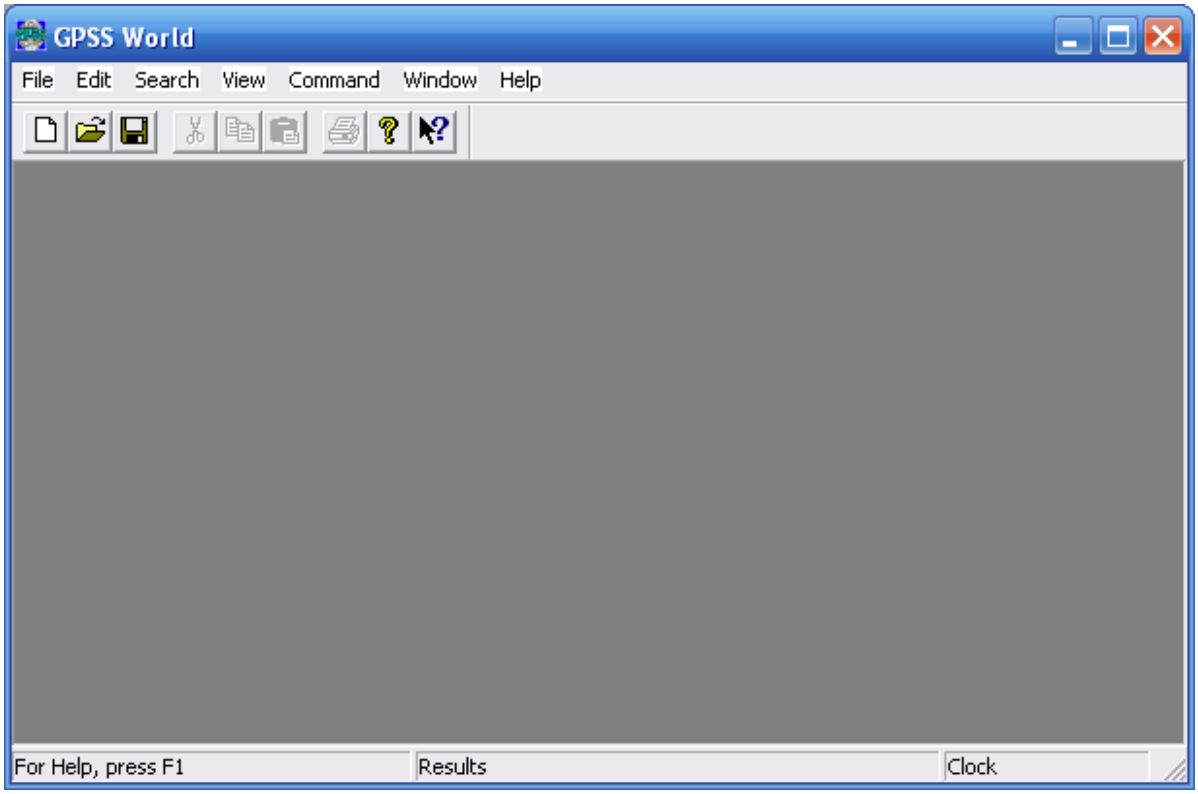


Рисунок 9.10 – Основне меню системи GPSS Word

Основне вікно системи включає декілька компонент.

1. У верхній частині розміщена стрічка заголовка.

2. Нижче розміщено основне меню, а ще нижче – панель інструментів, за якою розміщена клієнтська область.

3. В самому низу головного вікна розміщена стрічка стану, яка розділена на три частини:

- ліва частина стрічки – показує підказку з інформацією про використовуваний пункт меню;
- середина – показує повідомлення про помилки;
- права частина стрічки – призначена для відображення модельного часу в процесі виконання моделі.

Для формування нової моделі, необхідно у меню “File” вибрати “New” (див. рисунок 9.11). В результаті на екран монітора система виведе меню (рисунок 9.12). У цьому меню користувач має змогу набрати модель з допомогою мови GPSS.

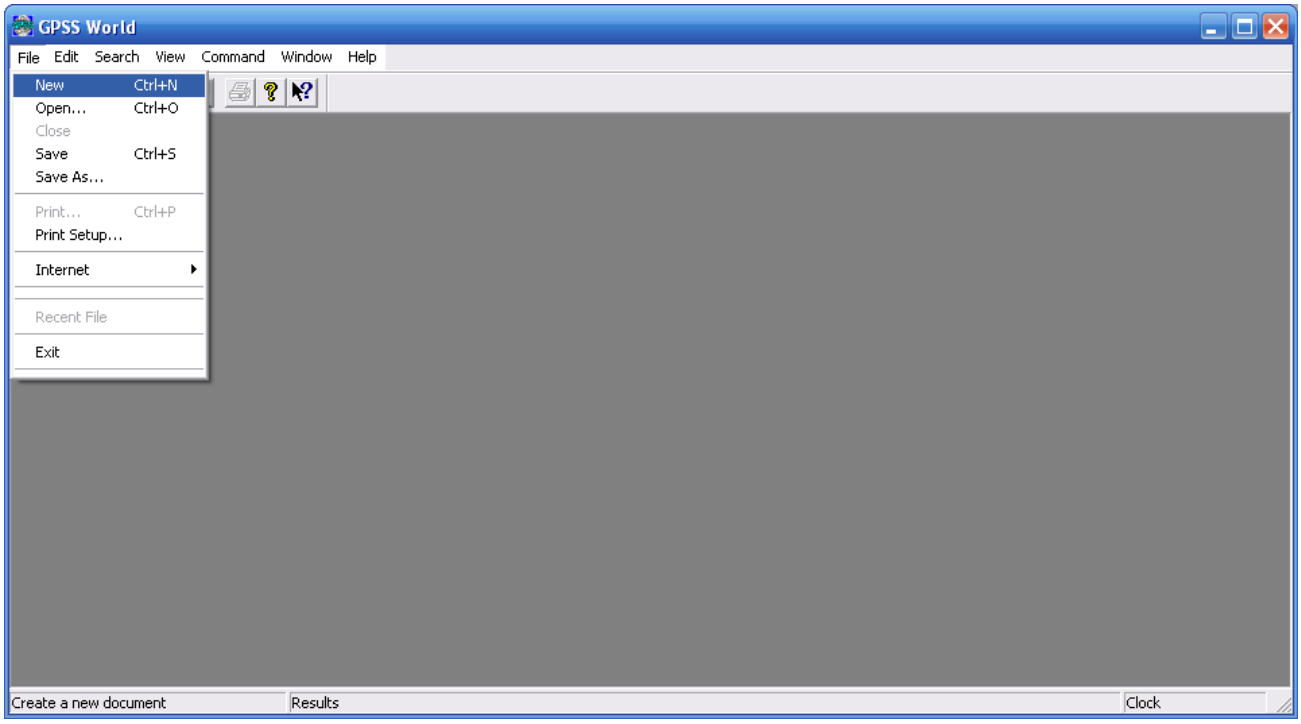


Рисунок 9.11 – Підсистема побудови нової моделі

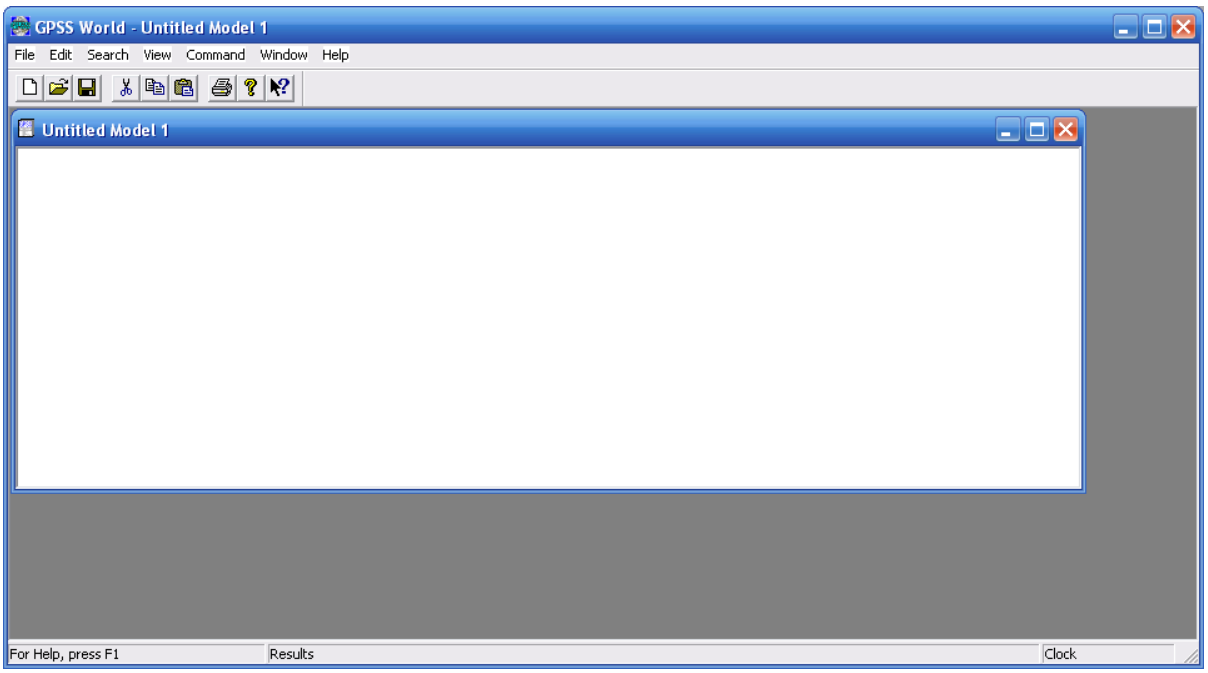


Рисунок 9.12 – Підсистема для введення коду моделі

Користувач має змогу вводити команди мови з допомогою системи (див.рисунок 9.13), а параметри команд (рисунок 9.14). Окрім того можна усі команди вводити з допомогою клавіатури.

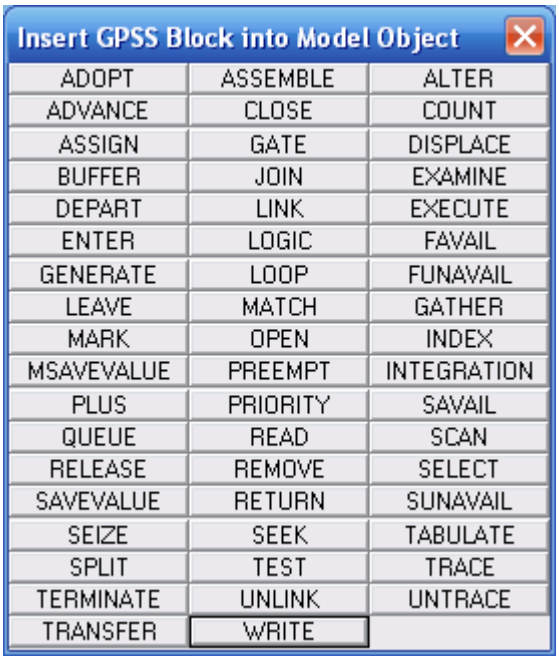


Рисунок 9.13 – Модель для автоматичного введення команди в модель

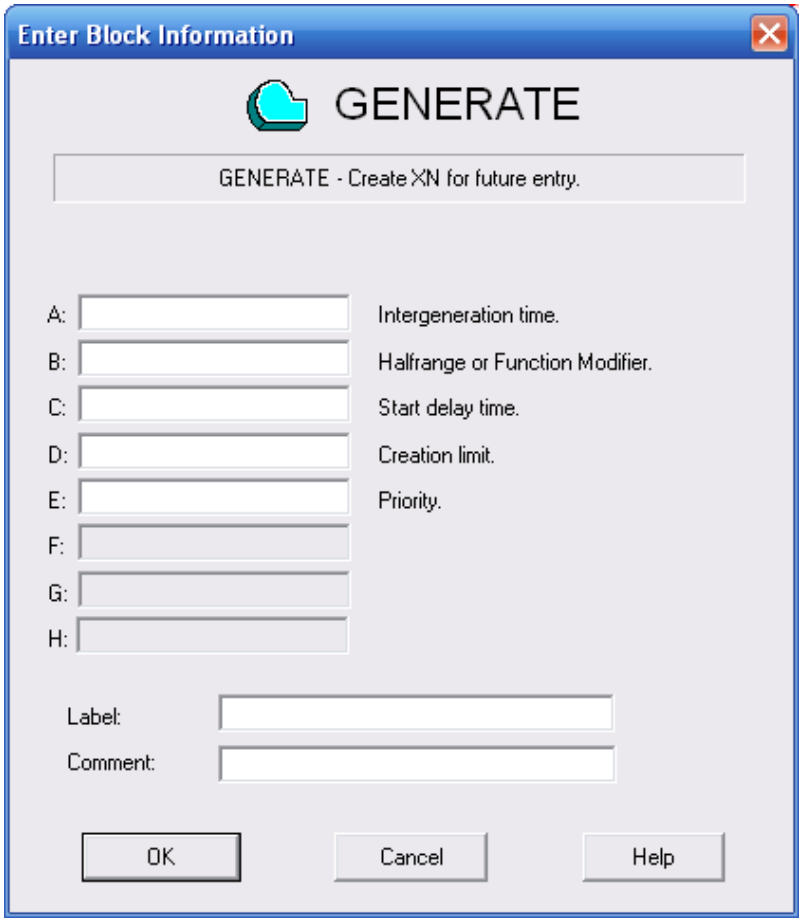


Рисунок 9.14 – Меню введення параметрів команди

Для кращого результату роботи з системою, розглянемо наступний приклад. Отже, скласти програму для моделювання процесу обслуговування автомобілів на заправці бензином протягом 13,3 год, які надходять по рівномірному закону розподілу з інтервалом 8±2 (хв.) одиниці часу, при їх обслуговуванні, яке також описується рівномірним законом, з середнім часом обслуговування - 5±3 (хв.) одиниці. Визначити коефіцієнт використання пристрою обслуговування та середній час займання пристрою одним транзанктом.

Приклад побудованої моделі написаної з використанням мови GPSS зображено на рисунок 9.15.

