**ЗМІСТ**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

1

ІАЛЦ 462637.003 ПЗ

Розроб.

Мазур Р. Ф.

Перевір.

Русанова О.В.

Н. Контр.

Затверд.

Русанова О.В.

Пояснювальна записка

Літ.

Акрушів

2

НТУУ „КПІ” ФІОТ

[1 Розробка програмного продукту 2](#_Toc263729960)

[1.1 Опис інтерфейсу користувача 2](#_Toc263729961)

[1.2 Опис структур даних та алгоритмів 6](#_Toc263729962)

[1.2.1 Перевірка ациклічності введеного графа задачі 6](#_Toc263729963)

[1.2.2 Перевірка зв’язності вершин графа системи 7](#_Toc263729964)

[1.2.3 Алгоритм формування черги 1 7](#_Toc263729965)

[1.2.4 Алгоритм формування черги 2 8](#_Toc263729966)

[1.2.5 Алгоритм формування черги 3 9](#_Toc263729967)

[1.2.6 Алгоритм генерації випадкового графа задачі 10](#_Toc263729968)

[1.2.7 Алгоритм призначення за пріоритетами 11](#_Toc263729969)

[1.2.8 Алгоритм «сусіднього» призначення з урахуванням усіх вузлів 11](#_Toc263729970)

[1.2.9 Алгоритм моделювання 12](#_Toc263729971)

[2 Дослідження роботи алгоритмів планування 14](#_Toc263729972)

[2.1Методика збору та аналізу статистичних даних 14](#_Toc263729973)

[2.2 Дослідження ефективності роботи алгоритмі в меш-топології 15](#_Toc263729974)

[2.3 Вплив параметрів системи на результати планування 24](#_Toc263729975)

[Висновки 26](#_Toc263729976)

# 1 Розробка програмного продукту

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

2

ІАЛЦ 462637.003 ПЗ

## 1.1 Опис інтерфейсу користувача

Головне меню програми містить наступні пункти:

* Файл
  + Новий граф задачі
  + Новий граф системи
  + Зберегти
  + Завантажити
* Моделювання
  + Старт моделювання
  + Вибір черги задач
  + Вибір алгоритму призначення
  + Генерація графу задачі
* Допомога

При роботі з редактором, можна створювати графи задач та систем. Перевірка коректності введення даних про граф виконується динамічно. У графі задач початкові вершини (без вхідних дуг) відображуються синім кольором, кінцеві оранжевим, інші зеленим. Якщо вершина входить до циклу, то вона відображується з червоним тлом (рис. 1.1).

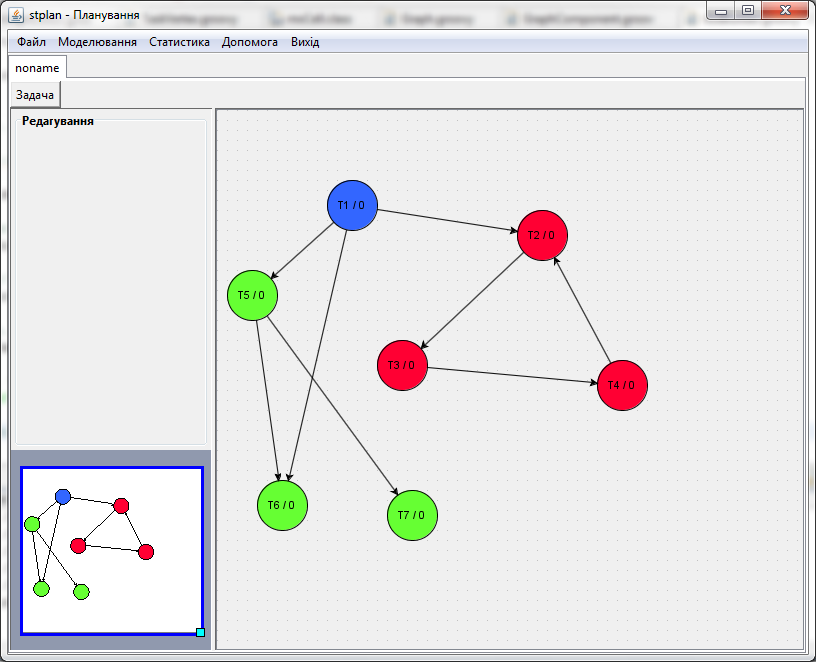
**

Рис 1.1 Редагування графа задачі

У редакторі системи вершини відображуються на червоному фоні, якщо вони не зв’язані з жодною іншою вершиною (при умові, що загальна кількість вершин більша за 2, рис. 1.2).

