Análisis de rendimiento de un cluster heterogéneo formado por placas Raspberry Pi

Performance analysis of a heterogeneous cluster consisting of Raspberry Pi boards

Trabajo de Fin de Grado
Grado en Ingeniería de Computadores

Facultad de Informática
Universidad Complutense de Madrid
2019/2020
Francisco Javier Abarca Calderín

Dirigido por:
Alberto Núñez Covarrubias
Pablo Cerro Cañizares

Agradecimientos

A mi familia por su confianza.

A mi pareja por su ayuda y su apoyo incondicional.

A mis amigos por aguantarme en los momentos de más presión.

A Alberto y a Pablo por su paciencia y su buen hacer, ayudándome y aconsejándome en cada momento.

Muchas gracias a todos.

Resumen

En el año 1965 Gordon E. Moore formuló la conocida ley de Moore, la cual expone que cada dos años - aproximadamente - se duplicaría el número de transistores por unidad de superficie de un circuito integrado, presagiando, de esta manera, la evolución que llevaría la tecnología desde entonces. Esta predicción, se ha estado cumpliendo prácticamente hasta nuestros días, donde parece que se ha llegado a un punto de saturación en el que no se puede avanzar tan rápido como se ha estado haciendo anteriormente. Para conseguir mejores rendimientos se está optando por la paralelización de procesos, lo cual, siempre ha estado presente en los *clusters* o superordenadores utilizados por centros de investigación, empresas y universidades.

En 2011, con la finalidad de estimular el aprendizaje de informática en las escuelas, surge Raspberry Pi, una placa computadora de bajo coste y de tamaño reducido. A partir de entonces ha surgido una comunidad muy activa involucrada en un amplio rango de proyectos. Entre otras posibilidades, estas placas pueden ejecutar Linux, permiten su interconexión para paralelización de procesos, permitiendo así, el despliegue de un *cluster* personal con un presupuesto limitado y que ocupe muy poco espacio, algo impensable hace unos años.

Con el avance de la tecnología, es probable que al extender un *cluster* ya existente, se puedan añadir placas más potentes a las actuales. De esta forma, conseguiremos aumentar significativamente el rendimiento del *cluster*. En este proyecto se realiza un estudio de rendimiento de un *cluster* de bajo presupuesto formado por placas Raspberry Pi 3 y Raspberry Pi 4. Este *cluster* está formado por las dos últimas versiones de la placa, las cuales, tienen diferente potencia. De esta forma, es posible realizar experimentos con diferentes configuraciones del *cluster*, realizando ejecuciones sobre placas iguales o mezclando distintos tipos de placas. El objetivo consiste en comparar distintas configuraciones que utilicen un único tipo de placa (homogéneas) con uno que utilice distintos tipos de placa (heterogéneo), analizando las diferencias de rendimiento de distintos experimentos y comparando estos rendimientos con el de un ordenador personal. Para la elaboración de este análisis, se ejecutarán varias aplicaciones, entre ellas, prácticas de la asignatura PSD y benchmarks conocidos y adoptados por la comunidad científica.

Palabras clave: *cluster*, MPI, Raspberry Pi, pruebas, rendimiento y distribuido

Abstract

In 1965 Gordon E. Moore formulated Moore's famous law, which states that - approximately - every two years the number of transistors per unit area of an integrated circuit would double, thus foreshadowing the evolution that the technology would bring since then. This prediction has been practically fulfilled until our days, where it seems that a saturation point has been reached and, therefore, it is not possible to advance as fast as it has been done before. In order to achieve better performance, the parallelization of processes is being chosen, which has always been present in the clusters or supercomputers used by research centers, companies and universities.

In 2011, with the aim of stimulating computer learning in schools, Raspberry Pi, a low-cost and small-sized computer board, was created. Since then, a very active community has emerged involving a wide range of projects. Among other possibilities, these boards can run Linux, allowing their interconnection for parallelization of processes, thus allowing the deployment of a personal cluster with a limited budget occupying very little space, something unthinkable a few years ago.

With the advance of technology, it is likely the cluster might be extended by adding more powerful boards. In this way, we will be able to significantly increase the performance of the cluster. In this project, a performance study is being carried out on a low-budget cluster made up of Raspberry Pi 3 and Raspberry Pi 4 boards. This cluster consists of the two latest versions of the board, which have different power levels. Thus, it is possible to perform experiments with different configurations of the cluster, performing executions on the same boards or mixing different types of boards. The objective is to compare different configurations that use a single type of board (homogeneous) with one that uses different types of board (heterogeneous), analyzing the differences in performance of different experiments and comparing these performances with a personal computer. For the elaboration of this analysis, several applications are executed, among them, practices of the subject PSD and benchmarks known and adopted by the scientific community.

Keywords: *cluster*, MPI, Raspberry Pi, tests, performance y distributed computing

Índice

Índice de figuras 10
1. Introducción 11
1.1 Objetivo 11
1.2 Alcance y motivación 12
1.3 Plan de trabajo 12
1.4 Estructura del documento 14
1. Introduction 15
1.1 Goals 15
1.2 Scope and motivation 16
1.3 Workplan 17
1.4 Document structure 18
2. Elementos del *cluster* 19
2.1 Placas Raspberry Pi 19
2.2 Placas utilizadas en el montaje del *cluster* 20
2.2.1 Montaje de las placas en el *cluster* 21
2.3 Switches 22
2.4 Almacenamiento 22
2.5 Alimentación 23
3. Configuración del *cluster* 25
3.1 Configuración del front-end 25
3.1.1 Instalación de MPI 25
3.1.2 Configuración de red 27
3.1.3 Configuración de NFS 27
3.2 Configuración de los nodos 28
3.2.1 Configuración cliente NFS 28
3.3 Comunicación entre front-end y los nodos 29
3.4 Arranque sin HDMI 29
4. Configuración del proceso de evaluación 31
4.1 Configuración del PC 31

4.1.1 Configuración de directorios compartidos 31
4.2 Configuraciones del *cluster* 32
4.3 Aplicaciones 35
4.3.1 Multiplicación de matrices 35
4.3.2 Compresión distribuida 36
4.3.3 Filtrado de imágenes 36
4.3.4 NAS Parallel Benchmarks (NPB) 36
5. Análisis de rendimiento 37
5.1 Multiplicación de matrices 37
5.2 Compresión distribuida 38
5.3 Filtrado de imágenes 40
5.3.1 Filtrado estático 40
5.3.2 Filtrado dinámico 41
5.3.3 Comparativa ejecución estática-dinámica 42
5.4 NAS Parallel Benchmarks 43
5.4.1 Tiempo de ejecución y MOPS 43
6. Conclusiones y Trabajo futuro 47
6.1 Conclusiones 47
6.2 Trabajo futuro 48
6. Conclusions and future work 49
6.1 Conclusions 49
6.2 Future work 50

Índice de figuras

Figura 1 Diagrama de Gannt Semanas 1-7 13
Figura 2 Diagrama de Gantt Semanas 8-14 14
Figura 3 Esquema de la arquitectura del *cluster* 19
Figura 4 Raspberry Pi 3 Modelo B. 20
Figura 5 Raspberry Pi 4 Modelo B. 21
Figura 6 Carcasa *"Joy-it Tower Case"* 22
Figura 7 TP-Link TL-SG108E 22
Figura 8 D-LINK DGS-1008D 22
Figura 9 Alimentación USB 10 puertos 23
Figura 10 Alimentación Raspberry Pi 4 23
Figura 11 *Cluster* homogéneo de Raspberry Pi 4 sin front-end 32
Figura 12 *Cluster* homogéneo de Raspberry Pi 3 sin front-end 33
Figura 13 *Cluster* Heterogéneo de Raspberry Pi 3 y Raspberry Pi 4 con front-end 34
Figura 14 Multiplicación de matrices 37
Figura 15 Compresión distribuida 39
Figura 16 Filtrado estático de imágenes 41
Figura 17 Filtrado dinámico de imágenes 42
Figura 18 tiempo de ejecución del NAS Parallel benchmark 44
Figura 19 MOPS 44

1.Introducción

Este proyecto está enfocado, principalmente, en realizar un estudio del rendimiento de un *cluster* heterogéneo de bajo presupuesto formado por placas Raspberry Pi 3 y Raspberry Pi 4. Para ello, se han ejecutado diferentes experimentos y *benchmarks* con el objetivo de determinar su comportamiento, variando el número de procesos en cada ejecución y empleando diferentes configuraciones.

El proyecto se divide en dos partes. La primera parte consiste en la realización y montaje de un *cluster* heterogéneo extensible basado en placas Raspberry Pi, así como su despliegue para la ejecución distribuida de aplicaciones MPI. En esta parte se lleva a cabo tanto el montaje físico del *cluster* como la instalación de programas necesarios y configuración para la interconexión de las diferentes placas. Una vez realizada la configuración del *cluster*, se lleva a cabo la segunda parte, en la cual se realiza un análisis de rendimiento del mismo. Para ello, se utilizan distintos programas paralelos - basados en MPI - que son ejecutados en diferentes entornos, esto es, en configuraciones homogéneas y heterogéneas del *cluster* de placas Raspberry Pi, y en un ordenador personal de propósito general. Con ello, se pretende comparar el rendimiento de las distintas configuraciones del *cluster* propuestas con con otros sistemas existentes.

1.1 Objetivo

El objetivo de este proyecto es **realizar un estudio del rendimiento de un *cluster* heterogéneo formado por diferentes versiones de la placa Raspberry Pi**. Para ello se han realizado distintas pruebas enfocadas en diferentes aspectos, como CPU o E/S, para poder observar el comportamiento del *cluster* en cada caso. Para llevar a cabo la distribución del trabajo entre las diferentes placas del *cluster* se utiliza MPI (por las siglas en inglés de *Message Passing Interface*) que permite la distribución de tareas en varios procesos para llevar a cabo una paralelización de las mismas y, por lo tanto, un mejor rendimiento en las aplicaciones.

Para la realización de este estudio se han seguido los siguientes pasos:

- Configuración del *cluster* completo, formado por placas Raspberry Pi3 y Pi4. La infraestructura del *cluster* propuesto es flexible, lo que permite un cambio de placas de forma ágil y sencilla, tanto para su ampliación como para la sustitución de alguna de las placas en caso de avería. A su vez, el *cluster* es de carácter heterogéneo ya que está formado por los dos tipos de placas distintas. Cabe destacar que, con el objetivo de realizar un análisis de rendimiento del *cluster*

más extenso y completo, se han considerado configuraciones homogéneas de la plataforma.

- Selección de las diferentes configuraciones del *cluster* donde se analizará el rendimiento y en las cuales se van a ejecutar las aplicaciones MPI, así como la elección de aplicaciones MPI que se utilizarán para realizar el estudio del rendimiento.