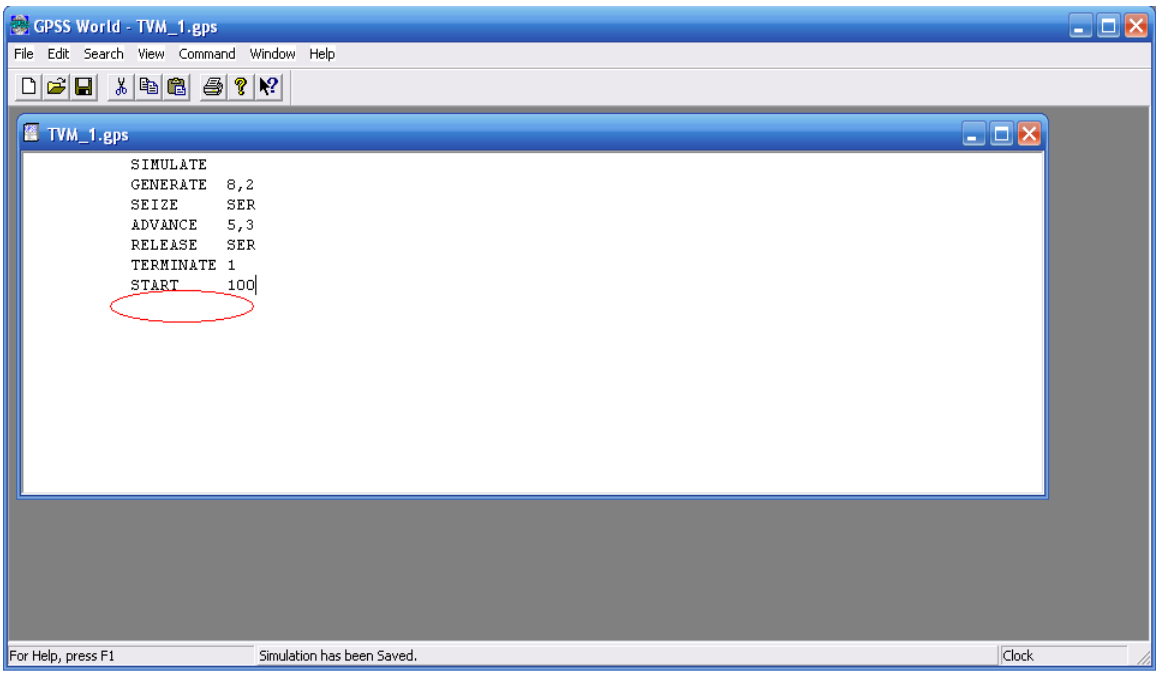


Рисунок 9.15 – Приклад меню з кодом моделі

Для перевірки коректності та правильності моделі необхідно Command→ Create → Simulation.

Результати моделювання зображено на рисунок 9.16 та рисунок 9.17.

З отриманих результатів слідує, що черга відсутня, середній час обслуговування одного транзанкта складає 4,991 хвилини, а коефіцієнт завантаження пристрою обслуговування – 0,613.

Розв’яжемо ще одну задачу з кількома каналами опрацювання даних. Отже, необхідно скласти програму для моделювання процесу проходження запитів від давачів на блок опрацювання, що включає два процесори, які надходять по рівномірному закону розподілу з інтервалом 8±2 одиниці часу, при їх обробці, яка також описується рівномірним законом, з середнім часом обробки – 5±3 одиниці. Запити можуть оброблятися на одному з двох пристроїв; на першому – з часом 5±3 одиниці, на другому – 7±2 одиниці. Обробка на першому пристрої має вищий пріоритет. Запити надходять на обробку до блоку THIS з ймовірністю **0,7** і до блоку THAT з ймовірністю **0,3**.

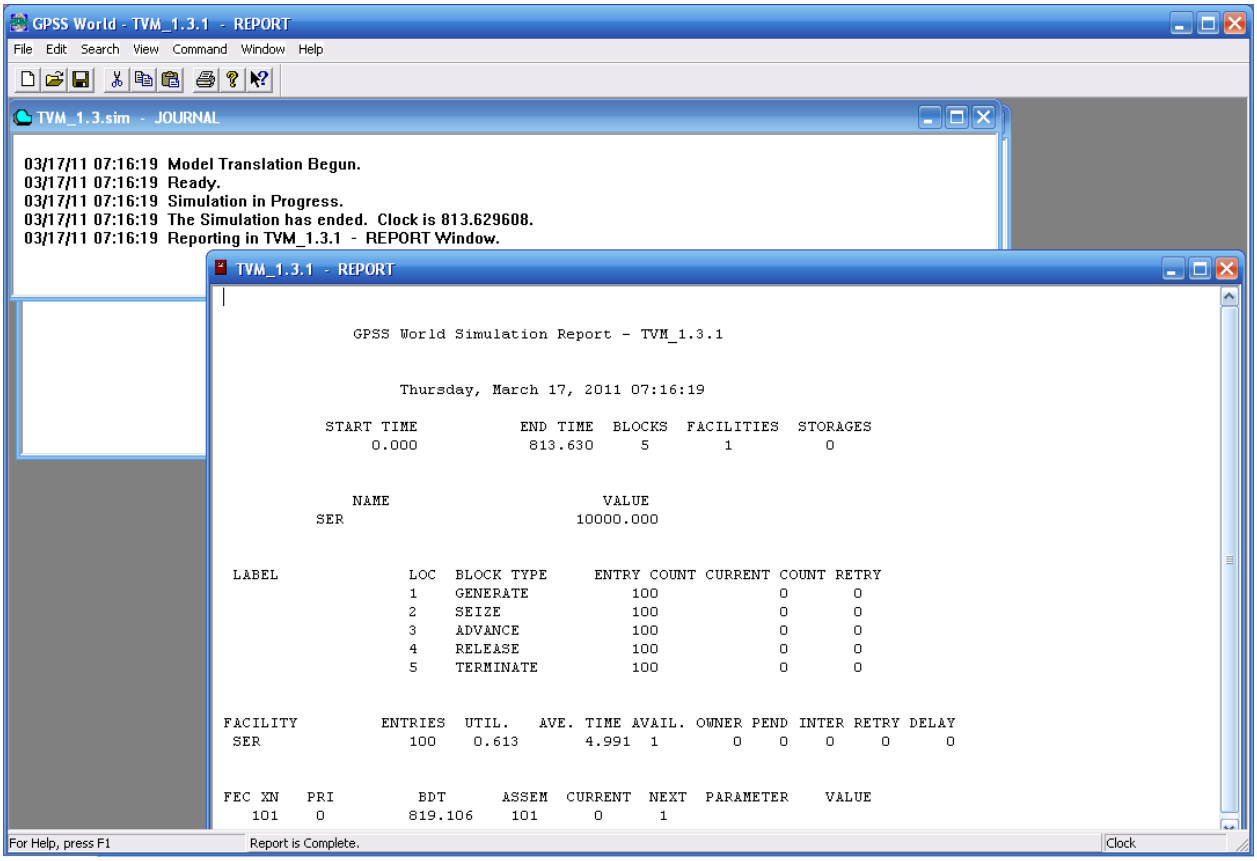


Рисунок 9.16 – Результати дослідження моделі

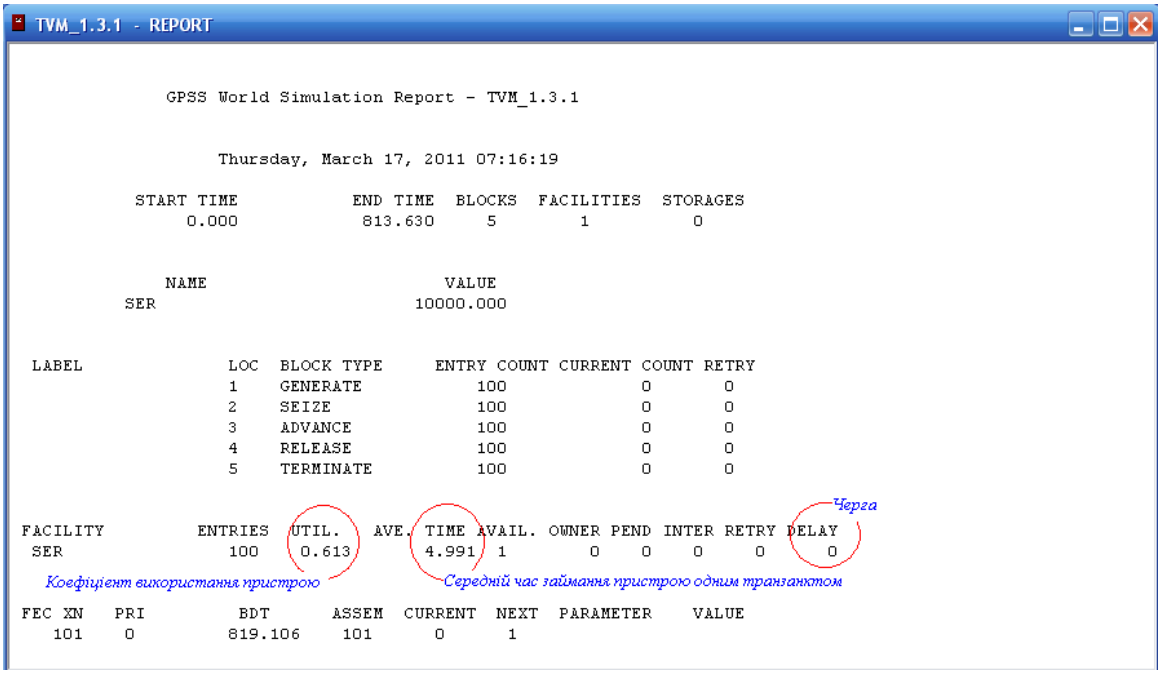


Рисунок 9.17 – Результати дослідження моделі

Отримані результати моделювання зображено на рисунок 9.18 і рисунок 9.19.

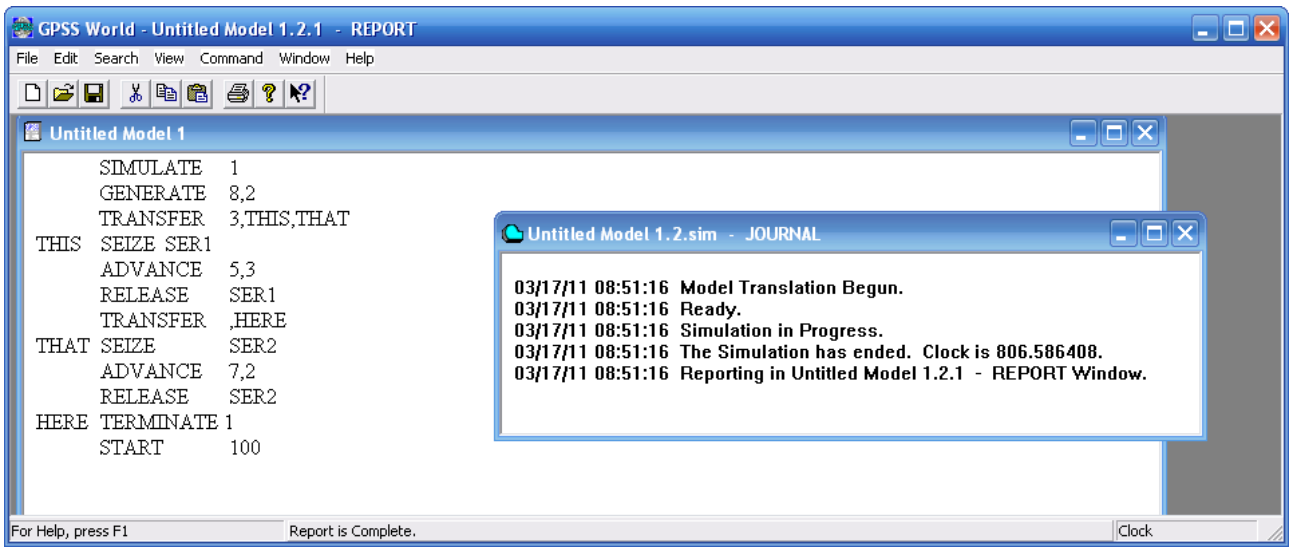


Рисунок 9.18 – Приклад меню з командами моделі

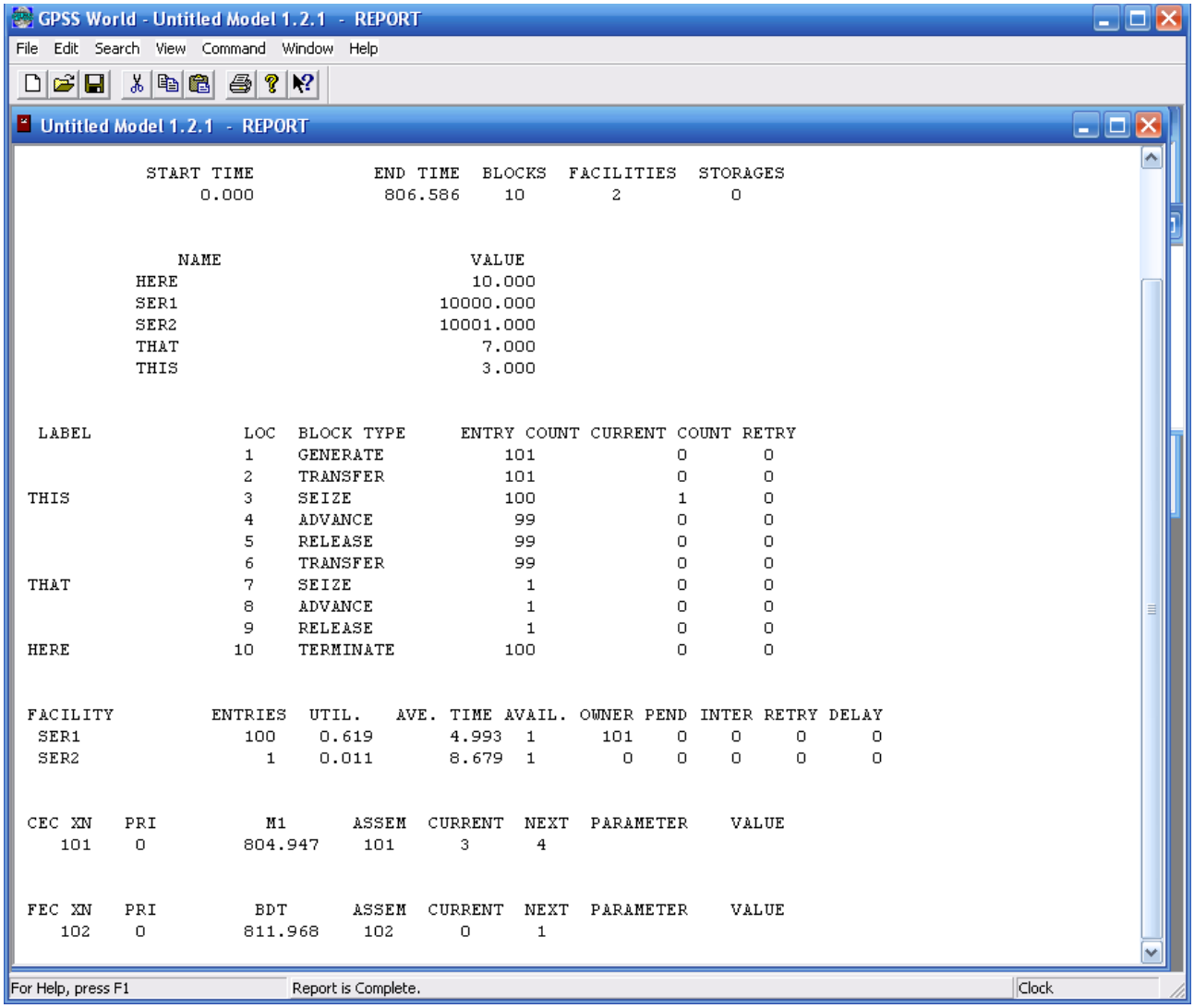


Рисунок 9.19 – Результати дослідження двоканальної системи

Отже, отримані дані дають змогу стверджувати, що час опрацювання на першому процесорі складає 4,992 секунди, а на другому 8,679 сек. Параметр завантаження – для першого 0,619, а для другого – 0,011.

