Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра системного проектування

Розподілені обчислення та інтернет-технології

Лабораторна робота №3

Виконала:

студентка групи ДА-11мп

Молчанова В.С.

Київ – 2021 р.

### **Мій комп’ютер**

Тип комп’ютера: ПК

Процесор: Intel(R) Core(TM) i7-9700K

Материнська плата: Gigabyte Technology H310M S2H 2.0

Чіпсет: Intel H310

Тактова частота: 3.60 GHz

8 фізичних ядер

RAM: 32Gb

Комп’ютер відноситься до класу MIMD за класифікацією Флінна.

### **Кластер КПІ**

Кластер НТУУ «КПІ» налічує 44 вузли з двома чотириядерними процесорами Intel Xeon E5440 (2,83 ГГц та 8 Гб оперативної пам'яті в кожному) та 78 вузлів з двома двоядерними процесорами Intel Xeon 5160 (3,00 ГГц та 4 Гб оперативної пам'яті в кожному), тобто в цілому 624 обчислювальних ядра. При цьому використані мережа обміну даними InfiniBand й операційна система Linux CentOS release 5.2.

Також кластер НТУУ «КПІ» має додаткову система з 128 ядрами, тобто з 16-ма восьмиядерними процесорами Intel Xeon E534 (2,33 ГГц, 8 Гб оперативної пам'яті та диском ємністю 500 Гб у кожному), на якій встановлено операційну систему MS Windows Server 2008 HPC Edition. Таким чином, вперше в Україні користувачам, які звикли до ОС Windows і відповідних прикладних програм, надається можливість застосовувати паралельні обчислення за допомогою програми розпаралелювання MS MPI 2.0.1551.

### Суперкомп’ютер зі списку RANK 500 (варіант 13)

| **Site:** | [**National Super Computer Center in Guangzhou**](https://www.top500.org/site/50365/) |
| --- | --- |
| **Manufacturer:** | **NUDT** |
| **Cores:** | **4,981,760** |
| **Memory:** | **2,277,376 GB** |
| **Processor:** | **Intel Xeon E5-2692v2 12C 2.2GHz** |
| **Interconnect:** | **TH Express-2** |
| **Performance** | |
| **Linpack Performance (Rmax)** | **61,444.5 TFlop/s** |
| **Theoretical Peak (Rpeak)** | **100,679 TFlop/s** |
| **Nmax** | **9,773,000** |
| **Power Consumption** | |
| **Power:** | **18,482.00 kW (Submitted)** |
| **Power Measurement Level:** | **1** |
| **Software** | |
| **Operating System:** | **Kylin Linux** |
| **Compiler:** | **icc** |
| **Math Library:** | **Intel MKL-11.0.0** |
| **MPI:** | **MPICH2 with a customized GLEX channel** |

### Кластер

В якості кластера було вирішено використовувати систему з декількох докер-контейнерів, запущених за допомогою docker-compose. В якості базового образу використовувався образ dispel4py/docker.openmpi, який містить в собі установлений інтерпретатор мови Python та бібліотеку для розподілених обчислень mpi4py . Фінальний Docker-образ:

# syntax=docker/dockerfile:1

FROM dispel4py/docker.openmpi

COPY script.py script.py

COPY collect\_hosts collect\_hosts

COPY run\_mpi run\_mpi

RUN chmod +x run\_mpi

RUN chmod +x collect\_hosts

RUN pip install numpy

Файл docker-compose.yml:

services:

mpi\_head:

image: docker-mpi

ports:

- "60037:22"

links:

- mpi\_node

mpi\_node:

image: docker-mpi

Можна побачити, що при запуску контейнерів буде створено контейнер mpi\_head, до якого можна підключитися з машини-хоста і на якому буде запускатися програма, та кілька контейнерів mpi\_node, на яких будуть проводитися обчислення.

На головному контейнері буде запущено скрипт collect\_hosts, який збирає імена обчислювальних контейнерів в файл:

#!/bin/bash

touch machines

for i in $(eval echo {1..${1:-1}})

do

echo "python-mpi-mpi\_node-$i" >> machines

done

Та скрипт run\_mpi який запускає програму на виконання на зазначених в файлі machines хостах:

#!/bin/bash

./collect\_hosts ${1:-1}

mpiexec -hostfile machines -n ${1:-1} python script.py

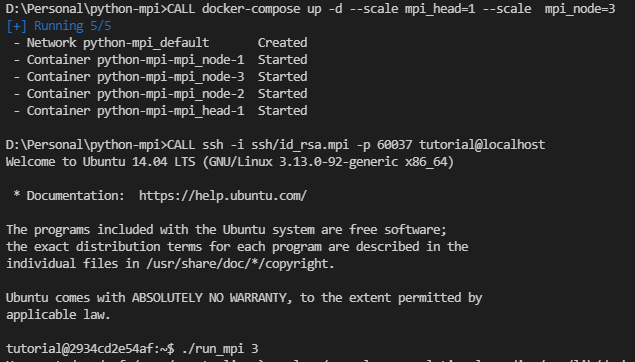
Запускаємо контейнери та підключаємося до головного по ssh за допомогою скрипту (1 головний контейнер та 3 обчислювальних):

CALL docker rmi docker-mpi

CALL docker build --tag docker-mpi .