- Ejecución de las aplicaciones MPI. Este objetivo consiste en la ejecución de las diferentes pruebas seleccionadas tanto en la configuración heterogénea del *cluster* como en las dos configuraciones homogéneas del *cluster*. El uso de infraestructuras de distinto carácter, permite realizar un estudio más exhaustivo del *cluster* propuesto.

- Finalmente, se realiza un análisis comparativo de rendimiento entre el *cluster* propuesto y un ordenador genérico. Para ello, los resultados del estudio realizado en los objetivos anteriores se comparan con la ejecución de las aplicaciones seleccionadas sobre un PC de propósito convencional.

1.2 Alcance y motivación

El alcance de este proyecto es académico. Este proyecto nos permite observar el rendimiento de un *cluster* de bajo presupuesto utilizando diferentes configuraciones. Los resultados de las diferentes ejecuciones permiten observar la escalabilidad de un sistema de estas características y mostrar la diferencia de rendimiento de estos *clusters* con el de un PC convencional. El *cluster* podrá ser utilizado en sus distintas variantes por los alumnos de la facultad, de forma que puedan ejecutar fácilmente aplicaciones sobre éste para realizar análisis de rendimiento.

Actualmente - debido generalmente a motivos de seguridad - no se permite la ejecución de aplicaciones MPI en diferentes equipos dentro de la facultad. Esta limitación hace que el *cluster* propuesto resulte de especial utilidad para los alumnos, permitiendo la ejecución de sus prácticas en un entorno real en vez de simulado. A su vez, podrán estudiar su funcionamiento y rendimiento en función del número y versión de las placas utilizadas, todo esto sin necesidad del uso de software externo.

La utilización de placas Raspberry Pi ha dado lugar a diferentes proyectos, tanto por parte de particulares, como por parte de empresas. La conocida empresa Oracle, en septiembre de 2019 en el evento *Oracle's OpenWorld conference* en la ciudad de San Francisco, un *cluster* formado por 1.060 placas Raspberry Pi 3 conectadas entre sí por 22 switches siendo capaz de ejecutar 4.240 procesos de forma simultánea.

1.3 Plan de trabajo

La planificación de este proyecto está dividida en tres partes diferenciadas, tal y como se muestra en la Figura 1 y en la Figura 2.

La primera parte, formada por las semanas 1-2, consiste en una organización general del trabajo a realizar. Durante la primera semana se estudia el proyecto de una manera general, analizando las tecnologías a utilizar, cómo se iba a realizar el montaje, y qué configuraciones necesitaban ser modificadas para el funcionamiento del *cluster*. En la segunda semana se organizan las diferentes tareas detectadas en la semana anterior y se recaba información para llevarlas a cabo correctamente.

La segunda parte, que abarca desde la semana 3 a la semana 5 y a la que se le suma más adelante la semana 8, consiste en el montaje, configuración y despliegue del *cluster*. Durante las semanas 3 y 5 se lleva a cabo el montaje y configuración del *cluster* de Raspberry 3, mientras que en la semana 8 se realiza el montaje y configuración del *cluster* de Raspberry Pi 4 que se retrasa con respecto a lo demás por motivos logísticos ajenos al proyecto.

En la tercera parte, formada por la semana 6 y las semanas 9-12, se realiza la definición y ejecución de las diferentes pruebas. La definición se realiza durante la semana 6. Sin embargo, hasta la semana 9 - cuando ya se dispone de todas las placas debidamente configuradas - no puede comenzar la ejecución de los experimentos.

La cuarta parte, formada por las semanas 7-8 y 10-14, consiste en la documentación del proyecto. En la semana 7 y la semana 8, se define la estructura del documento y, a su vez, se comienza con la documentación de la configuración del *cluster*, la cual se lleva a cabo previamente. Entre las semanas 10 y 14 durante la ejecución de las pruebas, se redacta en detalle la memoria del proyecto, donde se añaden y se analizan los resultados de las diferentes pruebas que se ejecuten.

	Semana 1	Semana 2	Semana 3	Semana 4	Semana 5	Semana 6	Semana 7
Parte 1							
Analisis del trabajo							
Organización de tareas							
Parte 2							
Montaje cluster RP3							
Configuracion del cluster							
Pruebas cluster funcional							
Montaje del cluster RP4							
Parte 3							
Definicion de pruebas							
Ejecucion de pruebas							
Parte 4							
Documentación							

Figura 1 Diagrama de Gannt Semanas 1-7

	Semana 8	Semana 9	Semana 10	Semana 11	Semana 12	Semana 13	Semana 14
Parte 1							
Analisis del trabajo							
Organización de tareas							
Parte 2							
Montaje cluster RP3							
Configuracion del cluster							
Pruebas cluster funcional							
Montaje del cluster RP4							
Parte 3							
Definicion de pruebas							
Ejecucion de pruebas							
Parte 4							
Documentación							

Figura 2 Diagrama de Gantt Semanas 8-14

1.4 Estructura del documento

Este documento está dividido en seis capítulos con el siguiente contenido:

- Capítulo 1: En este capítulo se realiza un prólogo al trabajo realizado explicando de forma introductoria el plan de trabajo seguido, el alcance, la motivación y un resumen del objetivo del proyecto.
- Capítulo 2: Este capítulo detalla el material utilizado para la elaboración del proyecto, así como las tecnologías utilizadas.
- Capítulo 3: Este capítulo presenta la topología y la configuración usada para el correcto funcionamiento del *cluster*.
- Capítulo 4: En este capítulo se explican las diferentes combinaciones de placas que se van a emplear en el proyecto y los programas que se ejecutarán para realizar el análisis.
- Capítulo 5: En este capítulo se comparan los resultados obtenidos de las distintas ejecuciones de las aplicaciones seleccionadas en los diferentes entornos.
- Capítulo 6: Este capítulo incluye las conclusiones generales del proyecto.

1. Introduction

This project is mainly focused on carrying out a performance study of a low-budget heterogeneous cluster formed by Raspberry Pi 3 and Raspberry Pi 4 boards. For this purpose, different experiments and benchmarks have been carried out in order to determine their behavior, varying the number of processes in each execution and using different configurations.

The project is divided into two parts. The first part consists of the realization and assembly of an extensible heterogeneous cluster based on Raspberry Pi boards, as well as its deployment for distributed execution of MPI applications. In this part, it is carried out either the physical assembly of the cluster and the installation of the required programs and configuration for the interconnection of the different boards. Once the configuration of the cluster is done, the second part is carried out, in which a performance analysis of the cluster is done. For this purpose, different parallel programs - based on MPI - are executed in different environments, in homogeneous and heterogeneous configurations of the Raspberry Pi board cluster, and in a general-purpose personal computer. The objective is to compare the performance of the different proposed cluster configurations with other existing systems.

1.1 Goals

The objective of this project is to carry out a performance study of a heterogeneous cluster that consists of different versions of the Raspberry Pi board. For this purpose, different tests have been carried out focused on different aspects, such as CPU or I/O, in order to observe the behavior of the cluster in each case. To carry out the distribution of work between the different boards of the cluster, we use MPI (Message Passing Interface) which allows the distribution of tasks in several processes to carry out a parallelization of them and achieving a better performance in applications.

The following steps have been followed to carry out this study:

- Complete cluster configuration, consisting of Raspberry Pi3 and Pi4 boards. The infrastructure of the proposed cluster is flexible, which allows a quick and easy change of boards, both for expansion and for the replacement of any of the plates in case of failure. At the same time, the cluster is heterogeneous since it is formed by two different types of boards. It should be noted that, in order to perform a more detailed and complete performance analysis of the cluster, homogeneous configurations of the cluster have been considered.

- Selection of the different configurations of the cluster where the performance will be analyzed and the MPI applications will be executed, as well as the choice of MPI applications that will be used to carry out the performance study.
- Execution of MPI applications. This objective consists in the execution of the different tests in both the heterogeneous configuration of the cluster and in the two homogeneous configurations of the cluster. The use of different character infrastructures allows a more exhaustive study of the proposed cluster.
- Finally, a comparative analysis of performance between the proposed cluster and a generic computer is carried out. For this purpose, the results of the research carried out in the previous objectives are compared with the execution of the selected applications on a conventional purpose PC.

1.2 Scope and motivation

The scope of this project is academic. This project allows us to observe the performance of a low budget cluster using different configurations. The results of the different executions allow us to observe the scalability of a system of these characteristics and show the performance of these clusters compared to a conventional PC. The cluster can be used in its different variants by students of the faculty, so they can easily run applications on it for performance analysis.

Currently - generally due to security reasons - it is not allowed to run MPI applications on different computers within the computer science faculty of the Complutense university of Madrid. This limitation makes the proposed cluster particularly useful for students, allowing the execution of their practices in a real environment instead of a simulated one. At the same time, they will be able to study its functionality and performance according to the amount and version of the boards used.

The use of Raspberry Pi boards has led to different projects, both by individuals and companies. The well-known company Oracle, in September 2019 at Oracle's OpenWorld conference in the city of San Francisco, showed cluster formed by 1,060 Raspberry Pi 3 boards connected by 22 switches and capable of executing 4,240 processes simultaneously.

1.3 Workplan

The planning of this project is divided into three distinct parts, as shown in Figure 1 and Figure 2.

The first part (weeks 1-2), consists of a general organization of the work to be done. During the first week the project is studied in a general way, analyzing the technologies to be used, how the assembly was going to be done, and what configurations needed to be modified for the operation of the cluster. In the second week the different tasks detected in the previous week are organized and information is gathered to carry them out correctly.

The second part, from week 3 to week 5 and adding week 8 later, consists of the assembly, configuration and deployment of the cluster. During weeks 3 and 5 the assembly and configuration of the Raspberry 3 cluster is carried out, while in week 8 the assembly and configuration of the Raspberry Pi 4 cluster is carried out, which is delayed with respect to the rest due to logistical reasons unrelated to the project.

In the third part (week 6 and weeks 9 to 12), the definition and execution of the different tests is carried out. The definition is done during week 6. However, it is not until week 9 - when all the boards are properly configured - where the execution of the experiments can begin.

The fourth part (weeks 7-8 and 10-14), consists of project documentation. In week 7 and week 8, the document structure is defined and, at the same time, the documentation of the cluster configuration is started. Between weeks 10 and 14 during the execution of the tests, the project memory is written in detail, where the results of the different tests executed are added and analyzed.

