Міністерство освіти і науки України

Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук

Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» грудня 2022 р.

Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему: «МОДЕЛЬ МЕРЕЖЕВОГО ПЛАНУВАННЯ ІТ-ПРОЄКТУ НА ОСНОВІ МЕТОДІВ МУЛЬТИПАРАЛЕЛЬНОЇ ОБРОБКИ ІНФОРМАЦІЇ»

Захищено на засіданні
Атестаційної комісії № 45
протокол № __ від __.12.2022 р.
Оцінка _____ / _____
Голова Атестаційної комісії
_____**МІНУХІН С. В.**

Виконав:

студент 2 курсу, групи КУ– 61
за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології.
Галузь знань 15 – Автоматизація та приладобудування
Толстолузький Євген Дмитрович _____

Керівник:
д.т.н., професор; професор кафедри теоретичної та прикладної системотехніки
Мірошник Марина Анатоліївна ____

Рецензент: д.т.н., доцент, завідувач кафедри безпеки інформаційних систем і технологій
Рассомахін Сергій Геннадійови ____

Харків – 2022

АНОТАЦІЯ

Пояснювальна записка до магістерської кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел і чотирьох додатків. Загальний обсяг роботи складає 76 сторінок, із яких 62 сторінки основної частини з 25 рисунками, 1 таблицею, 20 найменуваннями списку використаних джерел та двома додатками.

Метою кваліфікаційної роботи є скорочення часу мережевого планування виконання ІТ-проєкту за рахунок використання методів мультипаралельної обробки.

Об'єкт дослідження – процес мережевого планування виконання ІТ-проєкту.

Предмет дослідження – методи та моделі мережевого планування виконання ІТ-проєкту.

У кваліфікаційній роботі описується робота та задачі бізнес аналітика та менеджера по проєктам. Розглядаються існуючі засоби та методології розрахунків показників проєкту, побудови мережевих графів, а також було запропоновано нову модель та її програмну реалізацію. У роботі були наведені рекомендації по застосуванню розробленого програмного рішення.

Область застосування – розробка веб-додатків. Розроблений програмний продукт може широко використовуватися в сфері малого та середнього та великого ІТ-бізнесу.

Ключові слова: ПРОЄКТ, КОМП'ЮТЕРНА МОДЕЛЬ. АВТОМАТИЗАЦІЯ.МЕНЕДЖЕР ПО ПРОЄКТАМ, МЕРЕЖЕВИЙ ГРАФ, МЕТОДИ МУЛЬТИПАРАЛЕЛЬНОЇ ОБРОБКИ, ЕКСПЕРТНІ ОЦІНКИ.

ABSTRACT

An explanatory note to the master's attestation work is created in the introduction, three sections, conclusions, a list of sources used and four additional substances.

The total volume of work is 76 pages, of which 62 pages of the main part with 25 figures, 1 table, 20 names of the list of used sources and two additions.

The purpose of the qualification work is to reduce the time of network planning for the implementation of an IT project due to the use of multiparallel processing methods.

The object of the study is the process of network planning of the implementation of IT – of the project.

The subject of research is methods and models of network planning execution of an IT project.

The qualification work describes the work and tasks of a business analyst and project manager. Existing apps and methodologies for calculating project indicators, for building network graphs are considered, and a new model and its software implementation were proposed. Recommendations for the application of the developed software solution were given in the paper.

The field of application is the development of web applications. The developed software product can be widely used in the field of small and medium and large IT businesses

Keywords: PROJECT, COMPUTER MODEL. AUTOMATION. PROJECT MANAGER, NETWORK GRAPH, METHODS OF MULTI-PARALLEL PROCESSING, EXPERT EVALUATIONS.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

СЧС	–	семантико-числові специфікації
BF	–	базовий СЧС файл
CF	–	файл зв'язку СЧС
ti-jcp	–	середній час виконання задачі
i-j2	–	дисперсія роботи
АЛП	–	арифметико-логічний пристрій
АП	–	апарат з пам'яттю
ЕОМ	–	електронна обчислювальна машина
ЕП	–	елемент пам'яті
ЗЗМ	–	зв'язано–зовнішня множина
КММ	–	кодово-матричний метод
КОІ	–	канал обміну інформацією
ОС	–	обчислювальна система
РМЧ	–	реальний масштаб часу
ТТХ	–	тактико-технічні характеристики
ЧПГС	–	часова паралельна граф схема
УЛМ	–	універсальний логічний модуль
PERT	–	Project Evaluation and Review Technique
СЧС_АЧПП	–	семантико-числовий синтезатор адаптивних часових паралельних процесів

ВСТУП

Етап проєктування є невід'ємною частиною успішного та тривалого проєкту. Інформаційні технології дозволяють спростити і автоматизувати цей процес.

Метою будь-якого проєкту є створення нового, та здебільшого унікального продукту. Робота над цим проєктом включає обчислювальну, комунікаційну підтримку, інформаційні ресурси та персонал, що відповідальний за підтримку динамічних інформаційних моделей.

Процес створення та контролю планів, раціональний розподіл ресурсів і задач, зведення комунікацій між підрозділами, створення рівноваги між проєктними обмеженнями на весь час його реалізації називається керуванням проєкту.

Кількість ризиків змінюється в залежності від часу виконання проєкту. Чим більше часу займає проєктування, тим більше їх виникає. Ризики впливають на терміни розробки проєкту, на можливий прибуток та втрати. До них також можна віднести ефективність праці, кількість людей, їх похибки, якість програмного та технічного забезпечення, якісну оцінку часу роботи. Базуючись на цьому, можна сказати, що процесс автоматизування процесів, пов'язаних з проєктуванням, призводить до зменшення ризиків виникнення майбутніх помилок. Тому розробка автоматизованих рішень для керуванням проєктами є актуальним питанням, тому що такі програмні моделі допомагають мінімізувати витрати часу, розраховують можливі ризики та створюють ефективний план розвитку.

Одним з процесів, який можна було б автоматизувати є створення мережевого графу на етапі мережевого планування.

У даній роботі розглядається розробка автоматизованого процесу побудови мережевого графу та обчислення ризиків проєкту, користуючись

методами мультипаралельної обробки інформації. Завдяки цим методам можна перетворити послідовну задачу на паралельну.

Метою дослідження є скорочення часу мережевого планування виконання ІТ-проєкту за рахунок використання методів мультипаралельної обробки.

Об'єктом дослідження є процес мережевого планування виконання ІТ-проєкту.

Предметом дослідження є методи та моделі мережевого планування виконання ІТ-проєкту.

Щоб виконати в повній мірі зазначену мету дослідження, необхідно виконати такі завдання як:

1) провести аналіз методів мультипаралельної обробки інформації;
2) подати ІТ-проєкт у вигляді семантико-числових специфікацій;
3) створити програмну модель мережевого планування виконання ІТ-проєкту на базі методів мультипаралельної обробки;
4) провести випробування створеної моделі;
5) створити набір рекомендації для полегшення застосування розробленої моделі.

Таким чином, дана робота присвячена використанню методів мультипаралельної обробки інформації для побудови мережевого графа і оцінки ризиків проєкту.

РОЗДІЛ 1.
АНАЛІЗ МЕТОДІВ МУЛЬТИПАРАЛЕЛЬНОЇ ОБРОБКИ

1.1 Огляд роботи бізнес-аналітиків і проджект-менеджерів

Бізнес-аналіз – це процес поєднання аналізу даних із використанням певних методів і виконання завдань для визначення потреб бізнесу, а потім створення рекомендацій щодо змін і надання рішень, які несуть цінність для зацікавлених сторін. Багато рішень потенційно містять програмне забезпечення та компоненти на основі цифрових даних, але також можуть включати організаційні зміни, наприклад вдосконалення процесів, розробка нових політик і залучення до стратегічного планування [1].

Бізнес-аналітики – це агенти, які аналізують проєкт або організацію, документуючи її системи та процеси, оцінюючи її бізнес-модель, виявляючи вразливі місця та розробляючи рішення.

Головним чином завдяки незвичайній швидкості сучасного технологічного прогресу діловий світ 21-го століття є світом швидких і постійних змін. Інновації змінюють спосіб нашого життя та роботи, і підприємства повинні мати можливість адаптуватися до цих змін або залишитися позаду.

Бізнес-аналітики є ідеальними професіоналами для того, щоб провести будь-яку організацію через складну територію змін. Пройшовши ці зміни, компанії стають міцнішими, конкурентоспроможними та краще підготовленими для роботи в світі цифрового бізнесу, що постійно розвивається [2].

Бізнес-аналітики застосовують дисциплінований підхід до створення та керування змінами в організації. Вони виявляють вразливі місця, визначають потреби на основі зворотного зв’язку та спілкування із зацікавленими сторонами, організовують і впроваджують рішення та контролюють результати.

У сучасному конкурентному середовищі підприємства будь-якого розміру потребують усіх можливих переваг, які вони можуть отримати, а бізнес-аналітик є цінним ресурсом для визначення найкращих кроків як у тактичному, так і в стратегічному масштабі. Крім того, бізнес-аналітики є ідеальними менеджерами проєктів. Наймаючи такого працівника, компанії краще розуміють себе та свої потреби, а також те, як найкраще їх задовольнити. Якщо бізнес-аналіз не проводиться, то можна зустріти наступні недоліки:

1. Збільшення ризиків: Непередбачені обставини можуть призвести до провалу роботи;

2. Зростання цін на роботу: Погане керування ресурсами спричиняє додаткові витрати на роботу;

3. Збільшення тривалості роботи: Недостатня кількість ресурсів або їх неправильне планування на проєкті призводить до того, що робота займає більше часу [3].

Менеджер проєктів – це професіонал, який організовує, планує та виконує проєкти, працюючи в рамках обмежень, таких як бюджети та графіки. Менеджери проєктів керують цілими командами, визначають цілі проєкту, спілкуються із зацікавленими сторонами та доводять їх до закриття. Незалежно від того, чи проводить маркетингову кампанію, будує будівлю, розробляє комп'ютерну систему чи запускає новий продукт, керівник проєкту несе відповідальність за успіх чи провал проєкту.

Роль менеджера проєкту затребувана майже в кожній галузі. Давайте детальніше розглянемо, чим займаються менеджери проєктів, чому варто розглянути можливість кар'єри в управлінні проєктами та як почати.

Проєкт зазвичай поділяють на чотири різних фази: початок, планування, виконання та закриття.

Протягом усього життєвого циклу проєкту керівник проєкту несе відповідальність за:

1) визначення обсягу проєкту;

2) дотримання графіку;
3) планування вартості проєкту та дотримання бюджету;
4) керування ресурсами проєкту (включаючи команди та працівників);
5) документування прогресу проєкту;
6) спілкування із зацікавленими сторонами;
7) оцінка ризиків;
8) вирішення проблем;
9) провідна гарантія якості.

Велике розмаїття завдань означає, що немає двох однакових днів на роботі (або двох проєктів). У будь-який день менеджер може проводити співбесіди та наймати нових талантів, керувати командними зборами, перерозподіляти ресурси для покриття неочікуваних витрат або інформувати зацікавлених сторін про хід проєкту.

Основними рисами, які повинен мати проджект менеджер, є:

1) лідерство: Керування командою для досягнення мети;
2) комунікація: Дуже часто виступає у ролі першої лінії спілкування для членів команди, постачальників, зацікавлених сторін і клієнтів;
3) організація: Здатність розставляти пріоритети та багатозадачність щоб забезпечити безперебійну роботу проєктів;
4) критичне мислення: Критичний аналіз і оцінка ситуації допомагає запобігти проблемам до того, як вони виникають;
5) почуття гумору: підхід до проєкту з позитивним настроєм може зняти стрес і зарядити енергією вашу команду.

Базуючись на цій інформації можна зазначити, що люди, які виконують обов'язки бізнес-аналітиків та проджект менеджерів є необхідними на проєкті, з їхньою допомогою процес планування стає більш стабільним, більш прогнозованим та швидшим [4–5].

1.2 Методи мультипаральної обробки інформації

Нині одними з основних шляхів підвищення ефективності обчислювальних систем, які не вимагають розробки покращеної елементної бази, прийнято вважати застосування методів паралельної обробки інформації.

Застосування паралельної обробки інформації забезпечує зменшення часу виконання алгоритмів, підвищення тактової частоти, покращення характеристик точності та достовірності обчислень, збільшення пропускної спроможності цифрових систем.

На даний час в літературі знайшли опис п’ять методів паралельної обробки інформації:

1) метод суміщення незалежних операторів (СО);
2) метод конвеєрної обробки (КО);
3) метод декомпозиційної обробки (ДО);
4) кодово-матричний метод (КММ);
5) метод суміші алгоритмів (СА).

Наразі тільки метод суміщення незалежних операторів використовується в усіх сучасних процесорах. Метод конвеєрної обробки на апаратному рівні застосовується в скалярних та суперскалярних процесорах. Декомпозиційний метод, також на апаратному рівні, застосовується в процесорах VLIW з паралелізмом на рівні команд, Very Long Instruction Word. Між іншим, дослідження, проведені на факультеті комп’ютерних наук Харківського національного університету в межах наукового напрямку «На виконання завдань перспективного плану розвитку наукового напряму «Технічні науки» Харківського національного університету імені В.Н. Каразіна» показали, що застосування раціональної сукупності методів паралельної обробки інформації (мультипаралельна обробка інформації) дозволяють підвищити ефективність обчислювальних систем різних класів.

Метод суміщення незалежних операторів.

Сутність методу полягає в одночасному початку виконання у кожний з дискретних моментів часу деякої кількості операторів алгоритму, для яких виконуються такі умови:

а) ці оператори не пов'язані інформаційними та/або керуючими зв'язками;

б) для кожного з таких операторів до моменту часу, що розглядається, є значення відповідних операндів.

Будемо використовувати надалі таке визначення методу суміщення операцій – це метод паралельної обробки, для якого:

- об'єктами операційного рівня є бітові коди чисел або частини кодів (до окремого біта), що розглядаються як неподільні цілі;
- об'єктами алгоритмічного рівня є фрагменти алгоритмів чи алгоритми, які розглядаються як неподільні цілі;
- операторами операційного рівня є загальноприйняті операції/функці обробки даних (при роботі з кодами чисел) та операції булевої алгебри (при роботі з однобітовими даними);
- операторами алгоритмічного рівня є операції над графами або часовими паралельними граф-схемами;
- часові відносини між (хоча б деякими) операторами задовольняють вимогу наявності принаймні однієї пари операторів Pi∈P та Pj∈P алгоритму P з непорожнім перетином їх інтервалів активності IAi і IAj

$$IA_i \cap IA_j \neq O \quad \text{для } P_i, P_j \in P, \qquad (1.1)$$

виконання яких починається одночасно, тобто для яких t_i^H t_j^H;

- характер часових відносин між інтервалами активності I(SDk) реалізації алгоритму в різних наборах вхідних даних SDk∈SD визначається наступним співвідношенням:

$$I(SDk) \in I(SDl) \neq O \qquad (1.2)$$

хоча б для деякої пари номерів k , lj вхідних наборів даних алгоритму де ki, l ∈ 1,2,...,sd; sd = | SD |, тобто має місце перетин часових інтервалів реалізацій алгоритму для різних наборів даних.

На рисунку 1.1 представлений приклад, що ілюструє метод суміщення незалежних операцій стосовно обчислення значення наступного виразу

$$Y = \frac{(A+B)*(C+D)}{(M*N)-(K/L)} \qquad (1.3)$$

який відповідає нагоді відсутності обмежень на потрібний час реалізації алгоритму.

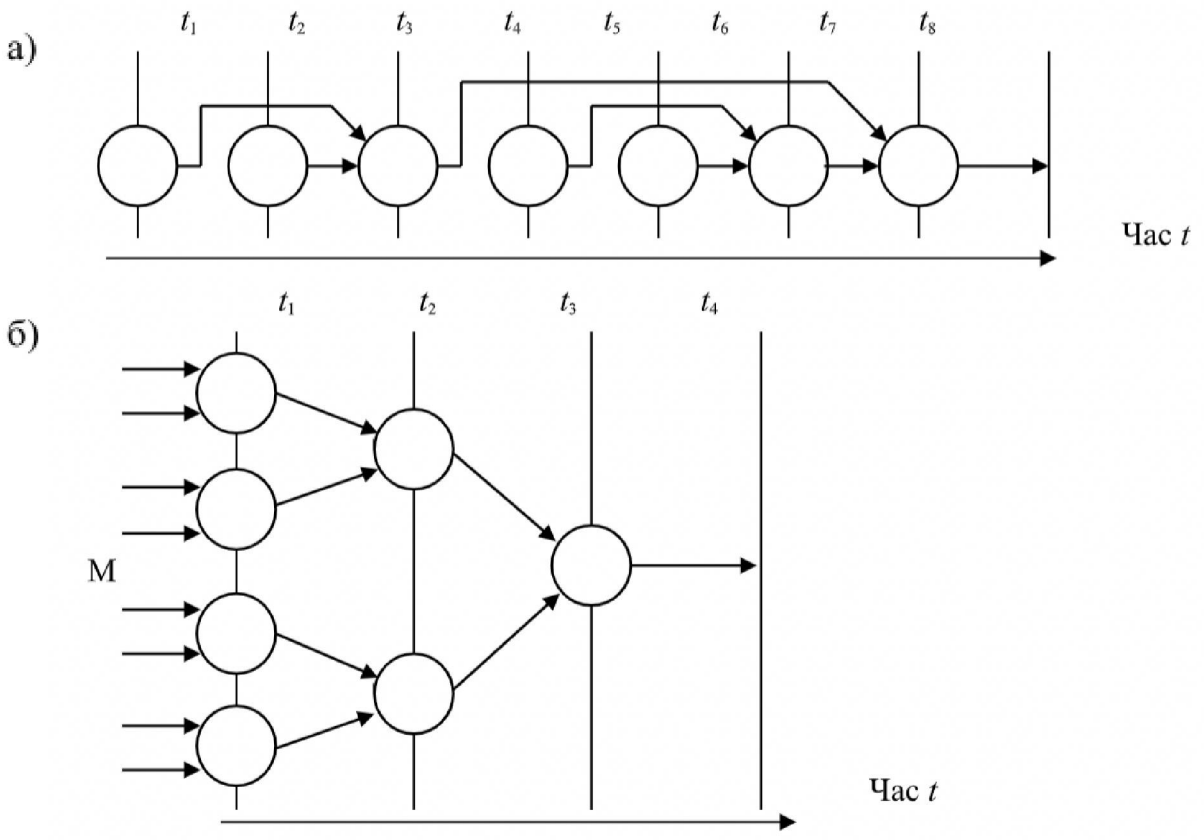

Рисунок 1.1 – Послідовна (а) та паралельна (б) часові моделі розв’язання задачі.

