## МІНІСТЕРСТВО НАУКИ І ОСВІТИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Курсовий проект

З дисципліни «Системне програмне забезпечення»

з теми: “ Зірка з комутатором ”

Студенту 4-го курсу групи ІО-24

напряму підготовки: 6.050102 Комп’ютерна інженерія.

Тарковський О.С.

Керівник: Сімоненко В.П.

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів:\_\_\_\_\_\_\_\_\_ Оцінка:ECTS\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (вчене звання, науковий ступінь,прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (вчене звання, науковий ступінь,прізвище та ініціали)

Київ – 2015

ЗМІСТ

1. Галузь застосування………………………………………………………4
2. Призначення розробки……………………………………………………4
3. Підстава для розробки……………………………………………………4
4. Вимоги до програмного забезпечення…………………………………..4

Изм.

Ліст

№ докум.

Підпис

Дата

Сторінк

1

###### ІАЛЦ.467449.002.ТЗ

Розробив

Тарковський

Перевірив

Сімоненко

Реценз.

Н. Контр.

Затведжено

**Технічне завдання**

Літ.

Сторінок

2

НТУУ “КПІ” ФІОТ

1. **Галузь застосування**

Изм.

Ліст

№ докум.

Підпис

Дата

Лист

2

ІАЛЦ. 467449.002.ТЗ

Розроблено моделюючий програмний продукт, який може застосовуватися для планування та оптимізації занурення комплексної задачі в розподіленої обчислювальної системі (РВС) для топології типу зірка з комутатором. Даний програмний продукт здійснює базове занурення задачі і дозволяє експериментальним шляхом знайти оптимальний розподіл завдань по процесорах. За допомогою програми, розробленої в даному курсовому проекті можна визначити таке занурення задачі, при якому співвідношення кількість процесорів - час виконання буде оптимальним.

1. **Призначення розробки**

Розроблений програмний продукт призначений для пошуку оптимального занурення комплексної задачі в РВС для топології типу зірка з комутатором, при якому співвідношення кількість процесорів - час виконання буде оптимальним.

1. **Підстава для розробки**

Підставою для розробки служить завдання на курсовий проект, а також закріплення отриманих знань з курсу «Системне програмне забезпечення».

1. **Вимоги до програмного забезпечення**

Провести повне моделювання занурення комплексної задачі в середовищі розробки Java, забезпечити моделювання для різних типів комутаторів, забезпечити засоби для можливості оптимізації базового рішення, забезпечити введення і збереження моделі комплексної задачі у вигляді графа.

Анотація

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

6 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

В курсовій роботі було розглянуто програму моделювання паралельної системи топології “зірка з комутатором ” в якій виконується планування за допомогою алгоритму CPFD (Critical Path Fast Duplication). Також, розглянуто відомі алгоритми розв’язання задачі та надано їх класифікацію.

В програмі було реалізовано, для заданої кількості процесорів, завантаження задачі у вигляді матриці зваженого ациклічного орієнтовного графу.

**ЗМІСТ**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

*7 - 8 -*

*ІАЛЦ.422605.003 ПЗ*

Введення…………………………………………………………………8

1. Постановка задачі……………………………………………………9
   1. Опис завдання…………………………………………………….9
   2. Опис топології……………………………………………………9
2. Розгляд існуючих методів…………………………………………..10
3. Обраний метод розв’язання………………………………………………………….11

3.1) Опис алгоритму CPFD…………………………………………………………………..11

3.2) Алгоритм рішення задачі…………………………………………………………….12

* + 1. Алгоритм визначення ярусу вершин……………………………12
    2. Алгоритм визначення top-level атрибута вершини…………….12
    3. Алгоритм визначення bottom-level атрибута вершини………...12
    4. Алгоритм пошуку критичного шляху…………………………...13
    5. Алгоритм визначення мінімального часу запущеної задачі на заданому процесорі……………………………………………………...13
    6. Алгоритм вирішення задачі………………………………………13

1. Дослідження графіка в середовищі моделювання…………………14
2. Висновок……………………………………………………………...17
3. Список літератури……………………………………………………18
4. Додаток 1. Лістінг……………………………………………………20

Введення

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

8 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

Розподілена обчислювальна система (РВС) являють собою набір обчислювального і мережевого устаткування, об'єднаного між собою певним способом, називаним топологією мережі, а також спеціального програмного забезпечення, що дозволяє користувачеві представляти всю систему як єдину паралельну обчислювальну середу.