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

3

ІАЛЦ 462637.003 ПЗ

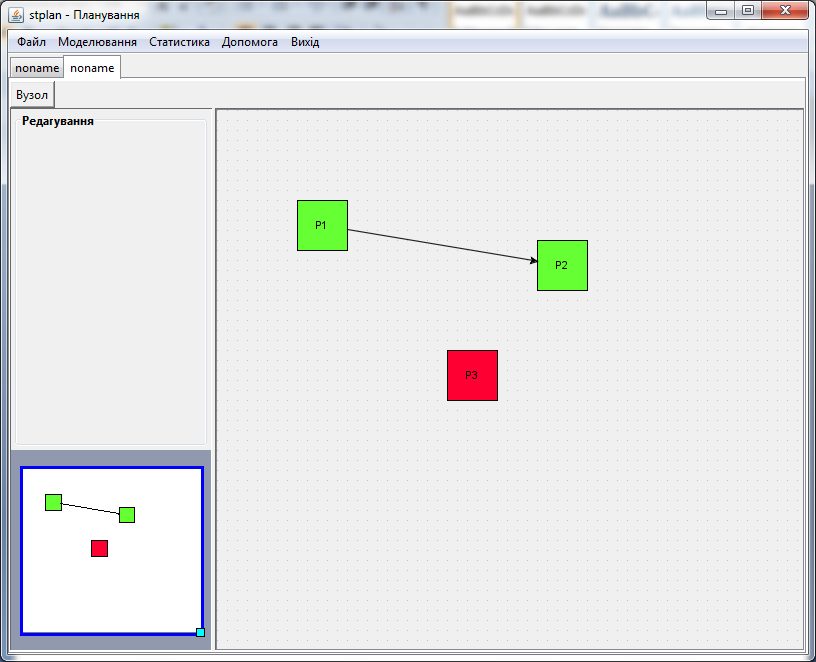
**

Рис 1.2 Редагування графа системи

Даний програмний реалізовує три алгоритми побудови черг:

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

4

ІАЛЦ 462637.003 ПЗ

1. у порядку спадання пронормованої суми критичних за часом та кількості вершин шляхів до кінця графа;
2. за часом готовності;
3. у порядку зростання критичного шляху за кількістю вершин від початку графа, а, у випадку рівних значень, за спаданням часу виконання.

Один із цих методів можна вибрати через відповідний пункт меню, та діалог, що з’являється (рис 1.3).

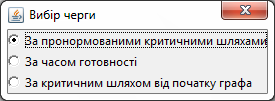


Рис 1.3 Діалог вибору черги задач

Серед алгоритмів призначення для MPP систем розроблений програмний продукт дозволяє вибрати для моделювання один із двох:

1. за пріоритетом процесорів;
2. алгоритм «сусіднього» призначення з урахуванням усіх процесорів.

Вибір алгоритму здійснюється також через відповідний пункт меню та діалог (рис. 1.4).

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

5

ІАЛЦ 462637.003 ПЗ

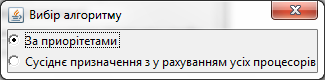


Рис 1.4 Діалог вибору алгоритму призначення

Для моделювання процесу планування необхідно відкрити у редакторах граф системи та граф задачі. Після цього, вибравши відповідний пункт меню, можна побачити вікно з діаграмою Ґанта (рис 1.5).



Рис 1.5 Вікно відображення процесу планування

Для покрокового моделювання можна використовувати кнопку «Наступний крок». Для автоматичного прогону всього процесу використовується кнопка «До кінця».

Праворуч від діаграми в даному вікні відображується упорядкована черга задач та список процесорів, упорядкований за їхнім пріоритетом.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

6

ІАЛЦ 462637.003 ПЗ

При дослідженні алгоритмів планування необхідно виконувати моделювання на великій кількості графів задач з різними параметрами. Для автоматизації даного процесу програмний продукт має можливість генерації випадкового графа за заданими параметрами, яка доступна через відповідний пункт меню. Параметри генерації графа задаються через діалог (рис 1.6).