Рисунок 1.1(а) демонструє варіант послідовного виконання завдання, що відповідає нагоді відсутності обмежень на потрібний час реалізації алгоритму.

Рисунок 1.1(б) показує варіант розв'язання задачі з використанням поєднання незалежних операцій, що відповідає вимозі досягнення мінімально можливого часу розв'язання. Між цими крайніми варіантами є кінцева кількість проміжних варіантів, що характеризуються різним часом вирішення задачі та відповідно різними витратами обладнання на паралельне виконання алгоритму.

Рисунок 1.2 показує область можливих оцінок скорочення Тріш (%) часу вирішення завдань при використанні методу суміщення операцій (називається також глобальним паралелізмом): ця область обмежена двома крайніми випадками – виконання послідовного (що не розпаралелюється) алгоритму та реалізації алгоритму, що повністю розпаралелюється та складається з незалежних операцій.

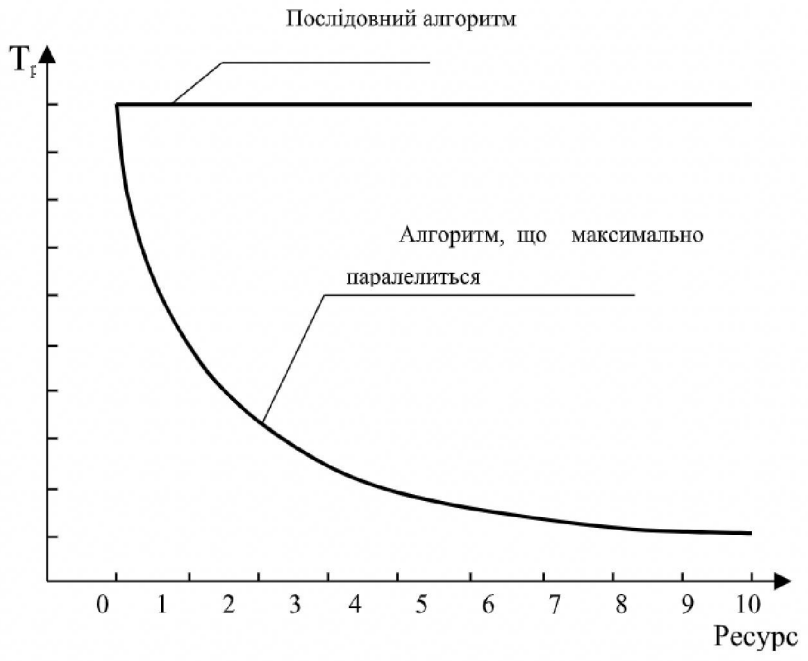

Рисунок.1.2 – Область скорочення часу виконання алгоритмів за рахунок суміщення незалежних операцій.

Метод поєднання незалежних операцій застосовується у всіх відомих багатопроцесорних ОС як одночасне виконання множиною процесорів сукупності «паралельних процесів», кожен із яких є фрагмент («нитка») операцій, які послідовно виконуються у межах загальної великої задачі.

Метод конвеєрної обробки

Метод Конвеєрної Обробки (КО) – це метод паралельної обробки даних, в якому в якості об'єктів виступають:

- алгоритми виконання операторів – загальноприйнятих операцій/функцій відомих мов програмування;
- об'єкти алгоритмічного рівня – фрагменти алгоритмів або алгоритми, що розглядаються як неподільне ціле.

Сутність методу полягає у виконанні для алгоритму наступних перетворень:

а) поділу часового алгоритму виконання операцій/функцій (на операційному рівні) або часового алгоритму розв'язання задачі (на алгоритмічному рівні) на "конвеєрні" фрагменти (F) рівної часової глибини ТК (такт конвеєра), такі, що кожен попередній фрагмент формує вхідні дані для суміжного наступного фрагмента, а параметр tн(Fj) початку кожного наступного фрагмента Fj визначається параметром tк(Fi) кінця попереднього фрагмента Fi(i,j $\in$ NF;NF= 0,1,...,nf 1; де nf=| NF| – кількість конвеєрних фрагментів або глибина конвеєра);

б) послідовної реалізації у часі фрагментів F0, F1, ..., Fnf-1 – у разі одиничної потужності (sd = 1) множини SD різних вхідних наборів даних алгоритму;

в) суміщення (при sd>1) інтервалів виконання "різнотипних" конвеєрних фрагментів алгоритму, що належать до різних наборів вхідних даних;

г) номери ρ поєднаних фрагментів Fρ, що відповідають вхідним наборам даних з номерами δ (δ=1,2,...,nf-1, nf, nf+1,…), задовольняють (у режимі роботи конвеєра) наступному співвідношенню ρ + δmod (nf +1) = nf.

Метод конвеєрної обробки характеризується введенням наборів вхідних даних з інтервалом дискретності ТК та виведенням результатів виконання алгоритму (для множини SD вхідних наборів даних) з інтервалом ТК.

На рис.1.3(а) та рис.1.3(b) представлений приклад, що ілюструє даний метод. Рис.1.3(а) демонструє варіант послідовного виконання одного алгоритму задачі для множини потужності різних наборів вихідних даних. На рис.1.3(b) показано варіант вирішення заданої множини завдань з використанням методу конвеєрної обробки.

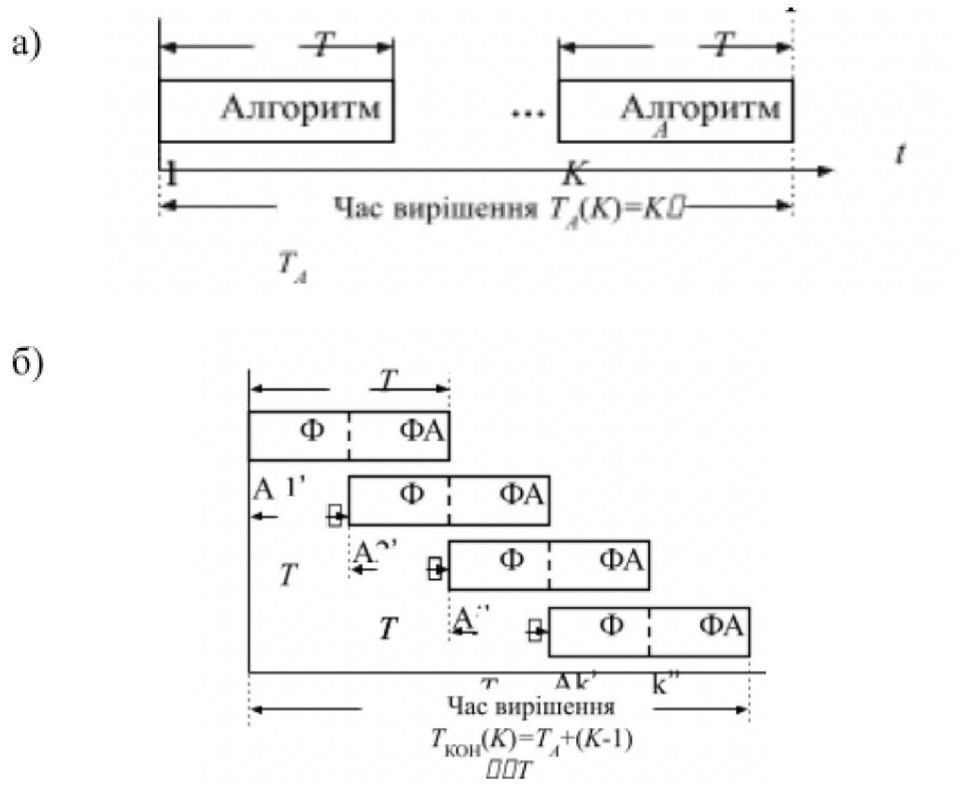

Рисунок.1.3 – Послідовне (а) та конвеєрне (б) виконання операцій/алгоритмів.

Метод грунтується на розбитті кожного алгоритму на n = TA/ΔT фрагментів (ΔT – інтервал введення даних, що задається), постановці

кожному фрагменту у взаємно однозначну відповідність пристрою («ступеня конвеєра»), послідовному з'єднанні зазначених частин відповідно до схеми з'єднання між собою відповідних фрагментів алгоритму, введення даних у конвеєрну структуру, що отримується, і виведення результатів рішення з тактовим інтервалом ΔT = TA/n.

Можливості зменшення часу виконання алгоритмів при конвеєрній обробці з тактовим інтервалом ΔT і часом одноразової реалізації кожного алгоритму, рівним TA (K – кількість виконуваних алгоритмів, nf = 6,12,24 – глибина / кількість фрагментів або ступенів конвеєра) показані на рис.1.4.

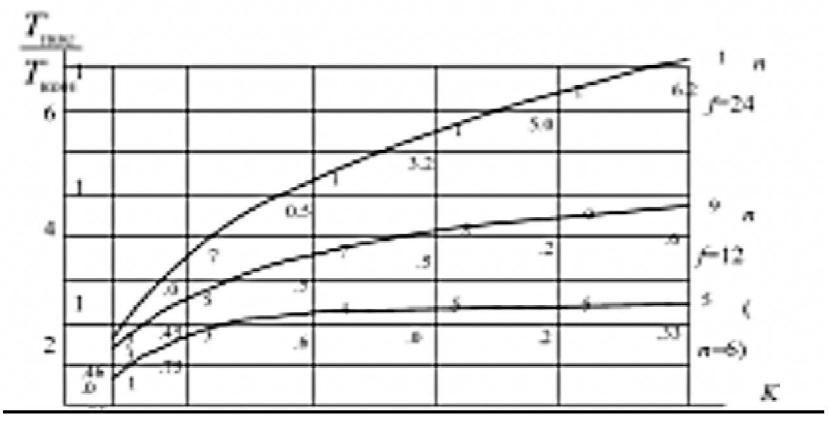

Рисунок. 1.4 – Зміна часу реалізації до алгоритмів при конвеєрній обробці залежно від кількості щаблів (глибини) nf конвеєра.

Метод суміші паралельних алгоритмів

Сутність методу паралельної суміші алгоритмів (СА) полягає у створенні на операційному рівні суміші завдань, використанні такої суміші для досягнення 100% завантаження обладнання та забезпечення за рахунок цього потенційно можливого підвищення ефективності реалізації аналізованої сукупності алгоритмів. Суміш алгоритмів формально розглядається і виконується як єдине завдання.

Суміш завдань використовується в тих випадках, коли окремо виконувані завдання, що використовують усі або частину розглянутих вище

методів паралельної обробки, не можуть забезпечити 100% завантаження обладнання та досягнення потенційного значення продуктивності.

На рис.1.5 та рис.1.6 представлені приклади, що ілюструють даний метод, при цьому для простоти викладу розглядається лише поєднання операцій суміші завдань A1 і A2 (без залучення інших методів паралелізму).

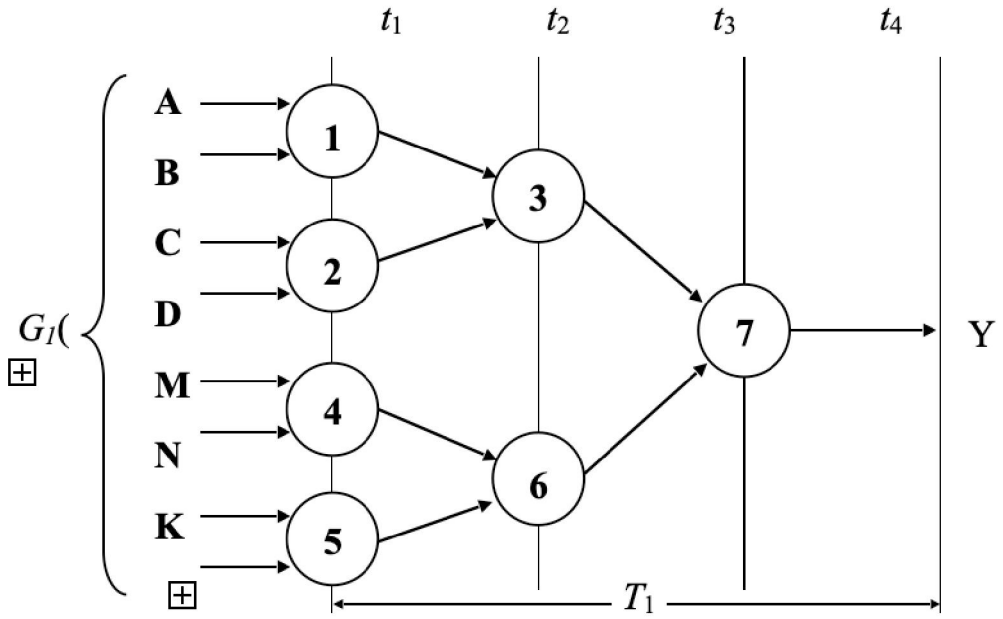

Рис.1.5 – Часовий паралельний граф G1(t) задачі A1.

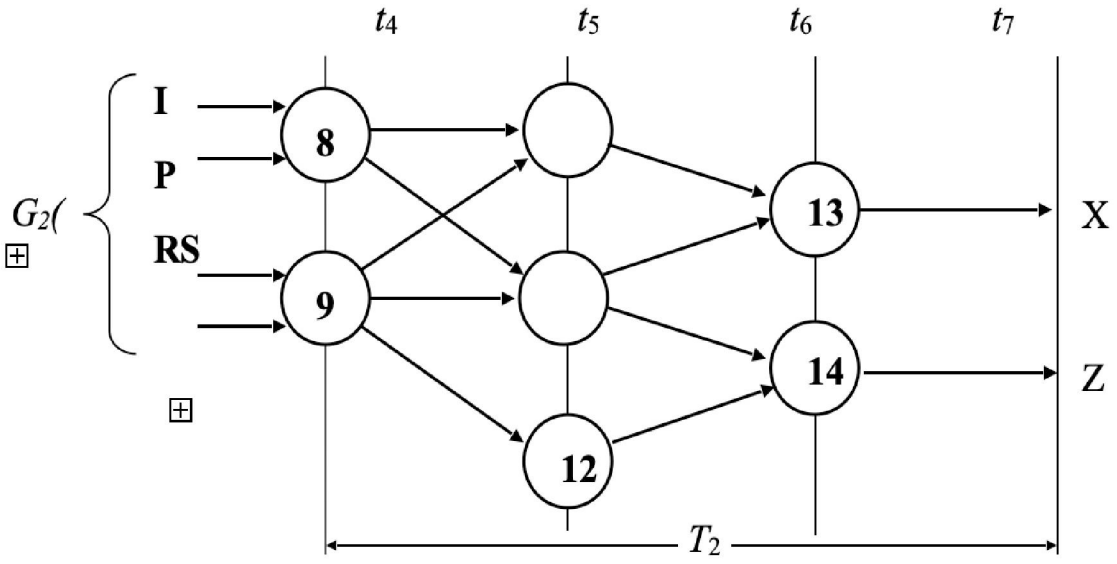

Рис.1.6 – Часовий паралельний граф G2(t) задачі A2.

На рис.1.5 та 1.6 представлен алгоритм у паралельній формі з наступними значеннями часу реалізації (у прикладі приймається, що обидві задачі та суміш задач реалізуються 4-х процесорною ОС): T1=3, T2=3, S1=0.58, S2=0.58 (S – коефіцієнт завантаження обладнання).

На рисунку 1.7 представлен часовий паралельний граф Gсм(t), побудований для суміші завдань і відображає одночасну реалізацію завдань, що розглядаються.