Оскільки в РВС загальний час вирішення поставленого завдання можна розділити на час рахунку і на час пересилань даних по мережі, виникає необхідність контролю часу пересилань. За певних умов продуктивність РВС може знизитися за рахунок збільшення загального часу пересилань. На загальний час пересилань крім усього значно впливає топологія РВС.

1. **Постановка задачі**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

9 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

* 1. **Опис завдання**

Завдання представляється у вигляді ациклічного орієнтовного графу з вагами вершин та переходів. Параметрами комп’ютерної системи є кількість процесорів та продуктивність кожного процесора

* 1. **Опис топології**

Зірка – топологія мережі, в якій всі абоненти підключені до явно виділеного центру. Таким центром може бути як окремий процесор (зірка з активним центром), так і комутатор, який використовується для пересилання інформаційних повідомлень з одного процесора на інший. В даній роботі розглянуто топологію “ зірка з комутатором ””. Переваги топології:

1)Вихід з ладу одного процесора не призводить до зупинки всієї системи.

2) Гарні можливості для масштабування.

3) Простота топології дозволяє використовувати більш прості алгоритми планування – адаптовані під конкретну архітектуру алгоритми груп BNP та UNC.

Недоліки:

1) Вихід з ладу центрального комутатора призводить до зупинки функціонування всієї системи.

2)Масштабування ускладнюється великим навантаженням на комутатор.

3) Порівняно із деякими топологіями (кільце, дерево), значно більша довжина ліній зв’язку.



Рис.1 – Топологія “зірка з комутатором” в багатопроцесорній системі

1. **Розгляд існуючих методів**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

10 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

Планування та відображення графів завдань на реальні процесорні архітектури залишаються одними з найбільш критичних задач у теорії паралельних та розподілених обчислень. Зважаючи на NP-повноту задачі, більша частина відомих та широковживаних алгоритмів використовують евристичний підхід для знаходження оптимальних рішень. Головною проблемою в такому випадку залишається непередбачуваність роботи на реальних багатопроцесорних системах, оскільки переважною частиною алгоритми тестуються на задачах невеликої обчислювальної складності та на системах з невеликою кількістю процесорів. Тим не менш, на сьогоднішній день розроблено низку алгоритмів планування, які можна поділити на декілька груп:

1. BNP (Bounded Number of Processors) – група алгоритмів, які напряму призначають задачі обмеженій кількості процесорів. Використання обмежено тільки повнозв’язними системами, стратегії маршрутизації не використовуються.
2. UNC (Unbounded Number of Clusters) – алгоритми, які розплановують ациклічний орієнтовний граф на будь-яку кількість кластерів (техніка отримала назву кластерізація). Суть методу полягає у послідовному об’єднанні двох вузлів графу, якщо це дозволяє зменшити загальний час виконання. Після виконання цієї процедури для всього графу, кожен отриманий кластер призначається окремому процесору. Перевагою подібного методу є можливість залучення великої кількості процесорів для вирішення масштабних задач, недоліком – необхідність додаткового кроку – постпроцесінгу. Також присутнє обмеження на повну зв’язність системи.
3. APN (Arbitrary Processor Network) – алгоритми, що враховують той факт, що процесори з’єднані певною мережевою топологією, тому окрім задачі занурення під завдань на процесори, вирішують завдання створення та планування мережевої маршрутизації комунікаційних повідомлень.
4. TDB (Task-Duplication Based) – алгоритми, які можуть використовуватись також на необмеженій кількості процесорів, але які використовують ідею копіювання певних завдань відразу на декілька процесорів для зменшення в подальшому часу, необхідного для пересилки результатів попереднього кроку обчислень на інший процесор. Особливий приріст продуктивності при використанні подібних алгоритмів спостерігається на задачах, що використовують так звану fork/joіn стратегію.