CALL docker-compose up -d --scale mpi\_head=1 --scale mpi\_node=3

CALL ssh -i ssh/id\_rsa.mpi -p 60037 tutorial@localhost



Запускаємо програму. Текст програми script.py:

from mpi4py import MPI

import numpy as np

comm = MPI.COMM\_WORLD

rank = comm.Get\_rank()

n\_processes = comm.size

if rank == 0:

print("Enter number of intervals: ")

n = int(input())

else:

n = None

n = comm.bcast(n, root=0)

h = 1.0 / n

sum = 0

for i in range(rank+1, n+1, n\_processes):

x = h \* i

sum += (4 / (1 + x\*x))

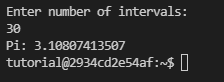
pi = np.array(h \* sum,'d')

result = np.array(0.0,'d')

comm.Reduce(pi, result, op=MPI.SUM, root=0)

if rank == 0:

print("Pi: {0}".format(result))

Результат виконання програми:  


### Контрольні вопроси:

#### 1. Які поняття покладені в основу класифікації Фліна? Назвіть і опишіть класи паралельних комп’ютерних систем по Фліну.

В основу класифікації Флінна вкладені поняття потоку команд та потоку даних. Вона зводиться до 4-х класів:

* **SISD** (single instruction — single data) — одиночний потік команд і даних. Паралелізм відсутній. До цієї категорії відносяться послідовні архітектури, у тому числі й [фон-нойманівского типу](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D1%84%D0%BE%D0%BD_%D0%9D%D0%BE%D0%B9%D0%BC%D0%B0%D0%BD%D0%B0).
* **SIMD** (single instruction — multiple data) — одиночний потік команд і декілька потоків даних. Паралелізм в таких архітектурах полягає в можливості одночасного виконання однієї й тої ж операції над декількома елементами даних. Досягається це централізованою видачею команд декільком обчислювальним пристроям спільним для них пристроєм управління. Найпоширенішими представниками архітектур типу SIMD є так звані векторні ЕОМ, оптимізовані для паралельного виконання однотипних операцій над елементами векторів і матриць. Спеціалізовані векторні процесори іноді вбудовуються в комп'ютери загального призначення. Зокрема, в багатьох сучасних мікропроцесорах вбудовані обмежені можливості векторних обчислень для обробки мультимедіа.
* **MISD** (multiple instruction — single data) — декілька потоків команд, одиночний потік даних. Фахівці дотепер не прийшли до єдиної думки щодо того, які архітектури відносити до даного класу. Деякі включають у нього конвеєрні обчислювачі, в яких цілісна операція розбивається на послідовність простіших етапів з суміщенням різних етапів для різних порцій даних у часі. Таким чином, в певному кожна порція даних від моменту завантаження з пам'яті до запису результату обробки проходить через декілька етапів обробки (стадій конвейера). Подібні принципи покладені в основу систолічних архітектур, які також відносяться до цього класу.
* **MIMD** (multiple instructions — multiple data) — декілька потоків команд і даних. Найпоширеніший на сьогоднішній день клас [паралельних архітектур](https://uk.wikipedia.org/w/index.php?title=%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%96_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B8_%D0%95%D0%9E%D0%9C&action=edit&redlink=1), в яких кожний процесор здатний працювати незалежно від інших над своїм завданням.

#### 2. Що таке багатопроцесорні комп’ютерні системи з пам'яттю? Що викликає некоректність обчислень в комп’ютерних системах з пам'яттю?

Багатопроцесорність ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Multiprocessing) — це використання двох або більше центральних [процесорів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80) (CPU) в одній [комп'ютерній](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) системі. Термін також відноситься до здатності системи підтримувати більше одного процесора і/або здатність [розділити завдання](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BD%D0%B8%D1%82%D0%B5%D0%B2%D1%96%D1%81%D1%82%D1%8C) між ними. Є багато варіацій на цю основну тему, і визначення багатопроцесорності може змінюватися в залежності від контексту, в основному від того, як визначені процесори (кілька ядер на одному кристалі, множина чипів в одному корпусі, множина корпусів в одному системному модулі тощо).