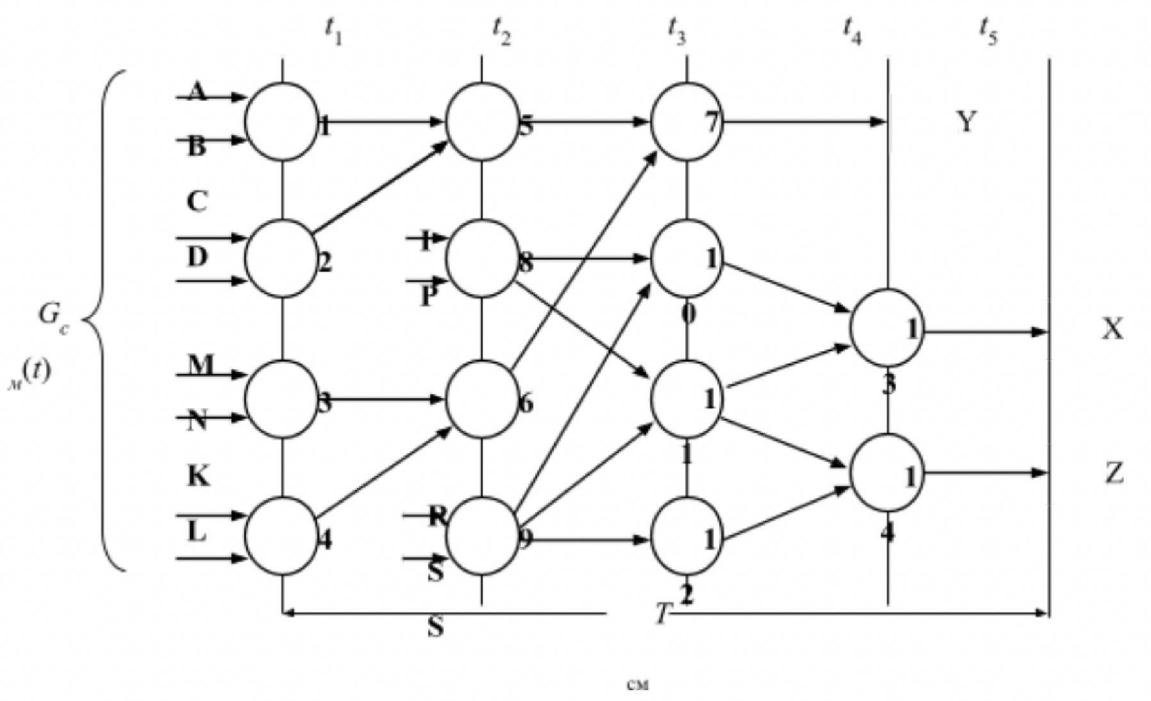

Рис.1.7 – Часовий паралельний граф Gсм(t) суміші завдань A1 та A2.

Кількість устаткування (в умовних універсальних процесорах) реалізації завдань відповідно до рис.1.5 і рис.1.6 дорівнює N = 4, сумарний час реалізації двох завдань T1,2 = 6 умовних тактів, коефіцієнт завантаження устаткування SΣ=0.58. Кількість устаткування реалізації суміші завдань відповідно до рис.1.7 дорівнює Nсм =4, час реалізації суміші дорівнює Tсм= 4 умовних такту, коефіцієнт завантаження устаткування Sсм = 0.87.

Добірка методів та головних особливостей їхнього застосування показано на рис.1.8 [6–8].

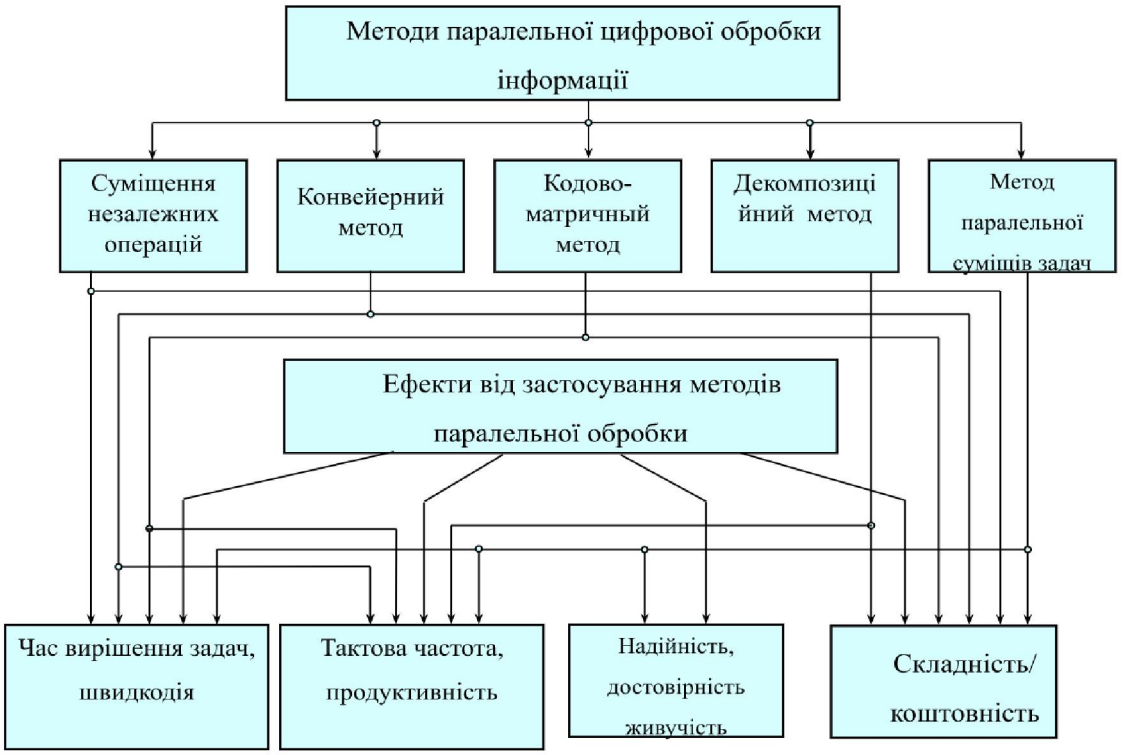

Рисунок 1.8 – Методи паралельної цифрової обробки інформації та ефекти від їхнього застосування [8].

1.3 Постановка задачі дослідження

У кваліфікаційній роботі буде необхідно розробити модель мережевого планування IT-проєкту базуючись на методах мультипаралельної обробки інформації.

Вхідні дані необхідні для виконання цієї задачі – це перелік робіт мережевого планування, у формі списка, їх тип, обмеження кожної з робіт і рамки виконання які залежать від зовнішніх факторів (кількість персоналу, їх кваліфікація, та ін.).

Облік системи: модель буде перетворювати перелік робіт у графи різних видів (послідовний та розгалужені), та проводити обчислення ризиків проєкту за допомогою методів мультипаралельної обробки інформації та методології PERT.

Система обмежень для даної задачі – це терміни виконання робіт, їх типи, ймовірні ризики проєкту.

Практична мета кваліфікаційної роботи – це менеджер задач, який зможе побудувати раціональну мережеву модель виконання робіт базуючись на вхідних даних та розраховує ризики проєкту.

Висновки за розділом 1

Мережеве планування – це важлива частина у процесі керування проєктами. Саме тому автоматизація процесу мережевого планування є актуальним і важливим питанням, вирішення якого може допомогти з пришвидшенням процесу розробки проєктів.

Методи мультипаралельної обробки інформації дозволяють перетворити послідовний алгоритм у паралельний, без сторонніх додатків.

Таким чином, використання методів мультипаралельної обробки інформації є доцільним для побудови мережевих графів. Необхідно розглянути існуючі програмні рішення побудови мережевих графів, провести аналіз методів ймовірної оцінки тривалості робіт і розробити модель програмного застосунку.

РОЗДІЛ 2.
РОЗРОБКА ПРОГРАМНОЇ МОДЕЛІ МЕРЕЖЕВОГО ПЛАНУВАННЯ ВИКОНАННЯ ІТ- ПРОЄКТУ

2.1 Методи ймовірнісної оцінки тривалості та вартості робіт

Оцінка вартості ІТ-проєкту допомагає спрогнозувати вартість проєкту. Команда проєкту використовує цей прогноз, щоб вирішити, чи має сенс виконувати проєкт. Кошторис включає аналіз необхідних ресурсів для проєкту.

Перед початком проєкту керівнику та спонсору проєкту потрібні дані, щоб визначити, чи є вартість проєкту ефективним використанням ресурсів з урахуванням бізнес-цілей. Оцінка вартості надає огляд необхідних ресурсів і обсягу потенційного проєкту, а також допомагає команді визначити, чи доступні необхідні ресурси для завершення проєкту [9].

Оцінка вартості ІТ-проєкту схожа на загальну оцінку вартості проєкту, але кожен ІТ-проєкт належить до однієї з двох основних областей: апаратного забезпечення та програмного забезпечення. Апаратні проєкти передбачають фізичні структури та постійні рішення, які включають модернізацію існуючої інфраструктури (або обладнання, або операційної системи, або хмарного сховища, або зміни будь-яких великомасштабних структур). Проєкти програмного забезпечення зосереджені на конкретному продукті, який з часом змінюється, наприклад, розробка та запуск нової програми або створення веб-сайту.

Кожен проєкт оцінки вартості ІТ відрізняється, але кожен вимагає чотирьох характеристик для створення якісного документу по суті. Ці характеристики кодифіковано та вдосконалено Управлінням інформаційної звітності уряду США як рекомендації щодо вдосконалення політики та практики агентства:

- Комплексність: витрати мають бути включені протягом усього життєвого циклу – від ініціації до закриття проєкту. Кошторис має повністю визначати проєкт, включаючи графік і технічні потреби. У документі має бути відображено вартість і графік, необхідні для виконання визначених результатів. Якщо є обмежені історичні дані, які можна використовувати як прецедент, у документі слід це зазначити та включати припущення.
- Добре задокументований: кошторис витрат спирається на детальну документацію. У кошторисі має бути описано, як ви отримали інформацію та розподілили кошти для виконання проєкту. Використовується структура розподілу робіт (WBS) і фіксуються вихідні дані, розрахунки та методологію, яка використовується для збирання вартості проєкту, щоб зацікавлені сторони могли перевірити будь-яку інформацію, яка надається, і відтворити процес для майбутніх проєктів. Документ потребує технічної основи, яка вимірює прогрес проєкту, і опис того, як організовані дані. Нарешті, оцінка витрат має бути переглянута та прийнята керівництвом, щоб підтвердити точність.
- Точність: треба переконатися, що створено неупереджену оцінку витрат. Необхідно створити оцінку, оцінивши найімовірніші витрати (з поправками на інфляцію) на основі історичних даних, включаючи подібні минулі проєкти. Якщо можливо, треба оновлювати документ фактичними витратами та графіком у міру просування проєкту.
- Надійний: обмеження оцінки мають бути включені через невизначеність, упередженість або припущення щодо даних. Треба проаналізувати чутливості, щоб показати перегляди на основі цих знайдених припущень, а також аналіз ризиків. Використовується діапазон витрат, від найнижчої до найвищої розрахункової ціни, щоб передати впевненість у досягненні результату проєкту та надати варіанти, враховуючи технічні, вартісні та графікові ризики. Перехресна перевірка результатів незалежною групою або внутрішньою командою [10].

На додаток до побудови вичерпної, добре задокументованої, точної та достовірної оцінки, керівник проєкту повинен пояснити мету документа зацікавленим сторонам і спонсору проєкту, а також те, що процес оцінки не передбачає, включаючи таке:

- Вирішення питань керування або контролю за проєктом, що призводять до розповзання обсягу.
- Прогнозування впливу проєкту на консенсус зацікавлених сторін.
- Отримання кращих результатів, ніж запропонована оцінка.
- Покладатися на документ як на єдиний фактор у прийнятті рішень.
- Використовуючи той самий метод для кожного проєкту.

Існує кілька поширених моделей оцінки вартості IT-проєктів, які також називають методології. Універсального рішення не існує; необхідно обрати методологію, визначивши потреби проєкту за допомогою IT-евристики, наприклад розмір проєкту, складність, необхідні ресурси та час підготовки.

Наприклад, деякі моделі базують оцінку на найменшому будівельному блоці, такому як рядок коду, тоді як інші порівнюють середній розмір попередніх проєктів. Наведені нижче евристики використовуються для того, щоб визначити, яка модель найкраще підходить для проєкту:

- Складність: Оцінка складності проєкту. Враховуються як суб'єктивні (на основі минулого досвіду), так і об'єктивні (наприклад, обмежені ресурси) фактори.
- Надійність методу: Хоча не обов'язково існує бажаний метод, деякі методи є більш надійними, ніж інші, за оцінкою їх точності.
- Необхідні ресурси: Деякі проєкти мають велику кількість історичних даних, до яких легко отримати доступ через загальнодоступні бази даних. Інші можуть мати суперечливі або неповні дані, що вплине на здатність проводити ретельні дослідження. Також можна зіткнутися з

проблемами персоналу, наприклад, працювати з новою або не досвідченою командою.

- Час на підготовку: приготування деяких методів потребує більше часу. Розклад, розмір і складність вашого проєкту впливають на робочі часові обмеження в рамках вашого виділеного графіка.
- Розмір проєкту: вимірювання розміра проєкту за допомогою одного з кількох поширених методів, таких як функціональні точки (одиниця вимірювання для приблизної функціональної бізнес-цінності продукту), визначення розміру футболки (інструмент оцінки, що визначає відносний обсяг роботи), або рядки коду.

Залежно від проєкту та обставин може знадобитися метод, який дає швидші результати. Основними цілями кошторису є надійність і точність.

Існує дві категорії методів оцінки IT-проєктів: алгоритмічні та неалгоритмічні . Алгоритмічні методи включають рівняння для кількісної оцінки зусиль. Неалгоритмічні методи використовують дані та аналіз [11–12].

Алгоритмічні методи

Алгоритмічні методи оцінки є формалізованими моделями оцінки. Ці методи використовують механічний процес, найчастіше формулу, створену на основі шаблонів в історичних даних, і є найбільш стандартизованими.

- Конструктивна модель витрат (COCOMO): Цей метод оцінює зусилля, витрати та графік за допомогою трирівневого процесу: основного, проміжного та детального. Для побудови він використовує складні алгоритмічні формули, отримані з історичних даних проєкту. Цей параметричний метод є одним із найточніших методів оцінки вартості проєкту та часто використовується для поступової розробки програмного забезпечення.

- Функціональна точка (FP): це галузевий стандарт для програмного забезпечення для визначення розміру. FP – це одиниця вимірювання, яка показує, наскільки функціональна інформаційна система для користувача в бізнесі. Він використовує минулі проєкти для розрахунку погодинної або грошової вартості.

Неалгоритмічні методи

Неалгоритмічні оцінки аналізують зібрані історичні дані, щоб отримати найкращу оцінку.

- Обчислення витрат на основі діяльності: цей метод враховує діяльність людей і обладнання, необхідних для доставки продукту. Цей стиль оцінює непрямі витрати пропорційно типу роботи, а також потреби в ресурсах для кожного виду діяльності.

- Аналогія: цей метод використовує попередні проєкти, щоб порівняти та оцінити вартість, розмір, складність або графік. Метод враховує подібності, відмінності та фактичні результати. Вам не потрібен експерт для цього методу, але ви повинні мати повністю точні історичні дані.

- Знизу вгору: ця методика починає оцінку з деталей нижнього рівня структури розподілу робіт (WBS) і переходить до вищого рівня. Цей метод забезпечує високий рівень точності завдяки своїй увазі до деталей.

- Delphi: це структурований і систематичний піднабір методів експертної оцінки для прогнозування за допомогою групи експертів. Комісія зважує свої думки та обговорює фактори, які впливають на оцінку. Метод передбачає, що точність структурованих групових взаємодій є більш надійною, ніж неструктурованих групових взаємодій. Доступні інші підмножини адаптацій Delphi, наприклад міні-Delphi або оцінка-розмова-оцінка (ETE) . Цей метод усуває політику та упередженість оцінки, але може зайняти багато часу.

- Планування покеру: у цьому ігровому методі команди використовують колоду карт для досягнення консенсусу. Це допомагає уникнути прив'язки , яка є ухилом до першої оцінки. Цей метод є різновидом широкосмугового Delphi, який найчастіше використовується в Scrum і XP у розробці Agile.
- Група процесів: це ще один широкосмуговий варіант Delphi. Тут команда оцінювання колективно створює WBS і деталізує будь-які припущення. На цій зустрічі окремі члени команди оцінюють зусилля та використовують ці прогнози на другій зустрічі, де досягають консенсусу.
- 3-точкова або техніка оцінки та перегляду програми (PERT): Цей метод оцінює невизначеності та ризики за допомогою наступних трьох точок оцінки:
  - Найімовірніше (M): Найбільш реалістична оцінка.
  - Оптимістичний (O): Оцінка найкращого сценарію.
  - Песимістичний (P): оцінка найгіршого сценарію.

Команда застосовує бали до двох формул розподілу, щоб обчислити оцінку.

Детальніше про методологію PERT.

PERT призначений для роботи над складними проєктами, які реалізуються за допомогою послідовних робіт або паралельно з другими проєктами. Мета PERT – це завершення проєктів в позначений срок та в рамках бюджету, так саме як і точна оцінка обсягу робіт на затрат на етапі визначення обсягів робіт.

