COLEGIO DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN INGENIERÍA EN SISTEMAS
ELECTRÓNICOS Y DE TELECOMUNICACIONES

Diseño e implementación de un clúster para realizar cómputo distribuido y paralelo utilizando software libre

TRABAJO RECEPCIONAL QUE

PARA OBTENER EL TÍTULO DE LICENCIADA EN

INGENIERÍA EN SISTEMAS ELECTRÓNICOS
Y DE TELECOMUNICACIONES

PRESENTA:

MARÍA DE LA PAZ CERVANTES EUGENIO

Director del trabajo recepcional
M. en C. Joel Yazbek Buendía Gómez

Codirector del trabajo recepcional
Dr. José Joaquín Lizardi del Angel

Ciudad de México, diciembre 2016.

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

Agradecimientos

A la Universidad Autónoma de la Ciudad de México, por ser la Universidad que sin prejuicios le abre sus puertas a todo aquel que disfruta del conocimiento, por brindarme la oportunidad de cursar una licenciatura, y sobre todo porque el estar en esta casa de estudios me ha permitido conocer a personas que me han enseñado a abrir mi mente al mundo.

A Joel Yazbek Buendía Gómez, estimado profesor y director de este trabajo, le agradezco por todo el apoyo que me ha brindado, por su paciencia, su entusiasmo y por elegirme para formar parte del proyecto del cual se deriva este trabajo.

A José Joaquín Lizardi del Angel, por su colaboración como codirector, por sus enseñanzas, por su confianza y por todo el apoyo que me ha otorgado.

A mis profesores y lectores, Magali Cortez Vázquez, José Ignacio Castillo Velázquez y Juan Carlos Aguilar Franco, por el tiempo que me han dedicado a lo largo de mi formación universitaria y por las observaciones que ayudaron a mejorar mi trabajo recepcional. A José Luis Quiroz Fabián, también lector de este trabajo, por sus comentarios y apoyo.

A la SECITI, por las facilidades otorgadas para realizar el proyecto *Laboratorio de sistemas distribuidos y de redes de alto desempeño*, y nuevamente a la UACM por el apoyo económico brindado para la impresión y empastado del trabajo recepcional.

Dedicatorias

A mis padres, Adolfo Cervantes De Gante y Silvia Eugenio Emeterio, por el apoyo que siempre me han dado para poder llegar a este momento de mi vida, porque este logro es también de ustedes, gracias por ser mis pilares, para ustedes no tengo más que amor, agradecimiento y respeto.

A mis hermanos, por su cariño. José Armando, por ser un apoyo más en casa, por la confianza que me has tenido, por compartir tus experiencias y por nuestros proyectos gastronómicos. Arturo, por tu forma tan característica de darme aliento para terminar este trabajo y por contarme tus aventuras.

A mi madrina, Gregoria De Gante, porque siempre has estado al pendiente de nuestra familia, por todo tu apoyo y amor.

A mi abuelo Joel De Gante -Coronel- y a mis primos, Ana Mallely Espinoza y José Roberto Marín, porque jamás los olvido, ustedes marcaron mi existencia con su presencia y su cariño.

A Areli Nájera, por ser una verdadera amiga, gracias por todo el apoyo y cariño que me has brindado, sobretodo por estar siempre conmigo en los momentos más difíciles. A Jorge Mijangos por ser un gran amigo y compañero de clase, por los desayunos que compartias y

por los momentos de crítica. A Fernando Octavo, que me acompañaste en la recta final de la carrera, gracias por tu apoyo, y por tus palabras de aliento. A Isabel Saucedo, que has sido mi amiga desde la adolescencia, por nuestros comienzos en la Universidad y por tu apoyo en esta parte final de la carrera.