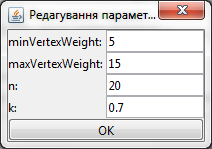


Рис 1.6 Параметри генерації випадкового графа

Статистичні дані, необхідні для дослідження ефективності використання алгоритмів, можуть бути отримані за допомогою скриптів, що працюють з модулями розробленого програмного продукту та виконують задачі автоматичної генерації множини випадкових графів, запуску процесу моделювання для цих графів та збір статистичних даних.

Так, розроблені скрипти, що генерують статистичні дані, представлені в даній роботі, забезпечують вихідні дані у форматі, зручному для подальшої обробки в Excel.

## 1.2 Опис структур даних та алгоритмів

### 1.2.1 Перевірка ациклічності введеного графа задачі

Усі вершини графа зберігаються у списку (A). Підтримуються множина вершин, які були перевірені (R). Вибираємо елемент A, що не належить R. Розглядаємо його, як вершину дерева та здійснюємо обхід у глибину. При обході відвідані вершини додаємо в множину R. Також, при обході формуємо поточний шлях від першої вершини до поточної (P). Якщо при обході зустрічається вершина, що належить R, то виявлено цикл. Вершини, що входять до циклу можна визначити, пройшовши по списку P з його кінця, поки не зустрінемо поточну вершину.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

7

ІАЛЦ 462637.003 ПЗ

### 1.2.2 Перевірка зв’язності вершин графа системи

Усі вершини графа зберігаються у списку (A). Підтримуються множина вершин, які були перевірені (R). Вибираємо перший елемент зі списку A. Розглядаємо його, як вершину дерева та здійснюємо обхід у глибину. При відвідуванні кожної з вершин додаємо її до множини R та вилучаємо з А. При цьому вважаємо, що зустрівши вершину, що належить R, ми досягли листка дерева. Після завершення обходу вершини, що залишилися в А є не коректними, оскільки ніяк не пов’язані з іншими.

### 1.2.3 Алгоритм формування черги 1

Дана черга формується у порядку спадання пронормованої суми критичних за часом та кількості вершин шляхів до кінця графа. Для формування черги за заданим алгоритмом необхідно розрахувати критичні шляхи для кожної з вершин графа. Ці дані можна отримати за допомогою одного повного обходу графу. Вибираючи одну з вершин графа, вважаємо її коренем дерева, здійснюємо його обхід в глибину, при цьому розраховуємо критичні шляхи як з часом так і за кількістю вершин. Після завершення обходу вибираємо наступну вершину, що не була досі відвідана. Повторюємо дану процедуру, враховуючи, що при повторному входженні у вершину, що вже була відвідана, обхід в поточному напрямку можна припинити, оскільки для даної вершини потрібні числові величини уже розраховані.

Таким чином, після обходу графа ми отримаємо для кожної з вершин її критичні шляхи за часом та кількістю вершин (Ткр.і, Lкр.і). Максимальні з цих величин є критичними шляхами графа (Ткр.г, Lкр.г). З отриманих даних розраховуємо коефіцєнт k = Ткр.і/ Ткр.г + Lкр.і/ Lкр.г. У порядку спадання даного коефіцієнта формується черга робіт.