Визначення обсягу роботи – це один з головних етапів роботи методології PERT. На цьому етапі створюється план дій для проєкту. Ця методологія враховує усі фактори, які ведуть до негативного результату або до зростання часу, яке необхідне для реалізації проєкту, і головні етапи його створення. Проводячи порівняння з методикою визначення об'ємів робіт, яку

використовує робоча група, PERT є вичерпною та враховує усе, від керування ресурсами до ефективності.

Події у методології PERT є етапами робочого процесу. Вони підпорядковані своїм правилам, наприклад вони не можуть бути розпочаті, поки не досягне закінчення кожна попередня подія та не вимагають ресурсів.

Властивості цих подій наступні:

1. Подія PERT це початок/завершення однієї або декількох дій;
2. Подія знаходиться безпосередньо перед іншою подією або подіями;
3. Подія відбувається безпосередньо після іншої події або подій.

Етапи проєктного планування PERT наступні:

1. визначити завдання;
2. визначити правильну послідовність;
3. оцінити терміни;
4. створити діаграму PERT;
5. встановити запас;
6. виконати оцінку критичного шляху.

Користуючись методологією PERT, під час розрахунку мережевих моделей, час робіт є випадковим, він підпорядковується своїм власним законам розподілу та володіє своїми числовими характеристиками, такі як:

1. середня тривалість роботи t_{i-j}^{cp};
2. дисперсія оцінки тривалості роботи (дисперсія роботи) σ_{i-j}^{2}.

Для визначення тривалості робіт на проєкті враховуються три оцінки часу:

1. песимістична t_{i-j}^{p} – оцінка тривалості при несприятливих умовах;

2. оптимістична t_{i-j}^{0} – оцінка тривалості при сприятливих умовах;

3. найбільш ймовірна t_{i-j}^{mv} – оцінка тривалості при звичайних умовах.

Середня тривалість роботи t_{i-j}^{cp} є математичним очікуванням:

$$t_{i-j}^{cp} = \frac{\left(t_{i-j}^{o} + 4 * t_{i-j}^{mv} + t_{i-j}^{p}\right)}{6} \quad (2.1)$$

Де t_{i-j}^{o} – оптимістичний час, t_{i-j}^{p} – песимістичний час, t_{i-j}^{mv} – очікуваний час.

Крім середньої тривалості роботи ще є поняття фактичної тривалості.

Фактична тривалість роботи може відрізнятися від очікуваної, через це необхідно знати дисперсію тривалості роботи:

$$\sigma_{i-j}^{2} = \left[\frac{t_{i-j}^{p} - t_{i-j}^{o}}{6}\right]^{2} \quad (2.2)$$

Середня тривалість роботи є найбільш вірогідную тривалістю роботи.

Для розрахунку вірогідності завершення роботи за певний термін необхідно зробити наступне:

1. Розрахувати середнє квадратичне відхилення по заданій формулі:

$$\sigma_{L\text{кр}} = \sqrt{\Sigma \sigma_{\text{кр}}^{2}} \ . \quad (2.3)$$

де $\sigma_{\text{кр}}$ – дисперсія роботи на критичному шляху

2. Обчислити аргумент функції інтеграла ймовірності по заданій формулі:

$$Z = \frac{\left(T_{L\text{кр}}^{\text{дир}} - T_{L\text{кр}}^{\text{ср}}\right)}{\sigma_{L\text{кр}}}, \quad (2.4)$$

де $T_{L\text{кр}}^{\text{дир}}$ – директивний час, за який проєкт має бути закінчений.

3. Знайти значення Ф(Z) по таблицям нормального розподілу (Додаток Г).

4. Обчислити вірогідність виконання проєкту за директивний час:

$$P\left(T_{L\text{кр}}^{\text{дир}} \leq T_{L\text{кр}}^{\text{ср}}\right) = [0,5 + 0,5 * \Phi(Z)] \quad (2.5)$$

Проаналізувавши існуючі методології ймовірнісної оцінки тривалості та вартості робіт, було обрано методологію PERT, через те, що вона вдовільняє усім потребам, є не дуже складною та є досить простою для перетворення у програмну реалізацію [13–16].

2.2 Аналіз існуючих програмних технологій для обчислення паралельних алгоритмів

Для паралелізму важливо розділити проблему на підрозділи, які не залежать від інших підрозділів (або менш залежні). Проблема, де субодиниці повністю незалежні від інших субодиниць, називається незручно паралельною.

Наприклад, поелементна операція над масивом. У цьому випадку операція повинна знати про конкретний елемент, який вона обробляє в даний момент.

За іншим сценарієм, проблема, яка розділена на підрозділи, повинна спільно використовувати деякі дані для виконання операцій. Це призводить до проблеми з продуктивністю через вартість зв'язку.

Існує два основних способи роботи з паралельними програмами:

Спільна пам'ять

У спільній пам'яті підблоки можуть спілкуватися один з одним через один простір пам'яті. Перевагою є те, що вам не потрібно явно обробляти

зв'язок, оскільки цього підходу достатньо для читання або запису зі спільної пам'яті. Але проблема виникає, коли декілька процесів отримують доступ і змінюють те саме розташування пам'яті одночасно. Цього конфлікту можна уникнути за допомогою методів синхронізації.

Розподілена пам'ять

У розподіленій пам'яті кожен процес повністю відокремлений і має власний простір пам'яті. У цьому сценарії зв'язок між процесами обробляється явно. Оскільки зв'язок відбувається через мережевий інтерфейс, він дорожчий порівняно зі спільною пам'яттю.

Можна виділити найпоширеніші технології паралеьльної обробки інформації:

- MPI
- OpenMP
- Python multiprocessing

Інтерфейс MPI

Інтерфейс передачі повідомлень (MPI) – це стандартизований засіб обміну повідомленнями між декількома комп'ютерами, на яких виконується паралельна програма в розподіленій пам'яті (рис 2.1).

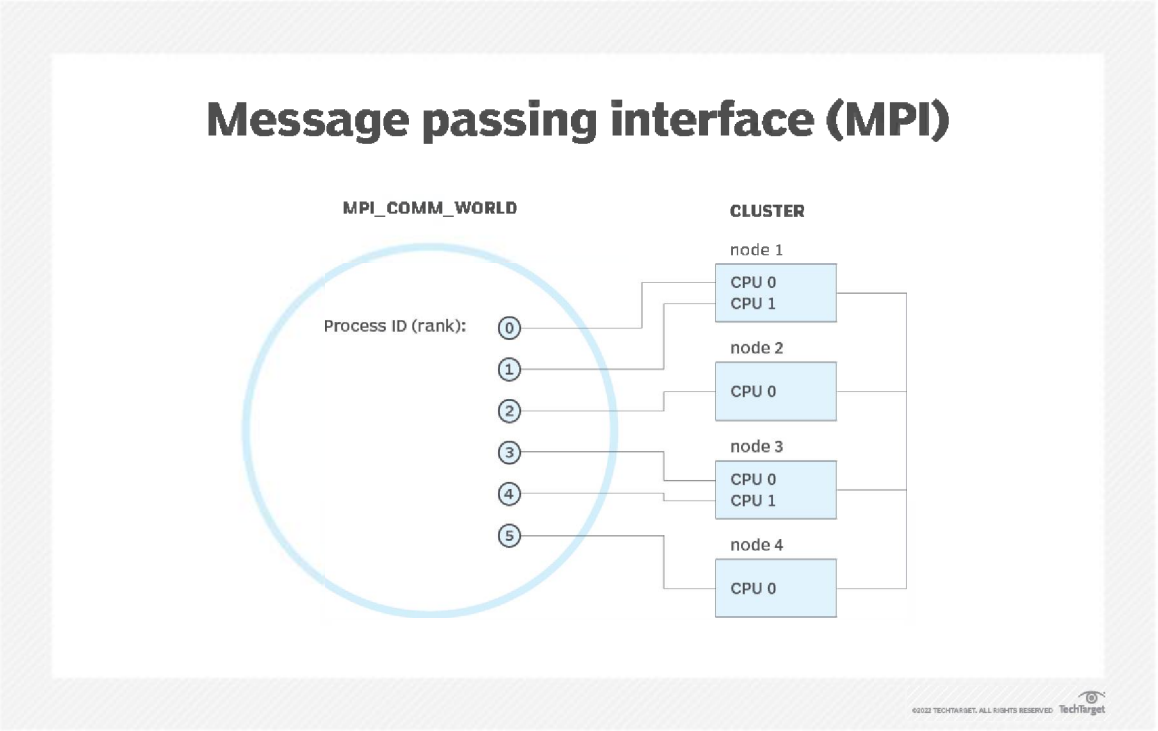

Рисунок 2.1 – Уявлення MPI.

У паралельних обчисленнях кілька комп'ютерів або навіть кілька ядер процесора в одному комп'ютері називають вузлами. Кожен вузол у паралельній структурі зазвичай працює над частиною загальної обчислювальної проблеми. Завдання полягає в тому, щоб синхронізувати дії кожного паралельного вузла, обмінюватися даними між вузлами та надавати команди та контролювати весь паралельний кластер. Інтерфейс передачі повідомлень визначає стандартний набір функцій для цих завдань. Сам термін передача повідомлення зазвичай стосується надсилання повідомлення до об'єкта, паралельного процесу, підпрограми, функції або потоку, яке потім використовується для запуску іншого процесу.

MPI не схвалений як офіційний стандарт жодною організацією зі стандартизації, наприклад Інститутом інженерів з електротехніки та електроніки (IEEE) або Міжнародною організацією стандартизації (ISO), але зазвичай він вважається галузевим стандартом і формує основа для більшості комунікаційних інтерфейсів, прийнятих програмістами паралельних обчислень. Розробники також створили різні реалізації MPI.

MPI визначає корисний синтаксис для процедур і бібліотек на мовах програмування, включаючи Fortran, C, C++ і Java.

Інтерфейс передачі повідомлень надає такі переваги:

1. Стандартизація. MPI замінив інші бібліотеки передачі повідомлень, ставши загальновизнаним галузевим стандартом.
2. Розробка широкою комісією. Хоча MPI може не бути офіційним стандартом, це все ж загальний стандарт, створений комітетом постачальників, розробників і користувачів.
3. Портативність. MPI реалізовано для багатьох архітектур розподіленої пам'яті, тобто користувачам не потрібно змінювати вихідний код під час перенесення програм на різні платформи, які підтримуються стандартом MPI.
4. Швидкість. Реалізація зазвичай оптимізована для обладнання, на якому працює MPI. Реалізації постачальників також можуть бути оптимізовані для власних апаратних функцій.
5. Функціональність. MPI розроблено для високої продуктивності на масивних паралельних машинах і кластерах. Базова реалізація MPI-1 має понад 100 визначених процедур.

Деякі організації також можуть розвантажити MPI, щоб зробити свої моделі програмування та бібліотеки швидшими.

Історія та версії MPI

Невелика група дослідників в Австрії почала обговорювати концепцію інтерфейсу передачі повідомлень у 1991 році. Семінар зі стандартів передачі повідомлень у середовищі розподіленої пам'яті, спонсорований Центром досліджень паралельних обчислень, був проведений через рік у Вільямсбурзі, Va. Для створення процесу стандартизації було створено робочу групу.

У листопаді 1992 року був створений проєкт для MPI-1, а в 1993 році стандарт був представлений на конференції Supercomputing '93. З додатковими відгуками та змінами версія MPI 1.0 була випущена в 1994 році.

Відтоді MPI був відкритий для всіх членів спільноти високопродуктивних обчислювальних машин, включаючи понад 40 організацій-учасників.

Старіший стандарт MPI 1.3, який отримав назву MPI-1, надає понад 115 функцій. Більш пізній стандарт MPI 2.2, або MPI-2, пропонує понад 500 функцій і значною мірою зворотно сумісний з MPI-1. Однак не всі бібліотеки MPI забезпечують повну реалізацію MPI-2. MPI-2 включив новий паралельний ввід-вивід, динамічне керуванням процесами, а також віддалені операції з пам'яттю. Стандарт MPI-3, випущений у листопаді 2014 року, покращує масштабованість, підвищує продуктивність, включає багатоядерну та кластерну підтримку та взаємодіє з більшою кількістю програм. У 2021 році The MPI Forum випустив MPI 4.0. Він представив розділені комунікації, новий інтерфейс інструментів, Persistent Collectives та інші нові доповнення. MPI 5.0 зараз знаходиться в стадії розробки.

Термінологія MPI: ключові поняття та команди

Наведений нижче список містить деякі основні поняття та команди MPI:

- Comm. Це об'єкти-комунікатори, які з'єднують групи процесів у MPI. Команди комунікатора надають закритому процесу незалежний ідентифікатор, організовуючи його як упорядковану топологію. Наприклад, команда для базового комунікатора включає MPI_COMM_WORLD.
- Color. Це призначає колір процесу, і всі процеси з таким самим кольором розташовані в одному комунікаторі. Команда, пов'язана з кольором, включає MPE_Make_color_array, який змінює доступні кольори.
- Key. Ранг або порядок процесу в комунікаторі базується на ключі. Якщо двом процесам надано однаковий ключ, порядок базується на ранзі процесу в комунікаторі.
- Newcomm. Це команда для створення нового комунікатора. MPI_COMM_DUP є прикладом команди для створення дублікату повідомлення з тими самими фіксованими атрибутами.

- Derived data types. Для функцій MPI потрібна специфікація того, який тип даних надсилається між процесами. MPI_INT, MPI_CHAR і MPI_DOUBLE допомагають у попередньому визначенні констант.
- Point-to-point. Це надсилає повідомлення між двома певними процесами. MPI_Send і MPI_Recv є двома поширеними методами блокування для повідомлень «точка-точка». Блокування означає, що процеси надсилання та отримання чекають, поки повне повідомлення буде правильно надіслано та отримано, щоб надіслати та завершити повідомлення.
- Collective basics. Це колективні функції, які потребують зв'язку між усіма процесами в групі процесів. MPI_Bcast є прикладом такого, який надсилає дані з одного вузла до всіх процесів у групі процесів.
- One-sided. Цей термін зазвичай використовується для позначення форми операцій зв'язку, включаючи MPI_Put, MPI_Get і MPI_Accumulate. Вони конкретно стосуються запису в пам'ять, читання з пам'яті та скорочення операцій з тією самою пам'яттю для різних завдань [15].

Інтерфейс OpenMP

OpenMP (Open Multi-Processing) – це інтерфейс прикладного програмування (API) для багатопроцесорного програмування спільної пам'яті на C, C++ і Fortran. OpenMP-розпаралелена програма запускається як послідовна програма, яка працює на одному обчислювальному ядрі. За вказівкою програміста програма породжує кілька потоків, які можуть працювати одночасно на окремих ядрах. Таким чином, роботу можна розподілити, щоб залучити більше ресурсів.

Треба зауважити, що нещодавно стандарт OpenMP було розширено, щоб дозволити перенесення обчислень на графічні процесори та інші прискорювачі. Однак ще не всі компілятори підтримують цю функцію, і існує подібний конкуруючий стандарт під назвою OpenACC, який стосується того самого випадку використання. Ми обмежимо цей урок багатоядерними обчисленнями без розвантаження.

Переваги:

1. підтримується великою кількістю багатоядерних архітектур спільної пам'яті (практично всі сучасні ЦП мають кілька ядер) і прискорювачів;
2. те саме джерело можна використовувати для компіляції коду OpenMP і не-OpenMP;
3. може бути легше реалізувати, ніж розпаралелювання MPI – кілька рядків коду можуть призвести до значного прискорення;
4. може бути більш ефективним, ніж MPI, оскільки дані в пам'яті можуть спільно використовуватися між кількома потоками.

Недоліки:

1. обмежені ресурсами одного обчислювального вузла;
2. можна створити «умови змагання», коли кілька потоків перезаписують роботу один одного, і такі помилки може бути важко налагодити;
3. необхідно остерігатися зовнішніх функцій, які не є «потоково-безпечними», оскільки ці функції чутливі до умов гонки.

Користування OpenMP

Щоб використовувати OpenMP, потрібно необхідно подати команду компілятору створювати потоки, додаючи директиви до коду, вказуючи, де і що розпаралелювати встановити кількість потоків, які будуть використовуватися під час виконання.

Директиви

Розпаралелювання з OpenMP реалізується за допомогою директив, які записуються як прагми (C/C++) або спеціально відформатовані коментарі (Fortran). OpenMP також надає додатковий інтерфейс прикладної програми (API), який дозволяє програмі налаштовувати та запитувати середовище виконання, наприклад, щоб дізнатися, скільки потоків працює паралельно та який ідентифікатор потоку виконує певний паралельний розділ. Для отримання додаткової інформації перегляньте останній стандарт OpenMP.

Директиви OpenMP у вихідному коді інтерпретуються компілятором. Той самий вихідний код можна використовувати для створення послідовної

або потокової версії програми, просто ввімкнувши або вимкнув перемикач компілятора OpenMP, а компілятор, що не є OpenMP, ігноруватиме директиви як невідомі прагми (C/C++) або як коментарі (Fortran).