Контрольні запитання до розділу 9

1. Що Ви розумієте під системою масового обслуговування?
2. Які види СМО Ви знаєте?
3. Які основні елементи СМО Ви знаєте?
4. Які основні параметри, що характеризують СМО Ви знаєте?
5. Що Ви розумієте під коефіцієнтом використання?
6. Які основні елементи включає граф стану СМО?
7. Які СМО називаються найпростішими?
8. Що Ви розумієте під стаціонарністю СМО?
9. Який граф називається розміченим?
10. Сформулюйте правило Колмогорова для побудови моделі на основі системи диференційних рівнянь щоб визначити ймовірності стану системи?
11. Що Ви розумієте під ординарністю СМО?
12. Яка особливість одно канальної СМО?
13. Яка особливість одно багатоканальної СМО?
14. За якими критеріями можна класифікувати СМО?
15. В чому особливість мови GPSS?
16. Які основні категорії включає мова GPSS?
17. Яке призначення таймера модельного часу в мові GPSS?
18. Які основні оператори мови GPSS Ви знаєте?
19. Яке призначення системи GPSS Word?
20. Яке призначення оператора GENERATE?
21. Яке призначення оператора SEIZE?
22. Яке призначення оператора TERMINATE?
23. Яке призначення оператора RELEAZE?

24. Яке призначення оператора ADVANCE?

25. Яке призначення оператора TRANSFER?

РОЗДІЛ 10. МОДЕЛЮВАННЯ КОМП’ЮТЕРНИХ СИСТЕМ І МЕРЕЖ

10.1 Проектування системи відеонагляду

Система відеонагляду (СВН) призначена для візуального спостереження і реєстрації (запису) поточних подій, в частині виконання наступних процесів:

- цілодобовий контроль подій, що відбуваються на території в цілому і окремих зонах, що вимагають дистанційного візуального контролю, і запис подій, що відбуваються в зоні спостереження відеокамери.

- можливість збереження архіву на зовнішніх носіях, перегляду і передачі архіву зображень стандартними засобами комп'ютерних технологій;

- можливість паралельного із записом перегляду відеоінформації, обробка і передача зображень по локальній мережі;

- можливість виведення зображення з заданої камери на екран монітора в повноекранному режимі;

- можливість віддаленого перегляду за допомогою сучасних комп'ютерних мереж та Інтернет;

- зниження небезпеки роботи на об'єкті.

Основними цілями створення СВН є:

- захист приміщень, продукції, території об’єкту від можливих вандальних дій, своєчасне реагування та припинення можливих терористичних і диверсійних дій, а також захист людей і майна від інших злочинних посягань;

- облік руху автотранспорту на території об’єкту та персоналу всередині приміщень;

- створення архіву (бази даних відеоінформації), контролю і документування поточних подій, з метою полегшення проведення розшукових, оперативно-слідчих та інших заходів (з пошуку та затримання зловмисників і визначення ступеня провини осіб, які притягуються до відповідальності);

- мінімізація збитку внаслідок вандалізму і крадіжок;

- цілодобовий контроль в режимі реального часу всіх подій, що відбуваються в полі зору зон СВН;

- збільшення продуктивності і ефективності праці, працівників служби охорони об'єкта і обслуговуючого персоналу.

Об'єктом СВН є події, що відбуваються в зонах території об’єкту, виробничих приміщеннях і складах, стоянці особистого автотранспорту співробітників, автотранспорту відвідувачів і клієнтів [117-120].

СВН включає в себе:

- цілодобовий контроль подій, що відбуваються в зонах відеонагляду;

- можливість збереження архіву на зовнішніх носіях, перегляду і передачі архіву зображень стандартними засобами комп'ютерних технологій;

- можливість паралельної із записом перегляду відеоінформації, обробки і передачі зображень по локальній мережі;

- можливість виведення зображення з заданої камери на екран монітора в повноекранному режимі;

- зниження небезпеки роботи на об'єкті.

Вимоги до обчислювального блоку:

- використовувати в якості основного обчислювального блоку одноплатний комп’ютер на базі сучасної елементної бази. Забезпечити зв’язок устаткування з обладнанням з ресурсом зберігання відеоінформації не менше 7 діб в максимально доступній для запису роздільній здатності (944x576).

- для розташування одноплатного комп’ютера використовувати серверну шафу, по габаритах не вище 22U, з можливістю додавання полиць шафи під обладнання;

- забезпечити використання безкоштовного програмного забезпечення, що не потребує продовження ліцензії на протязі всього терміну служби обладнання, повинні використовуватися всі функції роботи обладнання, за допомогою підключення будь-якого віддаленого комп'ютера локальної мережі, а також за допомогою Інтернет;

- забезпечити припустиму вартість системи відеонагляду за рахунок використання безкоштовного програмного забезпечення та сучасної елементної бази.

Відеокамери повинні бути кольорові, цифрові, з високою роздільною здатністю, що задовольняють всі сучасні вимоги і стандарти, що використовується в системах аналогового відеонагляду.

Використовувати 2 відеомонітора, розміру 19 дюймів, для установки:

- в серверній кімнаті;

- для установки в кімнаті охорони на КПП.

Закуплене обладнання повинно бути сертифікованим.

Джерелом живлення для відеокамер, повинні служити існуючі електричні щити і існуюча розводка електроживлення об’єкта.

Установка і монтаж відеокамер повинні здійснюватися на висоті відповідній до вимог до установки камер відеонагляду.

В системі відеонагляду об’єкта повинна бути передбачена система стабілізації напруги, для виключення наявності спотворень в системі електроживлення СВН і виключення додаткових наведень на систему відеонагляду, що спричиняє погіршення сигналу з відеокамер, а також надає вплив на якість запису відеоінформації. Система повинна бути захищена від перепадів напруги в результаті роботи силових агрегатів на виробництві.

Якість відтворення відеозображення з камер відеонагляду, а також якість записи з камер відеонагляду, не повинні змінюватися з впливом зовнішніх факторів, в т.ч. впливу перепадів напруги від силових агрегатів.

Важливою задачею на шляху розробки системи відеонагляду є вибір програмно-апаратної платформи, на основі якої будуть реалізовані усі її ключові функції. В даний час значної популярності набули міні-комп’ютери – так звані одноплатні комп’ютери. Одноплатним називається самодостатній комп'ютер, зібраний на одній друкованій платі, на якій встановлені мікропроцесор, оперативна пам'ять, системи вводу-виводу та інші модулі, необхідні для функціонування комп'ютера. Одноплатні комп'ютери є готовим

рішенням, яке дозволяє виключити етап розробки і виробництва в разі застосування процесорних модулів для створення несучої плати, отже, розробники системи концентруються тільки на програмних питаннях. Прикладами одноплатних комп'ютерів є Odroid, Cubieboard, Raspberry Pi, Orange Pi та ін.

Міні-комп'ютер Odroid-C2 компанії Hardkernel дає змогу створити повноцінний ігровий десктоп на базі Ubuntu 16.04 або Android 5.1 з підтримкою 4K-відео. Odroid-C2 комплектується 64-розрядним чіпом ARM Cortex-A53, що застосовуються в смартфонах, планшетах, десктопах і серверах, і 2 ГБ DDR3 RAM. Графічний процесор Mali-450 здійснює рендеринг потоку 4K відео H.265 з частотою до 60 fps. Для підключення 4K-дисплея обладнано роз'ємом HDMI 2.0. З інших портів в Odroid-C2 є Gigabit Ethernet, чотири USB 2.0, microSD, eMMC, а також GPIO, I2C, I2S і UART, корисні для розробки IoT-обладнання. У варіанті настільного ПК можлива установка радіатора і вентилятора.

Технічні характеристики одноплатного комп'ютера Odroid-C2 наведені у таблиці 10.1.

Таблиця 10.1 – Технічні характеристики комп'ютера Odroid-C2

Параметр	**Характеристика**
Процесор	Amlogic S905, SoC, ARM® Cortex®-A53 (ARM v8), 2.0 GHz, 4 ядра, технологія 28nm
Пам'ять	2 Gbyte DDR3 SDRAM
3D графічний прискорювач	ARM® Mali™-450 OpenGL ES 2.0 / 1.1 (3 x Pixel processors and 2 x Vertex shader processors)
Flash пам'ять	eMMC 5.0 Роз'єм модуля : 8~64 GB eMMC модуль (опціонально) Тримач MicroSD Card : 8 ~128 GB MicroSD UHS-1 (опціонально)
USB 2.0 Host	Високошвидкісний, роз'єм USB A x 4 ports

USB 2.0 Device/ OTG	Високошвидкісний, роз'єм USB A x 4 portsx 1 port
Ethernet/ LAN	10/ 100/ 1000Mbps Ethernet з роз'ємом RJ-45 (підтримка Auto-MDIX)
Видеовихід	HDMI2.0
Аудіовихід	HDMI/ I2S
Вхід відеокамери	Відеокамера USB-CAM 720p (опціонально)
Роз'єм розширення (I/O)	40 контактів (GPIO/ UART/ I2C/ ADC) 7 контактів (I2S)
WiFi	USB IEEE 802.11b/ g/ nWLAN (USB модуль з антеною -опціонально)
Живлення	Джерело живлення 5V 2A (опціонально)
Програмне забезпечення	Ubuntu 16.04 on Kernel 3.14 Android 5.1.x on Kernel 3.14 Повний вихідний код доступний через Github (Hardkernel).
Габарити и вага	85 x 56 x 18 mm приблиз. (Вес : 40 г. без радіатора, 56 г. з радіатором)

Зовнішній вигляд одноплатного міні-комп'ютера Odroid-C2 представлений на рисунку 10.1.

Рисунок 10.1 – Зовнішній вигляд міні-комп'ютера Odroid-C2

Ще одним відомим представником сімейства міні-комп’ютерів, які за своїми характеристиками можуть бути використані в якості програмно-апаратної платформи для пристрою розпізнавання облич є комп’ютер Cubieboard 4 (CC-A80).

Технічні характеристики Cubieboard 4:

- allwinner A80, Octa-Core big-LITTLE A15/A7, 28mm;
- CPU: Arm Cortex A15x4 up to 2.0GHz, A7x4 up to 1.3GHz;
- GPU: PowerVR 64-core G6230;
- 2GB, двоканальна 64-біт 1600MTPS;
- EMMC до 64GB по замовчуванню 8GB швидкість запису/читання 25MB/s;
- micro SD слот карти специфікація V2.0;
- USB 2.0 Хост x 4;
- USB 3.0 OTG x 1;
- HDMI Port A, HDMI V1.4;
- VGA, з підтримкою 1080P;
- 10M/100M/1000M Ethernet;
- Wi-Fi: 2.4G/5.8G, двох-діапазонний 300Mbps;
- bluetooth: BT4.0+EDR;
- IR приймач;
- підтримка батареї годинника реального часу 1220;
- живлення 5V/4A;
- підтримка живлення USB 3.0;
- підтримка батарейного живлення 3.7V Li-Po батареї;
- розмір:146*142*18мм.

Зовнішній вигляд одноплатного міні-комп’ютера Cubieboard 4 представлений на рисунку 10.2.

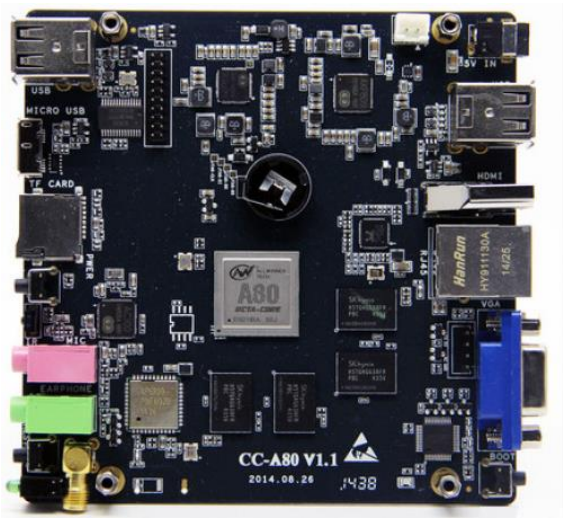


Рисунок 10.2 – Зовнішній вигляд міні-комп’ютера Cubieboard 4 (CC-A80)

Raspberry Pi 3 Model B+ (рисунок 10.3) побудований на базі однокристальної системи Broadcom BCM2837B0, що має в чотири ядра Cortex-A53 (ARMv8). Частота роботи SoC становить 1,4 ГГц, а об'єм оперативної пам'яті стандарту LPDDR2 становить 1 ГБ. Мережеві можливості представлені універсальним бездротовим модулем Wi-Fi 802.11ac/Bluetooth 4.2 і гігабітним контролером Ethernet (максимальна пропускна здатність інтерфейсу 300 Мбіт / с).

Для підключення периферії і джерела виведення зображення передбачено чотири порти USB 2.0, один HDMI, розширена колодка інтерфейсу GPIO на сорок контактів, роз'єми CSI і DSI для підключення сумісних камери і дисплея, універсальний аудіороз'єм/композитний відеовихід і слот для microSD карт. Живлення здійснюється через порт MicroUSB або контакти GPIO. Заявлена підтримка технології Power over Ethernet (PoE).

Спочатку одноплатний комп'ютер Raspberry Pi розроблявся як бюджетна система для навчання інформатиці в бідних країнах. Проте, успіх Raspberry Pi перевершив найсміливіші очікування авторів, і комп'ютер став дуже популярний у всьому світі. На сьогодні продано понад 19 млн оригінальних Raspberry Pi всіх модифікацій [67].