1.4 Document structure

This document is divided into six chapters with the following content:

- Chapter 1: In this chapter there is a prologue to the work carried out explaining in an introductory way the work plan followed, the scope, motivation and a summary of the objective of the project.
- Chapter 2: This chapter details the material used for the elaboration of the project, as well as the technologies used.
- Chapter 3: This chapter presents the topology and configuration used for the proper functioning of the cluster.
- Chapter 4: This chapter explains the different combinations of boards that will be used in the project and the programs that will be executed to perform the analysis.
- Chapter 5: This chapter compares the results obtained from the different executions of the selected applications in different environments.
- Chapter 6: This chapter includes the general conclusions of the project.

2. Elementos del *cluster*

En este capítulo se detallan los elementos utilizados en el *cluster*, así como el software utilizado para la realización y configuración del mismo. La Figura 3 muestra un esquema de la arquitectura general del *cluster*, donde se aprecian los distintos tipos de placa - Raspberry Pi3 y Raspberry Pi4 - además de los switches para poder conectar las mismas a la red de comunicaciones.

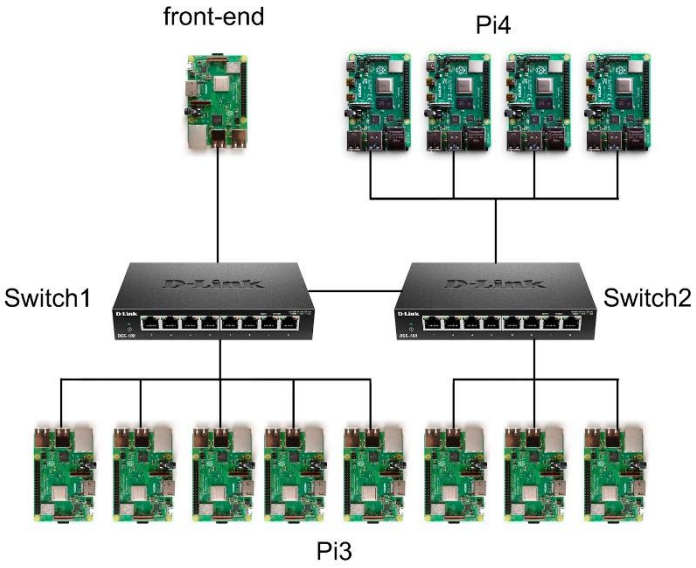

Figura 3 Esquema de la arquitectura del *cluster*

2.1 Placas Raspberry Pi

El *cluster* está formado por diferentes placas Raspberry Pi que son las encargadas de realizar el trabajo distribuido desde el front-end y realizar los diferentes cómputos establecidos por las aplicaciones MPI. Actualmente existen varios modelos de placas Raspberry Pi, los cuales son descritos a continuación:

Raspberry Pi 1: Model A+ y Model B+. Esta es la primera versión de Raspberry que sale al mercado, incluye 26 puertos GPIO, HDMI y RCA y una memoria RAM de 256 Mb en su primer modelo, el A. Este modelo cuenta con un procesador Single-Core de 700 MHz, una gráfica Broadcom Videocore IV y slot para tarjetas SD. En el modelo B + cuenta con 4 puertos USB, se le añade puerto Ethernet y se produce un aumento de su memoria RAM a 512Mb. Este modelo sustituye el slot para tarjetas SD, por soporte para tarjetas microSD.

Raspberry Pi 2: Model B. Esta versión de la placa Raspberry Pi cuenta con diversos cambios con respecto a la Raspberry Pi 1. Entre ellos, encontramos 40 pines GPIO, la

supresión del puerto RCA, el uso de 1Gb de memoria RAM compartida con la gráfica y la sustitución del procesador Single-Core de 700 MHz por un Quad Core de 900 MHz de frecuencia.

Raspberry Pi 3: Model A+, Model B y Model B+. Esta generación, supera por primera vez la frecuencia de 1GHz incluyendo un procesador QuadCore 1.2GHz. En esta generación se comienza a incluir Wi-Fi y bluetooth. En el caso de la Model B+, se aumenta la frecuencia del procesador a 1,4Ghz y se añade un puerto Gigabit.

Raspberry Pi 4: Model B. Esta placa tiene varias versiones, variando su memoria RAM entre 2Gb, 4Gb y 8Gb. Como el anterior modelo, cuenta con puerto Gigabit Ethernet y se sustituyen los puertos USB por USB 3.0. Con respecto al procesador esta versión viene equipada con un Quad-Core que funciona a una frecuencia de 1,5 GHz.

Raspberry Pi Zero: Zero y ZeroW. La Raspberry Pi Zero es una placa de un tamaño aún más reducido. Cuenta con unas medidas de 65mm x 31mm. Esta placa dispone de un procesador Single-Core de 1GHz, 512MB de memoria RAM, dos microUSB para alimentación y datos, y entradas mini HDMI y mini RCA. De este tipo de placas existe una segunda versión Raspberry Pi ZeroW que incluye Wi-Fi.

Las placas seleccionadas para el *cluster* son dos versiones diferentes de Raspberry Pi: Raspberry Pi 3 modelo B y Raspberry Pi 4 modelo B

2.2 Placas utilizadas en el montaje del *cluster*

La mayoría de las placas que conforman el *cluster* son placas Raspberry Pi 3 modelo B (ver Figura 4). En concreto contamos con 9 de estas placas. Este modelo de Raspberry Pi se lanzó al mercado en febrero de 2016 siendo el modelo menos potente que va a estar presente en el *cluster*, actualmente este tipo de placas tienen un precio de 40€.

Figura 4 Raspberry Pi 3 Modelo B.

Las especificaciones de este modelo son las siguientes:

- SoC: Broadcom BCM2837
- CPU: 1,2GHz 64-bit quad-core ARMv8
- RAM: 1GB LPDDR2
- Ethernet socket Ethernet 10/100 BaseT
- 802.11 b / g / n LAN inalámbrica y Bluetooth 4.1
- USB 4 x Conector USB 2.0

Al ser un *cluster* heterogéneo, también se utilizan 4 placas Raspberry Pi 4 modelo B (ver Figura 5). Existen varios modelos con variaciones en la capacidad de la memoria RAM, con una capacidad de 2GB, 4GB y 8GB y con un precio aproximado de 40€, 60€ y 80€ respectivamente. Este proyecto está llevado a cabo con placas de 4GB de RAM.

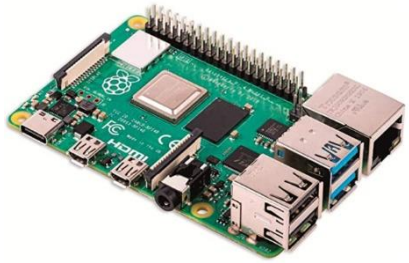

Figura 5 Raspberry Pi 4 Modelo B.

Las especificaciones de esta placa son las siguientes:

- SoC: Broadcom BCM2711
- CPU: 1.5GHz 64-bit quad-core Cortex-A72
- RAM: 4 GB LPDDR4 SDRAM
- Ethernet Gigabit Ethernet
- 802.11ac LAN inalámbrica y Bluetooth 5.0
- USB 2 x Conector USB 2.0
- USB 2 x Conector USB 3.0

2.2.1 Montaje de las placas en el *cluster*

Las placas están colocadas en una carcasa vertical minimizando así el espacio ocupado. El modelo de carcasa seleccionado es *"Joy-It Tower-Case para Raspberry Pi"* con capacidad para 7 placas Raspberry Pi (ver Figura 6). Esta carcasa tiene un precio aproximado de 9€

Figura 6 Carcasa *"Joy-it Tower Case"*

2.3 Switches

Los switches se encargan de la conexión entre las diferentes placas. En este caso tenemos dos switches de ocho puertos cada uno. Los modelos seleccionados han sido Switch TP-Link TL-SG108E (ver Figura 7) y D-LINK DGS-1008D (ver Figura 8) y tienen un precio de 30€ y 20€ respectivamente.

Figura 7 TP-Link TL-SG108E

Figura 8 D-LINK DGS-1008D

2.4 Almacenamiento

Las placas Raspberry Pi no cuentan con almacenamiento integrado. Por ello, es necesario contar con tarjetas microSD donde almacenar el sistema operativo. En este caso se han utilizado tarjetas microSD de 16Gb para cada una de las placas en el cluster. Estas tarjetas tienen un máximo de velocidad de lectura de 45 MB/s y tienen un precio aproximado de 9€.

2.5 Alimentación

Para la alimentación del módulo formado por Raspberry Pi 3 tanto del *cluster* como del switch se ha utilizado un cargador USB de 10 puertos que funciona a 5V y 2,4A (ver Figura 9). El modelo seleccionado ha sido el de *AmazonBasics*, ya que proporciona el voltaje y la intensidad necesarios para alimentar el *cluster*, ocupa poco espacio y cuenta con un interruptor de seguridad interno que permitirá proteger y apagar el *cluster* de una forma sencilla. Esta alimentación tiene un coste de 25€

Figura 9 Alimentación USB 10 puertos

Para las Raspberry Pi 4 se han utilizado los cargadores oficiales USB tipo C de 5,1V, 3A y con un coste de 11€ (ver Figura 10).

Figura 10 Alimentación Raspberry Pi 4

3. Configuración del *cluster*

En este capítulo se describen en detalle las distintas configuraciones del *cluster* utilizadas para la ejecución de las pruebas. Por un lado, se proporcionan las especificaciones de las diferentes placas Raspberry Pi que se utilizan como front-end y como nodos de cómputo. Por otro lado, se detalla la interacción del *cluster* con el front-end para llevar a cabo la ejecución de aplicaciones MPI.

3.1 Configuración del front-end

Al necesitar una instancia de los diferentes programas y una forma independiente de acceder al *cluster*, se ha decidido utilizar una placa que realizara el rol de front-end, desde la cual se ejecutan los experimentos y que sirve para actuar de intermediario entre el usuario y el *cluster*. Desde esta placa es desde la que se lanzan las pruebas y la que comparten los diferentes recursos con el resto del *cluster*. El front-end al igual que el resto de placas Raspberry Pi, necesita para funcionar la instalación de un sistema operativo. En este caso se ha utilizado Raspbian Buster Lite[1], un sistema operativo de Raspberry Pi basado en Debian Buster. Cabe destacar que se utiliza esta versión lite ya que no va a ser necesaria la utilización de una interfaz gráfica.

En este caso es importante tanto la ruta de instalación de MPI como la dirección IP que se le asigne, ya que para evitar la necesidad de instalar MPI en cada uno de los nodos de cómputo, este front-end comparte mediante NFS con el resto del *cluster*, tal y como se detalla en los siguientes puntos, la instalación de MPI, haciendo de esta forma el *cluster* más extensible.

3.1.1 Instalación de MPI

El primer paso para la configuración de nuestro equipo es la instalación de MPI (*Message Passing Interface*). MPI es una biblioteca de código abierto que se utiliza para la paralelización de tareas, la cual nos permitirá utilizar los nodos de nuestro *cluster* de manera simultánea en la ejecución las pruebas seleccionadas. Para ello, se ha seleccionado la distribución MPICH[2], que se trata de una implementación estándar de MPI de alto rendimiento. En este caso la instalación que se ha realizado es de la versión 3.1.