Директиви OpenMP завжди починаються з:

- C/C++: #pragma ompr;
- Fortran (вільна форма): !$omp;

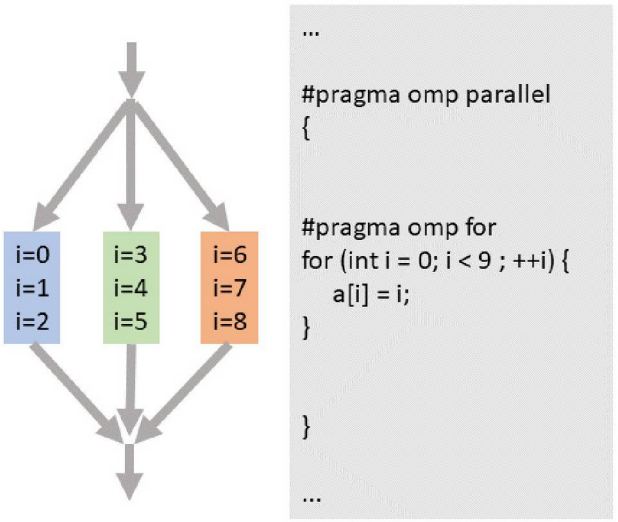

Рисунок 2.2 – Код та уявлення процесу розгалуження OpenMP.

За директивами йдуть назви директив і пункти, що керують розпаралелюванням і обробкою даних. Директиви OpenMP можуть складатися з кількох операторів і можуть бути розширені до кількох рядків за допомогою символів продовження рядка, таких як & (Fortran) або \ (C/C++) у кінці рядка.

Існують різні способи розподілу навантажень для паралельного виконання, найпоширенішим є паралельний цикл, представлений на рис 2.2.

Програма завжди запускається в послідовному режимі в одному потоці (єдина стрілка вгорі). За запитом створюється/зароджується кілька потоків (кілька стрілок угорі). У цьому конкретному випадку кількість потоків становить 3 (кольорові поля), і кожен потік виконує три ітерації (всього 9 ітерацій). Результати зберігаються в окремих елементах масиву a для кожного

індексу циклу i, тому ми не створюємо умови змагання, коли цикл виконується паралельно. Потім програма відновлює роботу в одному потоці (єдина стрілка внизу).

Оскільки OpenMP базується на моделі програмування спільної пам'яті, більшість змінних є спільними за замовчуванням. Інші змінні, такі як індекс циклу, мають бути приватними, тобто змінна може приймати різні значення для кожного потоку. Програміст визначає, які змінні є приватними, а які спільними [16].

Python multiprocessing

Multiprocessing – це бібліотека, яка підтримує процеси породження, подібного до потокового модуля. Multiprocessing є пакетом, який пропонує локальний і віддалений паралелізм, та ефективно обходить глобальне блокування інтерпретатора, тому що, використовує підпроцеси замість потоків. Через це багатопроцесорний модуль може дозволити програмісту використовувати багато процесорів на машині.

Multiprocessing є модулем, який не має аналогів у модулі потоків. Прикладом цього служить об'єкт Pool, котрий пропонує досить зручний засіб розгалуження виконання алгоритму для багатьох вхідних значень і розділяє вхідні дані між процесами.

У багатопроцесорній обробці процеси створюються об'єктом Process та викликом його методу start(). За цей процес відповідає API потоків.Thread.

Контексти та методи запуску

Залежно від платформи багатопроцесорність підтримує три способи запуску процесу.

В Unix системах використання методів запуску також запускає і процес відстеження ресурсів, котрий відстежує незв'язані системні ресурси, які створені програмними процесами. Після закінчення роботи усіх процесів засіб відстеження ресурсів прибирає об'єкт, який залишився. Їх не повинно бути, та якщо процес було знищено через сигналом, є ймовірність, що є «витік» ресурсів [17].

2.3 Аналіз існуючих програмних рішень для побудови мережевих графів та розрахунку ризиків проєкту

Програми побудови мережевих графів застосовуються менеджерами проєктів, за для відображення графів робіт, на етапах планування і виконання робіт. Це не тільки дозволяє поліпшити розуміння працівників, які працюють над проєктом, а ще і для керівництва, яке може проаналізувати цей граф та додати, або прибрати ту чи іншу роботу, змінюючи план робіт.

Зараз дуже поширеними є програми, або веб-додатки, які відповідають за структурне планування. Структурне планування – це етап планування, коли розписується лист задач, їхні зв’язки, терміни роботи. Нижче приведені найпопулярніші з них.

Програма Excel є продуктом компанії Майкрософт, яка працює з електронними таблицями та слугує для впорядкування даних і чисел користуючись функціями та формулами. Excel використовується по всьому світі і застосовується підприємствами різних розмірів [18].

На рисунку 2.3 можна побачити стандартний приклад інтерфейсу програми Excel.

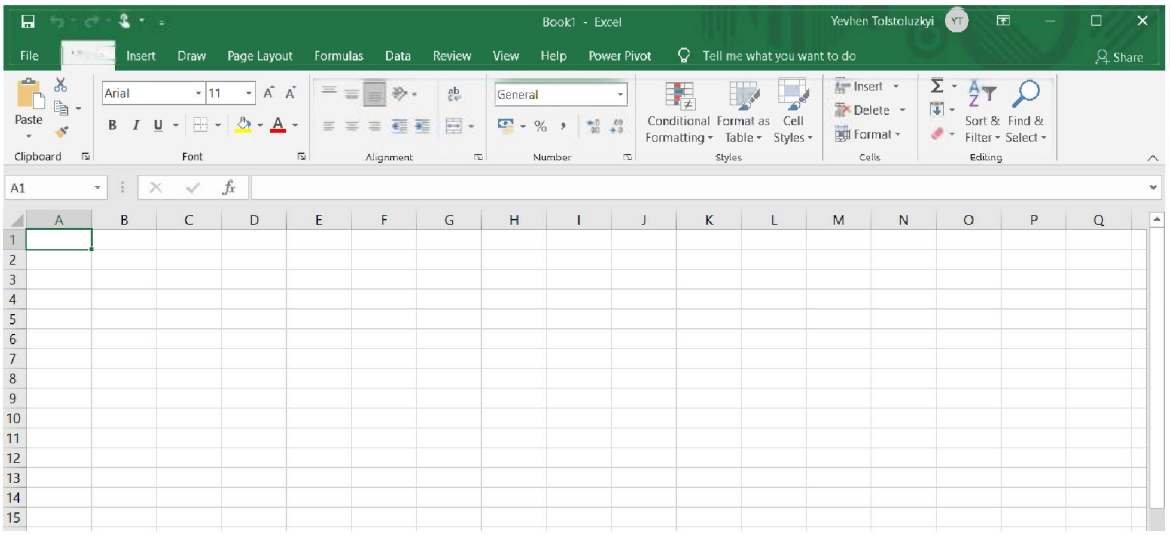

Рисунок 2.3 – Інтерфейс Excel.

Недоліки порівняно з моделлю, що пропонується:

1) низька швидкість обробки даних;
2) є платною;
3) потреба у додаткових функціях;
4) потребує більше роботи для підготовки даних.

Переваги:

1) багато додаткового функціоналу;
2) досить зрозумілий інтерфейс.

Також популярним додатком у середі IT-проєктів є додаток Jira. Це низка продуктів, які допомагають командам в організації роботи. Jira містить у собі продукти і варіанти розгортання, які спеціально розроблені для IT-команд, бізнес-команд і т. ін.

Застосовуючи продукти та додатки, створені на платформі Jira, команди можуть займатися плануванням, керуванням, та відстежувати та призначати роботу і створювати звіти. В програмі можна виконувати майже всю необхідну роботу: від agile-розробки та підтримки клієнтів, до організації стартапів і великих компаній [19].

Ці програмні продукти мають у собі вбудовані шаблони проєктів для різноманітних цілей і ефективно співпрацюють один з одним, завдяки чому всі учасники команди можуть отримати змогу зручно та ефективно взаємодіяти. Інтерфейс має вид зображений на рисунку 2.4.

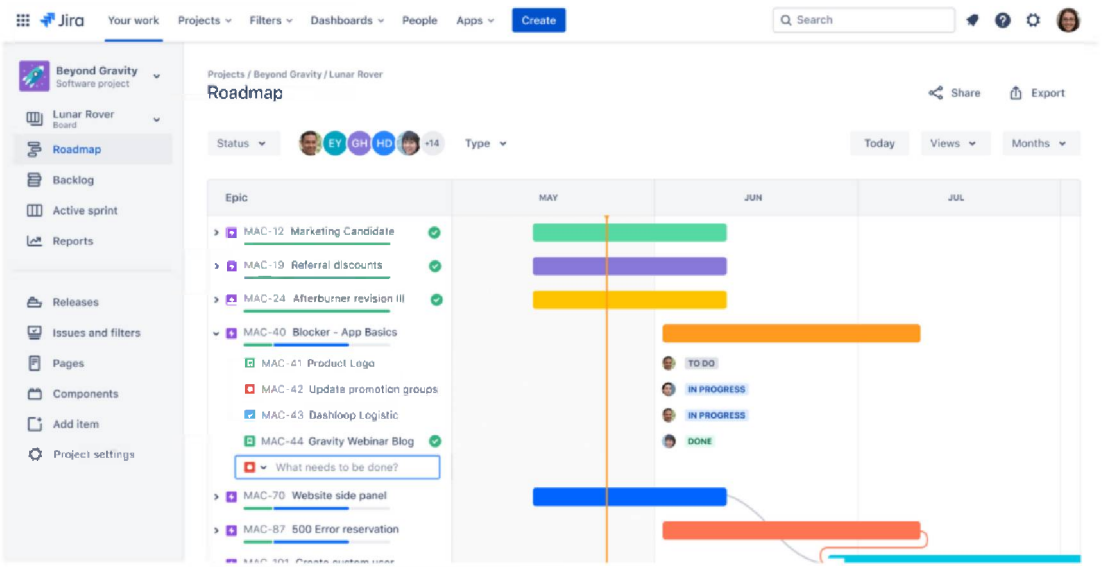

Рисунок 2.4 – Інтерфейс Jira.

Недоліки порівняно з моделлю, що пропонується:

1) складний інтерфейс;
2) призначена для великих компаній
3) є платною.

Переваги:

1) велика кількість додаткового і необхідного функціоналу;
2) дуже зручна для дуже великих проєктів.

Доволі розповсюдженим видом відображення процесу планування є дошки. Формат подання інформації у цьому вигляді є популярним і зручним.

Саме такий підхід реалізовано в застосунку Trello. Ця програма є однією з найпопулярніших зараз систем керування проєктами в режимі онлайн, яка має великий попит серед стартапів та невеликих компаній. Вона дозволяє досить ефективно організовувати роботу базуючись на методології канбан.

Основні якості Trello це гнучкість і універсальність, роблять її комфортною системою організації задач для невеликих команд. При правильній структурі компанії в Trello команда з 50 людей може досить

комфортно працювати, а менеджер – впоратися з контролем усіх ключових завдань і процесів.

Інтерфейс зображений на рисунку 2.5 є зручним та інтуїтивно зрозумілим для користувача, а завдяки досить широкому функціоналу Trello, ця програма на теперішній час є однією з найбільш популярних онлайн-менеджерів завдань.

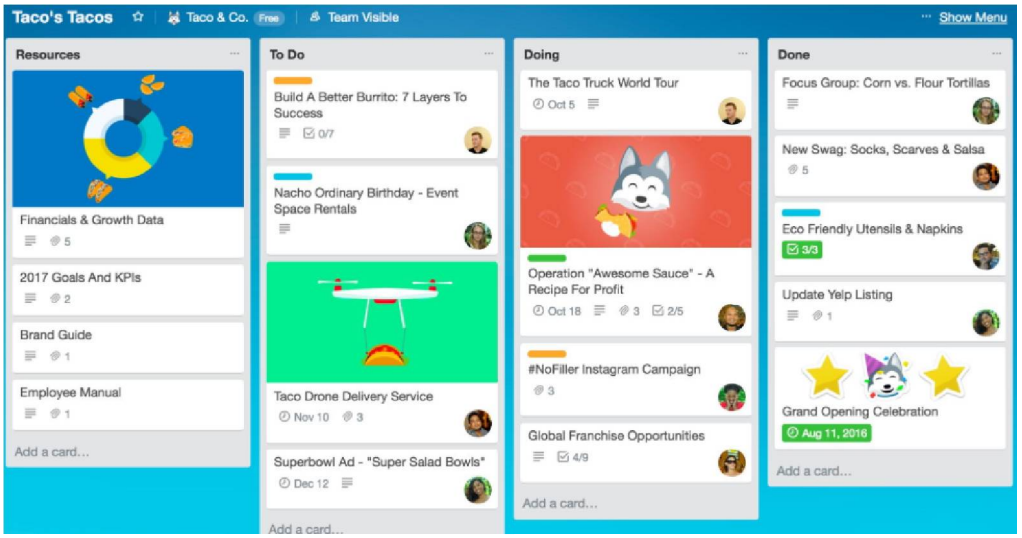

Рисунок 2.5 – Приклад інтерфейсу Trello.

Недоліки порівняно з моделлю, що пропонується:

1) повільно працює з великими проєктами;
2) є платною для команд більш ніж на 10 осіб;
3) не містить у собі усіх необхідних функцій для оцінки ризиків.

Переваги:

1) зрозумілий інтерфейс;
2) проста у експлуатації.

Проаналізувавши існуючі засоби, були зроблені висновки про те, що необхідно проджект менеджерам для поліпшення та полегшення їхньої роботи [20].

2.3 Модель побудови мережевого графу на основі методів мультипаралельного обчислення інформації

Розроблена модель представлена у вигляді нотації IDEF0 на рисунку 2.6, та містить наступні блоки:

1) блок для запису у лист задач;
2) блок для синтезу і трансляції СЧС;
3) блок для вибору методу мультипаралельної обробки інформації;
4) блок для розпаралелювання;
5) блок для оцінки показників ефективності;
6) блок для динамічної зміни ресурсів;
7) блок для верифікації;
8) блок для візуалізації.

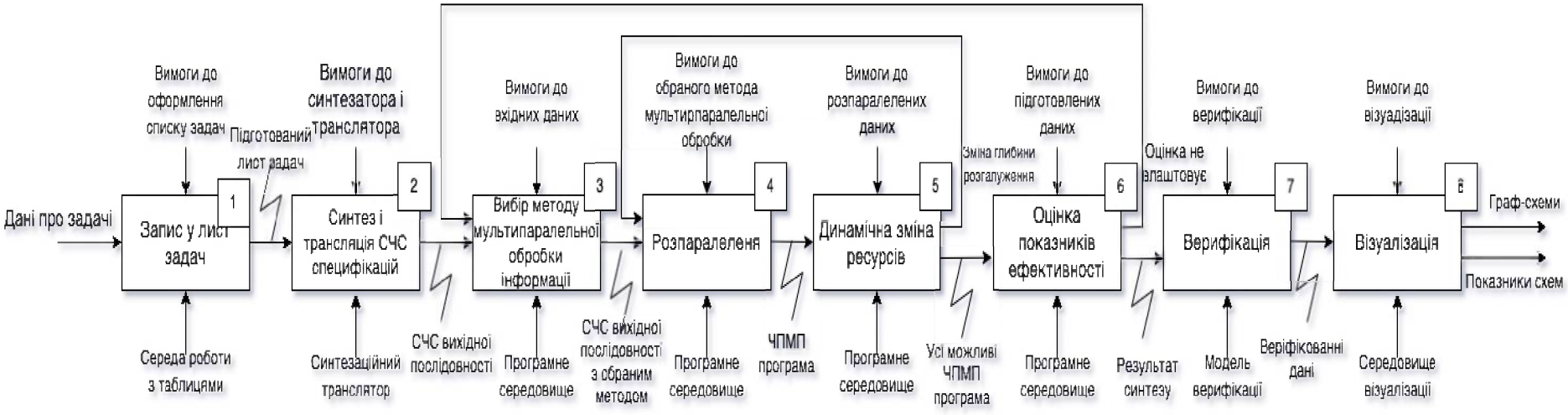

Рисунок 2.6 Модель у нотації IDEF0

Розглянемо опис моделі у нотації IDEF0 детальніше.

Блок 1: запис у лист задач. У цьому блоці відбувається процес обговорення проєкту, розбиття його на задачі, ставляться терміни та кошторис. Йде обговорення та попередньої оцінки задач, їх час виконання, залежності одна від іншої, важливість виконання кожної. Цей процес можна назвати структурним плануванням. Ця робота може бути виконана у будь-якому середовищі, яке працює з таблицями, наприклад Excel. Після оформлення листа задач, ці дані передаються у блок 2 синтезу та трансляції СЧС. Треба зазначити, що лист задач має бути формалізованим згідно з вимогами його оформлення. Тобто, у блоці 1 відбувається формування формалізованого листа задач, згідно з ідеями замовника.

Блок 2: синтез і трансляція СЧС. У цьому блоці відбувається створення програми на мові C++, згідно з формалізованим листом задач. Після цього програма подається на синтезаційному трансляторі у якому відбувається генерація таблиць СЧС, з якими оперують методи мультипаралельної обробки інформації, за для конвертації у послідовного алгоритма у паралельний. Тобто, блок 2 приймає лист задач та перетворює його на СЧС таблиці для того щоб далі могло відбуватися розпаралелювання у синтезаційному трансляторі.