Рисунок 10.3 – Зовнішній вигляд комп’ютера Raspberry Pi 3 Model B+

Orange Pi – це серія open-source одноплатних комп'ютерів компанії Shenzhen Xunlong Software CO. Підтримує операційні системи: Android 4.2, Android 4.4, Ubuntu, Debian, Fedora, Raspbian, ArchLinux, OpenSUSE, OpenWrt і інші ОС. Процесори: AllWinner A20 SoC і AllWinner H3, і від 512 до 2 Гб пам'яті DDR3 SDRAM.

Orange Pi PC PLUS (рисунок 10.4) – це багатофункціональний і потужний одноплатний міні-комп'ютер з відкритим вихідним кодом. Він використовує Allwinner H3 SOC, і має 1 ГБ DDR3 SDRAM. Він може працювати під управлінням Android 4.4, Ubuntu, Debian, rasberry Pi. На платі комп'ютера є 8Гб flash EMMC пам'яті програм і слот для підключення microSD карти пам'яті. Відмінною особливістю комп'ютера є те, що при збереженні функціональності і сумісності комп'ютер вийшов дуже малих розмірів. Міні-комп'ютер дуже вимогливий до якості живлення, тому рекомендується ретельно підходити до цього питання і використовувати тількі якісні та надійні блоки живлення.

Рисунок 10.4 – Зовнішній вигляд міні-комп’ютера Orange Pi PC PLUS

Технічні характеристики Orange Pi PC PLUS:

- процесор: H3 четирьохядерний процесор Cortex-A7 H.265/HEVC 4 K 1.6ГГц;
- графічний процесор: Mali400MP2 GPU @ 600 МГц з підтримкою OpenGL ES 2.0;
- оперативна пам’ять: SDRAM 1 ГБ DDR3 (використовується сумісно з GPU);
- вбудована пам’ять програм: 8 ГБ EMMC Flash пам’ять;
- зовнішня пам’ять програм: TF карта (Макс. 64 ГБ);
- мережевий інтерфейс: 10/100/1000 M Ethernet RJ45;
- безпровідний інтерфейс: WI-FI Realtek RTL8189ETV, IEEE 802.11 b/g/n;
- відео вхід: CSI роз’єм камери з підтримкою 8-бітних YUV422 CMOS камер CCIR656 протоколу для NTSC і PAL;
- роздільна здатність відео-захоплення: до 1080p @ 30fps;
- аудіо вхід: вбудований мікрофон;
- відео вихід: HDMI з HDCP і підтримкою HDMI CEC та інтегрований інтерфейс CVBS. Виходи можна використовувати одночасно;
- аудіо вихід: 3.5 мм роз’єм і HDMI;
- живлення: вхід постійного струму;
- напруга живлення: 5В;
- USB інтерфейс: три USB 2.0 HOST порти і один USB 2.0 OTG;
- кнопки: Скидання, Живлення, Відновлення і Uboot;
- низькорівнева периферія: 40 Pins GPIO інтерфейс, сумісний ї Raspberry Pi B+;
- світлодіоди: індикатор живлення та індикатор стану;
- підтримувані OS: Android, Lubuntu, Debian, Raspbian;
- розмір: 85 мм × 55 мм;
- вага: 38 г.

В якості керуючого пристрою у розробленій системі відеонагляду буде використано одноплатний комп’ютер. Реалізована система повинна здійснювати моніторинг приміщення на рух та виконувати певний сценарій дій при його виявленні. Зокрема одразу після виникнення руху система повинна автоматично вмикати відеокамеру в режим запису на визначений проміжок часу. Під час запису відеофайл зберігається на віддаленому сервері через комп’ютерну мережу. Поруч із записом відео, при виявленні руху, має відбуватися автоматичне оповіщення про ситуацію, шляхом автоматичного надсилання повідомлення відповідного змісту на електронну пошту власнику.

Структура розробленої системи відеонагляду показана на рисунок 10.5.

Рисунок 10.5 – Загальна структура підсистеми відеонагляду на основі міні-комп’ютера

Структурний аналіз має на меті виявити статичні характеристики системи шляхом поділу її на елементи й підсистеми різного рівня та аналізу зв’язків між ними. Об’єктами структурного аналізу є варіанти структур системи, які формуються у процесі її декомпозиції. Сутністю структурного синтезу є побудова (реорганізація) системи, яка має бажані властивості. Структурний синтез проводиться з метою обґрунтування множини елементів, зв’язків, відношень, що забезпечать у сукупності максимальну відповідність системи заданим вимогам. Розв’язок задачі синтезу структур можна представити у вигляді І-АБО дерева.

Ключовими та обов’язковими елементами системи відеонагляду (СВ0) є одноплатний комп’ютер (ОК1), відеокамера (В1), давач руху (ДР1) та зв’язок (З1).

Приклад синтезу альтернативи структури системи відеонагляду:

СВ0 = ОК1 + В1 + ДР1 + З1

Альтернатива І:

ОК1 = RaspberryPi2 + Linux3 + Raspbian4 + Whezzy5 + ООП3 + Python4

В1 = USB2 + Автофокус3

ДР1 = IR2

З1 = Безпровідний2 + WiFi_адаптер2 + Роутер3

Альтернатива ІІ:

…

Множина згенерованих альтернатив структури системи відеонагляду у вигляді І-АБО дерева показана на рисунку 10.6.

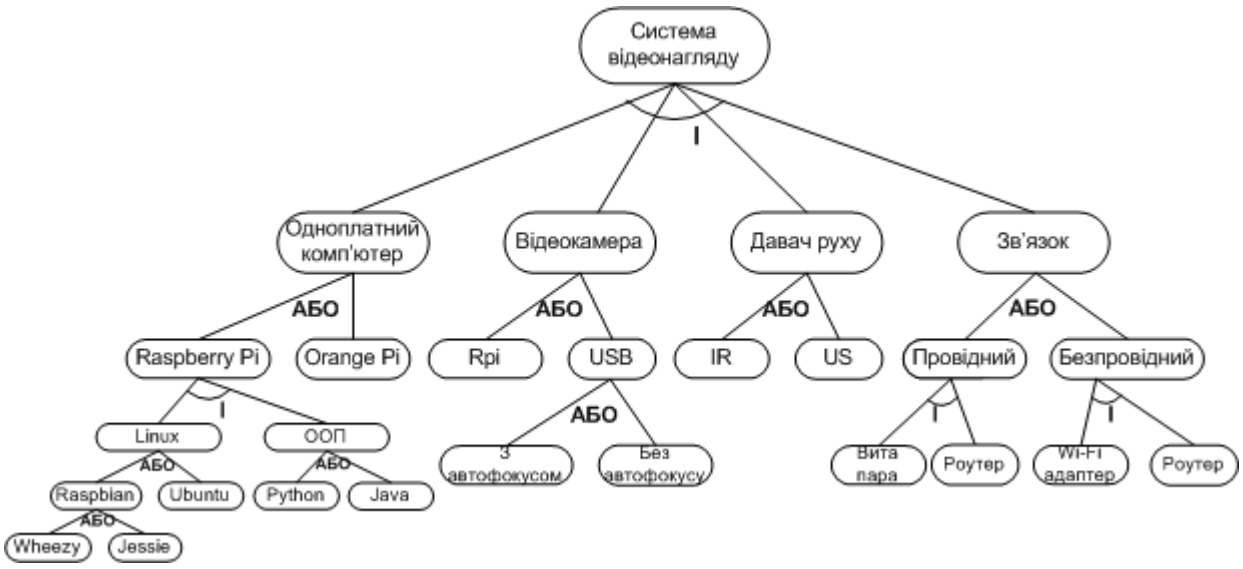


Рисунок 10.6 – Синтез множини структур з використанням І-АБО дерева

Графічне зображення Альтернативи І наведено на рисунку 10.7.

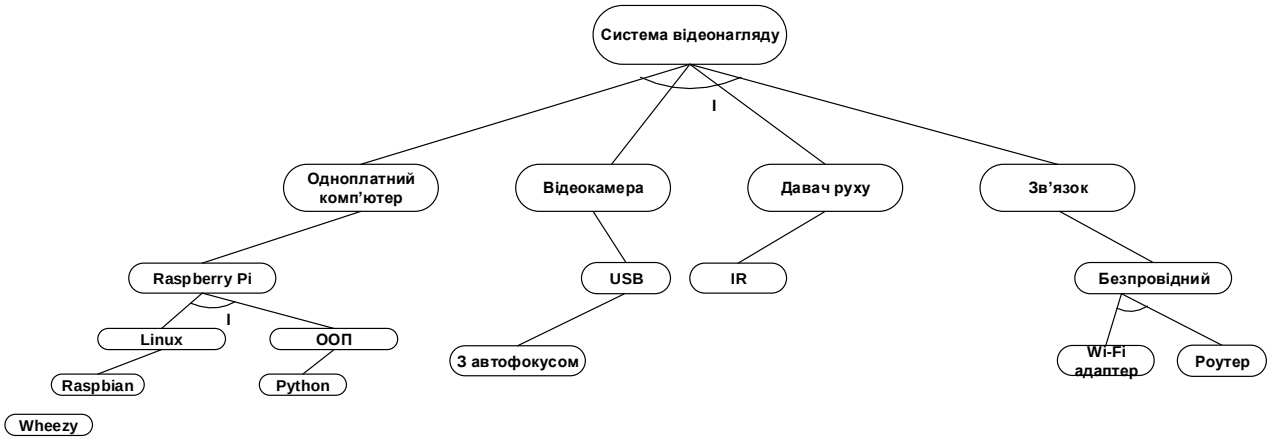


Рисунок 10.7 – Графічне зображення однієї з альтернатив структури системи

Розроблення складних технічних систем неможливо реалізувати без спеціальних методів та сучасних засобів проектування. Процес автоматизованого проектування передбачає розв'язання задач синтезу, оптимізації та аналізу і базується на технічному завданні.

Для знаходження оптимального рішення проведемо автоматизований пошук альтернатив за допомогою програми Optimizer.

Для оптимізації пошуку оптимальної структури системи відеонагляду було згенеровано певну кількість альтернатив структури і задано кількість критеріїв. Альтернатив структури системи згенеровано 20, а критеріїв обрано 3 – ціна, надійність, швидкодія.

Після запису критеріїв переходимо до основного вікна програми де ми задаємо характеристики усіх можливих альтернатив. Підбираємо вагові коефіцієнти, виходячи з таких міркувань:

- ціна найбільш вагомий коефіцієнт, тому її коефіцієнт рівний 0.5;
- також досить важливою є надійність розроблюваної системи, тому коефіцієнт надійності встановлено 0.35;
- третім критерієм є швидкодія системи, на даний критерій призначено коефіцієнт 0.15.

Для пошуку оптимального рішення обрано адитивний критерій пошуку, який передбачає врахування усіх параметрів. Результат роботи програми Optimizer показано на рисунку 10.8.

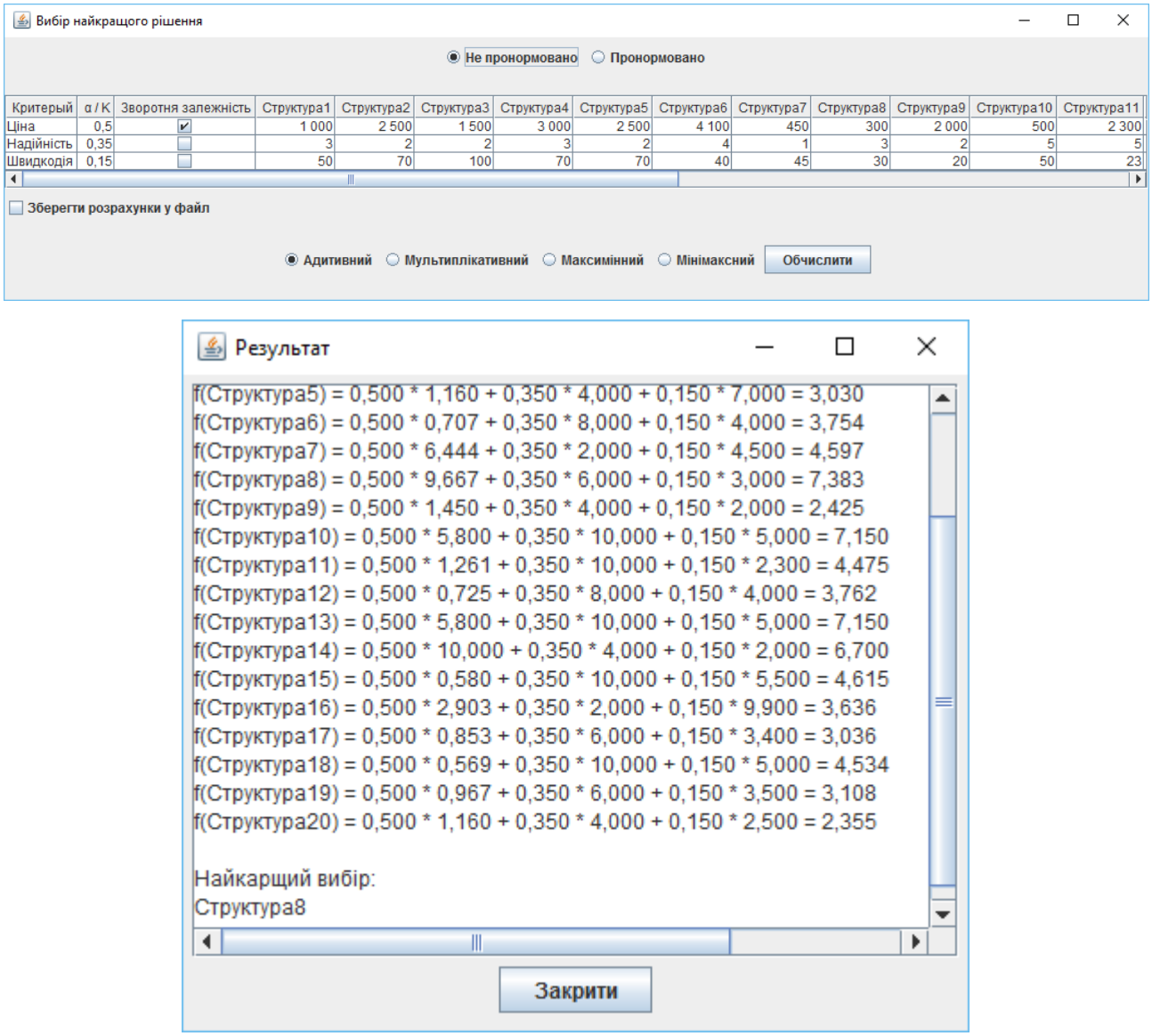


Рисунок 10.8 – Пошук оптимального варіанту структури з адитивного критерію

Оптимізована структура роботи системи передбачає проведення аналізу проектних рішень. Для розв’язання задач даного класу використано модель на основі теорії мереж Петрі.