A mis amigos del Museo de la Luz, que llegaron a mi vida para llenarla de risas, de buenos momentos, de mucho cariño y sobre todo de fraternidad, además de contribuirme con las historias y los conocimientos que cada uno tiene en su área. Especialmente quiero mencionar a Mayra Torres por ser siempre tan linda, cariñosa y protectora, a Miguel Jiménez por tu buena vibra y tu actitud siempre positiva, a Manuel Gutiérrez por ese buen humor, y las "quihuboles"que te dejaste infringir por mí, a Flaherthy Cota por adoptarme y ser mi padrino, por todas las historias que me contaste y por estar siempre al pendiente del Guapo, a Isabel Vázquez por detectar siempre las buenas ofertas :) por la confianza que me has tenido y por ser la fabulosa amiga que eres, finalmente a Nuria Fuentes, por ser bebé, porque siempre estuviste al pendiente de mí, por escucharme y sobretodo por tu amistad.

A Jorge Fabián Hernández Cervantes, mi historiador y filósofo de cabecera, por ser mi compañero de vida estos años, porque a pesar de todo ha logrado prevalecer lo bueno, por estar siempre involucrado en mi formación académica, por apoyarme y alentarme a ser una buena ingeniera que no solo pelea con números... por Chiapas y Tuxpan, por el fútbol, por todo el conocimiento que me has compartido y las experiencias que hemos vivido, porque siempre estuviste conmigo, por tu cariño y ternura.

Resumen

En este trabajo, se presenta el diseño e implementación de un clúster para realizar cómputo paralelo por medio de software libre. El clúster se implementó en el laboratorio B-404 del plantel San Lorenzo Tezonco de la Universidad Autónoma de la Cuidad de México y se denominó Xexelo0.

Para la construcción de Xexelo0 se adquirieron cuatro servidores de la marca SUPERMICRO con procesadores Intel Xeon, y se siguió la arquitectura de un clúster tipo Beowulf con tecnología Ethernet a 10 Gbps, el sistema operativo utilizado es CentOS, el middleware está formado con Mpich y algunas bibliotecas de Intel que son especializadas para realizar procesamiento en paralelo.

Para comprobar el funcionamiento del clúster, se instaló, configuró y ejecutó el modelo de predicción climática WRF que está diseñado para ejetutarse de modo paralelo, además se realizó el test linpack con el objetivo de conocer el rendimiento en Gflops de Xexelo0.
Los resultados del modelo WRF y del test linpack son la prueba de que se logró implementar exitosamente un clúster tipo Beowulf, que es funcional y capaz de resolver problemas complejos (como lo es la predicción climática) en pocos segundos a través de la paralelización.

Contenido

Lista de figuras

Lista de tablas

Capítulo 1

Introducción

El uso de clústeres representa una herramienta confiable para la solución eficaz de problemas complejos en diversas áreas de la investigación y desarrollo de la ciencia y la tecnología, tanto en la industria como en el ámbito académico. El uso de los clústeres se remonta a la década de los noventa del siglo XX, su aceptación se debe a que ofrecen poder de cómputo que se vuelve cada vez más potente, (ya que en el primer listado del TOP500, realizado en junio de 1993 el clúster del primer lugar tenía un rendimiento de 59.7 Gflops, mientras que el primer lugar del TOP500, realizado en junio de 2016, presenta un rendimiento de 93 Pflops) el ancho de banda disponible aumenta, además, la escalabilidad y el tiempo de resolución de un problema disminuye.

La programación distribuida hace posible el funcionamiento de los clústeres. Dentro de este tipo de programación se encuentran el cómputo paralelo y el cómputo distribuido, en ambos se llevan a cabo diversos procesos a la par con el uso de varios procesadores interconectados, pero en el cómputo paralelo se encuentran normalmente sistemas homogéneos que se componen de equipos que comparten características como la capacidad de almacenamiento y la velocidad de transferencia entre otras, y se realizan tareas que son parte de un sólo objetivo,

mientras que en el cómputo distribuido los sistemas son heterogéneos y se pueden ejecutar tareas con objetivos distintos.

Los clústeres trabajan con problemas paralelizados, lo cual significa que un problema es dividido en partes llamadas tareas, de las cuales se derivan dos conceptos importantes, proceso objeto e hilo. El proceso objeto es una tarea independiente que opera sobre su propio espacio de memoria. Un hilo es una tarea que comparte el mismo código de programa y el mismo espacio de memoria con otros hilos [1]. De lo anterior se deduce que un problema se divide en tareas llamadas procesos y que éstos a su vez contienen uno o más hilos de ejecución.