Блок 3: вибір методу мультипаралельної обробки інформації. На цьому етапі обирається той метод який буде застосовуватися для розпаралелювання. У моделі може бути застосован один з ниступних методів мультипаралеьної обробки інформації:

1. метод суміщення незалежних робіт;
2. конвеїрний метод;
3. декомпозиційний метод;
4. метод суміші.

Ці методи розставленні по мірі ускладнення, та наступний метод

обирається, якщо на етапі оцінки обраний метод має незадовільні показники.

Метод суміщення незалежних робіт.

Цей метод полягає у одночасному початку виконання деякої кількості робіт у кожний з дискретних моментів часу, для яких виконуються такі умови:

1) ці роботи не пов'язані послідовними зв'язками;
2) для кожного з таких робіт до моменту часу, що розглядається, попередня робота є завершеною.

Тобто, спеціаліст, працює над задачами у одній гілці від початку до кінця, і на кожну окрему гілку потрібен свій спеціаліст.

Конвеєрний метод

Даний метод полягає у тому, що розглядається не окрема робота, а група (фрагмент), який і виконується спеціалістом. У тому випадку, якщо на проєкті працюють спеціалісти різних рівнів кваліфікації (Junior, Middle, Senior, Architector) одна і та сама робота може бути виконана різними працівниками по різному та за різний проміжок часу. Тому у конвеєрному методі, більш кваліфікований працівник виконує найбільш тривалі фрагменти проєкту, у той час коли найлегший, або найкоротший фрагмент виконує менш кваліфікований спеціаліст.

Декомпозиційний метод

Цей метод полягає є нащадком конвеєрного методу. Він полягає у тому, що одна робота може бути поділена на кілька частин між різними працівниками. Тобто, якщо є складна задача (робота) над якою працює менш кваліфікований співробітник, то йому в поміч кожен день на деякий час допомагає більш кваліфікована людина. Ця допомога може бути як поясненням задачі, постановкою конкретних шагів, у пояснені незрозумілих моментів, або з допомогою реалізації роботи. Таким чином, трохи знижуючи швидкість виконання задачі більш кваліфікованим спеціалістом, значно

прискорюється швидкість роботи менш кваліфікованого спеціаліста. Цей метод показує себе тим краще чим проєкт є більшим.

Метод суміші

Даний метод базується на методі суміщення незалежних робіт. У ньому головною ідеєю є розрахунок максимальної нагрузки на співробітника та комбінованності працівників над задачами. Тобто, простою працівника, коли він чекає поки інший закінчить свою частину не має бути, або цей відрізок часу має бути мінімальним.

Якщо розглядається не сама задача роботи над проєктом, а процес розпаралелювання звичайної програми, то існує також матрично-кодовий метод, який є непридатним до задач планування.

Блок 4: розпаралелювання. На цьому етапі відбувається побудова множини задач, її подальше розбиття на підмножини які формуються на різних часових ярусах, розрахунок моментів початку та закінчення задач, прорахунок можливості розгалуження, запуску різних задач на одному і тому ж самому часовому відрізку. Відбувається попередня оцінка складності та тривалості реалізації сжатої моделі. Формується сжата СЧС модель, розраховується час та складність виконання. Тобто, блок 3 приймаючи СЧС таблиці перетворює їх на часову модель у середовищі програмування, згідно з обраним методом мультипаралельної обробки інформації.

Блок 5: оцінка показників ефективності. У цьому блоці відбувається процес оцінки показників ефективності часопараметризованої мультипаралельної програми (ЧПМП) і її характеристик ефективності згідно з показниками ЧПМП та розрахунок критичного шляху, показників ризиків та тривалості згідно з методологією PERT.

Якщо оцінки ефективності не влаштовують, то відбувається повторне розпаралелення, але користуючись іншим методом мультипаралельної обробки. У випадку коли показники усіх методів не задовольняють вимогам, виводиться найкращий з можливих варіантів методу та розпаралелення.

Блок 6: динамічної зміни ресурсів. Семантико-числовий підхід до синтезу моделей часопараметризованих паралельних процесів забезпечує гарну адаптивність процесів базуючись на врахуванні в динаміці виконання задач зміни ресурсу цифрових систем семантико-числових параметрів.

Блок розраховує можливу глибину розгалуження для кожного проєкту і передає показник розгалуження на блок розпаралелення, який в свою чергу робить розгалуження базуючись на даних. На блок оцінки ефективності подаються усі моделі, від послідовної до максимально розгалуженої. На рисунку 2.7 представлен алгоритм методу синтезу динамічних паралельних процесів зі змінним ресурсом цифрових систем.

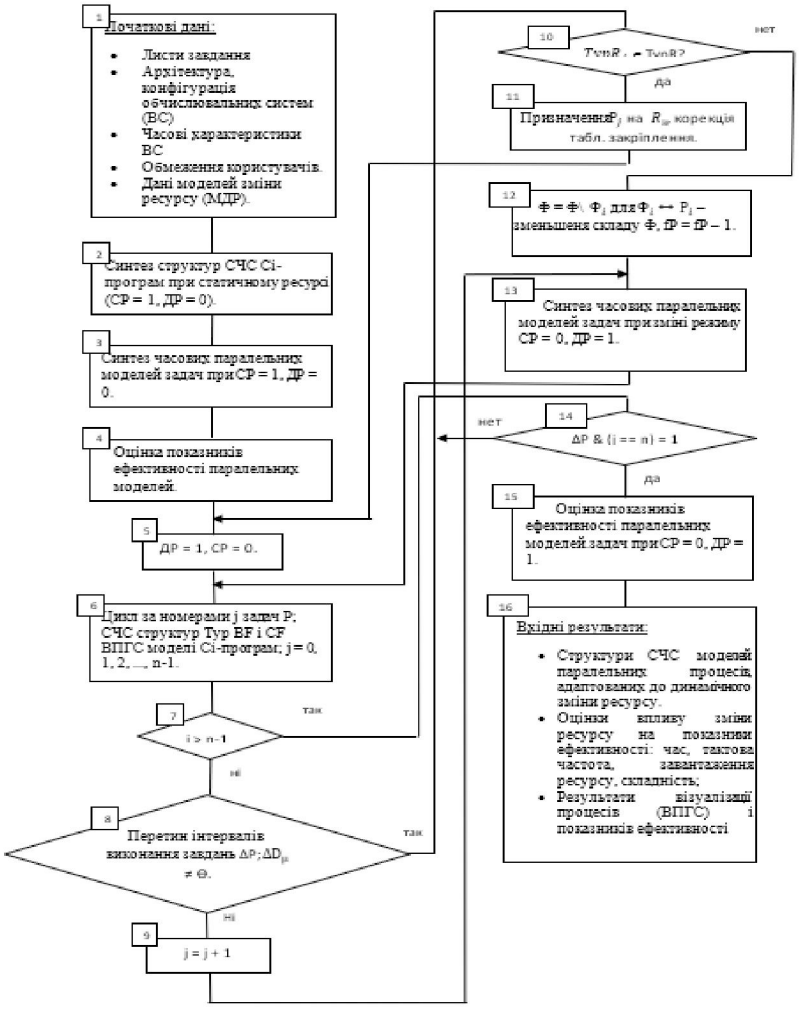

Рисунок 2.7. Узагальнений алгоритм методу синтезу динамічних паралельних процесів зі змінним ресурсом цифрових систем

Блок 7: верифікація. У цьому блоці відбувається перевірка коректності структур семантико-числових специфікацій, статичних та часових результатів формального синтезу часопараметризованих рішень для обраного методу. У якості вхідних даних верифікатора можуть використовуватися:

1. Структур СЧС моделей;
2. Часові структури СЧС;
3. Графи;
4. Графічні специфікації часопараметризованих моделей.

Також відбувається:

- компіляціонна верифікація структур семантико-числової специфікації, Сі-графа та Сі-програми;
- верифікація структур семантико-числовихспецифікацій часопаралельної моделі Сі-програми;
- верифікація статичних структур СЧС паралеьльної обробки
- перифікація часових відношень операторів часо-параментризованної паралельної програми;
- перевірка відповідності показників ефективності заданим вимогам та обмеженням.

Блок 8: візуалізація. На цьому етапі відбувається побудова візуальної складової моделі. Середа візуалізації приймає підготовленні дані та на їхній основі будує часові графи проєкту та відображає дані розрахунків ризиків та ймовірної тривалості проєкту за даних умов.

Висновки до розділу 2

У розділі було проаналізовано та порівняно існуючі методи ймовірнісної оцінки тривалості та вартості проєкту, обрано методологію

PERT, через його точність, та простоту реалізації. Були проаналізовані існуючі технології паралельної розробки та порівняно їх з методами мультипаралельної обробки. Були розглянуті існуючі програмні рішення для мережевого планування та створення графів робот, проаналізовано їх переваги та недоліки згідно з моделлю, що розробляється. Було представлено схему модель програмного рішення та були описані блоки цієї моделі, вимоги до них, та середовища у яких ці блоки повинні бути.

Потрібно провести випробування створеної моделі, показати приклади її роботи та розробити рекомендації щодо застосування.

РОЗДІЛ 3.
ВИПРОБУВАННЯ МОДЕЛІ ТА РОЗРОБКА РЕКОМЕНДАЦІЙ

3.1 Результати роботи

Усе планування проєктів починається з якісного та добре вивіреного листа задач. Якщо це новий проєкт, або субпроєкт великої компанії, завжди треба проаналізувати продукт, що створюється. На сам перед складається MVP проєкт. MVP (мінімально життєздатній продукт) – це рання версія продукту, що вирішує хоча б одну з задачу потенційного клієнта. Розробивши такий план, створюються задачі, які необхідно втілити, щоб проєкт існував і був працездатний. Тому від якості та коректності часової оцінки та важкості задач з цього плану залежить вірогідність закінчення проєкту у задані терміни.

Для моделі, яка розробляється необхідно щоб дані були сформовані згідно з вимогами, в іншому випадку робота по побудові мережевого графа та оцінці ризиків та термінів виконання не відбудеться.

Вимоги до листа наступні:

1) задачі записуються горизонтально одна за одною без розривів.;
2) дозволяється мати у підготовленій таблиці перший рядок – позначення приведених нижче показників;
3) показники задач займають місце один за одним по горизонталі та мають чітку структуру (наведено на рис 3.1);
4) максимальна кількість задач у листі обумовлена максимальної ємністю змінних у мові C++(приблизно 20000 задач).

Індекси	Задачі	Описання	Оптимістичний час	Песимістичний час	Фаза виконання	Оцінка важкості	Зв'язки
1	Розробка дизайну головної		14	25		3	0
2	Створення стилю та розмітки головного сайту		4	10		2	1
3	Додавання функцій на сторінку		7	15		4	2
4	Створення бази знань		3	5		2	0
5	Підключення бази знань до сайту		5	9		3	4
6	Оптимізація та організація бази знань		15	28		5	4
7	Заповнення бази знань даними о користувачах і їх ролях		10	20		4	5
8	Додання авторизації на сайт		8	14		4	7
9	Створення мобільної версії сайту		5	18		3	8,3

Рисунок 3.1 – Лист задач складений згідно вимогам.

Після створення та форматування цього листа, фахівець виділяє необхідні дані, а саме мінімальний та максимальний час та номер задачі. Це мінімально необхідні дані, з якими працює модель.

Наступним кроком є розробка Сі програми на основі зв'язків, завдань у листі задач (рис. 3.2).

```c
#include <stdio.h>
void main(void){
    int task1,task2,task3;
    int task4,task5,task6;
    int task7,task8,task9;
    int begin,end;
    scanf('%d',&begin);
    task1=begin;
    task2=task1;
    task3=task2;
    if(begin==0){
        task5++;
        task7=task5;
        task8=task7;
    }
    else task6++;
    task9=task3+task8;
    printf('%4d\n',task6);
    printf('%4d\n',task9);
}
```

Рисунок 3.2 – код Сі-програми.

Після створення програми відбувається трансляція та синтез СЧС таблиць, які генеруються автоматично. Ці таблиці, базовий файл та файл зв'язків робіт представлено на рисунку 3.3 та рисунку 3.4

N	TYP	NSJ	SJD	NWJ	WJD	RES
0	0	-1	0	0	2	begin
1	10	0	1	2	1	Task1
2	11	1	1	3	1	Task2
3	12	2	1	4	1	Task3
4	13	3	1	5	2	Task4
5	14	4	1	7	1	Task5
6	15	5	1	8	1	Task6
7	16	6	1	9	1	Task7
8	17	7	1	10	1	Task8
9	18	8	2	11	1	Task9
10	1	10	2	-1	0	end

Рисунок 3.3 – Базовий файл графа планування виконання робіт BF.

N	JSD	SPJD	JWD	WPJD
0	-1	0	1	1
1	-1	1	-1	4
2	-1	2	-1	2
3	-1	0	-1	3
4	-1	4	-1	9
5	-1	4	6	5
6	-1	5	-1	6
7	-1	7	-1	7
8	9	3	-1	10
9	-1	8	-1	8
10	11	9	-1	9
11	-1	6	-1	10

Рисунок 3.4 – Файл зв’язків графа планування виконання робіт CF.

Потім відбувається розпаралелення згідно обраного метода мультипаралельної обробки. На першому кроці відбувається розрахунок послідовної моделі (при паралельності равній одиниці), потім блок динамічної зміни ресурсів розраховує можливість поглиблення,збільшення кількості паралельних процесів (збільшення кількості спеціалістів, які працюють одночасно), якщо поглиблення можливе, то передаємо новє значення параметра паралелізації у блок розпаралелення, та робимо новий розрахунок до поки не дійдемо до максимально розгалуженого варіанта, який можливий за приведеними даними.

Далі дані прямують у блок оцінки ефективності, у якому відбувається їхня оцінка та розрахунок ризиків згідно з методологією PERT. Маючи дані про роботи та їхні зваємозв’язки можна розрахувати критичних шлях, середню тривалість робіт, середньоквадратичне відхилення та вірогідності успішного завершення проєкту за певний термін.

Розглянемо більш детальніше цих розрахунків для методу суміщеня незалежних операцій.

Розрахуемо критичний шлях для двох випадків, для максимально паралельної реалізації та для послідовної.

Розглядаючи послідовну реалізацію, варто зазначити, що критичний шлях це весь шлях, тому середнєквадратичне відхилення розраховується за формулою 3.1:

$$\sigma^2_{L\text{кр}} = 4,9 \qquad (3.1)$$

Розрахувавши критичний шлях для паралельної реалізації маємо наступне: критичний шлях складатиме: (0:4)(4:5)(5:7)(7:8)(8:9), а середнєквадратичне відхилення розраховується за формулою 3.2.

$$\sigma^2_{L\text{кр}} = 2,9 \qquad (3.2)$$

Наступним кроком розрахуємо аргумент функції інтегралу вірогідностей, при директивному часі, який задали окремо та який дорівнює $T^{\text{дир}}_{L\text{кр}}$=115, Також необхідно розрахувати середній час критичного шляху.

Для послідовного він складатиме $T^{\text{ср}}_{L\text{кр}}$=114.8.

Для параллельного – $T^{\text{ср}}_{L\text{кр}}$=53.2.

Отримавши ці дані можна обчислити тривалість робіт при вірогідності завершення проєкту у 95% (β=0.95). Така вірогідність є достатньою для багатьох проєктів, але цей показник може бути налаштований для кожного проєкту окрему у разі запиту зі сторони замовника.

Користуючись формулою 2.4, обчислимо час успішного завершення проєкту:

Для послідовного варіанту він складає $T_{L\text{кр}}$=123.

Для параллельного варіанту – $T_{L\text{кр}}$=58.

Як можна побачити, при паралельній реалізації проєкту. ми маємо виграш більш ніж в два рази.

Візуалізатор будує часопаралельні графи. Після побудови отримуємо графі різної мірі розгалуженності.

Після цього приведені часопаралельні графи виконання проєкту при різних умовах оптимістичності:

1) Послідовна мережева модель при оптимістичному часі виконання (Рис 3.5);

2) Паралельна мережева модель при оптимістичному часі виконання (Рис 3.6);

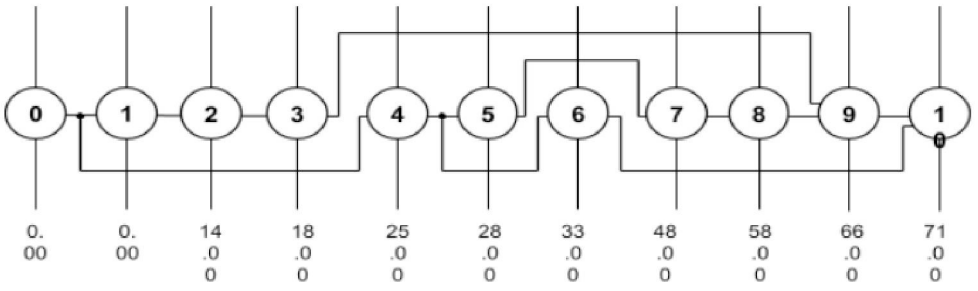

Рисунок 3.5 – Часова послідовна мережева модель планування виконання робіт при мінімальному часі реалізації задач.