Зокрема на рисунку 10.9 наведено структурну модель для аналізу роботи модуля системи відеонагляду. Структурна модель реалізована в середовищі

Woped. Граф досяжності станів у випадку спрацювання давача руху зображено на рисунку 10.10.

Результати, які були отримані після проведення моделювання дають змогу стверджувати, що тупики в моделі відсутні, мережа Петрі є живою й усі стани досяжні. Відповідно, можна зробити висновок про те, що розроблена структура системи відеонагляду забезпечує технічне завдання.

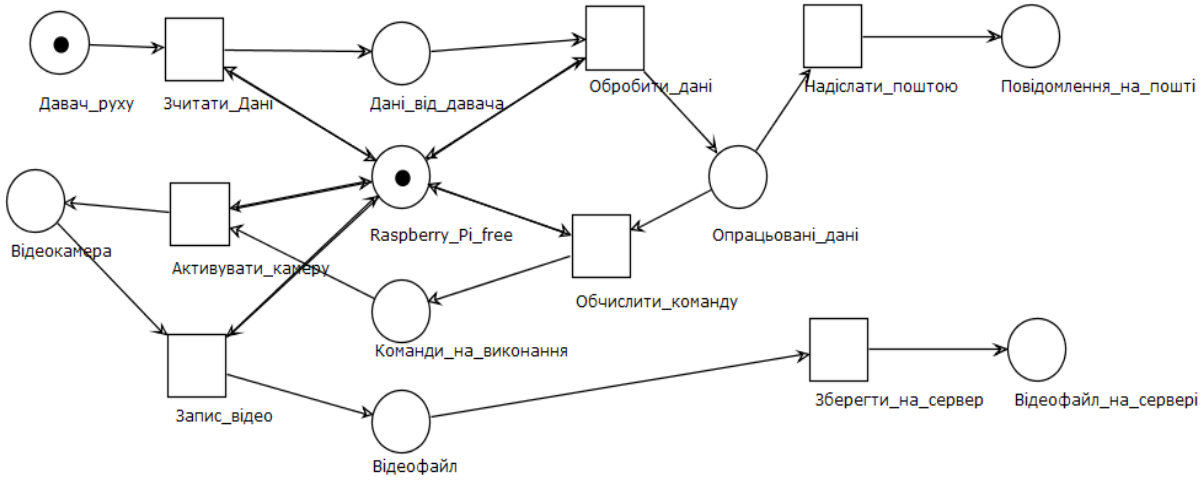


Рисунок 10.9 – Структурна модель підсистеми відеонагляду на основі мереж Петрі

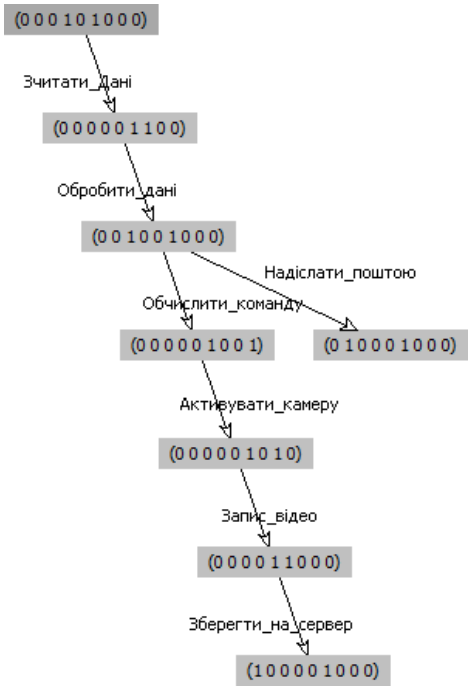


Рисунок 10.10 – Граф досяжності станів (при спрацюванні давача руху)

Роботу системи відеонагляду реалізовують програми, написані на Bash (поштовий скрипт, скрипт монтування) та Python (скрипт запису відео). Макет реалізованої системи відеонагляду показано на ристунку 10.11.

Зчитуванням показів давача руху, увімкненням камери та запуском поштового bash-скрипта займається python-скрипт. Поштовий скрипт запускається при виникненні руху і реалізує надсилання повідомлення на електронну пошту. Для його роботи використовується SMTP-сервер (поштовий). Bash-скрипт монтування реалізовує точку монтування і таким чином забезпечує можливість збереження відеофайлу одразу на жорсткий диск сервера. Усі скрипти та програмне забезпечення функціонують у середовищі ОС Raspbian Wheezy, під управлінням якої працює одноплатний комп'ютер Raspberry Pi. Відеофайли зберігаються з іменем поточної дати і часу.

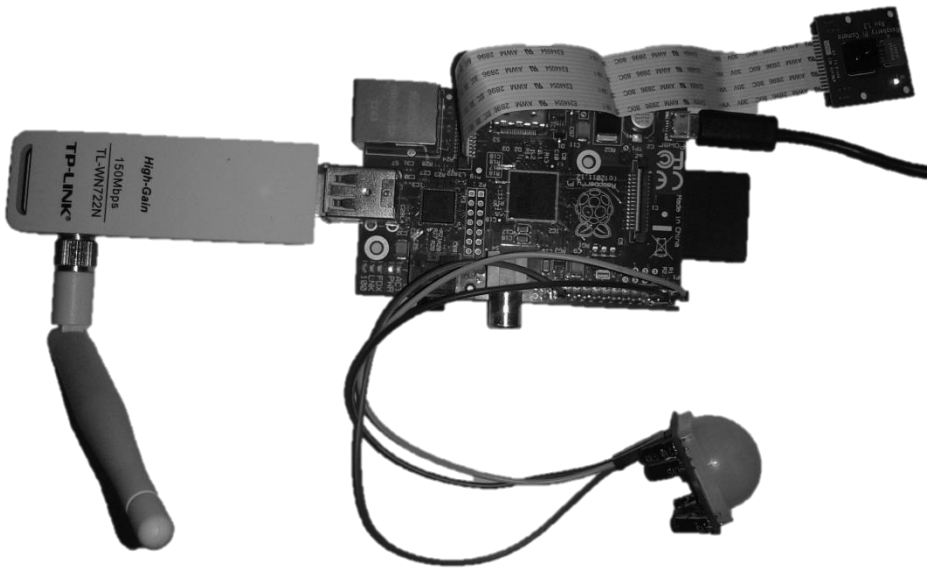


Рисунок 10.11 – Макет розробленої системи відеонагляду на основі одноплатного комп'ютера Raspberry Pi

В якості давача руху використано HC-SR501. Давач являє собою модуль, що складається із самого PIR-давача (Pyroelectric (Passive) InfraRed sensor) і схеми керування.

Для відеозйомки обрано модуль п'ятимегапіксельної камери від команди розробників комп'ютера Raspberry Pi. Камера підключається шлейфом до одного із спеціальних роз'ємів CSI на платі.

Для повноцінної роботи розробленої підсистеми вона має входити до складу мережі, що має вихід у Інтернет. Включення плати Raspberry Pi до мережі реалізовується через Ethernet-кабель або USB Wi-Fi модуль. В даному випадку використано TP-LINK Wi-Fi-адаптер.

Після реалізації системи відеонагляду, до складу якої увійшли плата Raspberry Pi з блоком живлення, давач руху та відеокамера, було здійснено тестування та верифікацію системи.

Після встановлення та налаштування, розроблена система функціонує за таким алгоритмом:

- моніторинг руху. Якщо в приміщенні, в якому відбувається моніторинг, давач руху реєструє рух, то перейти до кроку 2;
- здійснити оповіщення про подію на електронну поштову скриньку через Інтернет;
- здійснити запис 10-ти секундного відеофайлу;
- зберегти відеофайл на сервері через локальну комп’ютерну мережу.

10.2 Проектування мережі в Packet Tracer

Розробимо план IP адресації.

Таблиця 10.2 – План IP адрес мережі

№	Кінцевий пристрій	IP адреса
1	2	3
1	ВРС1	192.168.2.2
1	2	3

2	ВРС2	192.168.2.3
3	ВРС3	192.168.2.4
4	ВРС4	192.168.2.5
5	PC1	192.168.3.2
6	PC2	192.168.3.3
7	PC3	192.168.3.4
8	PC4	192.168.3.5
	Шлюз за замовчуванням	192.168.2.1
	Шлюз за замовчуванням	192.168.3.1

Встановлюємо перший комутатор Cisco 2960, комп'ютери. Використовуючи прямий кабель підключаємо послідовно комп'ютери до комуатора (рисунок 10.12).

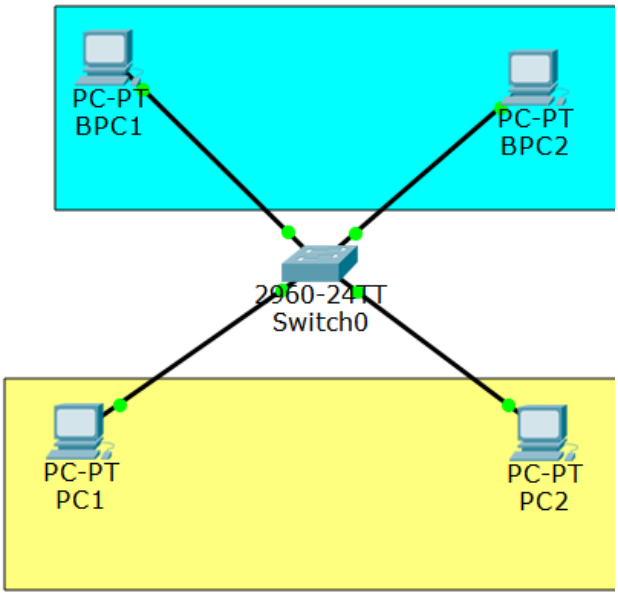


Рисунок 10.12 – Схема з одним комутатором

Станції ВРС1 і ВРС2 належать до одного сегмента бухгалтерії. А PC1 і PC2 належать до сегмента звичайних користувачів. Позначимо їх кольоровими прмякутниками. Розділимо трафік двох сегментів.

Налаштуємо перший комутатор Switch0.

Відкриваємо консоль. Переходимо в привілейований режим. Всі порти по замовчуванню в першому vlan.

```
Switch>en

Switch#show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
1    default                           active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                 Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                 Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                 Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                 Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                 Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                 Gig0/1, Gig0/2
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
1005 trnet-default                     active
```

Визначаємо vlan для станцій бухгалтерії. Створюємо vlan 2

```
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.


Switch(config)#vlan 2
Switch(config-vlan)#
Switch(conf ig-vlan)#name finance
Switch(config-vlan)#exit
```

```
Switch(config)#
```

Додаємо інтерфейси у новий vlan.

```
Switch(config)#interface fastEthernet 0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#exit

Switch(config)#int fa0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 2
Switch(config-if)#end
```

Перевіряємо налаштування vlan на комутаторі

```
Switch#show vlan brief

VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------

1    default                           active    Fa0/3, Fa0/4, Fa0/5, Fa0/6
                                                 Fa0/7, Fa0/8, Fa0/9, Fa0/10
                                                 Fa0/11, Fa0/12, Fa0/13, Fa0/14
                                                 Fa0/15, Fa0/16, Fa0/17, Fa0/18
                                                 Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                                 Fa0/23, Fa0/24, Gig0/1, Gig0/2

2    finance                           active    Fa0/1, Fa0/2
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
```

1005 trnet-default active

Створимо vlan 3 для другого сегмента

```
Switch(config)#vlan 3
Switch(config-vlan)#
Switch(conf ig-vlan)#name visitors
Switch(config-vlan)#exit
Switch(config)#
```

Додаємо інтерфейси у новий vlan.

```
Switch(config)#int fa0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#exit

Switch(config)#int fa0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 3
Switch(config-if)#end
```

Переглянемо створені vlan

```
2    finance                 active    Fa0/1, Fa0/2
3    visitors                active    Fa0/3, Fa0/4
```

Налаштуємо IP адреси робочих станцій першої групи враховуючи що перший номером в підмережі буде маршрутизатор відповідно таблиці 10.2.

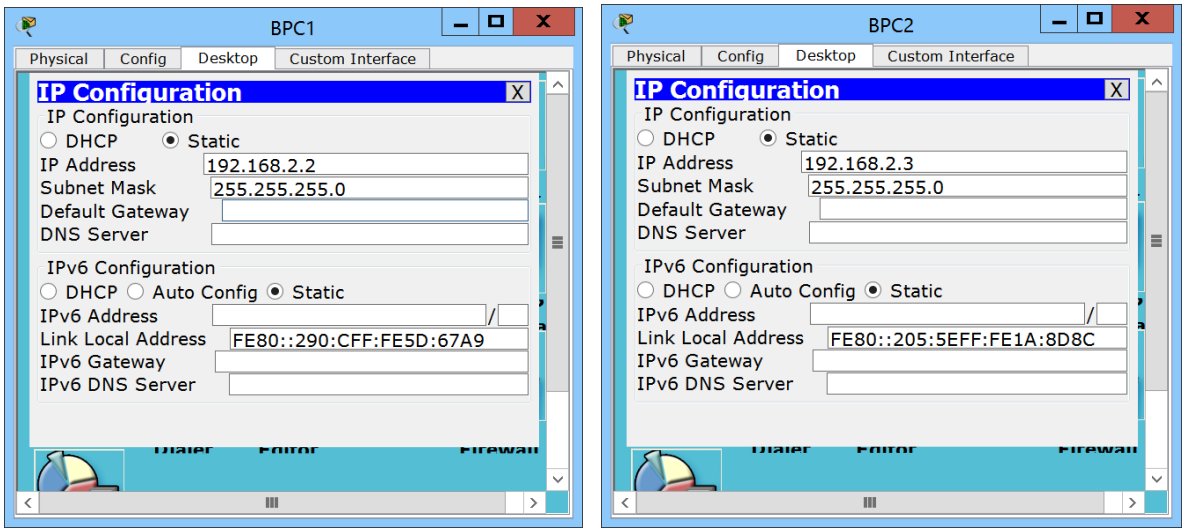


Рисунок 10.13 - Налаштування робочих станцій

Перевіримо з’єднання другої робочої станції BPC2 із першою BPC1.