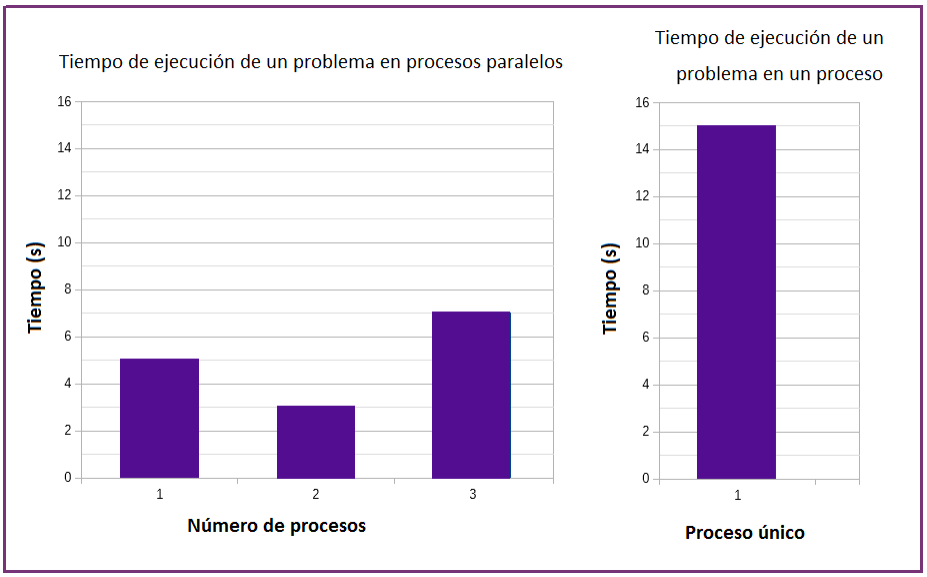

Figura 1.1: Tiempos de ejecución de un problema en paralelo y en serie.

Al dividir un problema se impacta directamente en el tiempo de ejecución, esto se puede ilustrar con el ejemplo hipotético de la figura 1.1, en la que se muestran los tiempos de ejecución de un problema en paralelo y también se muestra el tiempo de ejecución del mismo problema

en un sólo proceso, es decir de modo serial. Se puede notar que el tiempo de ejecución con un proceso tarda 15 segundos, mientras que la ejecución en paralelo ofrece tres tiempos, uno por cada proceso, que son de 5, 3 y 7 segundos, de los cuales se toma como referencia el tiempo mayor, ya que en ese tiempo, los otros dos procesos paralelos ya finalizaron. Por tanto, aunque el tiempo representativo de una ejecución en paralelo es el mayor de los obtenidos, se muestra que es menor al tiempo de la ejecución de un problema en serie.

Una situación que ilustra el uso de un clúster se presentó en la astronomía. En 1986 en el número 92 del The Astronomical Journal se publicó el artículo The outer solar system for 200 million years, resultado de la implementación de un sistema llamado Planetario digital utilizado para calcular la estabilidad de las orbitas de los planetas de nuestro sistema solar, incluido Plutón, ya que en ese año aún se consideraba parte del sistema solar, este Planetario digital estaba compuesto por un anillo de diez computadoras, una de ellas fungió como procesador central, mientras que las otras nueve representaron a los planetas, estas computadoras fueron llamadas Procesadores planetarios y cada una de ellas contenía la información de cada planeta.

La tarea del Planetario digital fue calcular las trayectorias de los cinco planetas más lejanos del Sol en un intervalo de 112 millones de años hacia el pasado y hacia el futuro, los resultados arrojaron que en ese tiempo habría estabilidad en las órbitas, es decir, que no habrá intersección entre ellas. Este resultado se obtuvo en seis días gracias a que la velocidad de cálculo del Planetario digital fue de 10 Mflops [1]. Los responsables de este artículo aseguraron que si hubieran utilizado una computadora DECVAX 11/780 (esta computadora fue presentada en 1977 [2]) en lugar del planetario digital el cálculo habría tardado un año [3].

[1] 10 Mflops equivalen a 10 millones de operaciones aritméticas de punto flotante por segundo.