[1] https://www.raspberrypi.org/downloads/raspberry-pi-os/
[2] https://www.mpich.org/

Para realizar la instalación, comenzamos creando un directorio, descargando y descomprimiendo el software para su posterior instalación. Es Esto se realiza con los siguientes comandos:

```
mkdir mpich
cd mpich
wget http://www.mpich.org/static/downloads/3.1/mpich-
3.1.tar.gz
tar xfz mpich-3.1.tar.gz
```

A continuación, creamos un directorio donde se va a establecer la instalación de MPI y lo configuramos como se muestra a continuación. Para la instalación es necesario también el compilador `gfortran`, que no viene por defecto instalado en Raspbian.

```
sudo mkdir /home/rpimpi
sudo mkdir /home/rpimpi/mpi-install
sudo apt-get install gfortran
cd /home/pi/mpi-build
sudo /home/pi/mpich2/mpich-3.1/configure
-prefix=/home/rpimpi/mpi-install
sudo make
sudo make install
```

Una vez realizada la instalación debemos añadir la ruta de MPI al PATH para poder utilizarlo, para ello hay que editar el fichero `.bashrc` y la ruta del compilado de la instalación de MPI:

```
PATH=$PATH:/home/rpimpi/mpi-install/bin
```

Una vez añadido al *path*, reiniciamos la máquina virtual y nos aseguramos de que la instalación se ha realizado correctamente viendo si la consola nos reconoce el comando “mpiexec”.

3.1.2 Configuración de red

Para la configuración de red, se ha decidido la utilización de direcciones IP estáticas, esto, permite un mayor control para la ejecución de los experimentos haciendo que sea más fácil reconocer a qué placa pertenece cada una de las direcciones IP, al front-end se le va a añadir la dirección 169.254.12.1. Para ello editamos el fichero `/etc/network/interfaces` dejándolo de la siguiente manera:

```
auto eth1
iface lo inet loopback
iface eth1 inet static
address 169.254.12.1
netmask 255.255.255.0
network 169.254.12.0
```

Para asegurarnos del correcto funcionamiento, reiniciamos el dispositivo para ejecutar el comando `ifconfig`, que indicará los datos de las interfaces de red mostrando la IP establecida en el fichero.

3.1.3 Configuración de NFS

Para la correcta configuración del *cluster*, es necesaria la instalación y configuración de NFS, de esta forma, se realiza un montaje de directorios compartidos entre las placas del *cluster*. En primer lugar, se debe instalar el paquete `nfs-kernel-server` utilizando el siguiente comando:

```
sudo apt-get install nfs-kernel-server
```

para exportar el directorio se debe crear una entrada en el fichero `/etc/exports` añadiendo lo siguiente:

```
/home/rpimpi *(rw,sync,no_subtree_check)
```

El asterisco se utiliza para que el directorio se comparta en todas las redes a las que tiene acceso el front-end. Los distintos flags significan:

- `rw`: los nodos pueden leer y escribir en la carpeta compartida.
- `sync`: se aplican los cambios sólo cuando se hace *commit*.
- `no_subtree_check`: este flag se establece por seguridad para que los nodos o quien tenga acceso a ese recurso compartido no pueda visualizar los directorios anteriores a la carpeta compartida.

Una vez finalizados estos pasos, el contenido del directorio `/home/rpimpi` del front-end queda compartido con el resto de los nodos.

3.2 Configuración de los nodos

Para configurar los nodos se necesita instalar - en las tarjetas SD de cada placa - el sistema operativo tal y como se ha realizado en el front-end.

Una vez instalado es necesario configurar la red para cada nodo con su propia ID, tal como se describe en el apartado 3.1.2. En nuestro caso, las Raspberry Pi 3 tendrán las IPs comprendidas en el rango 169.254.12.2-169.254.12.9, mientras que las placas Raspberry Pi 4 tendrán las direcciones entre 169.254.12.10-169.254.12.13. Con las IPs estáticas nos será más sencillo configurar las diferentes ejecuciones y estudiar el rendimiento de los diferentes nodos.

3.2.1 Configuración cliente NFS

Para poder ejecutar aplicaciones MPI sin instalar las bibliotecas en cada uno de los nodos de forma individual, necesitamos acceder al directorio compartido establecido en el punto 3.1.3 del presente documento. Para ello tenemos que instalar el cliente NFS en todos los nodos para poder acceder desde ellos a los directorios compartidos por el front-end, tal y como se detalla a continuación:

```
sudo apt-get install nfs-common
```

Una vez instalado, la configuración se realizará editando el fichero `/etc/fstab`, valdría con crear una carpeta donde montar la carpeta compartida y añadir en `/etc/fstab` la siguiente entrada:

```
mkdir /home/rpimpi

169.254.12.1:/home/rpimpi /home/rpimpi nfs
```

Para comprobar su correcto funcionamiento bastaría con acceder a la carpeta creada y comprobar que los archivos mostrados corresponden a los del front-end.

3.3 Comunicación entre front-end y los nodos

La comunicación entre el front-end y los nodos se realiza por red. En este caso se utiliza ssh y, para ello, es necesario crear claves ssh para establecer la comunicación entre front-end y los diferentes nodos. Este proceso se puede llevar a cabo directamente desde el front-end con la siguiente secuencia de instrucciones:

```
ssh-keygen
cp id_rsa.pub frontEnd
ssh pi@169.254.12.2
ssh-keygen
cd .ssh
cp id_rsa.pub pi01
scp 169.256.12.1:/home/pi/.ssh/frontEnd
cat frontEnd >> authorized_keys
exit
```

Esto sería un ejemplo para uno de los nodos, en este caso el de IP 169.256.12.2, pero habría que repetirlo con todos los nodos del *cluster*.

3.4 Arranque sin HDMI

Las placas del *cluster* no están conectadas a ninguna interfaz gráfica. Esto es debido a que no es necesaria una interacción del usuario, puesto que el front-end es el encargado de la distribución del trabajo. Sin embargo, Raspbian por defecto no permite el arranque si no detecta un cable HDMI conectado. Para poder realizar el arranque de las placas Raspberry Pi sin interfaz gráfica es necesario modificar el archivo `/boot/config.txt` añadiendo lo siguiente:

```
hdmi_force_hotplug = 1
```

4. Configuración del proceso de evaluación

En este apartado se detallan los experimentos que se van a realizar sobre diferentes configuraciones del *cluster*. Además, se realizan los experimentos en un PC convencional para hacer un estudio comparativo.

4.1 Configuración del PC

En este caso, se utilizará un equipo de sobremesa para realizar un estudio comparativo del rendimiento de los diferentes programas ejecutados tanto en PC como en las diferentes configuraciones del *cluster*.

Al no contar con un equipo con un equipo Linux se ha decidido la virtualización por medio del software "Virtualbox", creando una máquina virtual con el sistema operativo "Raspberry Pi Desktop", que es una distribución de Raspberry Pi OS especial para PC y MAC de 32 bits.

Las características de la máquina virtual se detallan a continuación:

- CPU Intel core i5-8600K 3.6GHz de 4 núcleos.
- 8Gb Ram DDR4 2400MHz
- Disco duro HDD SATA 6Gb/s

Para la configuración se deben realizar los pasos establecidos en el punto 3.1.1 y 3.1.2 para para la instalación de MPI y la configuración de red, respectivamente. En este caso el PC está configurado con la dirección IP 169.254.12.15.

4.1.1 Configuración de directorios compartidos

El *cluster* no dispone de conexión a Internet. Sin embargo, está conectado a una red privada local, por lo que es necesario proporcionar una vía de acceso a los diferentes recursos y aplicaciones requeridos. Por ello, se debe establecer acceso a una carpeta compartida con el equipo, en este caso: `/home/shared` (configurada en el punto 3.1.3) y compartir con el front-end las aplicaciones y recursos necesarios para ejecutar en el *cluster*. Con lo cual, el fichero `/etc/exports` para compartir el directorio /home/shared, se actualiza de la siguiente manera:

```
/home/shared *(rw,sync,no_subtree_check)
```

Para que se pueda tener acceso a estos recursos compartidos desde el front-end se deben seguir las instrucciones establecidas en el punto 3.2.1 se debe crear una nueva carpeta en el front-end `/home/shared` dónde realizar el montaje del directorio compartido con el equipo. Una vez hecho esto se actualiza el fichero `/etc/fstab` añadiendo la dirección IP del equipo seguido de dos puntos y su directorio a compartido, y separado por un espacio se añade el directorio del front-end en el que se quiere realizar el montaje del directorio:

```
169.254.12.15:/home/shared /home/shared nfs
```

Para comprobar su correcto funcionamiento bastaría con acceder a la carpeta creada desde el front-end y comprobar que los archivos mostrados corresponden a los del equipo.

4.2 Configuraciones del *cluster*

Para realizar el análisis de rendimiento, se han seleccionado tres aplicaciones MPI. Por un lado, estas aplicaciones se ejecutan en configuraciones homogéneas, es decir, la aplicación se ejecuta en un mismo tipo de placa. Por otro lado, se propone utilizar una configuración heterogénea, formada por los dos tipos de placa. Así, contamos con tres configuraciones definidas.

Figura 11 *Cluster* homogéneo de Raspberry Pi 4 sin front-end

La primera configuración - homogénea - está formada por 4 placas Raspberry Pi 4 (ver Figura 12). Al contar con procesadores Quad-core al igual que las placas Raspberry Pi 3, esta configuración puede ejecutar 16 procesos simultáneamente.

La segunda configuración, al igual que la anterior, es una configuración homogénea. En particular, esta configuración cuenta con 8 placas Raspberry Pi 3. La Figura 11 muestra esta configuración, donde se aprecian las 8 placas con 4 núcleos cada una, por lo que se pueden procesar 32 procesos de forma simultánea.