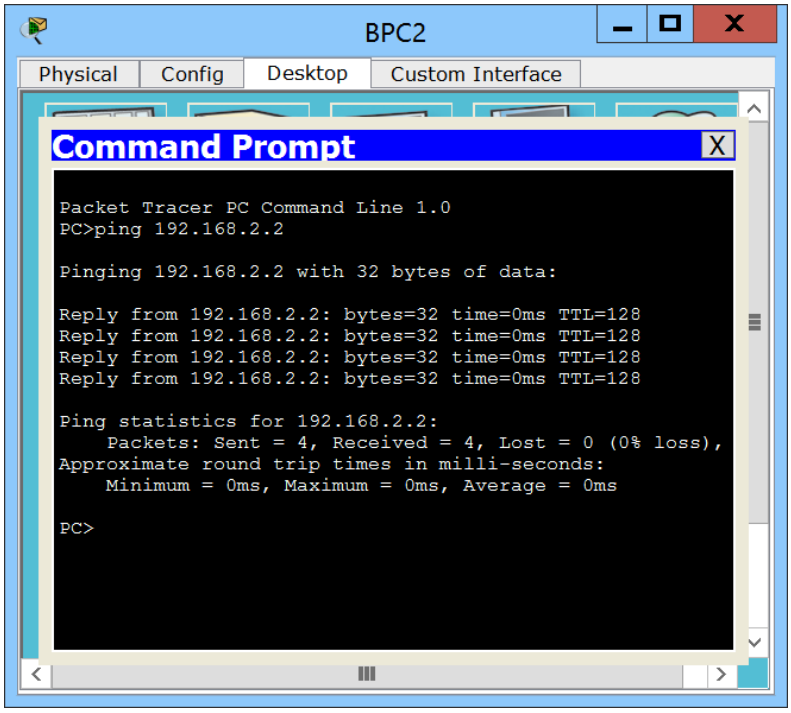


Рисунок 10.14 - Тестування з’єднання

Налаштуємо IP адреси робочих станцій другої групи

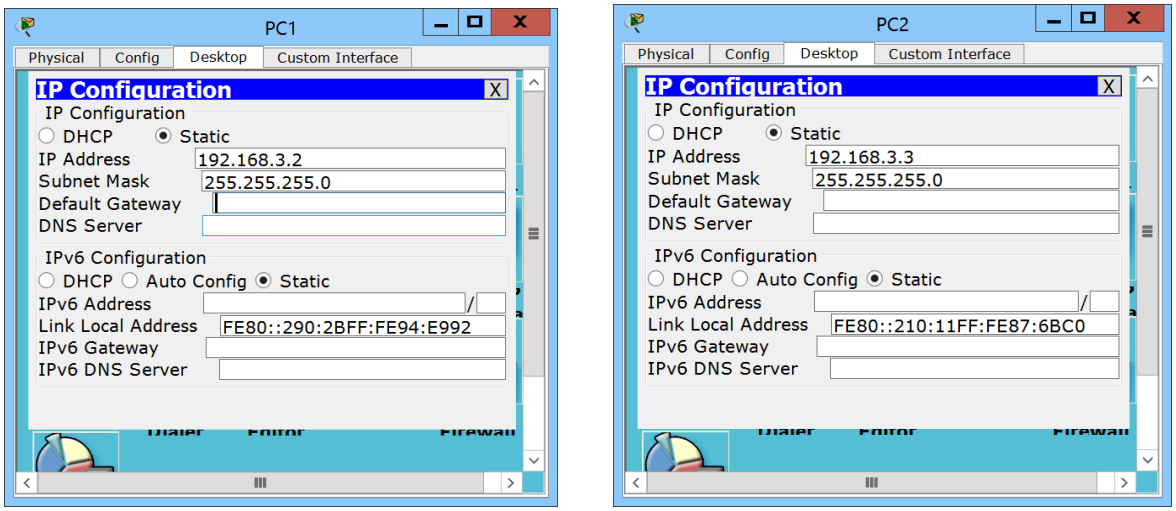


Рисунок 10.15 - Налаштування робочих станцій

Перевіримо чи ізольовано трафік сегментів.

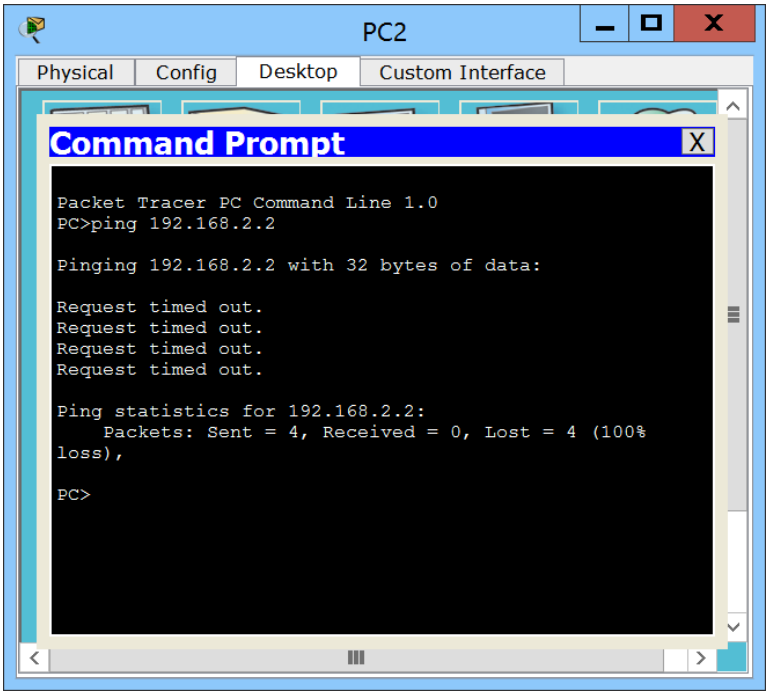


Рисунок 10.16 - Тестування з’єднання

Перегялнемо таблицю MAC адрес

Vlan Mac Address Type Ports

---- ----------- -------- -----

2 0005.5e1a.8d8c DYNAMIC Fa0/2

2 0090.0c5d.67a9 DYNAMIC Fa0/1

Додамо в проект другий комутатор із робочими станціями шляхом копіювання уже налаштованого. З’єднаємо гігабітні інтерфейси комутаторів кроскабелем.

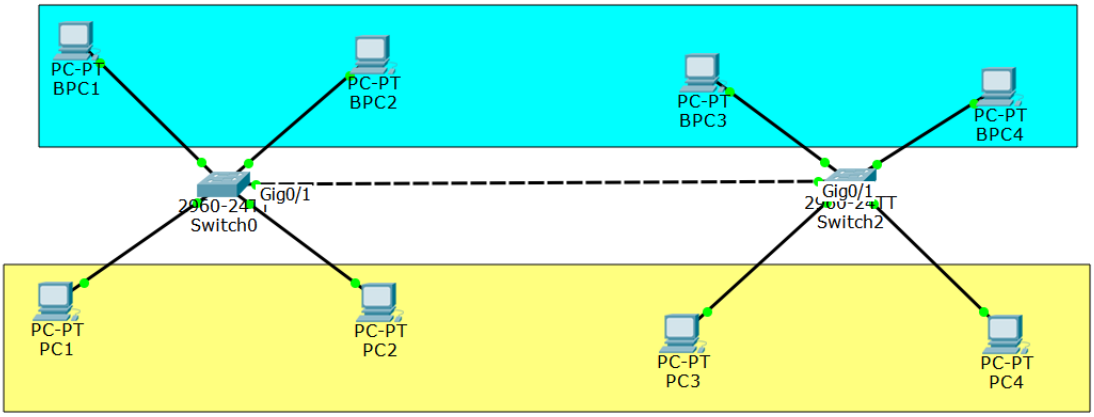


Рисунок 10.17 – Схема мережі

Перевіримо конфігурацію другого комутатора.

```
Interface FastEthernet0/1
 switchport access vlan 2
 switchport mode access
!
interface FastEthernet0/2
 switchport access vlan 2
 switchport mode access
!
interface FastEthernet0/3
 switchport access vlan 3
 switchport mode access
!
interface FastEthernet0/4
 switchport access vlan 3
```

switchport mode access

Як бачимо потрібні налаштування збереглись. Порти доступу налаштовані. Налаштуємо на 1 комутаторі trunk порт gig0/1 і дозволимо йому пропускати VLAN 2 і 3.

Switch#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#int gig0/1

Switch(config-if)#switchport mode trunk

Switch(config-if)#

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up

Switch(config-if)#switchport trunk allowed vlan 2,3

Switch(config-if)#exit

Switch(config)#

Налаштуємо 2 комутатор транк порт gig0/1 і дозволимо йому пропускати VLAN 2 і 3

Switch>en

Switch#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#int gig0/1

Switch(config-if)#switchport mode trunk

Switch(config-if)#switchport trunk allowed vlan 2,3

Switch(config-if)#exit

Якщо переглянути VLAN інформацію то побачимо що перший гігабіт порт пропав із VLAN1.

```
VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
1    default                           active    Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                                 Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                                 Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                                 Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                                 Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                 Gig0/2
2    finance                           active    Fa0/1, Fa0/2
3    visitors                          active    Fa0/3, Fa0/4
```

Первіримо конфігурацію з допомогою show run.

```
Interface GigabitEthernet0/1
 switchport trunk allowed vlan 2-3
 switchport mode trunk
!
interface GigabitEthernet0/2
```

Збережемо налаштування

```
Switch#wr mem
```

Протестуємо з’єднання на комп’ютері ВРС3 із комютерами в даній робочій групі.

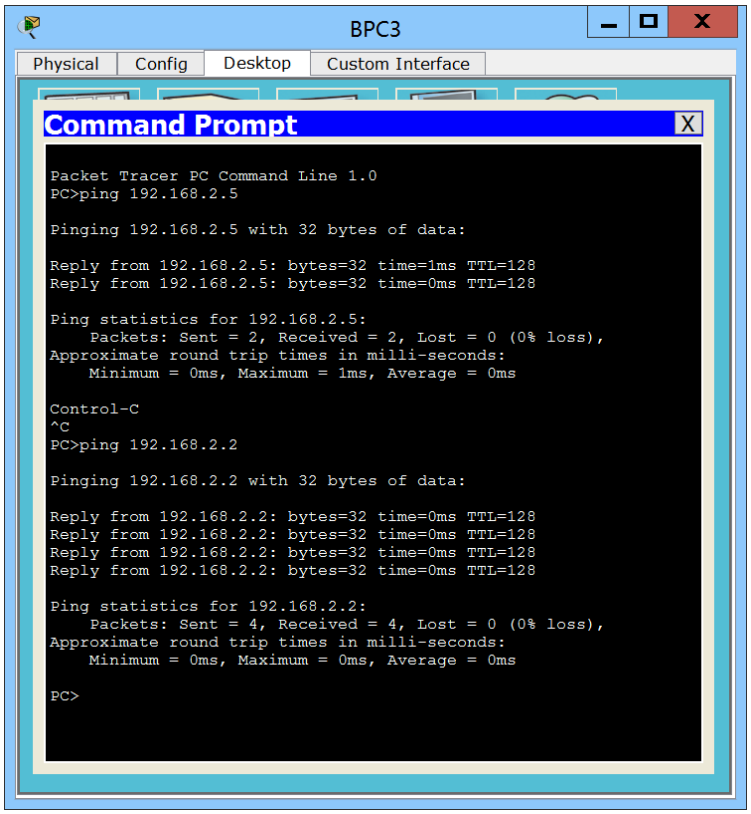


Рисунок 10.18 – Тестування зєднання

Комутатори налаштована вірно. Trunk порт працює.

Додаємо маршрутизатор Cisco 1841.

Таблиця 10.3 - План підключень

Підключення	Router0	Switch0
1	Fa0/1	Fa0/24

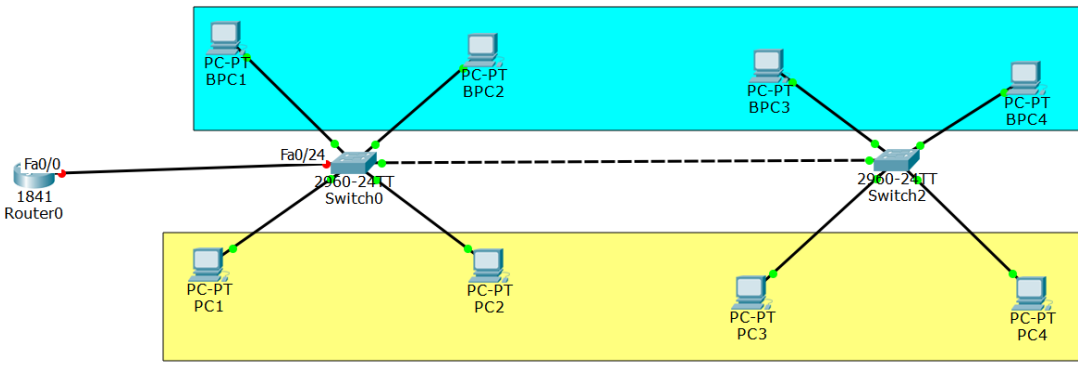


Рисунок 10.19 – Остаточна схема мережі

Налаштування додаткового транка на Switch0.

Switch>en

Switch#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#int fa0/24

Switch(config-if)#switchport mode trunk

Switch(config-if)#switchport trunk allowed vlan 2,3

Switch(config-if)#end

Switch#

Switch#wr mem

Building configuration...

[OK]

Підключаємося в консоль маршрутизатора Router0

Would you like to enter basic management setup? [yes/no]: n

First, would you like to see the current interface summary? [yes]:y

Current interface summary

Interface	IP-Address	OK?	Method	Status	Protocol
FastEthernet0/0	unassigned	YES	manual	administratively down	down
FastEthernet0/1	unassigned	YES	manual	administratively down	down
Vlan1	unassigned	YES	manual	administratively down	down

Configuring global parameters:

```
  Enter host name [Router]: router0
  The enable secret is a password used to protect access to
  privileged EXEC and configuration modes. This password, after
  entered, becomes encrypted in the configuration.
  Enter enable secret: router0

  The enable password is used when you do not specify an
  enable secret password, with some older software versions, and
  some boot images.
  Enter enable password: router0
% Please choose a password that is different from the enable secret
  Enter enable password: router00

  The virtual terminal password is used to protect
  access to the router over a network interface.
  Enter virtual terminal password: router000
Configure SNMP Network Management? [no]:n
Configuring interface parameters:

Do you want to configure Vlan1  interface? [no]:n
Do you want to configure FastEthernet0/0  interface? [no]:n
Do you want to configure FastEthernet0/1  interface? [no]:n

The following configuration command script was created:

!
hostname router0
enable secret 5 $1$mERr$pbNer2V864ZdOTnUYdjKo1
enable password router00
line vty 0 4
```

password router000
!
interface Vlan1
 shutdown
 no ip address
!
interface FastEthernet0/0
 shutdown
 no ip address
!
interface FastEthernet0/1
 shutdown
 no ip address
!
end

[0] Go to the IOS command prompt without saving this config.
[1] Return back to the setup without saving this config.
[2] Save this configuration to nvram and exit.