Некоректність обчислень в багатопроцесорних системах з пам’яттю найчастіше відбувається через спроби одночасного доступу декількох процесів до одного участку пам’яті або некоректно побудований алгоритм, який не враховує особливості багатопроцесорних систем і через це, наприклад, команди виконуються в неправильному порядку.

#### 3. Які переваги і недоліки комп’ютерних систем з передачею повідомлень?

#### 4. Визначте три класи технічної реалізації багатопроцесорних комп’ютерних систем.

**Матричні ПКС** - це КС, що складаються з великої кількості універсальних процесорів, призначених для повнорозрядної або для однорозрядної обробки інформації. Працюють усі процесори під управлінням єдиного пристрою управління, тобто в один і той же момент часу усі процесори виконують одну й ту ж саму операцію над різними даними. Ці системи орієнтовані на реалізацію природного паралелізму. Вони відносяться до систем з розділеною (локальною) пам'яттю, а значить процесори можуть бути об'єднані у різні топології з необмеженою масштабованістю. Найбільш поширеними топологиями для матричних систем є кільце, решітка і тор. Сьогодні такі архітектури використовуються для графічних процесорів та прискорювачів нейромереж.

**Асоціативні ПКС** відрізняються від матричних тим, що в якості оперативної пам'яті використовується асоціативна пам'ять.

**Векторні системи** будуються на основі одного або множини векторних процесорів. У свою чергу векторний процесор включає набір одно функціональних конвеєрів, які використовують загальну, або розподілену пам'ять. Такий тип оперативної пам'яті впливає на обмеженість масштабованості подібних процесорів. Якщо векторна система складається із множини векторних процесорів із роздільною пам'яттю, то масштабованість таких систем необмежена. Векторні системи орієнтовані на реалізацію як природного паралелізму, так і паралелізму суміжних операцій, а у разі багатопроцесорних варіантів векторних систем, що складаються з безлічі векторних процесорів, то і на реалізацію середньо та крупнозернистого паралелізму.

#### 5. Що таке симетричні мультипроцесори (SMP)?

Симетричне мультипроцесування (англ. Symmetric Multiprocessing, або SMP) — це архітектура багатопроцесорних комп'ютерів, в якій два або більше однакових процесорів підключаються до загальної пам'яті. Більшість багатопроцесорних систем сьогодні використовують архітектуру SMP.

SMP-системи дозволяють будь-якому процесору працювати над будь-яким завданням незалежно від того, де в пам'яті зберігаються дані для цього завдання; за належної підтримки операційною системою, SMP-системи можуть легко переміщувати завдання між процесорами ефективно розподіляючи навантаження. З іншого боку, пам'ять набагато повільніша процесорів, які до неї звертаються, тож навіть однопроцесорним машинам доводиться витрачати значний час на отримання даних з пам'яті. У системах SMP ситуація ще більш ускладнюється, бо лише один процесор може звертатися до пам'яті в певний момент часу.

#### 6. Які особливості систем з масовим паралелізмом (MPP)?

Масово-паралельна архітектура (Massively Parallel Processor, MPP), також масивно-паралельна архітектура, масивно-паралельний процесор – архітектура паралельної ЕОМ з розподіленими блоками обчислень, зокрема розподіленою пам'яттю, тобто з наявною в кожного з процесорів власної пам'яті. Тому ці архітектури також називають архітектурами з розподіленою пам'яттю (хоча, якщо бути точнішим, клас архітектур з розподіленою пам'яттю є підкласом масово-паралельних архітектур).

#### 7. Дайте визначення обчислювального кластера.

Обчислювальний кластер — це масив серверів, об'єднаних деякою комунікаційною мережею. Кожний обчислювальний вузол має свою оперативну пам'ять і працює під керуванням своєї операційної системи.

#### 8. Опишіть види кластерів, їх особливості, дайте приклади кластерів.

**Кластери високої доступності**

Позначаються абревіатурою HA (англ. High Availability — висока доступність). Створюються для забезпечення високої доступності сервісу, що надається кластером. Надмірна кількість вузлів, що входять в кластер, гарантує надання сервісу у разі відмови одного або декількох серверів. Типове число вузлів — два, це мінімальна кількість, що приводить до підвищення доступності. Створено безліч програмних рішень для побудови такого роду кластерів.