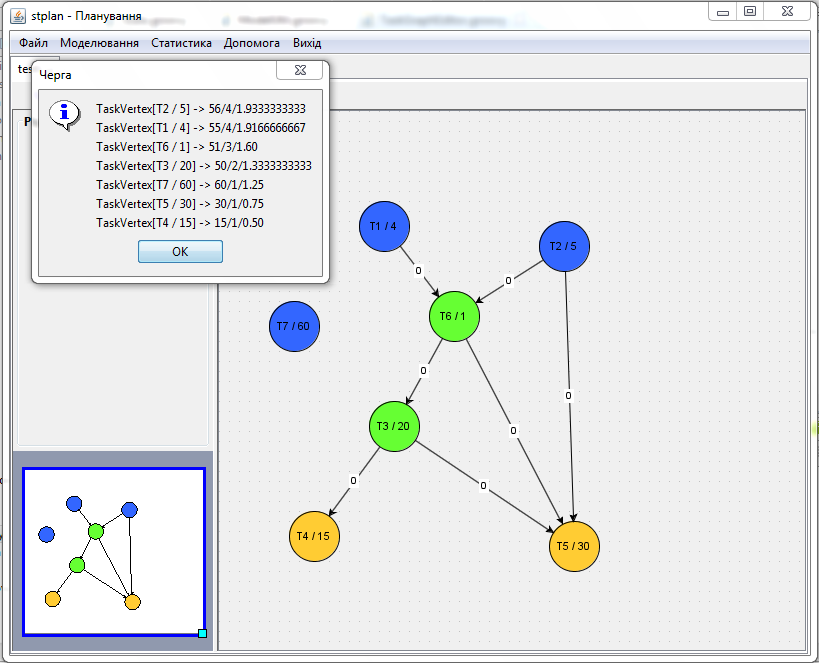
**

Рис 1.7 Приклад сформованої черги

### 1.2.4 Алгоритм формування черги 2

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

8

ІАЛЦ 462637.003 ПЗ

Даний алгоритм часто використовується в динамічному плануванні. Вершини графа задачі потрапляють у чергу за мірою своєї готовності. Таким чином, на початку черга складається з вершин, що не мають вхідних дуг. Далі, під час роботи алгоритму призначення, вершини, що беруться на виконання видаляються із черги, а вершини, які стають готовими до виконання, додаються в неї.

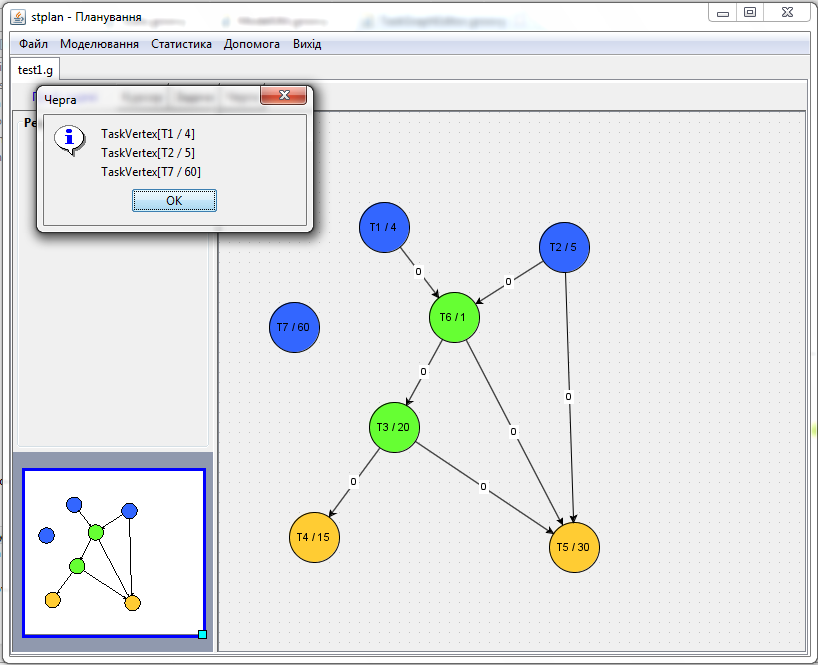


Рис 1.8 Приклад сформованої черги

### 1.2.5 Алгоритм формування черги 3

Елементи цієї черги упорядковуються в порядку зростання критичного шляху за кількістю вершин від початку графа, а, у випадку рівних значень, за спаданням часу виконання. Для формування черги за заданим алгоритмом необхідно розрахувати критичні шляхи для кожної з вершин графа. Ці дані можна отримати за допомогою одного повного обходу графу. Вибираючи одну з вершин графа, вважаємо її коренем дерева, здійснюємо його обхід в глибину у напрямку протилежному напрямку дуг у графі, при цьому розраховуємо критичні шляхи за кількістю вершин. Після завершення обходу вибираємо наступну вершину, що не була досі відвідана. Повторюємо дану процедуру, враховуючи, що при повторному входженні у вершину, що вже була відвідана, обхід в поточному напрямку можна припинити, оскільки для даної вершини потрібні числові величини уже розраховані.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

9

ІАЛЦ 462637.003 ПЗ

Таким чином, після обходу графа ми отримаємо для кожної з вершин її критичний шлях за кількістю вершин. Після цього задача зводиться до сортування вершин у порядку зростання даних критичних шляхів та перевірки їхнього часу виконання для рівних значень.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