Enter your selection [2]: 0
% You can enter the setup, by typing setup at IOS command prompt

Press RETURN to get started!

Включимо інтерфейс **fa0/0**
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#**int fa0/0**

Router(config-if)#**no shutdown**

Router(config-if)#

%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit

Налаштовуємо субінтферфейси для одного фізичного інтерфейса. Кожному субінтерфейсу відповідами свій vlan. Також пропишемо IP адресу субінтферфейса.

Router(config)#int fa0/0.2

Router(config-subif)#

%LINK-5-CHANGED: Interface FastEthernet0/0.2, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0.2, changed state to up

Router(config-subif)#encapsulation dot1Q 2

Router(config-subif)#ip address 192.168.2.1 255.255.255.0

Router(config-subif)#no shutdown

Router(config-subif)#

Router(config-subif)#exit

Router(config)#

Router(config)#int fa0/0.3

Router(config-subif)#

%LINK-5-CHANGED: Interface FastEthernet0/0.3, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0.3, changed state to up

```
Router(config-subif)#encapsulation dot1Q 3
Router(config-subif)#ip address 192.168.3.1 255.255.255.0
Router(config-subif)#no shutdown
Router(config-subif)#
Router(config-subif)#exit
Router(config)#
```

Передивимося конфігурацію

```
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
!
interface FastEthernet0/0.2
 encapsulation dot1Q 2
 ip address 192.168.2.1 255.255.255.0
!
interface FastEthernet0/0.3
 encapsulation dot1Q 3
 ip address 192.168.3.1 255.255.255.0
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
```

shutdown

!

Пінгуємо шлюз із робочої станції 1 комутатора (рисунок 10.20).

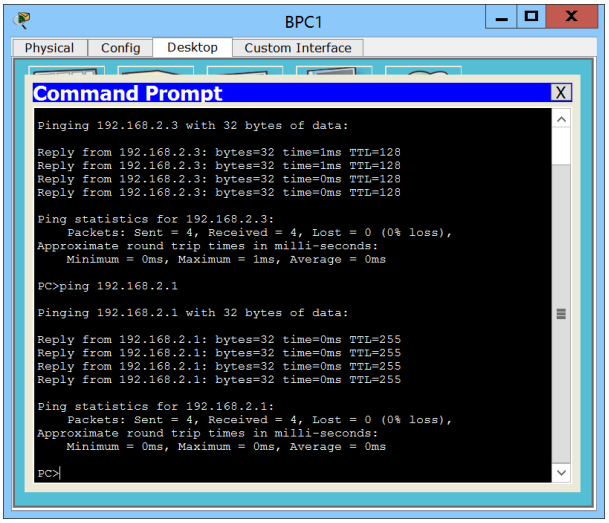


Рисунок 10.20 – Тестування зєднань

Так само пінгуємо з 4 робочої станції. З’єднання є.

Без прописаного шлюзу станції з різних робочих груп не пінгуються. Якщо прописати шлюзи по замовчуванню то станції пінгують одна одну.

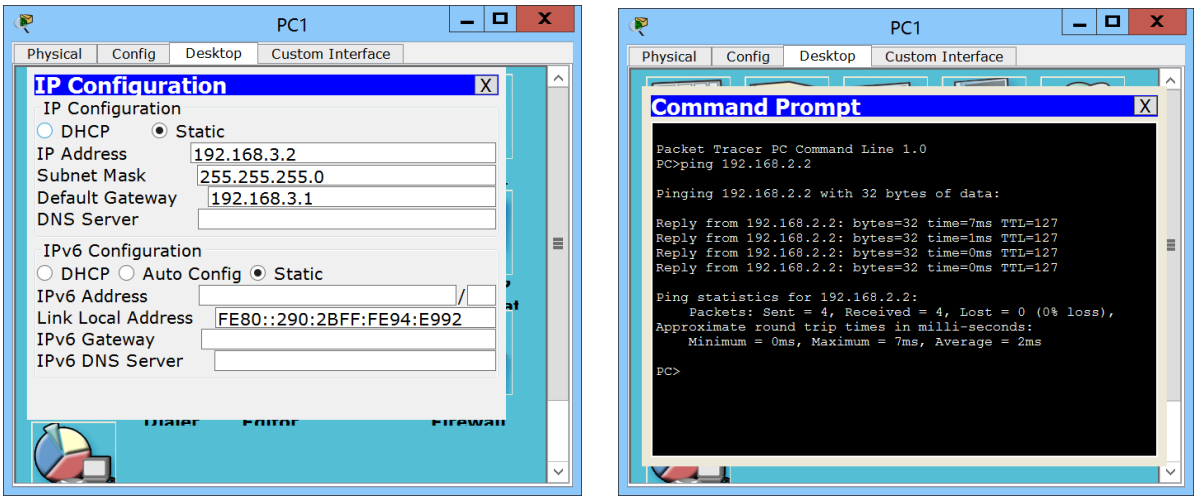


Рисунок 10.21 – Тестування зєднань

Практичні завдання

Створити 2-х рівневу комп’ютерну мережу з виродженим ядром. В якості кінцевих пристроїв виступають робочі станції котрі входять в декілька робочих груп. Відповідні групам сегменти мережі повинні мати різні IP підмережі. Маршрутизатор повинен забезпечити маршрутизацію між сегментами. Комутатори повинні бути зв’язані trunk зв’язком.

Таблиця 1 – Склад мережі

Номер рівня	Обладнання	К-ість
2	Маршрутизатор Cisco 2960	1
1	Комутатор	2

Таблиця 2 – Склад робочих груп

№ групи	Призначення і назва підмережі	IP адреса мережі
1	Бухгалтерія (finance)	192.168.2.0/24
2	Відвідувачі (visitors)	192.168.3.0/24

Кількість робочих станцій в кожному сегменті мережі підключених до комутаторів вказано в таблиці 3

Таблиця 3 – Розподіл кінцевих пристроїв по комутаторам

№ групи	Всього кінцевих пристроїв	Комутатор 1	Комутатор 2	
1	4	2	2	
2	4	2	2	

Для тестування з’єднань пристроїв використовувати команду ping. Навести результати тестування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Березький О.М., Дубчак Л. О., Цмоць І. Г. Проектування комп'ютерних систем на програмованих логічних інтегральних схемах: навч. посіб. Тернопіль: ТНЕУ, 2014. 163 с.

2. Цмоць І.Г., Березький О.М., Ігнатєв І. В. Шляхи підвищення швидкодії обробки зображень в комп'ютерні системі з графічним процесором. *Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова*. 2016. Т.76 С. 201-208.

3. Цмоць І.Г., Березький О.М., Ігнатєв І. В. Особливості розпаралелювання та реалізації алгоритмів обробки зображень на графічних процесорах. *Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту (ISDMCI'2016)*: матеріали Міжнародної наукової конференції, 24–28 травня 2016 р. м. Залізний Порт – Херсон: ПП Вишемирський В.С., 2016. С. 222-234.

4. Tsmots I., Berezkyi O., Ihnatiev I., Gumovska I. Implementation of image processing algorithms based on GPU. *Computer Sciences and Information Technologies (CSIT'2016)*: proceedings of the XIth International Scientific and Technical Conference, 6-10 September, 2016. Lviv, 2016. P. 27□29.

5. Tsmots I., Rabyk V., Berezky O., Lukaschuk Y., Teslyuk V. Development of modules of neuro-like cryptographic encryption and decryption of data and their implementation on FPGA. *Experience of Designing and Application of CAD Systems (CADSM 2021)*: proceedings of 2021 IEEE 16th International Conference, 22-25 Fabruary 2021. Jaroslaw, Poland, 2021. Pp. 53–57.

6. Цмоць І.Г., Березький О.М., Ігнатєв І. В. Структура пристрою обміну на базі багатопортової пам'яті для нейроорієнтованих комп'ютерних систем. *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України «Моделювання та інформаційні технології»*. 2014. Вип. 73. С. 67–72.

7. Цмоць І.Г., Березький О.М., Ігнатєв І. В. Принципи побудови та базова структура нейроорієнтованих комп'ютерних систем реального часу. *Вісник Хмельницького національного університету*. 2014. №6 (219). С. 173-179.

8. Цмоць І.Г., Теслюк В.М., Ваврук І.Є., Березький О. М. Прогнозування руху мобільної робототехнічної системи. *Збірник наукових праць інституту проблем моделювання в енергетиці ім. Г.Є. Пухова НАН України «Моделювання та інформаційні технології»*. 2012. Вип. 66. С. 174–178.

9. Цмоць І.Г., Красовський В.Б., Березький О. М. Особливості реалізації реконфігуровних нейросистем реального часу. *Моделювання та інформаційні технології*. 2012. Вип. 65. С. 119–125.

10. Teslyuk T., Teslyuk V., Denysyuk P., Tsmots I., Berezsky O., Melnyk M. Synthesis of Neurocontroller for Intellectualization Tasks of Process Control Systems. *Experience of Designing and Application of CAD Systems (CADSM)*: proceedings of 15 th International Conference. February 26 – March 2, 2019. Polyana (Svalyava), UKRAINE, 2019. Pp. 6/39-6/42

11. Denysyuk P., Tesluyk T., Kernytskyy A., Teslyuk V., Tsmots I., Berezsky O. Interface-Sensitive Method of Synthesis of Microcontroller- Based System Structures. *Experience of Designing and Application of CAD Systems (CADSM)*: proceedings of 15 th International Conference. February 26 – March 2, 2019. Polyana (Svalyava), UKRAINE, 2019. Pp. 21-24

12. Berezsky O., Melnyk G., Batko Yu., Pitsun O. Regions matching algorithms analysis to quantify the image segmentation results. *Sensors & Transducers*. 2017. Vol. 208, Issue 1. P. 44-50.

13. Березький О. М. Методи та алгоритми перетворення контурів зображень в афінному просторі. *Вісник Національного університету «Львівська політехніка». Комп'ютерні науки та інформаційні технології*. 2009. № 638. С. 185-189.

14. Березький О. М., Батько Ю.М., Мельник Г.М. Комп'ютерна система аналізу біомедичних зображень. *Вісник Національного університету «Львівська*

політехніка». Комп'ютерні науки та інформаційні технології. 2009. № 570. С. 84-89.

15. Berezsky O., Datsko T., Verbovy S. The intelligent system for diagnosing breast cancers based on image analysis. *Information Technologies in Innovation Business (ITIB)*: proceedings of International conference. 7-9 October, 2015. Kharkiv, Ukraine, 2015. P. 27-30.

16. Berezsky O., Dubchak L., Pitsun O. Access Distribution in Automated Microscopy System. *The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM 2017)*: Proceedings of the 14 th International Conference. 21-25 February 2017. Lviv, Ukraine, 2017. P. 241-243.

17. Berezsky O., Verbovyy S., Dubchak L., Datsko T. Fuzzy system diagnosing of precancerous and cancerous conditions of the breast. *Computer Sciences and Information Technologies (CSIT'2016)*: Proceedings of the XIth International Scientific and Technical Conference. 6-10 September, 2016. Lviv: Ukraine, 2016. Pp. 200–203.

18. Березький О.М., Цмоць І.Г. Система автоматизації медичних експериментів. *Вісник Національного університету «Львівська політехніка». Комп'ютерні науки та інформаційні технології*. 2011. № 694. С. 113-121.

19. Березький О.М., Цмоць І.Г. Принципи побудови та архітектура комп'ютерних засобів аналізу і синтезу зображень. *Вісник Хмельницького національного університету. Технічні науки*. 2011. №2. С.160- 168.

20. Березький О.М., Цмоць І.Г. Методи, алгоритми та НВІС-структури для множення матриці на вектор у реальному часі. *Вісник Хмельницького національного університету. Технічні науки*. 2007. Т. 1 (93), № 3. С. 134–140.]

21. Березький О.М., Цмоць І.Г. Методи та НВІС-структури для множення матриці на матрицю у реальному часі. *Комп'ютерні системи проектування. Теорія і практика*. 2007. № 591. С. 63–75.

22. Aleksander M.B. *et al*. Implementation technology software-defined networking in Wireless Sensor Networks. *Intelligent Data Acquisition and Advanced*

Computing Systems: Technology and Applications (IDAACS): Proc. of 2015 IEEE 8th International Conference, 2015. Pp. 448-452.

23. Vasylkiv N., Dubchak L., Lendyuk T., Turchenko I., Shylinska I., Aleksander M. Tasks distribution for students testing based on fuzzy logic. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proc. of 2017 9th IEEE International Conference, 2017. Pp. 26-29.

24. Komar M. *et al*. High performance adaptive system for cyber attacks detection. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proc. of 2017 9th IEEE International Conference, 2017. Pp. 853-858.

25. Romanets I., Sachenko A., Dubchak L. Method of Protection Against Traffic Termination in VoIP. *Electronics, Computers and Artificial Intelligence (ECAI)*: Proc. of 2018 10th International Conference, 2018. Pp. 1-5.

26. Vasylkiv N., Dubchak L., Turchenko I., Ivashchuk I., Savchyshyn R. Fuzzy Estimation Method of Information System Providing Part Influence on the Functioning Quality. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proc. of 2019 10th IEEE International Conference, 2019. Pp. 980-984.

27. Vasylkiv N., Turchenko I., Dubchak L. Fuzzy Model of the IT Project Environment Impact on its Completion. *Advanced Computer Information Technologies (ACIT)*: Proc. of 2020 10th International Conference, 2020. Pp. 302-305.

28. Енциклопедія соціогуманітарної інформології / за заг. ред. проф. К. І. Бєляков. Київ: Видавничий дім «Гельветика», 2020. Т. 1. 472 с.