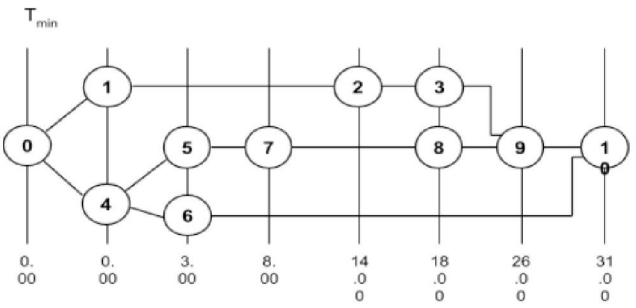

Рисунок 3.6 – Часова паралельна мережева модель планування виконання робіт при мінімальному часі реалізації задач.

3) Послідовна мережева модель при песимістичному часі виконання (Рис 3.7);

4) Паралельна мережева модель при песимістичному часі виконання (Рис 3.8);

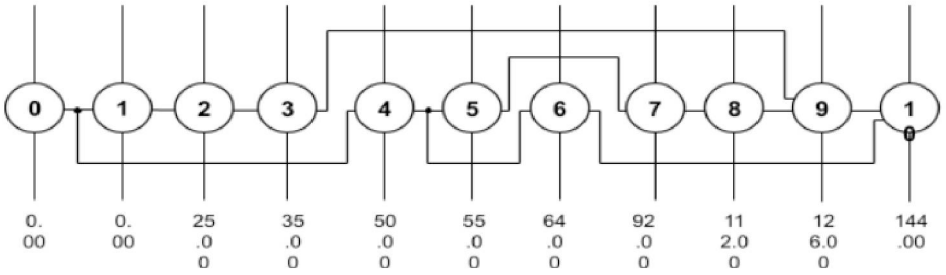

Рисунок 3.7 – Часова послідовна мережева модель планування виконання робіт при максимальному часі реалізації задач

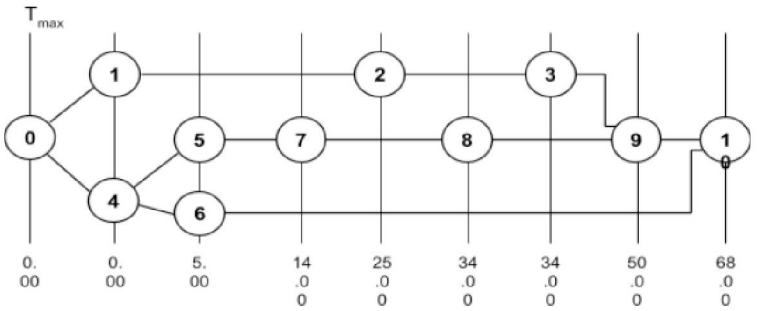

Рисунок 3.8 – Часова мережева паралельна модель планування виконання робіт при максимальному часі реалізації задач

На цих графах можна побачити, що розгалужена реалізація має значний виграш у термінах виконання проєкту, порівнюючи з послідовним.

При песимістичних обставинних терміни проєкту можуть зайняти максимально виділений час на розробку.

Візуалізація показників методології PERT на рис. 3.9.

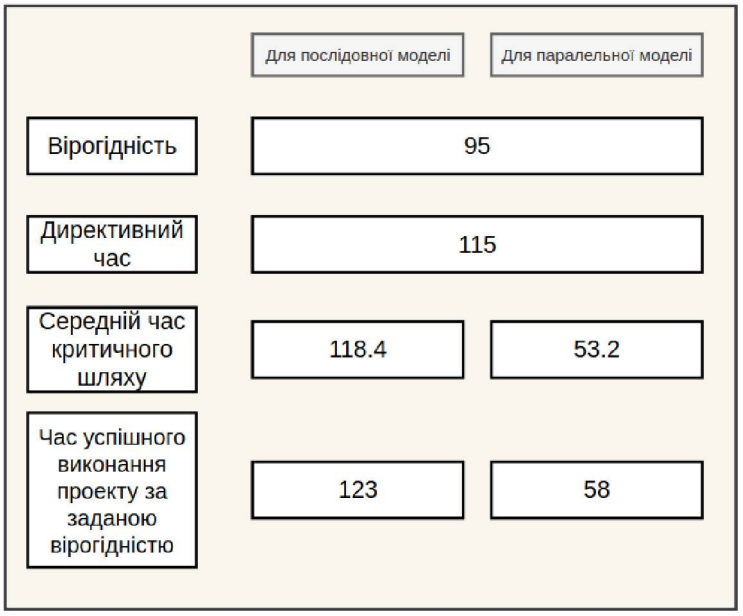

Рисунок 3.9 – Показники проєкту у візуалізаторі.

Послідовні мережеві графи виконання робіт відрізняються лише часом, не змінюючи послідовність виконання робіт, у той час як при паралельному плануванні відрізняються також і місцеположення деяких задач на часопаралельному графі. Це говорить про те, що в процесі роботи в залежності від успіхів в просуванні та розробці проєкту, деякі роботу можуть бути переміщені задля покращення якості продукту, або для пришвидшення його виконання у разі затримок в розробці.

3.2 Рекомендації по застосуванню

На сам перед ця програма буде корисна для менеджерів проєктів та бізнес аналітиків. Вона є інструментом для прискорення їхньої роботи, пропонує найвигідніший варіант серед усіх можливих базуючись на запропонованому листі задач.

Користувачам необхідно зіставити план задач згідно з вимогами та розрахувати значення термінів виконання задач основуючись на знаннях експертів.

Висновки до розділу 3

Були випробувано моделі, які доводять працездатність та ефективність пропонуємого рішення, також були розроблені рекомендації до застосування цієї програми.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи були розглянуті процеси планування, а саме побудови мережевого графа, виконання ІТ-проєкту, базуючись на методах мультипаралельної обробки.

Було проведено оцінку термінів виконання робіт та резервів часу, завдяки чому було зроблено перетворення листа задач у паралельний алгоритм та допомогло знизити терміни виконання проєкту.

Оцінка часових та ймовірнісних показників процесу виконана за допомогою методології PERT, яка базується на даних, необхідних для побудови мережевих графів.

Було проведено аналіз існуючих засобів паралелізації, зроблено висновки та виділені переваги та мінуси порівняно з обраним методом.

Були проаналізовані існуючі програмні рішення для побудови мережевих графів, було проведено порівняння цих рішень з розробленою програмної моделлю.

Було розроблено програмну модель для побудови мережевих графів завдяки методам мультипаралельної обробки, було описано блоки моделі.

Були проведені випробування розробленої програмної моделі які показали, що при застосуванні методів мультипаралельної обробки інформації для побудови мережевих графів і методології PERT для ймовірнісної оцінки тривалості робіт, ефективність обчислювального процесу зростає як і знижується час на виконання робіт.

Майже повністю нівелюється негативний вплив людей на етапі мережевого планування, – особливо, при формуванні мережевого графу, тому, що автоматично побудована схема є більш точною.

Порівнюючи мережеве планування із застосуванням методів мультипаралельної обробки зі звичайним мережевим плануванням можна відмітити візуалізацію чіткої часової граф схеми, на якій відображено у який час буде закінчена та чи інша задача, орієнтуючись на працівників, що працюють паралельно.

Були розроблені рекомендації, щодо застосування розробленої моделі проджект менеджерами та бізнес аналітиками у IT-компаніях, що дозволяє зменшити терміни і витрати на виконання проєкту.

Результати досліджень були апробовані в доповідях на міжнародних конференціях. Доповідь на тему «Розробка та верифікація СЧС-моделі мережевого планування» за авторством Толстолузький, Бердніков, Мороз, Толстолузька, Будько. Доповідь на тему «Розробка моделі мережевого планування із застосуванням методів мультипаралельної обробки інформації» за авторством Толстолузький. Доповідь на тему «Модель мережевого планування із застосуванням методів мультипаралельної обробки інформації» за авторством Толстолузький, Мірошник. Дослідження були створені базуючись на роботі бакалавра по суміжній темі.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Авінаш К. Діксіт, Баррі Дж. Нейлбафф. Мистецтво стратегії. Путівник до успіху в житті та бізнесі від експертів теорії гри [Текст] : Авінаш К. Діксіт, Баррі Дж. Нейлбафф ; переклад з англ. Анастасії Богоніс.– Львів : Видавництво Старого Лева, 2019. – 608 с.

2. Джейсон Фрайд, Девід Хайнемайєр Хенссон. Rework. Ця книжка змінить ваш погляд на бізнес. Клуб Сімейного Дозвілля, 2022. – 176с

3. What is a business analyst? A key role for business-IT efficiency [Електронний ресурс]/ What is a business analyst – 2022 – Режим доступу до ресурсу: https://www.cio.com/article/276798/project-management-what-do-business-analysts-actually-do-for-software-implementation-projects.html (дата звернення – 21.06.2022).

4. Peter Hobbs. Project Management : Dorling Kindersley Limited, Nov 30, 2021 – Business & Economics . – 96 pages.

5. What is Project Management [Електронний ресурс]/ freshwork – 2020 – Режим доступу до ресурсу: https://freshservice.com/what-is-project-management (дата звернення – 21.06.2022).

6. Поляков Г.А., Шматков С.И., Толстолужская Е.Г., Толстолужский Д.А. (2012). Синтез и анализ параллельных процессов в адаптивных времяпараметризованных вычислительных системах. – Х.: ХНУ имени В.Н. Каразина, 2012. – 672с.

7. Хьюз К., Хьюз Т. (2004). Параллельное и распределенное программирование на C++.: Пер. с англ. – М.: Издательский дом «Вильямс», 2004. – 672с.

8. Grama, A., Gupta, A., Kumar V. Introduction to Parallel Computing, Second Edition. – Harlow, England: Addison-Wesley. January 2003. 656p.

9. Методи та засоби оцінки тривалості робіт [Електронний ресурс]/ студопеція – 2018 – Режим доступу до ресурсу: https://studopedia.com.ua/1_418620_delikatnoe-reshenie-delikatnoy-problemi.html (дата звернення – 15.09.2022).

10. Управление проектом. Основы проектного управления : учебник / кол авт.; У66 под ред. проф. М.Л. Разу. – М.КНОРУС, 2006. – 768 с.

11. Project Manager. Project Managemen guide [Електронний ресурс]/ ProjectManager – 2020 – Режим доступу до ресурсу: https://www.projectmanager.com/project-management (дата звернення – 17.09.2022).

12. Интуит. Сетевое планирование [Електронний ресурс]/ Интуит – 2013 – Режим доступу до ресурсу: https://intuit.ru/studies/courses/496/352/lecture/8389 (дата звернення – 18.09.2022).

13. IBM. Planning for network management. [Електронний ресурс] / IBM – 2016 – Режим доступу до ресурсу: https://www.ibm.com/docs/en/power5?topic=communications-planning-network-management (дата звернення – 22.09.2022).

14. LinkedIn. What is PERT and how can we use it? . [Електронний ресурс]/ LinkedIn – 2018 – Режим доступу до ресурсу: https://www.linkedin.com/pulse/what-pert-how-can-we-use-dave-fourie-pmp-prince2-/ (дата звернення – 01.10.2022).

15. Message passing interface (MPI) [Електронний ресурс]/ TechTarget – 2016 – Режим доступу до ресурсу: https://www.techtarget.com/searchenterprisedesktop/definition/message-passing-interface-MPI (дата звернення – 11.10.2022).

16. OpenMP [Електронний ресурс]/ NeSI – 2016 – Режим доступу до ресурсу: https://nesi.github.io/perf-training/python-scatter/openmp (дата звернення – 11.10.2022).

17. M ultiprocessing – Process-based parallelism [Електронний ресурс]/ Python – 2014 – Режим доступу до ресурсу: https://docs.python.org/3/library/multiprocessing.html (дата звернення – 11.10.2022).

18. Microsoft. Excel Help & Training [Електронний ресурс]/ Microsoft – 2020 – Режим доступу до ресурсу: https://support.microsoft.com/ru-ru/excel (дата звернення – 01.11.2022).

19. Atlassian. Jira Software [Електронний ресурс]/ Microsoft – 2016 – Режим доступу до ресурсу: https://www.atlassian.com/software/jira (дата звернення – 01.11.2022).

20. Trello [Електронний ресурс]/ Trello – 2017 – Режим доступу до ресурсу: https://trello.com/en (дата звернення – 01.11.2022).

ДОДАТОК А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) Магістр
Галузь знань: Автоматизація та приладобудування
Спеціальність: 151 – Автоматизація та комп'ютерно-інтегровані технології.

ЗАТВЕРДЖУЮ
Завідувач кафедри теоретичної та прикладної системотехніки
_____ д.т.н., проф. Шматков С. І.
«10» грудня 2021 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Толстолузького Євгена Дмитровича
(прізвище, ім'я, по батькові студента)

1. Тема роботи Модель мережевого планування ІТ-проекту на основі методів мультипаралельної обробки інформації
керівник роботи Мірошник Маріна Анатоліївна д.т.н., професор; професор кафедри ТПС.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___» _____20__ року №_____

2. Строк подання студентом роботи _30 листопада 2022р_____

3. Перелік питань, які потрібно розробити
1. Аналіз методів мультипаралельної обробки;
2. Подання ІТ-проекту у вигляді семантико-числових специфікацій;
3. Розробка моделі ІТ-проекту на базі семантико-числових специфікацій;
4. Випробування розробленної моделі;
5. Розробка рекомендації що до застосування моделі.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1.	Аналіз літератури і патентів	19.10.2021 - 09.01.2022
2.	Дослідження застосування семантико-числових специфікацій для опису ІТ- проекту	10.01.2022 - 01.02.2022
3.	Опис характеристик мережевого графа ІТ- проекту	02.02.2022 - 30.03.2022
4.	Розробка програмної моделі ІТ- проекту	01.04.2022 - 01.05.2022
5.	Тестування та апробація моделі, що розробляється	01.05.2022 - 25.05.2022
6.	Розробка рекомендацій стосовно застосування моделі у навчальному процесі	26.05.2022 - 20.06.2022
7.	Розробка інструкцій для користувача	21.06.2022 - 15.07.2022
8.	Оформлення пояснювальної записки	16.07.2022 - 01.09.2022
9.	Оформлення звіту за результатами науково-дослідницької практики	01.09.2022 - 30.09.2022
10.	Оформлення звіту за результатами переддипломної практики	01.10.2022 - 15.10.2022
11.	Підготовка доповіді на тему дипломної роботи на науково-технічну конференцію	16.10.2022 - 01.11.2022
12.	Підготовка до попереднього захисту роботи	02.11.2022 - 14.11.2022
13.	Надання пояснювальної записки науковому керівнику	15.11.2022 - 23.11.2022
14.	Надання пояснювальної записки науковому рецензенту	24.11.2022 - 30.11.2022

5. Дата видачі завдання 10. 12. 2021

Студент Толстолузький Є. Д. _____
ініціали, прізвище підпис

Керівник роботи Мірошник М.А _____
ініціали, прізвище підпис

ДОДАТОК Б

Технічне завдання
на розробку програмного вироби
«Модель мережевого планування ІТ-проекту на основі методів
мультипаралельної обробки інформації»

Назва розділу	Назва і зміст підрозділу
1. Введення	1.1. Назва програмного вироби – Модель мережевого планування ІТ-проекту на основі методів мультипаралельної обробки інформації. 1.2. Галузь застосування – управління проектами.
2. Підстава для розробки	2.1. Навчальний план за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології. 2.2. Завдання на дипломну роботу магістра(представить як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3. Призначення розробки	3.1. Мета розробки програмного вироби – скорочення часу мережевого планування виконання ІТ-проекту за рахунок використання семантико-числових специфікацій. 3.2. Призначення програмного вироби для автоматизації будування мережевого графа. 3.3. Початкові дані для розробки: лист задач, їх характеристики.
4. Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: 1) представляти з себе програмну реалізацію 2) додати вірогідну оцінку тривалості проекту 3) надавати граф схеми (мережеві графи) 4.2. Вимоги до надійності: Можна побудувати мережевий граф з 200 000 задач 4.3. Вимоги до умов експлуатації офісні приміщення. 4.4. Вимоги до складу і параметрів технічних засобів Персональний комп'ютер у повній комплектації (ноутбук) 4.5. Вимоги до інформаційної та програмної сумісності забезпечити сумісність з усіма обчислювальними засобами. 4.6. Вимоги до маркування та упаковки відсутні. 4.7. Вимоги до транспортування і зберігання відсутні. 4.8. Спеціальні вимоги не пред'являються.
5. Вимоги до програмної документації.	Програмною документацією до виробу «Модель мережевого планування ІТ-проекту на основі методів мультипаралельної обробки інформації» вважати: 1) Справжнє Технічне завдання на розробку програмного вироби (представити у вигляді Додатку Б до пояснювальної записки до дипломної роботи). 2) Програму і методику випробувань розробленого програмного вироби (представити у вигляді Додатку В до пояснювальної записки до дипломної роботи). 3) Опис програмного вироби (представити в розділі 3 пояснювальної записки до дипломної роботи). 4) Текст програми (представити в Додатку Г до пояснювальної записки до дипломної роботи).

<table>
<tr><td>6. Техніко-економічні показники</td><td>В даному розділі можуть бути представлені:

1) порівняльний структурно-функціональний аналіз представити в розділі 1 пояснювальної записки до кваліфікаційної роботи.