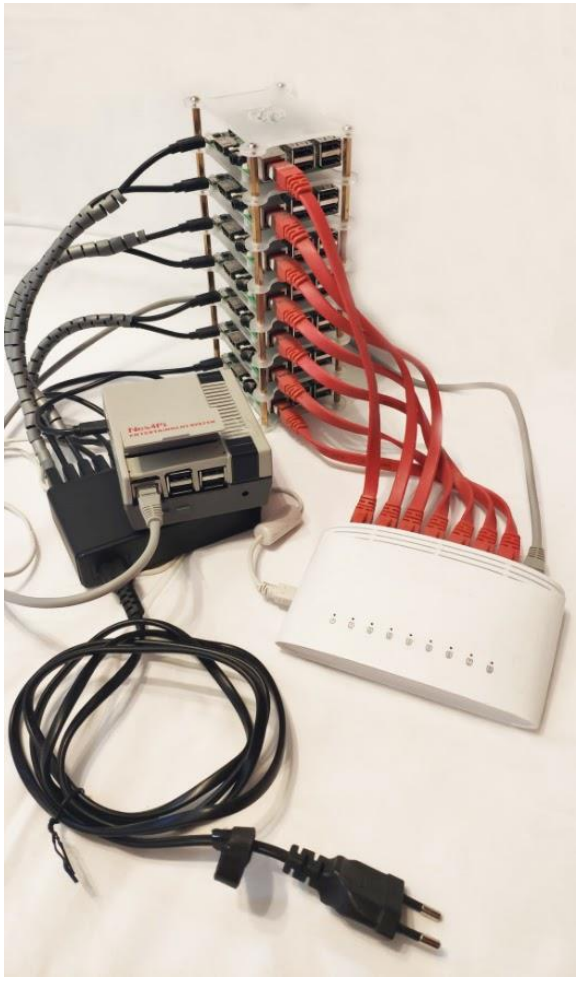

Figura 12 *Cluster* homogéneo de Raspberry Pi 3 sin front-end

Por último, la tercera configuración (Figura 13) es heterogénea, es decir, está formada por placas de diferentes tipos, concretamente, esta configuración cuenta con 12 placas en total, 8 raspberry Pi 3 y 4 Raspberry Pi 4 como puede verse en la Figura 13, en esta Figura se puede observar tanto la parte del cluster de Raspberry Pi 3, de la Figura 11, como el *cluster* formado por placas Raspberry Pi 4, los switches de ambos clusters están conectados con un cable de red, y en medio, con una carcasa transparente se observa el front-end, que es el encargado de la distribución del trabajo. Con esta configuración pueden ejecutarse 48 procesos de forma simultánea.

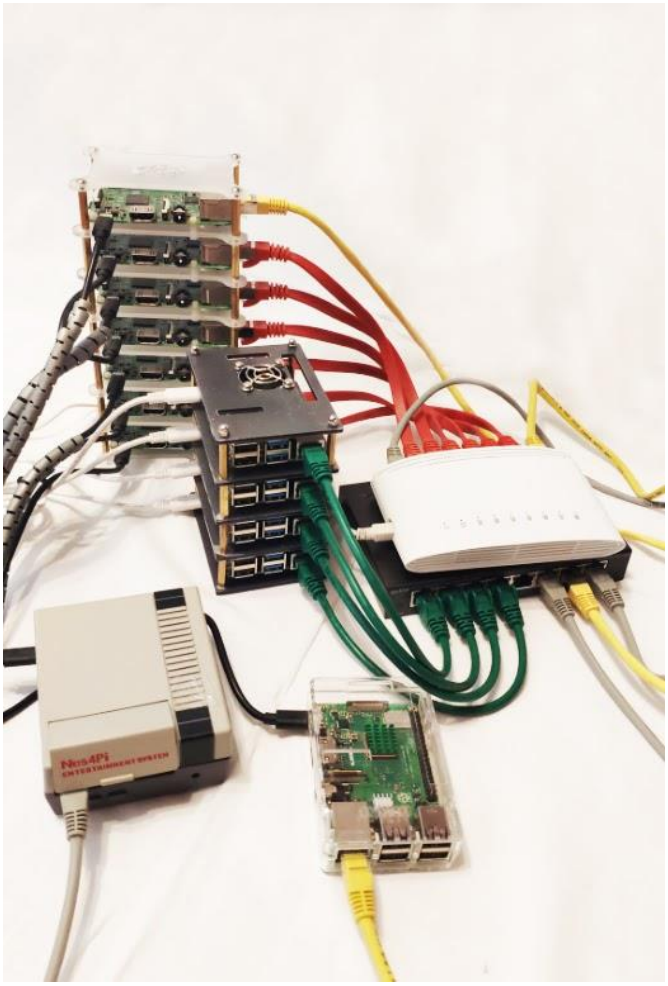

Figura 13 *Cluster* Heterogéneo de Raspberry Pi 3 y Raspberry Pi 4 con front-end

Para la ejecución de las pruebas en las diferentes configuraciones, se han generado tres archivos en el front-end denominados `machinefile`, `machinefileRP3` y `machinefileRP4`, los cuales contienen las direcciones IP de todos los nodos, los nodos Raspberry Pi 3 y los nodos Raspberry Pi 4, respectivamente. El fichero `machinefil`e es el utilizado para la configuración heterogénea, a la cual se le hará referencia a partir de ahora en el presente documento como *config hetPis*. Este fichero tiene el siguiente formato:

```
169.245.12.2:4
169.245.12.3:4
169.245.12.4:4
169.245.12.5:4
169.245.12.6:4
169.245.12.7:4
169.245.12.8:4
169.245.12.9:4
169.245.12.10:4
169.245.12.11:4
169.245.12.12:4
169.245.12.13:4
```

Como se puede observar, el fichero incluye las direcciones IPs en los que se va a realizar la ejecución seguida de dos puntos y el número de cores que van a ejecutar en cada uno de ellos. Las configuraciones homogéneas contendrán únicamente las direcciones de las placas Raspberry Pi del mismo tipo: En el caso del de la configuración homogénea de placas Raspberry Pi 3, que se referencia más adelante como *config homRP3*, su fichero machinefileRP3 contendrá las direcciones comprendidas entre 169.254.12.2 y 169.254.12.9; y la configuración homogénea formada por placas Raspberry Pi 4, a la cual se le hará referencia como *config homRP4*, su fichero machinefileRP4 contendrá las direcciones comprendidas entre 169.254.12.10 y 169.254.12.13

Los resultados obtenidos de las diferentes ejecuciones se contrastan en el siguiente capítulo con el tiempo de ejecución de las mismas pruebas en el PC especificado en el punto 3.1.

4.3 Aplicaciones

Para realizar un análisis completo, es necesario ejecutar diferentes experimentos con diferentes objetivos para observar las variaciones de rendimiento en diversos entornos, como puede ser cómputo, comunicación o lectura/escritura.

4.3.1 Multiplicación de matrices

La primera aplicación elegida consiste en la multiplicación de matrices de números enteros. La multiplicación de matrices es un problema con una complejidad exponencial, por lo que es un buen medidor para establecer un benchmark fijándonos en la potencia de cómputo de las distintas placas.

Para esta prueba se ha utilizado un código alojado en GitHub[3] con pequeñas variaciones para aumentar el tamaño de las matrices a multiplicar y para evitar un error de surge al reservar en memoria una matriz de más de 600x600.

El propio programa al acabar indica el tiempo de cómputo únicamente de las instrucciones MPI, sin contar con la generación aleatoria de la matriz. Para esta prueba se ha utilizado una matriz de 3000x3000 números enteros

[3] https://gist.github.com/kmkurn/39ca673bb37946055b38

4.3.2 Compresión distribuida

La segunda aplicación que se ha seleccionado es la compresión distribuida de un fichero. Esta prueba consiste en realizar una compresión en paralelo con los diferentes nodos utilizando MPI. Para ello hemos utilizado el compresor paralelo llamado MPIBZIP2, que contiene una implementación MPI del algoritmo BZIP2[4] de compresión. Para la realización de esta prueba, se ha utilizado un archivo de audio .wav de 1.9GB, que es comprimido por el *cluster* en paralelo.

4.3.3 Filtrado de imágenes

Esta aplicación forma parte de las prácticas de la asignatura PSD de la facultad de informática y consiste en la aplicación de un filtro de escala de grises a blanco y negro en ficheros con formato BMP.

Con esta aplicación se realizan dos tipos de ejecuciones, una ejecución estática y una dinámica. En la ejecución estática se selecciona el número de nodos deseado y las filas de la imagen se dividen en los nodos disponibles, es decir, se realiza el reparto de carga entre los nodos antes de comenzar con la distribución del trabajo, estableciendo, de esta manera, la misma carga de trabajo a todos los nodos sin importar su potencia. En la ejecución dinámica, se determina el tamaño de los datos que se envían en cada ocasión, haciendo así que los nodos con mayor capacidad de cómputo tengan más carga de trabajo y se pueda obtener un mayor rendimiento. Con estas dos ejecuciones se espera obtener resultados que contrasten el comportamiento del *cluster* heterogéneo cuando se carga de mayor trabajo a las placas Raspberry Pi 4, por su mayor capacidad de cómputo, que a las Raspberry Pi 3.

4.3.4 NAS Parallel Benchmarks (NPB)

La última aplicación es el NAS Parallel Benchmark[5]. Se trata de un pequeño grupo de programas diseñados por la NASA, en el cual se ejecutan diferentes algoritmos para analizar el rendimiento de los sistemas distribuidos, en este caso se utilizará el benchmark dedicado a computación paralela.

Con esta ejecución, se puede observar cómo varían los MOPS (millones de operaciones por segundo) de las distintas configuraciones y asignándoles diferentes nodos.

[4] https://rc.byu.edu/wiki/index.php?page=Parallel+BZIP2+Compression

[5] https://www.nas.nasa.gov/publications/npb.html

5. Análisis de rendimiento

En este apartado se muestran los resultados de las pruebas mencionadas en las diferentes configuraciones del *cluster*. En cada una de ellas se muestra una tabla de resultados con el tiempo en minutos que tardó cada sistema en cada ejecución y un gráfico para poder hacer un mejor contraste de los datos de la tabla.

5.1 Multiplicación de matrices

Para la prueba de multiplicación de matrices se ha generado una matriz aleatoria de 3000 filas y 3000 columnas, la cual se divide a partes iguales entre el número de procesos seleccionados. La medición del tiempo no contempla la generación de la matriz aleatoria. Específicamente, el programa únicamente muestra el tiempo de ejecución desde el comienzo de la operación de multiplicación hasta que se obtiene el resultado.