Відмовостійкі кластери та системи взагалі будуються по трьом основним принципам:

* З холодним резервом або активний / пасивний. Активний вузол виконує запити, а пасивний чекає його відмови і включається в роботу, коли така відбудеться. Приклад — резервні мережеві з'єднання, зокрема, алгоритм зв'язуючого дерева. Наприклад зв'язку DRBD та HeartBeat.
* З гарячим резервом або активний / активний. Всі вузли виконують запити, в разі відмови одного навантаження перерозподіляється між рештою. Тобто кластер розподілення навантаження з підтримкою перерозподілу запитів при відмові. Прикладами є практично всі кластерні технології, наприклад, Microsoft Cluster Server. OpenSource проект OpenMosix.
* З модульною надмірністю. Застосовується тільки у випадку, коли простій системи абсолютно неприпустимий. Всі вузли одночасно виконують один і той же запит (або частини його, але так, що результат досяжний і при відмові будь-якого вузла), з результатів береться будь-який. Необхідно гарантувати, що результати різних вузлів завжди будуть однакові (або відмінності гарантовано не вплинуть на подальшу роботу). Приклади — RAID та Triple modular redundancy.

**Кластери розподілу навантаження**

Принцип їх дії будується на розподілі запитів через один або кілька вхідних вузлів, які перенаправляють їх на обробку в інші, обчислювальні вузли. Початкова мета такого кластера — продуктивність, однак, у них часто використовуються також і методи, що підвищують надійність. Подібні конструкції називаються серверними фермами. Програмне забезпечення (ПЗ) може бути як комерційним (OpenVMS, MOSIX, Platform LSF HPC, Solaris Cluster Моава Cluster Suite, Maui кластера Scheduler), так і безкоштовним (OpenMosix Sun Grid Engine, Linux Virtual Server).

**Обчислювальні кластери**

Кластери використовуються в обчислювальних цілях, зокрема в наукових дослідженнях. Для обчислювальних кластерів вагомими показниками є висока продуктивність процесора в операціях над числами з рухомою комою (flops) і низька латентність об'єднувальної мережі, і менш вагомими — швидкість операцій введення-виведення, яка більшою мірою важлива для баз даних та web-сервісів. Обчислювальні кластери дозволяють зменшити час розрахунків, порівняно з одиночним комп'ютером, розбиваючи завдання на паралельно виконувані гілки, які обмінюються даними через мережу. Одна з типових конфігурацій — набір комп'ютерів, зібраних із загальнодоступних компонентів, з встановленою на них операційною системою Linux, і пов'язаних мережею Ethernet, Myrinet, InfiniBand або іншими відносно недорогими мережами. Таку систему прийнято називати кластером Beowulf.

Окремо виділяють високопродуктивні кластери (позначаються англ. абревіатурою HPC Cluster — High-performance computing cluster). Список найпотужніших високопродуктивних комп'ютерів (також може позначатися англ. абревіатурою HPC) можна знайти в світовому рейтингу ТОП500.

**Системи розподілених обчислень (grid)**

Такі системи не прийнято вважати кластерами, але їх принципи в значній мірі схожі з кластерною технологією. Їх також називають grid-системами. Головна відмінність — низька доступність кожного вузла, тобто неможливість гарантувати його роботу в заданий момент часу (вузли підключаються і відключаються в процесі роботи), тому завдання повинне бути розбите на ряд незалежних один від одного процесів. Така система, на відміну від кластерів, не схожа на єдиний комп'ютер, а служить спрощеним засобом розподілу обчислень. Нестабільність конфігурації, в такому випадку, компенсується великим числом вузлів.

**Кластер серверів, організованих програмно**

Кластер серверів (в інформаційних технологіях) — група серверів, об'єднаних логічно, здатних обробляти ідентичні запит и і використовуються як єдиний ресурс. Найчастіше сервери групуються за допомогою локальної мережі. Група серверів володіє більшою надійністю і більшою продуктивністю, ніж один сервер. Об'єднання серверів в один ресурс відбувається на рівні програмних протоколів.

На відміну від апаратного кластера комп'ютерів, кластери організовувані програмно, вимагають:

* Наявності спеціального програмного модуля (Cluster Manager), основною функцією якого є підтримка взаємодії між усіма серверами — членами кластеру:
  + Синхронізації даних між усіма серверами — членами кластеру;
  + Розподіл навантаження (клієнтських запитів) між серверами — членами кластеру;
* Від уміння клієнтського програмного забезпечення розпізнавати сервер, що являє собою кластер серверів, і відповідним чином обробляти команди від Cluster Manager;
  + Якщо клієнтська програма не вміє розпізнавати кластер, вона буде працювати тільки з тим сервером, до якого звернулася спочатку, а при спробі Cluster Manager перерозподілити запит на інші сервери, клієнтська програма може взагалі позбутися доступу до цього сервера (результат залежить від конкретної реалізації кластера)

#### 9. Що таке комунікаційна мережа, які її основні параметри? Наведіть приклади комунікаційних мереж.