1.1. Definición y delimitación del problema

Actualmente se ha demostrado que para realizar investigaciones de frontera se necesita de sistemas que resuelvan, por ejemplo, simulaciones que las computadoras comunes no pueden realizar. Debido a esto, el sector académico en el área de tecnología se ha dado a la tarea de implementar sistemas con gran poder de cómputo. Por ello, en este proyecto se propone implementar un clúster, en el cual se busca que el procesamiento de problemas computacionales se ejecute de manera paralela y distribuida.

Aunque la programación en paralelo es un factor importante en los sistemas distribuidos, no se abordará a profundidad, pese a ello, este trabajo finalizará en la evaluación de un programa con aplicación científica que trabaja en paralelo.

1.2. Contribuciones

Con el diseño e implementación de un clúster denominado Xexelo0, se pretende proporcionar a la Universidad Autónoma de la Ciudad de México de una red de servidores, en la cual se pueda realizar el procesamiento y la solución de problemas complejos a través de la paralelización de éstos. El clúster Xexelo0 proporcionará a las diversas áreas de investigación de la Universidad un sistema con una capacidad de cómputo mayor a la que utilizan actualmente, lo que contribuirá al desarrollo de la investigación científica y tecnológica. Para comprobar que el sistema implementado es capaz de resolver problemas complejos de modo paralelo, se propone ejecutar el modelo WRF que está diseñado para realizar el cálculo de variables climáticas (por ejemplo, la temperatura ambiente) de manera paralela.

1.3. Objetivos

El objetivo de este proyecto es diseñar, instalar y medir el rendimiento de un clúster de alto desempeño que opera con software libre, el cual será implementado con cuatro servidores de alto desempeño que tienen una capacidad de transferencia máxima de 10 Gbps, además se pretende comprobar que en el clúster Xexelo0 se pueden ejecutar aplicaciones científicas en paralelo. Esta implementación, que será la base del laboratorio de sistemas distribuidos en la UACM, se realizará a partir de la infraestructura con que cuenta actualmente el laboratorio B-404 y la que se pueda integrar con el presupuesto asignado por la SECITI[2].

1.3.1. Objetivos particulares:

1. Evaluar las prestaciones de los switches y servidores que formaran parte del clúster.

2. Diseñar una red de datos apropiada para Xexelo0 dentro del laboratorio B-404.

3. Instalar un sistema operativo adecuado para el funcionamiento del clúster.

4. Instalar y configurar las bibliotecas necesarias para ejecutar el modelo WRF.

5. Medir el rendimiento del clúster Xexelo0 con base a la prueba de Linpack.

6. Analizar el tiempo de procesamiento al ejecutar el modelo WRF, con base a diferente número de tareas en cada ejecución.

[2]La Secretaria de Ciencia, Tecnología e Innovación, financió parte del proyecto denominado *"Laboratorio de sistemas distribuidos y redes de alto desempeño", en la UACM.*
Convenio: UACM/SECITI 060/2013.

1.4. Estructura del documento

En el capítulo 2, se muestra un panorama general de lo qué es un clúster y de los elementos que lo componen, así como las principales características de los clústeres de las universidades más reconocidas de la Ciudad de México y del TOP500. Al final del capítulo se mencionan algunas características del modelo de predicción climática WRF. En el capítulo 3, se realiza la valoración de la infraestructura para la implementación del clúster Xexelo0, así como su diseño y la especificación de los pasos a seguir para su funcionamiento y la ejecución del modelo WRF. En el capítulo 4, se muestra la evaluación y análisis sobre el desempeño con base a la prueba Linpack y los resultados obtenidos de la configuración propuesta para la ejecución de WRF en Xexelo0. Finalmente, en el capítulo 5 se presentan las conclusiones y se deja el planteamiento de un escenario funcional para el posible trabajo futuro.

Capítulo 2

Marco teórico

En este capítulo se realiza una descripción de los elementos teóricos que resultan importantes para una mejor comprensión acerca de las características e importancia que tiene el uso de los clústeres. Se hace mención del TOP500 y de las características de algunos de estos clústeres incluyendo los clústeres de la UNAM, la UAM y el CINVESTAV. Finalmente, se describe de modo breve en qué consiste el modelo de predicción climática WRF.