	2	4	8	12	16	20	24	32	48
homPi3	60,66	52,42	26,9	18,99	14,72	12,49	9,98	8,48	
homPi4	17,61	14,89	9,37	5,98	4,48	5,66	5,49	4,43	
hetPi	56,97	51,59	28,19	18,56	14,43	11,76	10,44	8,4	6,6
PC	3,83	1,23	1,12	1,23	1,7	1,73	1,83	2,9	

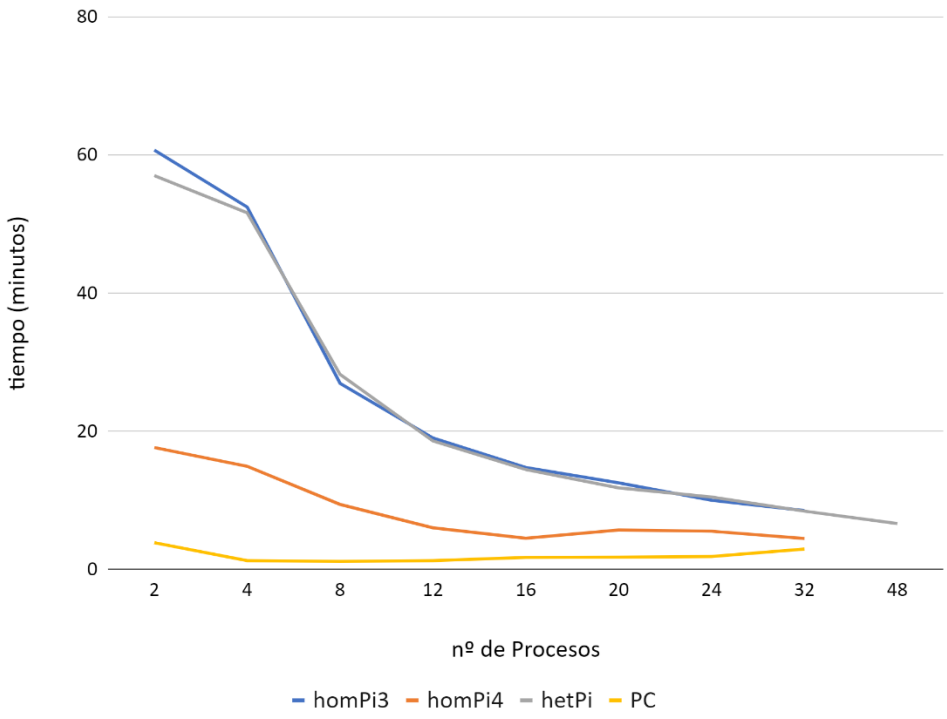

Figura 14 Multiplicación de matrices

Como se puede observar en el gráfico de la Figura 14, la mejora del rendimiento a medida que aumentan los procesos es notable. Se puede ver en el gráfico que con la configuración *config homPi3* se consigue reducir el tiempo de ejecución en más de 50 minutos, lo que significa una mejora del rendimiento en un 86,66%. Respecto a la CPU, se puede observar que el nuevo modelo de Raspberry Pi 4 tiene una mayor potencia de cómputo que el anterior modelo. Esto es debido a que la ejecución únicamente se lleva a cabo con dos procesos, uno realiza la distribución y otro el cómputo. A su vez, se puede observar que el tiempo de ejecución de la aplicación cuando se utiliza la *config homRP4* (17,61 minutos) es menor a un tercio del tiempo necesario para ejecutar la misma aplicación en la *config homRP3* (60,66 minutos).

En la configuración *config homRP4*, podemos observar que entre la ejecución de 16 procesos y la de 24 procesos el tiempo de ejecución crece en más de un minuto. Esto es debido a que al contar únicamente con 4 Raspberry Pi 4, con 4 núcleos cada placa, se pueden ejecutar únicamente 16 procesos de forma simultánea, esto da lugar a la realización de una nueva distribución de trabajo al acabar la primera ejecución de 16 procesos penalizando el rendimiento. De hecho, en esta configuración, se puede observar que con 32 procesos el tiempo de ejecución de la prueba es bastante similar al de 16. Esto puede darse debido a que al contar con 16 cores de ejecución y ejecutar 32 procesos, aunque el front-end tenga que distribuir en dos ocasiones, al ser múltiplo de 16 se aprovechan todos los cores durante la mayor parte del tiempo de ejecución de la prueba.

En cuanto al *cluster* heterogéneo formado por placas tanto Raspberry Pi 3 como Raspberry Pi 4, se muestra una curva en el gráfico muy similar a la configuración homogénea de Raspberry Pi 3. Esto se debe a que la carga de trabajo se distribuye por igual - de forma estática - en todas las placas, independientemente de si es una Raspberry Pi 3 o una Raspberry Pi 4. Esta aplicación no concluye hasta que todas las placas hayan realizado su parte del cómputo, por lo que, aunque contemos con placas más potentes y estas hayan completado su parte del cómputo, para la conclusión de la aplicación se hace necesaria la espera a la finalización del cómputo de las placas menos potentes. Así, las placas menos potentes penalizan el rendimiento global del sistema.

5.2 Compresión distribuida

Para esta prueba de rendimiento se ha utilizado MPIBZIP2, que es una herramienta de compresión paralela para ser utilizada por múltiples procesadores.

Se ha utilizado un archivo .wav, ya que este tipo de archivos tienen un amplio margen de compresión El archivo original era de 1,9 GB y tras la compresión se generó un archivo comprimido con el fichero de audio de 1,3GB.

	2	4	8	12	16	20	24	32	48
homPi3	28,53	11,01	8,35	7,88	8,01	8,03	8,01	8,03	
homPi4	12,01	4,66	3,36	3,21	3,16	3,2	3,28	3,33	
hetPi	29,88	11,08	8,43	8,38	8,08	8,03	8,01	8,11	8,2
PC	2,88	1,1							

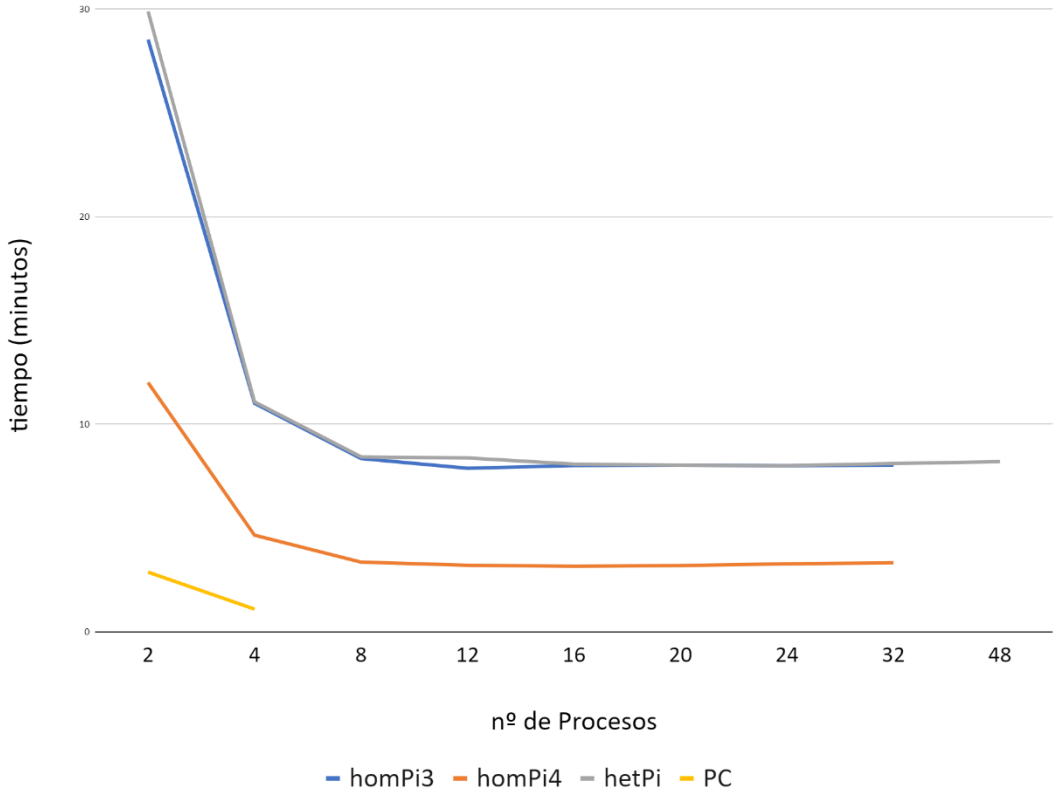

Figura 15 Compresión distribuida

En este experimento (ver Figura 15) se puede observar en todos los casos relacionado con el *cluster*, que la tendencia en primera estancia es descendiente y a partir de 8 procesos se mantiene, en lugar de seguir descendiendo. Esto es debido a que se puede estar produciendo un cuello de botella al distribuir el archivo entre los nodos del *cluster* causado por las características del front-end. Cabe recordar que el frontend utilizado es una placa Raspberry Pi 3, esta placa, tal como se indica en el capítulo 2 de este documento, tiene una interfaz ethernet 10/100 BaseT, lo que puede considerarse insuficiente para la distribución de archivos de este rango de tamaño.,

Tanto la configuración formada únicamente por Raspberry Pi 3 como la formada configuración heterogénea tienen unos tiempos muy similares. En las primeras ejecuciones en el *cluster* heterogéneo se han producido en las placas Raspberry Pi 3, y más tarde no se ha podido aprovechar la mayor potencia de la red debido al cuello de botella de la red mencionado anteriormente.

La configuración homogénea de Raspberry Pi 4, realiza la compresión bastante más rápido que las otras dos configuraciones y aunque se sigue produciendo el mismo cuello de botella de red, la ejecución con 16 procesos tarda únicamente un 9% más que la ejecución con dos procesos del PC. Del PC únicamente se muestran los tiempos con dos y cuatro procesos ya que al tener únicamente cuatro núcleos no se van a obtener mejoras de rendimiento a partir de cuatro procesos.

5.3 Filtrado de imágenes

Esta prueba se trata de una práctica de la asignatura Programación de Sistemas Distribuidos, la cual consiste en el filtrado de una imagen BMP en escala de grises, el cual a partir de dicha imagen genera otra nueva en blanco y negro con el *threshold* especificado. Hasta ahora, en el resto de pruebas se ha visto un rendimiento similar entre el *cluster* homogéneo de Raspberry Pi 3 y el *cluster* heterogéneo. Esta prueba consta de dos ejecuciones distintas por cada configuración establecida. En este punto, se mostrarán tres apartados, uno para cada ejecución y otro con la comparación de las diferentes ejecuciones.

La imagen utilizada, ocupa 501 MB y tiene unas dimensiones de 17648x9927 pixeles, con lo cual cada fila que se procese tendrá que procesar 17648 del ancho de la imagen. Por ello y para maximizar la explotación de recursos del *cluster* heterogéneo en la ejecución dinámica se ha determinado que a cada proceso se le enviaran 10 filas.

5.3.1 Filtrado estático

La ejecución estática, como se menciona anteriormente, realiza el reparto de la carga de trabajo antes de la distribución del mismo, con este tipo de filtrado en la configuración heterogénea *config hetPi,* se dotará de la misma carga de tanto a las placas Raspberry Pi 3, como a las Raspberry Pi4. En esta ejecución estática se dividen las filas de la imagen entre el número de procesos seleccionados.