10

ІАЛЦ 462637.003 ПЗ

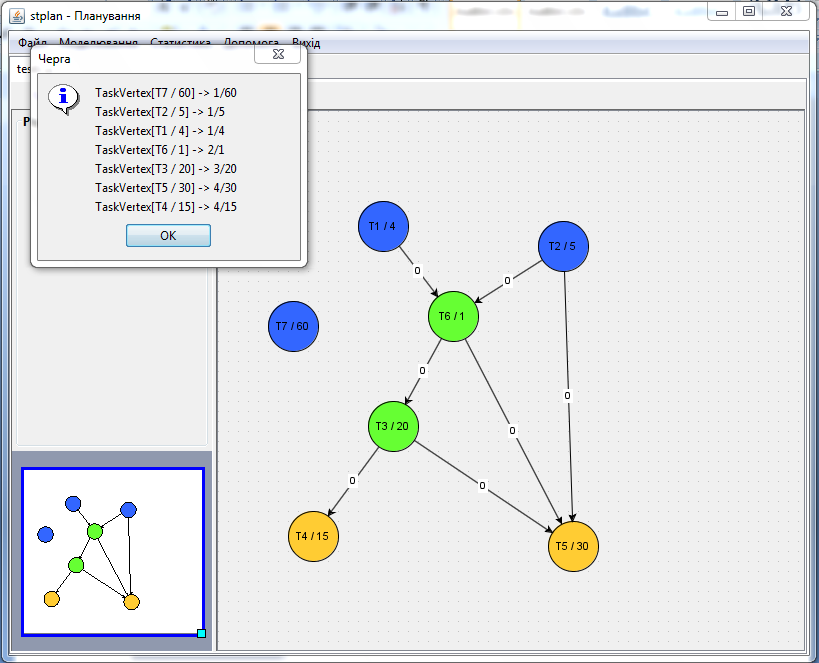


Рис 1.8 Приклад сформованої черги

### 1.2.6 Алгоритм генерації випадкового графа задачі

Першим кроком генерації є створення заданої кількості вершин з вагою, що відповідає заданим параметрам. Далі необхідно сформувати зв’язки між створеними вершинами таким, чином, щоб коефіцієнт

задовольняв задане значення. Для цього випадковим чином вибираються початкова та кінцева вершини для дуги, що генерується. Новий утворений шлях перевіряється на відсутність циклів. Якщо умова задовільна, то для дуги обирається її вага. Вага дуги також обирається випадковим чином, однак в початкових параметрах немає діапазону для генерації її значення. Це діапазон вираховується як чверть залишку від суми ваги вершин, яку залишилося компенсувати для забезпечення заданого коефіцієнта зв’язності. При цьому, регулюючи долю від цієї суми, що використовується для визначення діапазону генерації значення ваги дуги, можна відкоригувати кількість дуг у графі. Чим меншим буде цей діапазон, тим більша кількість дуг буде згенеровано у результуючому графі. Даний процес завершується, коли сума всіх дуг графа стає рівним необхідному значенню, а саме: .

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

11

ІАЛЦ 462637.003 ПЗ

### 1.2.7 Алгоритм призначення за пріоритетами

Визначення вузла, на який призначатиметься виконання задачі, відбувається шляхом вибору із вільних на даний момент вузлів того, який має найбільшу зв’язність.

Якщо два процесора мають однакову зв’язність, то алгоритм має право вибрати будь-який з них для призначення. Таким чином, можливі дві реалізації. Перша – з формуванням статичної черги процесорів за їхніми пріоритетами, яка не змінюється протягом всього часу, та вибором першого вільного з максимальним пріоритетом. Друга – з випадковим вибором серед вільних процесорів з максимальним пріоритетом. У розробленому програмному продукті реалізовано другий варіант.

### 1.2.8 Алгоритм «сусіднього» призначення з урахуванням усіх вузлів

Визначення вузла, на який призначатиметься виконання задачі, відбувається шляхом вибору такого вузла, який, забезпечить мінімальний час витрачений на пересилання даних. При цьому розглядаються не лише вільні вузли, а й зайняті. При оцінці часу старту виконання задачі на певному вузлі використовуються сума часу пересилань, який витрачається для підготовки даних на вибраному вузлі та час звільнення даного вузла від виконання вже призначених на нього задач. Для цього використовується формула

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

12

ІАЛЦ 462637.003 ПЗ



### 1.2.9 Алгоритм моделювання

На кожному кроці моделювання із черги вже запланованих задач вибирається перша, відповідно до часу завершення її виконання встановлюється час системи. Після вибірки завершеної задачі відмічаються нові готові вершини. Даний процес продовжується поки не з’явиться принаймні одна готова вершина. Із черги задач вибирається наступна задача, готова для виконання. Після цього виконуються наступні кроки:

1. Вибирається вузол, на який призначається виконання, алгоритмом, описаним вище.
2. Розв’язується задача організації пересилань даних, необхідних для виконання задачі на визначеному вузлі.
3. Визначається час початку виконання задачі.
4. Визначений вузол відмічається як «зайнятий»

При організації пересилань знаходиться маршрут від вузла-відправника до приймача. При цьому для кожного з вузлів відслідковується стан кожного з його лінків: фіксується час, протягом якого лінк займається та вузол, з яким проходить комунікацію. Час виконання задачі визначається після організації всіх пересилань. Для збереження даних про використання лінків вузла використовується клас NodeState. У нього міститься масив з описом стану кожного з лінків. Кількість елементів даного масиву рівна кількості лінків у вузла. Інформація про стан лінка містить мітки часу, протягом якого лінк є зайнятим та в якому напрямку ведеться обмін даними (це потрібно для підтримки моделювання дуплексності). Використовуючи ці дані, метод класу NodeState resolveStartTime() визначає час, коли пересилання даних будуть здійснені та на визначеному вузлі можна буде стартувати обчислення.

Алгоритм вибору процесора, на який виконується призначення, також повертає затримку, яку необхідно врахувати при плануванні задачі. Так, варіант «сусіднього» призначення, який реалізовано в програмному продукті, повертає, окрім вузла, на який здійснюється призначення, ще й час, через який таке призначення буде можливим. Тоді при плануванні, аналізується час, який займають пересилання, та вимушену затримку, що була визначена алгоритмом, після чого приймається остаточне рішення про час початку обчислення на вузлі.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

13

ІАЛЦ 462637.003 ПЗ

# 2 Дослідження роботи алгоритмів планування

## 2.1Методика збору та аналізу статистичних даних

У дослідженні роботи алгоритмів планування виділено два етапи:

1. порівняння ефективності роботи реалізованих у програмному продукті алгоритмів в системі з меш-топологією;

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

14

ІАЛЦ 462637.003 ПЗ

1. визначення впливу характеристик системи на результат планування.

2-гий етап виконується для алгоритму, що визнається кращим під часу етапу 1. Для першого етапу статистичні дані отримуються за рахунок прогону всіх 6 алгоритмів на випадкових графах із різною кількістю вершин.

Таким чином, для кожної з кількості вершин, що нас цікавлять, для всіх коефіцієнтів зв’язності від 0,1 до 0,9 з кроком 0,1 генерується по 10 графів задач. Кожен із цих графів завантажується в систему з меш-топологією (рис. 2.1). Після завантаження в систему визначаються наступні характеристики:

* кофіцієнт прискорення;
* коефіцієнт ефективноств системи;
* коефіцієнт ефетиіності алгоритму;
* час, витрачений на моделювання.

Для 10 графів з одноковою зв’язністю визначаються середні значення вище перерахованих величин. Ці середні дані наведенні у таблицях в розділі 2.2.

Для визначення кращого алгоритму використовувався наступний підхід. Для кожного з алгоритмів розраховується узагальнюючий критерій, що рівний сумі по кожній із вище зазначених величин середніх значень для всіх значень коефіцієнта зв’язності. За отриманими даними будується графік, на основі якого робиться висновок про доцільність використання алгоритму планування на вибраній топології.

Для кращого алгоритму процедура повторюється лише з тоб зміною, що змінними є параметри системи, а кількість вершин залишається сталою. При цьому кількість вершин була рівною 9N.