29. Класифікація паралельних комп’ютерних систем. URL: https://uk.wikipedia.org/wiki/Класифікація_паралельних_комп%27ютерних_систем

30. Класифікація Дункана. URL: https://uk.wikipedia.org/wiki/Класифікація_Дункана

31. Основні компоненти комп’ютерних систем. URL: https://klaster.ua/ua/stati-i-obzory/osnovn-komponenti-kompyuternih-sistem/

32. Будова сучасного електронного комп'ютера. URL: https://studfile.net/preview/5128368/page:2/

33. Сервер. URL: https://uk.wikipedia.org/wiki/Сервер

34. Лінія зв’язку. URL: https://uk.wikipedia.org/wiki/Лінія_зв%27язку

35. Перетворювач електричної енергії. URL: https://uk.wikipedia.org/wiki/Перетворювач_електричної_енергії

36. Програмне забезпечення. URL: https://uk.wikipedia.org/wiki/ Програмне_забезпечення

37. Мельник В.А. Стан та перспективи розвитку високопродуктивних обчислювальних систем. URL: http://ena.lp.edu.ua:8080/bitstream/ ntb/12231/1/15_СТАН%20ТА%20ПЕРСПЕКТИВИ%20РОЗВИТКУ.pdf

38. Микитишин А.Г., Митник М.М., Стухляк П.Д., Пасічник В.В. Комп’ютерні мережі: навч. посіб. Львів, «Магнолія 2006», 2013. 256с.

39. Тарнавський Ю.А., Кузьменко І.М. Організація комп’ютерних мереж: підручник для студ. спеціальності 121 «Інженерія програмного забезпечення» та 122 «Комп’ютерні науки». Київ: КПІ ім. Ігоря Сікорського, 2018. 259с.

40. Peterson L., Davie B. Computer Networks a systems approach. Fifth Edition. Elsevier, 2012. 921 p.

41. Stallings W. Data and Computer Communications. Pearson, 2013. 912 p.

42. Демида Б.А., Обельовська К.М., Яковина В.С. Основи адміністрування LAN у середовищі MS Windows: навч. посіб. Львів: Видавництво Львівської політехніки, 2013. 488 с.

43. Сенів М.М., Яковина В.С. Безпека програм та даних : навч. посіб. Львів: Видавництво Львівської політехніки, 2015. 256 с.

44. Сучасний тлумачний словник української мови: 60 000 слів: за заг. Ред. д-ра філол. наук, проф. В.В. Дубічинського. Харків: ВД «ШКОЛА», 2007. 832 с.

45. Словник іншомовних слів: за редакцією член-кореспондента АН УРСР О.С. Мельничука. Київ, Головна редакція Української радянської енциклопедії, 1977. 776 с.

46. Параметри оптимізації. URL: http://wiki.tntu.edu.ua/параметри_оптимізації

47. Теслюк В.М. Математичне моделювання в САПР: Ч.1. Конспект лекцій з курсу “Математичне моделювання в САПР” для студентів базового напрямку “Комп’ютерні науки”. Львів: Видавництво Національного університету “Львівська політехніка”, 2009. 64 с.

48. Словарь по кибернетике: Св. 2000 ст.: Под ред. В.С. Михалевича, 2-е изд.-К.: Гл.ред.УСЭ им.М.П.Бажана, 1989. 751с.

49. Steuer R.E. Multiple Criteria Optimization: Theory, Computations, and Application. New York : John Wiley & Sons Inc., 1986. ISBN 047188846X.

50. Sawaragi, Y. Theory of Multiobjective Optimization. *Mathematics in Science and Engineering*. Vol. 176. Orlando, FL : Academic Press Inc., 1985. ISBN 0126203709.

51. Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen та Roman Slowinski Multiobjective Optimization: Interactive and Evolutionary Approaches. Lecture Notes in Computer Science. — Springer, 2008. ISBN 3-540-88907-8.

52. Лотов А.В., Поспелова И.И. Многокритериальные задачи принятия решений: Уч. пособие. М.: Издат. отдел ф-та ВМиК МГУ, МАКС Пресс, 2008. 197 с.

53. Малышкин В.Э. Основы параллельных вычислений. ЦИТ СГГА, 2003. URL: http://www.ssga.ru/metodich/paral1/contents.html

54. Питерсон Дж.. Теория сетей Петри и моделирование систем. Москва: Мир, 1984. 264 с.

55. Котов В. Е. Сети Петри. Москва: Наука, 1984. 160 с.

56. Теслюк В. М. та ін. Розробка математичних моделей МЕМС на основі мереж Петрі для системного рівня автоматизованого проектування.

Моделювання та інформаційні технології. Збірник наукових праць ІПМЕ ім. Г. Є. Пухова НАН України. Київ: 2008. Вип. 46. С.120 – 126.

57. Lobur M., Teslyuk V., Zaharyuk R., Antonyuk V. Using petri nets in MEMS design. *Computer Simulation In Machine Design – COSIM2006* : Proc. of the XI Inter. Conf. Krynica Zdrój, Poland, 2006. P. 209 – 213.

58. Teslyuk V., Romanyuk A., Matviykiv T. Petri networks application during mems design process. *Perspective technologies and methods in MEMS design (MEMSTECH'2007)* : Proc. of the IIId international conference of young scientists. Lviv - Polyana, 2007. P. 159 – 160.

59. Lobur M., Teslyuk V., Zaharyuk R., Antonyuk V. Using petri nets in MEMS design. *Journal Machine Dynamics Problems*. Poland, Warsaw University of Technology, 2006. Vol. 30, No. 4. P. 29 – 36.

60. Teslyuk V., Omari Tarik Al , Alshavabkekh Hamza, Denysyuk P., Melnyk M. Computer – aided design of MEMS at system level. *CAD in Machinery Design – Inplementation and Educational Problems*: Proc. of the XV. Naleczow, Poland, 2007. P. 35 – 37.

61. Zaharyuk R., Teslyuk, Karkulyovskyy V. Using petri nets in capacitive microaccelerometers design. *Perspective Technologies and Methods in MEMS Design*: Proc. of the III-d Intern. Conf. Lviv – Polyana, 2007. P. 149 – 150.

62. Teslyuk V., Al - Shavabkeh Hamza, Pereyma M., Al Omari Tarik. The formalization of the MEMS automated design process by usage of Petri Networks. *Perspective technologies and methods in MEMS design (MEMSTECH'2007)*: Proc. of the IIId international conference of young scientists. Lviv - Polyana, 2007. P. 133 – 134.

63. Формалізація процесу автоматизованого проектування МЕМС з допомогою мереж Петрі / Теслюк В. та ін. *Інтелектуальні системи прийняття рішень та прикладні аспекти інформаційних технологій*: Матер. наук.-техн. конф. 2007. Херсон: Вишемирський В.С. Том. 2. С. 230 – 233.

64. Теслюк В.М., Загарюк Р.В. Застосування мереж Петрі при автоматизованому проектуванні інтегральних акселерометрів на системному

рівні. *Комп'ютерні технології друкарства*: збірник наук. праць УАД. Львів, 2007. № 18. С. 64 – 72.

65. Computer-Aided Design of MEMS at System Level / Teslyuk V. et al. *Journal Machine Dynamics Problems*. Poland, Warsaw University of Technology, 2007. Vol. 31, No. 3. P. 92 – 104.

66. Фомин Б.Ф., Яковлев В.Б. Автоматика и управление в технических системах : в 11 кн. Кн. 3. Моделирование производственных систем. Київ: Вища шк., 1992. 191 с.

67. Карлин С. Математические методы в теории игр, программировании и экономике. Москва: Мир, 1964. 312 с.

68. Вентцель Е.С. Введение в исследование операций. Сов. радио, 1964. 198 с.

69. Пономаренко О.І., Пономаренко В.О. Системні методи в економіці, бізнесі й менеджменті. К.: Либідь, 1995. 320 с.

70. Пономаренко О.І., Перестюк М.О., Бурим В.М. Основи математичної економіки. К.: Інформтехніка, 1995. 262 с.

71. Горелик В.А., Ушаков М.А. Исследование операций. М.: Машиностроение, 1986. 288 с.

72. Прохоров А.В., Ушаков В.Г., Ушаков Н.Г.. Задачи по теории вероятностей: Основные понятия. Предельные теоремы. Случайные процессы. М.: Наука, 1986. 326 с.

73. Коваленко И.Н., Кузнецов Н.Ю., Шуренков В.М. Случайные процессы. Справочник. Киев: Наук. думка, 1983. 366 с.

74. Розанов Ю.А. Теория вероятностей, случайные процессы и математическая статистика. М.: Наука, 1989. 320 с.

75. Сучасний тлумачний словник української мови: 60 000 слів / За заг. ред. д-ра філол. наук, проф. В.В. Дубічинського. Х.: ВД «ШКОЛА», 2007. 832 с.

76. Словник іншомовних слів. / За редакцією член-кореспондента АН УРСР О.С. Мельничука. Київ, Головна редакція Української радянської енциклопедії, 1977. 776 с.

77. Теслюк В.М. Математичне моделювання в САПР: Ч.1. Конспект лекцій з курсу “Математичне моделювання в САПР” для студентів базового напрямку “Комп’ютерні науки”. Львів: Видавництво Національного університету “Львівська політехніка”, 2009. 64 с.

78. Словарь по кибернетике: Св. 2000 ст./Под ред. В.С. Михалевича, 2-е изд.-К.: Гл.ред.УСЭ им.М.П.Бажана, 1989. 751с.

79. Steuer, R.E. Multiple Criteria Optimization: Theory, Computations, and Application. New York: John Wiley & Sons, Inc , 1986. ISBN 047188846X.

80. Sawaragi, Y. Theory of Multiobjective Optimization. *Mathematics in Science and Engineering*. Vol. 176 Orlando, FL : Academic Press Inc , 1985. ISBN 0126203709.

81. Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen та Roman Slowinski Multiobjective Optimization: Interactive and Evolutionary Approaches. Lecture Notes in Computer Science. Springer, 2008. ISBN 3-540-88907-8.

82. Лотов А.В., Поспелова И.И. Многокритериальные задачи принятия решений: Уч. пособие. М.: Издат. отдел ф-та ВМиК МГУ, МАКС Пресс, 2008. 197 с.

83. Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. М.: Наука, 1982. 256 с.

84. Штойер Р. Многокритериальная оптимизация. Теория, вычисления, и приложения. М.: Радио и связь,1992. 504 с.

85. Кини Р.Л., Райфа Х. Принятие решений при многих критериях: предпочтения и замещения. М.: Радио и связь, 1981. 560 с.

86. Ларичев О. И. Теория и методы принятия решений. М.: Логос, 2000. 350 с.

87. Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. М.: Наука, 1982. 195 с.

88. Соболь И.М. Выбор оптимальных параметров в задачах со многими критериями. М.: Наука, 1981. 108 с.

89. Подиновский В.В., Гаврилов В.М. Оптимизация по последовательно применяемым критериям. М.: Сов. радио, 1975. 194 с.

90. Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. М.: Наука, 1982. 254 с.

91. Лотов А.В., Бушенков В.А., Каменев Г.К., Черных О.Л. Компьютер и поиск компромисса. Метод достижимых целей. М.: Наука, 1997. 239 с.

92. Таха Х. Введение в исследование операций: В 2-х кн. Кн. 1. Пер. с англ. М.: Мир, 1985. 479 с.

93. Таха Х. Введение в исследование операций: В 2-х кн. Кн. 2. Пер. с англ. М.: Мир, 1985. 496 с.

94. Теслюк В.М., Андрійчук М.І. Конспект лекцій з курсу «Методи синтезу та оптимізації» для студентів базового напряму «Комп'ютерні науки», Ч.1. Львів, 2005. 64 с.

95. Теслюк В.М. Моделі та інформаційні технології синтезу мікроелектромеханічних систем: Монографія. Львів: Видавництво ПП "Вежа і Ко", 2008. 192 с.

96. Норенков И. П. Основы автоматизированного проектирования: учеб. для вузов. 2-е изд., перераб. и доп. М.: Изд. МГТУ им. Н. Э. Баумана, 2002. 336 с.

97. Норенков И. П. Системы автоматизированного проектирования : [учеб. пособие для вузов] : В 9 - ти кн. Кн.1. Принципы построения и структура / И. П. Норенков. – М. : Высш. шк., 1986. – 127 с.

98. Петренко А. И. Основы автоматизации проектирования. К.: Техніка, 1982. 295 с.

99. Корячко В. П., Курейчик В.М., Норенков И. П. Теоретические основы САПР: учеб. для вузов. М.: Энергоатомиздат, 1987. 400 с.

100. Тихонов А. Н., Самарский А. А. Уравнения математической физики. М.: Наука, 1966. 724 с.

101. Самарский А.А. Теория разностных схем. М.: Наука, 1983. 616 с.

102. Самарский А.А. Численные методы. М.: Наука, 1989. 432 с.

103. Годунов С.К., В.С. Рябенький. Разностные схемы. М.: Наука, 1973. 400 с.

104. Молчанов И.Н. Машинные методы решения прикладных задач. Дифференциальные уравнения. К.: Наукова думка, 1988. 344 с.

105. Калиткин Н.Н. Численные методы. М.: Наука, 1978. 512 с.

106. Сегерлинд А. Применение метода конечных элементов. М.: Мир, 1979. 392 с.

107. Норри Д., Ж. Де Фриз. Введение в метод конечных элементов. М.: Мир, 1981. 304 с.

108. Зенкевич О. Метод конечных элементов в технике.: Пер.с англ. М.: Мир, 1975. 572 с.

109. Соболь И.М. Метод Монте - Карло. М.: Наука. Гос. изд. физ.-мат. лит., 1985. 80 с.

110. Ермаков С.М. Метод Монте – Карло и смежные вопросы. М.: Наука. Глав. ред. физ. - мат. лит., 1975. 472 с.

111. Волынский Б.А., Бухман В.Е. Модели для решения краевых задач. М.: Гос. изд. физ.-мат. лит., 1960. 452 с.

112. Теория подобия и размерностей. Моделирование / П. М. Алабужев и др. М.: Высш. шк., 1968. 208 с.

113. Веников В.А. Теория подобия и моделирования. М.: Высш. шк., 1976. 479 с.

114. Гухман А.А. Введение в теорию подобия. М.: Высш. шк., 1973. 296 с.

115. Гухман А.А. Применение теории подобия к исследованию процессов тепло-массообмена. М.: Высш. шк., 1974. 328 с.

116. Толковый словарь по вычислительным системам / Под. ред. В. Иллингуорта и др. : Пер. с англ. А. К. Белоцкого и др.; Под. ред. Е. К. Масловского. М.: Машиностроение, 1991. 560 с.

117. Джеффирс Г. Методи математической физики. Вип. 2 / Г. Джеффирс, Б. Свирлс. М.: Мир, 1970. 352 с.

118. Краснов М.Л. Интегральные уравнения. М.: Наука, 1975. 303 с.

119. Адамар Ж. Задача Коши для линейных уравнений с частными производными гиперболического типа. М.: Наука, 1978. 351 с.

120. Проблемы Гильберта: Под ред. П.С.Александрова. М.: Наука, 1969. 239 с.