	2	4	8	12	16	20	24	32	48
homPi3	18,46	6,4	2,93	2,31	1,83	1,66	1,41	1,33	
homPi4	10,76	3,71	1,66	1,13	0,85	0,9	0,95	0,98	
hetPi	18,45	6,43	2,93	1,93	1,73	1,38	1,23	1,1	1,06
PC	3,98	1,33							

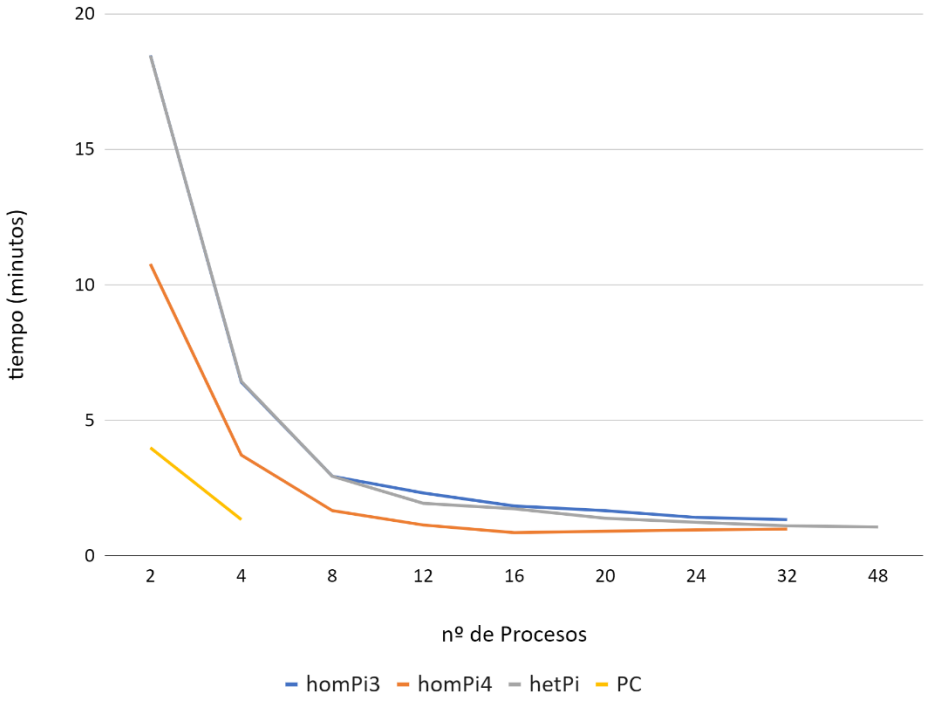

Figura 16 Filtrado estático de imágenes

Con esta ejecución estática mostrada en la Figura 16, ya se puede ver una mejoría de rendimiento significativa en los tres sistemas. Por primera vez, se puede observar que las diferentes configuraciones del *cluster* superan o igualan al PC en rendimiento, en el caso de la configuración *config homRP3,* se iguala en la ejecución con 32 procesos, la configuración *config homRP4* lo supera a partir de la ejecución con 12 procesos y la configuración heterogénea *config hetPi* obtiene también un mejor rendimiento que el PC a partir de la ejecución con 24 procesos.
Se puede observar como en todas las ejecuciones anteriores, que el sistema heterogéneo y el que está formado por Raspberry Pi 3 manejan unos tiempos prácticamente iguales.

5.3.2 Filtrado dinámico

La ejecución dinámica realiza el reparto de trabajo a medida que se va ejecutando la aplicación, estableciendo una mayor carga de trabajo en los nodos más potentes. En esta ejecución dinámica se determinará un número de filas, que son las que se enviarán a cada nodo cada vez que se haga una distribución de trabajo. Cada vez que un proceso termine de ejecutar el trabajo que le ha sido asignado, recibirá más filas para procesar, haciendo así que los nodos más rápidos tengan más carga de trabajo. De esta manera, el objetivo es obtener un rendimiento mayor con el *cluster* heterogéneo dándole una mayor prioridad a las placas Raspberry Pi 4.

	2	4	8	12	16	20	24	32	48
homPi3	18,7	6,41	2,86	1,93	1,5	1,31	1,11	1,05	
homPi4	10,66	3,65	1,65	1,1	0,86	0,9	0,9	0,91	
hetPi	18,7	6,38	2,88	1,6	1,3	1,06	0,91	0,8	0,78
PC	3,98	1,43							

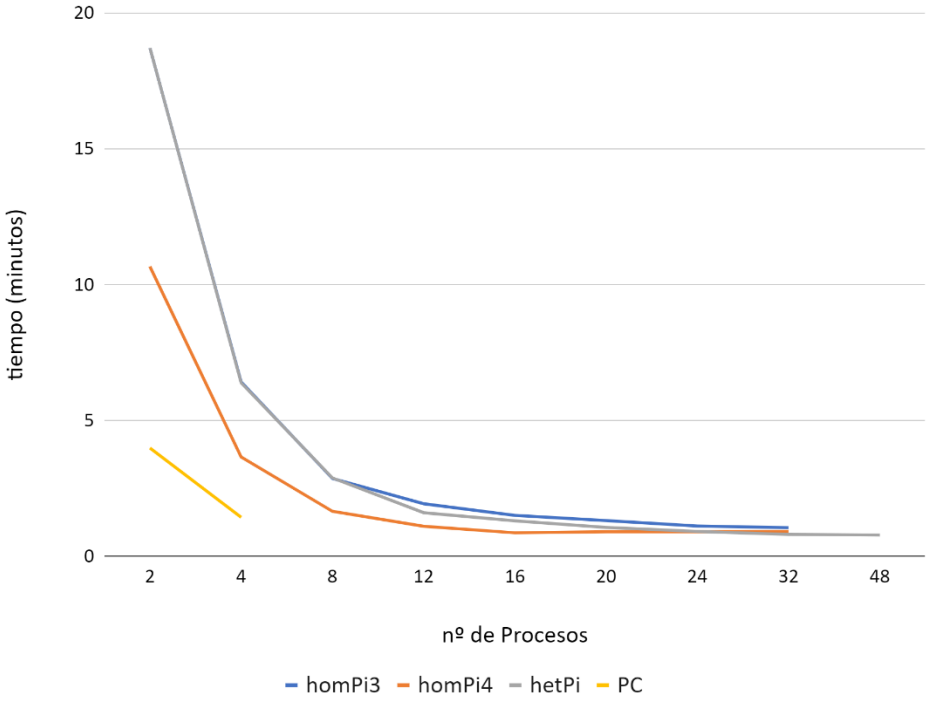

Figura 17 Filtrado dinámico de imágenes

Con esta prueba, por primera vez se obtiene un mejor rendimiento con el *cluster* heterogéneo que con las configuraciones homogéneas. De esta forma, se ha reducido el tiempo de ejecución en 18 minutos entre la ejecución con dos procesos y la ejecución con 32 procesos.

5.3.3 Comparativa ejecución estática-dinámica

Como se ha podido observar en estas dos ejecuciones la mejora al ir incrementando el número de procesos es notable, tanto en la ejecución estática como en la dinámica se igualan o se superan los tiempos del PC.

Si se comparan los resultados de las dos ejecuciones se puede ver como con menor número de procesos, el comportamiento es prácticamente el mismo tanto en la ejecución estática como en la dinámica y a medida que vamos aumentando procesos se van optimizando las ejecuciones vamos obteniendo mejores resultados en la ejecución dinámica.

A diferencia de la distribución estática, la dinámica, aprovecha los cores más rápidos de las diferentes placas, que, aunque en los *clusters* homogéneos la diferencia de tiempo no sea muy notoria, en el heterogéneo si lo es. Esto es debido a que la ejecución dinámica de 32 procesos reduce en un 20% el tiempo de ejecución en comparación con la distribución estática utilizando el mismo número de procesos.

Con esto llegamos a la conclusión de que una distribución adecuada del trabajo incrementa el aprovechamiento de los recursos, y a su vez mejora el rendimiento medio del sistema. Esto se puede observar especialmente en el caso de sistemas heterogéneos, ya que, en caso contrario, al no priorizar la utilización de las placas con mayor rendimiento el rendimiento se ve afectado negativamente.

5.4 NAS Parallel Benchmarks

Esta prueba puede dividirse en dos partes. En la primera se muestran los tiempos de ejecución del benchmark y en la segunda se presenta la potencia de las distintas configuraciones medida en MOPS. Este benchmark contiene distintas aplicaciones:

- IS: Ordenación de enteros y accesos aleatorios a memoria
- EP: Computación paralela
- CG: Gradiente conjugado, acceso a memoria irregular y comunicación
- MG: Multi-Grid en la secuencia de mallas
- FT: Transformación rápida de Fourier 3D

Para medir el tamaño de los problemas contamos con las siguientes clases de problema:

- Clase S: Problemas pequeños, para test rápidos
- Clase W: Problemas para una estación de trabajo de los 90
- Clases A,B,C: Tests estándar, el tamaño se incrementa en 4x de una a otra
- clases D,E,F: Test grandes, el tamaño aumenta en 16x de una clase a otra

Para llevar a cabo estas pruebas se ha utilizado la versión 3.4.1 del benchmark, utilizando la aplicación de computación paralela EP y la clase C, la más grande de los test de tamaño estándar. En este benchmark, también se ha realizado en la configuración heterogénea una ejecución con 48 hilos, para ver el incremento del benchmark usando toda la capacidad de cómputo disponible en el *cluster*.