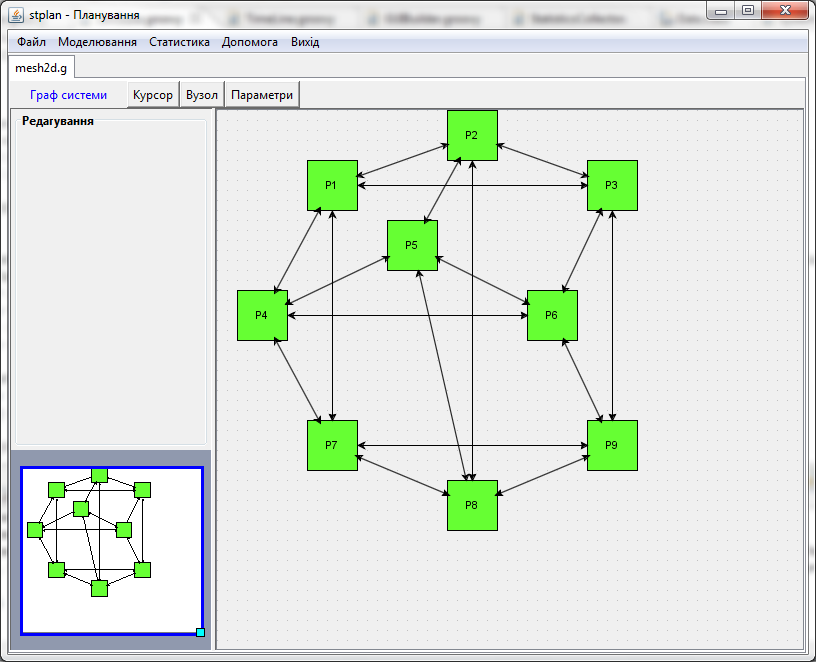


Рис 2.1 Меш-топологія

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

15

ІАЛЦ 462637.003 ПЗ

## 2.2 Дослідження ефективності роботи алгоритмі в меш-топології

Наведемо нижче статистичні дані – результати моделювання роботи алгоритмів планування. Розглядаються алгоритми:

1. Черга 1, призначення за пріоритетами
2. Черга 2, призначення за пріоритетами
3. Черга 3, призначення за пріоритетами
4. Черга 1, «сусіднє призначення»
5. Черга 2, «сусіднє призначення»
6. Черга 3, «сусіднє призначення»

Таблиця 2.1 N=9



Таблиця 2.2 N=18

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

16

ІАЛЦ 462637.003 ПЗ



Таблиця 2.3 N=27

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

17

ІАЛЦ 462637.003 ПЗ



Таблиця 2.4 N=36



Таблиця 2.5 N=45



Таблиця 2.6 N=54



Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

18

ІАЛЦ 462637.003 ПЗ

Таблиця 2.7 N=63



Таблиця 2.8 N=72



Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

19

ІАЛЦ 462637.003 ПЗ

Таблиця 2.9 N=81



Таблиця 2.10 N=90



Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

20

ІАЛЦ 462637.003 ПЗ

Переглянувши дані по високій зв’язності графа (k=0,1), можна побачити, що алгоритми планування в результаті уповільнюють розв’язання задачі, що пояснюється додатковим часом, який закладається на пересилання даних.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

21

ІАЛЦ 462637.003 ПЗ

Можна побачити, що алгоритми, що вибирають цільовий вузол за пріоритетом, дають дуже низькі результати, що пояснюється випадковою вибіркою серед процесорів з однаковими пріоритетами, у результаті чого послідовна задача розкладається на кілька процесорів, що лише уповільнює її розв’язання за рахунок великої кількості пересилань даних.

Алгоритми із сусіднім призначенням працюють значно краще, хоча й вони не завжди оптимально опрацьовують графи з суто послідовною природою.

Цілком зрозумілим є зростання коефіцієнтів прискорення та ефективності при зменшенні зв’язності задачі, що можна просліжкувати на вертикалях в усіх таблицях вище.

Для аналізу ефективності використання цих алгоритмів побудуємо графіки за сумарним критерієм для кожного з параметрів, що визначався (рис. 2.2-2.5).

Рис 2.2 Коефіцієнт прискорення

Ідеальне значення, яке може прийняти сумарний коефіцієнт прискорення рівний 90.

Рис 2.3 Коефіцієнт ефективності системи

Ідеальне значення, яке може прийняти сумарний коефіцієнт ефективності системи рівний 10.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

22

ІАЛЦ 462637.003 ПЗ

Рис 2.4 Коефіцієнт ефективності алгоритму

Ідеальне значення, яке може прийняти сумарний коефіцієнт ефективності алгоритму також рівний 10.

Рис 2.5 Сумарний час, витрачений на планування

У даному випадку кращим показником є мінімальний витрачений час, однак для статичного планування це не є критичним питанням.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

23

ІАЛЦ 462637.003 ПЗ

Графіки показують, що з ростом розмірності задачі усі алгоритми значно краще «показують себе»: ростуть коефіцієнти прискорення та ефективності системи, хоча й падає ефективність алгоритму, оскільки при збільшенні «об’єму» графа задачі все ваще стає наблищитись у його завантаженні в систему до критичного шляху через нестачу обчислювальних ресурсів (тобто зменшується простій вузлів системи, однак у ході роботи уже готові задачі змушені чекати в черзі на звільнення обчислювального ресурсу).