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

11 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

1. **Обраний метод розв’язання**

Оскільки розглядається система з топологією «зірка з комутатором», в якій в можлива пересилка між будь-якими двома компонентами системи за допомогою центрального комутатора, задача планування дещо спрощується через відсутність необхідності детального планування стратегії маршрутизації, тому за основу можна взяти алгоритм групи TDB. Було обрано алгоритм CPFD (Critical Path Fast Duplication) – алгоритм швидкої копіювання критичного шляху, який потребує певної модифікації для можливості використання в системах з обмеженим числом процесорів та для простої стратегії маршрутизації в топології “зірка”. Алгоритм було обрано через те, що він забезпечує гарний результат розпаралелювання, але більші затрати часу на планування.

* 1. **Опис алгоритму CPFD**

Зазначений алгоритм розбиває всі вершини графу на три групи:

1. Вершини критичного шляху (CPN – Critical Path Nodes). Критичним шляхом називається найдовший за часом виконання шлях від початкової вершини у графі до кінцевої.

2. Вершини IBN (In-branch nodes), вершини, з яких існує шлях до вершин критичного шляху, отже виконання яких є необхідним для виконання найдовшого завдання.

3. Вершини OBN (Out-branch nodes), вершини, які не мають відносяться до перших двох груп.

Найбільший пріоритет надається вершинам першої групи, оскільки саме найдовший шлях багато в чому визначає загальний час виконання програми, і саме час виконання завдань критичного шляху є мінімально можливим часом виконання всієї програми. Далі за пріоритетом йдуть вершини IBN, оскільки їх виконання є необхідним для максимально швидкого початку виконання вершин критичного шляху. Вершини третьої групи є найменш пріоритетними і плануються в останню чергу, оскільки мало впливають на загальний час роботи програми.

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

12 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

* 1. **Алгоритм рішення задачі**
     1. Алгоритм визначення ярусу вершин

Для реалізації алгоритму використовується рекурсивний метод визначення ярусу вершини:

1. Розглядається вершина ni
2. Наявність батьківського рівня (якщо немає ,то рівень=0 )
3. Якщо ярус вершини ni<level+1 ,то ярус вершини ni=level+1
4. Для кожного наступного рівня визначити ярус

Алгоритм визначення ярусів для всіх вершин графу:

1. Встановити ярус всіх вершин рівні нулю
2. Скласти перелік вершин в котрих відсутні батьківські вершини
3. Для кожного елемента визначити ярус
   * 1. Алгоритм визначення top-level атрибута вершини
4. Скласти перелік вершин в порядку збільшення ярусу toplist
5. Для кожної вершини ni в toplist використовувати алгоритм:

Max=0;

if t\_level(nx)+w(nx)+c(nx, ni) > max then

max = t\_level(nx)+w(nx)+c(nx, ni)

t\_level(ni) = max

* + 1. Алгоритм визначення bottom-level атрибута вершини

1. Скласти перелік вершин в порядку зменшення ярусу bottomlist
2. Для кожної вершини ni в toplist використовувати алгоритм:

Max =0

if b\_level(ny)+c(ni, ny) > max then

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

13 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

max = b\_level(ny)+c(ni, ny)