5.4.1 Tiempo de ejecución y MOPS

nº de procesos	2	4	8	12	16	20	24	32	48
homPi3	11,6	10,3	4,3	3,58	2,54	2,15	1,82	1,4	
homPi4	5,61	2,85	1,68	0,964	0,71	1,16	0,94	0,71	
hetPi	10,93	5,4	2,7	1,82	1,73	1,1	0,9	0,76	0,53
PC	1,9	0,96							

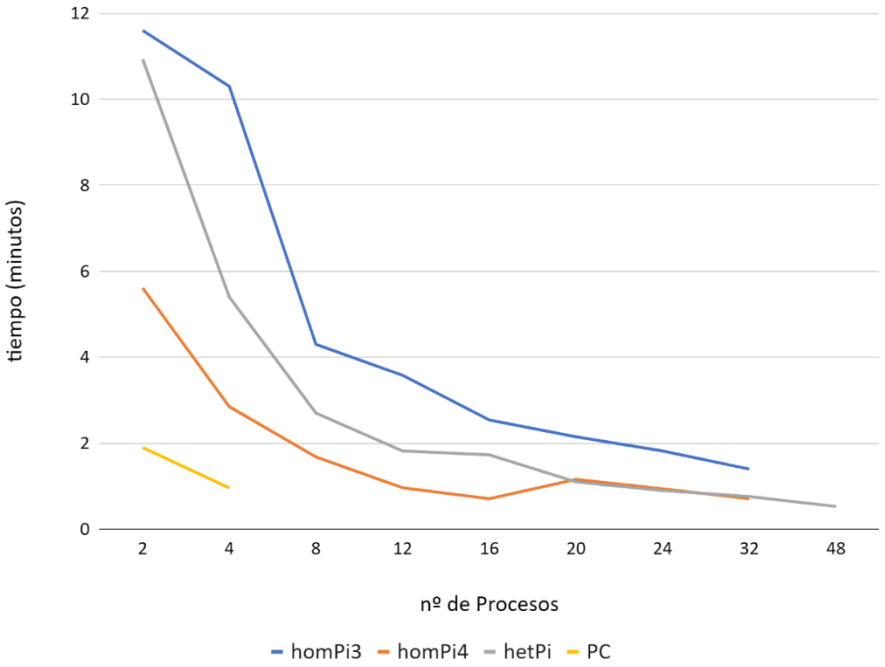

Figura 18 tiempo de ejecución del NAS Parallel benchmark

nº de procesos	2	4	8	12	16	20	24	32	48
homPi3	12,24	13,9	27,06	39,9	56,3	64,64	78,81	101,66	
homPi4	25,48	50,14	100,69	148,51	200,29	122,89	150,9	200,65	
hetPi	13,09	26,16	53,2	78,47	104	129	156	185,95	261
PC	75,24	147,44							

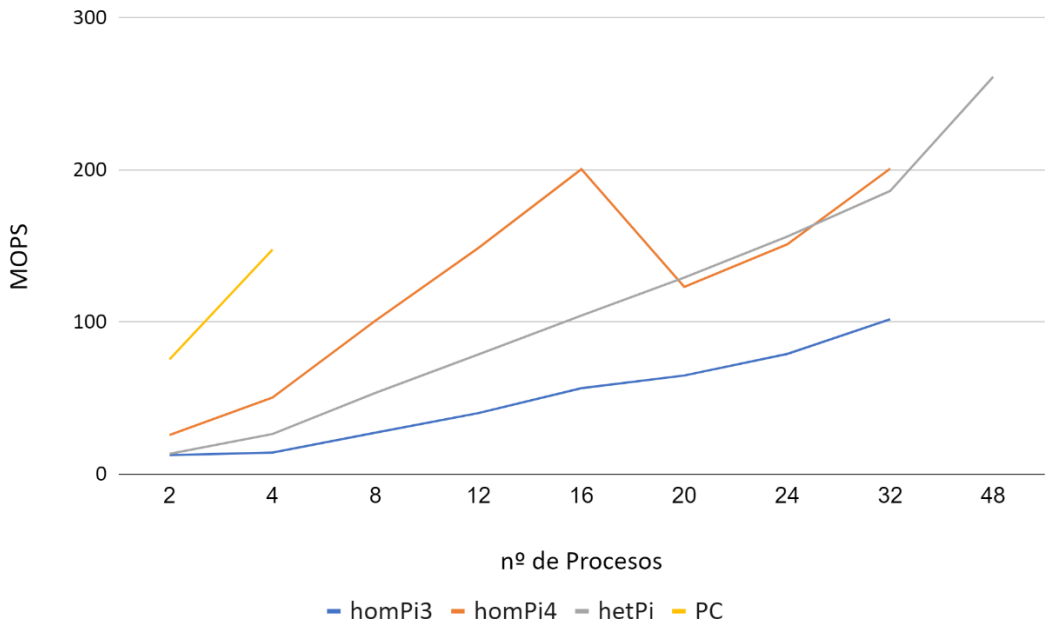

Figura 19 MOPS

Como puede se puede observar en las dos gráficas (Figura 18 y Figura 19), tanto en la de tiempo, como en la de MOPS, está muy diferenciado cada tipo de *cluster* y se puede observar de manera muy clara la diferencia de disminución de tiempo y del aumento de MOPS de las distintas configuraciones según cada tipo de placa.

Se puede observar que con 12 núcleos del *cluster* de Raspberry Pi 4, igualamos la potencia y los tiempos del PC. Este caso se repite con el *cluster* heterogéneo, que alcanza unas cifras similares con 24 procesos, sin embargo, el *cluster* formado por Raspberry Pi 3 obtiene un rendimiento menor si se compara a los cuatro núcleos del PC. En este caso, también se ha ejecutado un benchmark con 48 procesos para poder ver la potencia total del *cluster* heterogéneo. Con esto, se ha reducido el tiempo de ejecución del benchmark casi a la mitad y se han aumentado los MFLOPs en un 77% en comparación con cuatro núcleos de un PC convencional.

6. Conclusiones y Trabajo futuro

En esta sección se proporcionan las conclusiones de este proyecto y algunas líneas de trabajo futuro.

6.1 Conclusiones

Dada prácticamente por obsoleta la ley de Moore que predice la reducción de tamaños de chip, el próximo paso para seguir avanzando es la distribución de procesos. La paralelización de tareas está cada vez más presente en la actualidad, donde los procesadores con más número de cores van tomando un mayor protagonismo en el ámbito particular. Además, cabe destacar que, durante los últimos años, la irrupción de dispositivos de bajo presupuesto que cuentan con procesadores de hasta cuatro y ocho núcleos, han generado una tendencia que promueve una buena relación entre rendimiento y coste.

Este TFG es una muestra de cómo avanza la tecnología y como con un presupuesto bajo y en un espacio reducido. Hoy en día, la adquisición de un *cluster* de las características propuestas está al alcance de gran parte de la sociedad, sin embargo, hace algunos años solo podría haber estado en grandes empresas o universidades, y nunca ser obtenido por un particular, tanto por temas económicos como de espacio.

Tras la realización de este proyecto, se ha podido observar que no en todos los casos la sustitución de parte de un sistema distribuido con mejor hardware, en nuestro caso el *cluster* heterogéneo, va a generar mejoras en su rendimiento. Para que esto ocurra, es necesario un mayor aprovechamiento del hardware, y para ello hay que utilizar una distribución de carga dinámica, donde se dé mayor peso a los núcleos más rápidos de los procesadores disponibles.

Me ha parecido un TFG interesante de realizar ya que puede ayudar a diferentes estudiantes de la asignatura “Programación de sistemas distribuidos” de la universidad a ver sus prácticas MPI ejecutadas en un entorno real. De esta manera, la escalabilidad de las distintas aplicaciones que hagan y cómo varía su rendimiento en función de la configuración seleccionada en cada ejecución puede ser fácilmente analizada. Este hecho proporciona una mejor experiencia en el aprendizaje del funcionamiento de un sistema distribuido, ya que, en lugar de ver su comportamiento en simulaciones, puede verse en un *cluster* físico.

6.2 Trabajo futuro

Durante la realización de este proyecto han sido detectadas distintas situaciones, las cuales han generado distintas líneas de trabajo futuro.

Una de ellas está relacionada con la ejecución de las aplicaciones. Actualmente, para lanzar una ejecución es necesario realizar tareas manualmente, tales como el envío de archivos al front-end o la configuración de la infraestructura en la cual se puede ejecutar, todo esto sin una interfaz gráfica que permita llevar a cabo estas tareas de forma sencilla e intuitiva. Este proceso podría automatizarse por medio de una GUI con la cual se pueda acceder y manejar el *cluster* de manera visual, haciendo más sencilla la ejecución de pruebas en el mismo.

Además, se contempla ampliar el *cluster* para para observar las variaciones de rendimiento y compararlas con las ejecuciones mostradas en este documento. La idea es realizar experimentos creando nuevas configuraciones de la infraestructura, incluyendo nuevas versiones diferentes de placas Raspberry Pi, como las de bajo presupuesto Raspberry pi Zero o con las siguientes versiones que sean lanzadas en un futuro.

6. Conclusions and future work

This section provides the conclusions of this project and some lines of future work.

6.1 Conclusions

Given that Moore's law predicting chip size reduction is practically obsolete, the next step to move forward is distributed computing. The parallelization of tasks is increasingly present today, where the processors with more cores are taking a greater role in the particular field. In addition, it is worth noting that in recent years, the emergence of low-budget devices with processors of up to four and eight cores has generated a trend that promotes a good relationship between performance and cost.

This TFG is an example of how technology advances and how with a low budget and in a reduced space, nowadays, the acquisition of a cluster with the proposed features is within the reach of much people, however, some years ago it could only have been in large companies or universities, and never be obtained by a private individual, both for economic and space issues.

After the completion of this project, it has been observed that not in all cases the replacement of part of a distributed system with better hardware, in our case the heterogeneous cluster, will generate improvements in performance. For this to happen, it is necessary to make better use of the hardware, and to do so, it is necessary to use a dynamic load distribution, where more weight is given to the fastest cores of the available processors.

I found this an interesting TFG to do because it can help different students of the "Distributed Systems Programming" course of the university to see their MPI practices executed in a real environment. In this way, the scalability of the different applications how their performance varies depending on the configuration selected in each execution can be easily analyzed. This provides a better experience in learning how a distributed system works, since instead of seeing its behavior in simulations, it can be seen in a physical cluster.

6.2 Future work

During the realization of this project, different situations have been detected, which have generated different lines of future work.

One of them is related to the execution of the applications. Currently, to launch an execution it is necessary to perform tasks manually, such as sending files to the front-end or configuring the infrastructure in which it can run, all this without a graphical interface that allows to carry out these tasks in a simple and intuitive way. This process could be automated by means of a GUI with which the cluster can be accessed and managed visually, making it easier to execute tests on the cluster.

In addition, it is contemplated to expand the cluster to observe the performance variations and compare them with the executions shown in this document. The idea is to carry out experiments creating new configurations of the infrastructure, including new different versions of Raspberry Pi boards, such as the low-budget Raspberry pi Zero or with the next versions to be launched in the future.

Bibliografía

- https://www.raspberrypi.org
- https://www.raspberrypi.org/downloads/raspberry-pi-os
- https://www.mpich.org
- https://wiki.archlinux.org/index.php/NFS
- https://gist.github.com/kmkurn/39ca673bb37946055b38
- https://rc.byu.edu/wiki/index.php?page=Parallel+BZIP2+Compression
- https://www.nas.nasa.gov/publications/npb.html
- https://www.zdnet.com/article/oracle-this-1060-raspberry-pi-supercomputer-is-worlds-largest-pi-cluster/
- https://www.ibm.com/support/knowledgecenter/SSWRJV_10.1.0/lsf_admin/EAS_AFS_install_config_benchmark_programs.html
- Daniel Quiñones Sánchez y Miguel Romero Martínez - Diseño y despliegue de un *cluster* de bajo presupuesto para el desarrollo de las prácticas de Programación de Sistemas Distribuidos.