За наведенеми діаграмами досить легко побачити, що кращі результати дає 4 алгоритм (зі складним алгоритмом формування черги та «сусіднім» призначенням), хоча на його роботу йде значно бульше часу за інші алгоритми (рис. 2.5).

Варто також відзначити, що алгоритми, які використовують простіший алгоритм черги 2 (за мірою готовності) дають кращі результати за алгоритми зі більш склдною чергою 3.

## 2.3 Вплив параметрів системи на результати планування

Для алгоритму, що показав кращі результати, проведемо дослідження впливу параметрів системи на результати його роботи. Статистичні дані наведено нижче в таблиці 2.11. Ці дані є середніми значеннями від отриманих від прогону 10 графів задач для кожної із зв’язностей. Розглядалися графи із 45 вершин.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

24

ІАЛЦ 462637.003 ПЗ

Таблиця 2.11 Вплив параметрів системи



Для аналізу отриманих даних побудуємо гістограми для сумарних критеріїв (рис. 2.6-2.7).

Рис 2.6 Коефіцієнт прискорення

Ідеальне значення для Кп становить 90, а для коефіцієнтів ефективності – 10.

Рис 2.7 Кофіцієнти ефективності

У таблиці та діаграмах показано зміну коефіцєнтів прискорення та ефективності залежно від кількості лінків у системі та наявності дуплексного режиму (п/д – півдуплекс, д - дуплекс).

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

25

ІАЛЦ 462637.003 ПЗ

За наведеними даними чітко видно, що збільшення кількості лінків у системі веде до збільшення її продуктивності, однак таке нарощення має певну планку насичення. Таким чином, можна зробити висновок, що для універсальних систем ідеальною була б трасп’ютерна природа з кількістю лінків, що відповідає зв’язності вузла системи. Однак для спеціалізованої системи має сенс оцінити типи задач, які будуть розв’язуватися на ній та визначити оптимальну кількість лінків, яка може бути меншою за ту, що в трасп’ютерах, а це, у свою чергу, приведе до зменшення вартості такої системи.

Введення дуплексного режиму також значно покращує продуктивність системи, однак слід враовувати, що в даному випадку при моделюванні в дуплексному режимі не зменшувалася пропускна здатність каналів, що завжди має місце на практиці.

# Висновки

З ходом виконання завдання на курсовий проект було розроблено програмний продукт, призначений для дослідження алгоритмів планування. Він дозволяє за допомогою графічного редактора задавати вхідні дані у вигляді графів задач та систем, виконує автоматичну верифікацію ввдених даних та надає можливості покрокового та автоматичного моделювання процесу планування. Результати планування відображуються в вигляді діаграм Ґанта.

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

26

ІАЛЦ 462637.003 ПЗ

За допомогою скриптів, що працюють з модулями розробленого програмного продукту, було згенеровано статистичні дані для подальшого дослідження ефективності алгоритмів плануванні, що були задані варіантом.

Після аналізу отриманих статистичних даних можна зробити наступні висновки.

При високому ступені зв’язності графа розв’язання задачі може уповільнитися за рахунок появи паразитних пересилань даних.

Кращі показники в роботі дає алгоритм з використанням «сусіднього» призначення з урахуванням усіх процесорів та чергою, що впорядковує вершини графа задачі у порядку спадання пронормованої суми критичних за часом та кількості вершин шляхів до кінця графа.

При збільшенні об’єму робіт зменшується простій системи, що відображується в рості коефіцєнтів ефективності системи та прискорення. Однак, через нестачу обчислювальних ресурсів, коефіцієнт ефективності алгоритму спадає, оскільки готові до виконання задачі все довже очікуюють в черзі.

При збільшенні кількості лінків у системі (кількості каналів введення-виведення) її ефективність зростає. Однак для певних класів задач таке нарощення може перейти в фазу насичення до моменту досягнення трасп’ютерної природи. Тому, для спеціалізованих систем, має сенс визначити цей момент насичення для зменшення вартості системи.

Застосування дуплексного режиму також дає позитивний результат, що відображується в рості ефективності. Однак потрібно враховувати зменшення пропускної здатності каналів у такому випадку.