2) Вимоги до розрахунку техніко-економічних показників не потрібні.</td></tr>
<tr><td>7. Стадії і етапи розробки</td><td>
<table>
<tr><th>№ з/п</th><th>Назва етапів роботи</th><th>Термін виконання етапів роботи</th></tr>
<tr><td>1.</td><td>Аналіз літератури і патентів</td><td>19.10.2021 - 09.01.2022</td></tr>
<tr><td>2.</td><td>Дослідження застосування семантико-числових специфікацій для опису IT- проекту</td><td>10.01.2022 - 01.02.2022</td></tr>
<tr><td>3.</td><td>Опис характеристик мережевого графа IT- проекту</td><td>02.02.2022 - 30.03.2022</td></tr>
<tr><td>4.</td><td>Розробка програмної моделі IT- проекту</td><td>01.04.2022 - 01.05.2022</td></tr>
<tr><td>5.</td><td>Тестування та апробація моделі, що розробляється</td><td>01.05.2022 - 25.05.2022</td></tr>
<tr><td>6.</td><td>Розробка рекомендацій стосовно застосування моделі у навчальному процесі</td><td>26.05.2022 - 20.06.2022</td></tr>
<tr><td>7.</td><td>Розробка інструкцій для користувача</td><td>21.06.2022 - 15.07.2022</td></tr>
<tr><td>8.</td><td>Оформлення пояснювальної записки</td><td>16.07.2022 - 01.09.2022</td></tr>
<tr><td>9.</td><td>Оформлення звіту за результатами науково-дослідницької практики</td><td>01.09.2022 - 30.09.2022</td></tr>
<tr><td>10.</td><td>Оформлення звіту за результатами переддипломної практики</td><td>01.10.2022 - 15.10.2022</td></tr>
<tr><td>11.</td><td>Підготовка доповіді на тему дипломної роботи на науково-технічну конференцію</td><td>16.10.2022 - 01.11.2022</td></tr>
<tr><td>12.</td><td>Підготовка до попереднього захисту роботи</td><td>02.11.2022 - 14.11.2022</td></tr>
<tr><td>13.</td><td>Надання пояснювальної записки науковому керівнику</td><td>15.11.2022 - 23.11.2022</td></tr>
<tr><td>14.</td><td>Надання пояснювальної записки науковому рецензенту</td><td>24.11.2022 - 30.11.2022</td></tr>
</table>
</td></tr>
<tr><td>8. Порядок контролю і приймання</td><td>1) Перевірку ходу розробки програмного виробу керівнику робіт виконувати раз в 2 тижні.

2) Випробування програмного продукту провести відповідно до програми та методики випробувань на базі комп’ютерного класу.

3) Захист розробленої моделі провести на засіданні Атестаційної комісії.

4) Пояснювальну записку подати на паперових носіях в 1 примірнику і в електронному вигляді в 1 примірнику на CD R компакт-диску.</td></tr>
</table>

Виконавець
студент групи КУ-61

Толстолузький Є.Д.

Замовник
д.т.н., професор; професор кафедри ТПС

Мірошник М.А.

ДОДАТОК В

Програма і методика випробувань
програмного вироби
«Модель мережевого планування ІТ-проекту на основі методів мультипаралельної обробки інформації»

1 Об'єкт випробувань

1.1 Найменування випробуваного програмного вироби «Модель мережевого планування ІТ-проекту на основі методів мультипаралельної обробки інформації».

1.2 Область його застосування управління проектами.

2. Мета випробувань

Скорочення часу мережевого планування виконання ІТ-проекту за рахунок використання семантико-числових специфікацій.

3. Загальні положення

3.1 Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

3.2 Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3.3 Обсяг випробувань

Приймальні випробування програмного вироби проводяться в обсязі відповідному цієї Програми і методики випробувань.

3.4 Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

4.1. Вимоги до функціональних характеристик:

1) представляти з себе програмну реалізацію

2) додати вірогідну оцінку тривалості проекту

3) надавати граф схеми (мережеві графи)

4.2. Вимоги до надійності:

Можна будувати мережевий граф з 200 000 задач

4.3. Вимоги до умов експлуатації офісні приміщення.

4.4. Вимоги до складу і параметрів технічних засобів Персональний комп'ютер у повній комплектації (ноутбук)

4.5. Вимоги до інформаційної та програмної сумісності забезпечити сумісність з усіма обчислювальними засобами.

4.6. Вимоги до маркування та упаковки відсутні.

4.7. Вимоги до транспортування і зберігання відсутні.

4.8. Спеціальні вимоги не пред'являються.

5. Вимоги до програмної документації

Склад програмної документації, що подається на випробування, включає:

1) Технічне завдання на розробку програмного вироби (представлено в Додатку Б до пояснювальної записки до дипломної роботи).

2) Ця Програма і методика випробувань розробленого програмного вироби (представлена в Додатку В до пояснювальної записки до дипломної роботи).

3) Опис програмного вироби (представлено в розділі ___ пояснювальної записки до дипломної роботи).

6. Засоби і порядок випробувань

6.1 Засоби випробувань

Випробування проводяться на технічних засобах, яких персональний комп'ютер, ноутбук.

Випробування проводяться з використанням програмних засобів, яких EXCEL, VS Code, Visualisator.

6.2 Порядок проведення випробувань

6.2.1. Перевірка програмної документації.

Перевірка комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в технічному завданні документації.

6.2.1. Перевірка програмної документації.

Перевірка комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в технічному завд документації.

6.2.2. Перевірка якості програмної документації.

Перевірку здійснювати за критерієм відповідності вимогам єдиної системи програмної документації (ЄСПД).

6.2.3. Перевірка виконання програми.

Тест 1: Перевірка створення листа задач. По виду рисунку робиться висновок про працездатність моделі (рис. В.1).

Індекси	Задачі	Описання	Оптиміст ичний час	Песиміст ичний час	Фаза виконання	Оцінка важкості	Зв'язки
1	Розробка дизайну головної		14	25		3	0
2	Створення спило та розмітки головного сайту		4	10		2	1
3	Додавання функцій на сторінку		7	15		4	2
4	Створення бази знань		3	5		2	0
5	Підключення бази знань до сайту		5	9		3	4
6	Оптимізація та організація бази знань		15	28		5	4
7	Заповнення бази знань даними о користувачах і їх ролях		10	20		4	5
8	Додання авторизації на сайт		8	14		4	7
9	Створення мобільної версії сайту		5	18		3	8,3

Рисунок В.1 – Перевірка створення листу задач.

Висновок: перевірка створення листа задач пройшла успішно.

Тест 2: Перевірка створення Сі-програми з листа задач. По рисунку робиться висновок про працездатність моделі (рис. В.2).

```
#include <stdio.h>
void main(void){
    int task1,task2,task3;
    int task4,task5,task6;
    int task7,task8,task9;
    int begin,end;
    scanf('%d',&begin);
    task1=begin;
    task2=task1;
    task3=task2;
    if(begin==0){
        task5++;
        task7=task5;
        task8=task7;
    }
    else task6++;
    task9=task3+task8;
    printf('%4d\n',task6);
    printf('%4d\n',task9);
}
```

Рисунок В.2 – код Сі-програми.

Висновок: перевірка створення Сі-програми з листа задач пройшла успішно.

Тест 3: Перевірка створення СЧС таблиць з Сі-програми. По виду рисунку робиться висновок про працездатність моделі (рис В.3).

N	TYP	NSJ	SJD	NWJ	WJD	RES
0	0	-1	0	0	2	begin
1	10	0	1	2	1	Task1
2	11	1	1	3	1	Task2
3	12	2	1	4	1	Task3
4	13	3	1	5	2	Task4
5	14	4	1	7	1	Task5
6	15	5	1	8	1	Task6
7	16	6	1	9	1	Task7
8	17	7	1	10	1	Task8
9	18	8	2	11	1	Task9
10	1	10	2	-1	0	end

N	JSD	SPJD	JWD	WPJD
0	-1	0	1	1
1	-1	1	-1	4
2	-1	2	-1	2
3	-1	0	-1	3
4	-1	4	-1	9
5	-1	4	6	5
6	-1	5	-1	6
7	-1	7	-1	7
8	9	3	-1	10
9	-1	8	-1	8
10	11	9	-1	9
11	-1	6	-1	10

Рисунок В.3 – Базовий файл і файл зв'язків графа планування виконання робіт.

Висновок: перевірка створення створення СЧС таблиць з Сі-програми пройшла успішно.

Тест 3: Перевірка побудови мережевого графа базуючтсь на СЧС таблицях. По виду рисунку робиться висновок про працездатність моделі (рис. В.4).

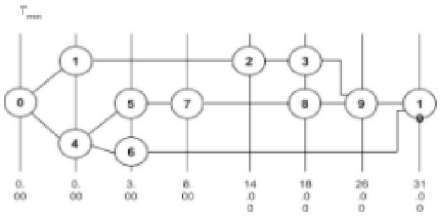

Рисунок В.4 – Часова паралельна модель планування виконання робіт.

Висновок: перевірка побудови мережевого графа базуючтсь на СЧС таблицях пройшла успішно.

Висновки: при вдалому виконанні всіх 4 тестів випробування розробленого додатку вважаються успішними.

Виконавець

студент групи КУ-61

Толстолузький Є.Д.

ДОДАТОК Г

Таблиця нормального розподілу Гаусса

	0,00	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09
0,0	0,5000	0,5040	0,5080	0,5120	0,5160	0,5199	0,5239	0,5279	0,5319	0,5359
0,1	0,5398	0,5438	0,5478	0,5517	0,5557	0,5596	0,5636	0,5675	0,5714	0,5753
0,2	0,5793	0,5832	0,5871	0,5910	0,5948	0,5987	0,6026	0,6064	0,6103	0,6141
0,3	0,6179	0,6217	0,6255	0,6293	0,6331	0,6368	0,6406	0,6443	0,6480	0,6517
0,4	0,6554	0,6591	0,6628	0,6664	0,6700	0,6736	0,6772	0,6808	0,6844	0,6879
0,5	0,6915	0,6950	0,6985	0,7019	0,7054	0,7088	0,7123	0,7157	0,7190	0,7224
0,6	0,7257	0,7291	0,7324	0,7357	0,7389	0,7422	0,7454	0,7486	0,7517	0,7549
0,7	0,7580	0,7611	0,7642	0,7673	0,7704	0,7734	0,7764	0,7794	0,7823	0,7852
0,8	0,7881	0,7910	0,7939	0,7967	0,7995	0,8023	0,8051	0,8078	0,8106	0,8133
0,9	0,8159	0,8186	0,8212	0,8238	0,8264	0,8289	0,8315	0,8340	0,8365	0,8389
1,0	0,8413	0,8438	0,8461	0,8485	0,8508	0,8531	0,8554	0,8577	0,8599	0,8621
1,1	0,8643	0,8665	0,8686	0,8708	0,8729	0,8749	0,8770	0,8790	0,8810	0,8830
1,2	0,8849	0,8869	0,8888	0,8907	0,8925	0,8944	0,8962	0,8980	0,8997	0,9015
1,3	0,9032	0,9049	0,9066	0,9082	0,9099	0,9115	0,9131	0,9147	0,9162	0,9177
1,4	0,9192	0,9207	0,9222	0,9236	0,9251	0,9265	0,9279	0,9292	0,9306	0,9319
1,5	0,9332	0,9345	0,9357	0,9370	0,9382	0,9394	0,9406	0,9418	0,9429	0,9441
1,6	0,9452	0,9463	0,9474	0,9484	0,9495	0,9505	0,9515	0,9525	0,9535	0,9545
1,7	0,9554	0,9564	0,9573	0,9582	0,9591	0,9599	0,9608	0,9616	0,9625	0,9633
1,8	0,9641	0,9649	0,9656	0,9664	0,9671	0,9678	0,9686	0,9693	0,9699	0,9706
1,9	0,9713	0,9719	0,9726	0,9732	0,9738	0,9744	0,9750	0,9756	0,9761	0,9767
2,0	0,9772	0,9778	0,9783	0,9788	0,9793	0,9798	0,9803	0,9808	0,9812	0,9817

ДОДАТОК Д

Лістинг коду побудови графу завдань

```
#include <vcl.h>
#include <stdio.h>
#pragma hdrstop
#include "Unit2.h"
#include "Unit1.h"
#include "Ek.h"
#define Ekran_Height 768   // Висота екрана
#define Ekran_Width 1020   // Ширина екрана
#define Ekran_Delta  200   // зміна висоты екрана
//#define koor_X      200
//#define koor_Y      200
#define delta_XY    100
#define d_kant      10
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm2 *Form2;
extern int mode,object;
extern Graphics::TBitmap* bmp1;  // Область ввода
extern Graphics::TBitmap* bmp2;  // Рабоча область
extern int i;
extern TRect MyRect,MyOther;
FILE *f2;
extern long color_fon;
extern long color_line;
extern long color_kant;
extern long color_oper;
extern long color_line_kant;
extern long color_line_oper;
extern int height;       // Всота
extern int width;        // Ширіна
```

Продовження 1 Лістингу 1

```
extern int delta;        // Зміна высоти
extern int razmer;       // Розмір оператора
extern int kant;         // Признак окантовки
extern int delta_kant;   // Розмер окантовки
extern int speed;        // Швидкість прокрутки
__fastcall TForm2::TForm2(TComponent* Owner)
   : TForm(Owner){
   Form2->Enabled = false;
   Form2->Visible = false;
   f2 = fopen("c:\\my_prog\\graph\\graph\\color.txt","r");
   fscanf(f2,"%ld %ld %ld %ld %ld",
         &color_fon,
         &color_line,
         &color_kant,
         &color_oper,
         &color_line_kant,
         &color_line_oper);
   fclose(f2);
   Form2->Panel2->Color = color_fon;
   Form2->Panel3->Color = color_line;
   Form2->Panel4->Color = color_kant;
   Form2->Panel5->Color = color_oper;
   Form2->Panel6->Color = color_line_kant;
   Form2->Panel7->Color = color_line_oper;
}
void __fastcall TForm2::Panel2DblClick(TObject *Sender){
   if(ColorDialog1->Execute())
      Panel2->Color = ColorDialog1->Color;
}
```

Продовження 2 Лістингу 1

```
void __fastcall TForm2::Button1Click(TObject *Sender){
//   bmp2->Canvas->Brush->Color=Form2->Panel2->Color;
   Form2->Enabled = false;
   Form2->Visible = false;
//   Init();
//   bmp2->Canvas->Brush->Color=Form2->Panel4->Color;
//   DrawOper();
   f2 = fopen("c:\\my_prog\\graph\\graph\\color.txt","w");
   fprintf(f2,"%ld %ld %ld %ld %ld",  //  Color
   Form2->Panel2->Color,Form2->Panel3->Color,
   Form2->Panel4->Color,Form2->Panel5->Color,
   Form2->Panel6->Color,Form2->Panel7->Color);
   fclose(f2);
}
void __fastcall TForm2::Init(){
   bmp2->Canvas->Pen->Width = 1;
   bmp2->Canvas->Pen->Color=RGB(128,128,128);
   bmp2->Canvas->MoveTo(0,0);
   bmp2->Canvas->Brush->Style = bsSolid;
   bmp2->Canvas->Rectangle(0,0,bmp1->Width,bmp1->Height + delta);
}
void __fastcall TForm2::DrawOper(int &i){
   bmp2->Canvas->Brush->Style = bsSolid;
   bmp2->Canvas->Brush->Color=color_kant;
   bmp2->Canvas->Ellipse(nom_x[i],
                  nom_y[i],
                  nom_x[i]+razmer,
                  nom_y[i]+razmer);
   if(kant==1)  {
      bmp2->Canvas->Brush->Color=color_oper;
```

Продовження 3 Лістингу 1

```
      bmp2->Canvas->Pen->Color=color_line_oper;
      bmp2->Canvas->Ellipse(nom_x[i]+delta_kant,
                  nom_y[i]+delta_kant,
                  nom_x[i]+razmer-delta_kant,
                  nom_y[i]+razmer-delta_kant);
   }
   bmp1->Canvas->Draw(0,0,bmp2);
   MyRect = Rect(0,0,width,height);
   MyOther = Rect(0,delta/2+8,width,height+delta/2+8);
   i = delta/2;
   bmp1->Canvas->CopyRect(MyRect,
                bmp2->Canvas,
                MyOther);

   Form1->Image1->Canvas->Draw(0,0,bmp1);
}
void __fastcall TForm2::Panel3DblClick(TObject *Sender){
   if(ColorDialog1->Execute())
     Panel3->Color = ColorDialog1->Color;
}
void __fastcall TForm2::Panel4DblClick(TObject *Sender){
   if(ColorDialog1->Execute())
     Panel4->Color = ColorDialog1->Color;
}
void __fastcall TForm2::Panel5DblClick(TObject *Sender){
   if(ColorDialog1->Execute())
     Panel5->Color = ColorDialog1->Color;
}
void __fastcall TForm2::Panel6DblClick(TObject *Sender){
   if(ColorDialog1->Execute())
```

Продовження 4 Лістингу 1

```
    Panel6->Color = ColorDialog1->Color;
}
void __fastcall TForm2::Panel7DblClick(TObject *Sender){
  if(ColorDialog1->Execute())
    Panel7->Color = ColorDialog1->Color;
}
void __fastcall TForm2::Button2Click(TObject *Sender){
  Form2->Enabled = false;
  Form2->Visible = false;
}
```