b\_level(ni) = w(ni)+max

* + 1. Алгоритм пошуку критичного шляху

1. Определить максимальное значение cp = top-level + bottom-level
2. Найти вершину ni с минимальным значением яруса, для которой t\_level(ni) + b\_level(ni) = cp
3. Добавить ni в список вершин критического пути
4. while существует такой потомок ny вершины ni, для которого t\_level(ny) + b\_level(ny) = cp do
5. Добавить ny в список вершин критического пути
6. Переназначить вместо вершины ni вершину ny
   * 1. Алгоритм визначення мінімального часу запущеної задачі на заданому процесорі.
7. Пусть ni – задача, для которой нужно определить время запуска на процессоре Px
8. Добавить пересылки e(ny, ni)
9. Определить DAT(ni, Px)
10. Определить ST(ni, Px)
11. Удалить пересылки e(ny, ni)
    * 1. Алгоритм вирішення задачі
12. Скласти перелік з домінуючими вершинами критичного шляху
13. Завантаження на головний процесор
14. Визначити для кожної задачі, чи існує процесор, на котрому час запуску менше за поточний. Якщо існує то перепризначити задачу на цей процесор.
15. **Дослідження графіка в середовищі моделювання**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

14 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

У даному розділі буде виконано занурення довільного графа завдань на систему із комутатором з загальною шиною. Моделювання та оптимізація занурення будуть виконані в розробленому середовищі моделювання з використанням запропонованого алгоритму.

На рис. 2 на графі у вершинах позначені їх номери, в дужках - тривалість завдання. На стрілках графа позначена вартість пересилки даних. 

Рис. 2 - Досліджуваний граф завдань

Результат занурення для 3 процесорів:

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

15 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

Такт: P1 P2 P3 Шина

0: T3 T5 - -

1: T3 T5 - -

2: T3 T5 - -

3: T3 T5 - -

4: T3 T5 - -

5: T3 T5 - -

6: T3 T5 - -

7: T4 T5 - 3 -> 6

8: T4 T5 - 3 -> 6

9: T4 T5 - -

10: T4 T6 - 5 -> 10

11: T4 T6 T10 -

12: T4 T6 T10 -

13: T4 T6 T10 -

14: T4 T6 T10 -

15: T4 T6 T10 -

16: T4 T6 T10 -

17: T4 T6 T10 -

18: T4 T6 T10 -

19: T4 T6 T11 10 -> 2

20: T7 T6 T11 -

21: T7 T2 T11 -

22: T7 T2 T11 -

23: T7 T2 T11 -

24: T7 T2 T11 -

25: T7 T2 T12 -

26: T7 T2 T12 -

27: T7 T2 T12 -

28: T7 - T12 -

29: T1 - T12 7 -> 8

30: T1 T8 T12 -

31: T1 T8 T12 -

32: T1 T8 T12 -

33: T1 T8 T12 -

34: T1 T8 T12 -

35: T1 T8 T12 -

36: T1 T8 T15 -

37: T14 T8 T15 -

38: T14 T8 T15 -

39: T14 T8 T15 -

40: T14 T8 - -

41: T14 T8 - -

42: T14 T9 - -

43: T14 T9 - -

44: T14 T9 - -

45: T14 T9 - -

46: T14 T9 - -

47: T14 - - -

48: T14 - - -

49: T14 - - -

50: T13 - - -

51: T13 - - -

52: T13 - - -

53: T13 - - -

54: T13 - - -

55: T13 - - -

56: T13 - - -

57: T13 - - -

**Висновок**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

16 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

В результаті виконання курсової роботи була написана програма, яка може використовуватися для визначення оптимального занурення реальної комплексної задачі на основі її моделі в розподіленої обчислювальної системі з топологією зірка з комутатором. При цьому за результатами існує можливість скорочення кількості процесорів і загального часу виконання за рахунок зменшення кількості пересилань.

**Список літератури**

*Изм*.

*Ліст*

*№ докум.*

# Підпис

# Дата

*Ліст*

17 *- 8 -*

*ІАЛЦ.422605.003 ПЗ*

1. Э. Таненбаум. Современные операционные системы. 4-е изд. СПб.: Питер, 2015
2. Н. А. Олифер, В. Г. Олифер. Сетевые операционные системы
3. В.Г. Олифер, Н.А. Олифер: «Компьютерные сети. Принципы, технологии протоколы».
4. Е.Н. Баня: «Компьютерные сети»