2.1. Redes de computadoras

Las redes de computadoras son conjuntos de computadoras interconectadas entre sí de modo que existe un intercambio de datos entre los elementos que conforman dicha red. Además de computadoras, se consideran componentes de una red equipos como servidores, switches, routers, impresoras, equipos móviles, etc.

Fue en 1964 cuando en ARPANET (Advanced Research Projects Agency Network) se conjuntó la comunicación de mensajes con la comunicación de paquetes, lo que permitió que un par de años después se iniciara el proyecto de comunicar por medio de redes de datos a las universidades de Estados Unidos. En diciembre de 1969 ya existía una red que conectaba a

cuatro universidades estadounidenses, estas son: UCLA (University of Carolina-Los Angeles), UCSB (University of Carolina-Santa Barbara), SRI (Stanford Research Institute) y la Universidad de Utah, esta red de datos fue la primera red de área extendida [4].

Las redes se pueden clasificar a partir de su extensión geográfica, de menor a mayor extensión estas redes son redes de área local (LAN), redes de área metropolitana (MAN) y redes de área extendida (WAN). Además de estas redes, están las redes de área personal (PAN) que interconectan dispositivos móviles cercanos al usuario.

Los elementos de hardware que interconectan las redes son las NICs, switches y routers. Las NICs son tarjetas de interfaz de red, que tienen la función de habilitar a las computadoras para que se puedan comunicar con otras ya que proporcionan el acceso a la red en la que se conectan, además estas tarjetas adaptan la velocidad de transmisión. Los switches o conmutadores establecen la comunicación entre los elementos de una red a través de una conmutación inteligente ya que tienen la capacidad de segmentar una red a nivel lógico para administrar el ancho de banda, para lo cual se auxilian de las direcciones físicas de las tarjetas de red. Los routers o encaminadores interconectan a dos o más redes utilizando direcciones de red, estos dispositivos tienen la capacidad de elegir las mejores rutas para el envío de información entre redes.

2.2. Sistemas distribuidos

Los sistemas en general se integran por diversos componentes que tienen una relación entre sí, y que trabajan en conjunto para llevar a cabo un objetivo. Hay sistemas de todo tipo, y en todas las áreas del conocimiento. Algunos sistemas en el área de las telecomunicaciones

son los sistemas de radio, de televisión, sistemas inalámbricos, telefónicos y los sistemas distribuidos de transmisión de datos, entre otros [5].

En la actualidad, se llevan a cabo ciertos procesos de información en diversas áreas del conocimiento como la economía, medicina, informática, astronomía, entre otras, que requieren de equipo que les permita llevar a cabo tareas de forma rápida y eficiente, que de manera general no pueden ser resueltas por computadoras comunes ya que no presentan las características necesarias para resolver tareas complejas como son: el procesamiento de grandes volúmenes de datos, y compartir recursos ya sea de software o de hardware. Para poder realizar las tareas referidas se hace uso de lo que se conoce como sistema distribuido.

La implementación de diversos sistemas distribuidos se puede utilizar para el desarrollo del cómputo de alto desempeño o HPC (High performance Computing) utilizado para llevar a cabo simulaciones de problemas complejos ofreciendo respuestas de manera rápida, además de impulsar resultados y descubrimientos, lo que deriva en el desarrollo e innovación de conocimiento y productos [6].

Uno de los sistemas distribuidos más popular es el Internet, de lo cual se puede inferir que un sistema distribuido es equivalente a una red de equipos de cómputo, sin embargo Colouris (2001) ofrece la siguiente definición:
"Un sistema distribuido es aquel en que los componentes hardware o software localizados en computadores unidos mediante una red comunican y coordinan sus acciones mediante el paso de mensajes."
Pese a que es una definición muy completa, Tanenbaum (2008) es más claro al decir que:

"Un sistema distribuido es una colección de computadoras independientes que dan al usuario la impresión de constituir un único sistema coherente."

De modo general, los sistemas distribuidos se forman con base al software, donde la base de cada elemento es su sistema operativo, la parte superior son las aplicaciones que pueden estar en uno o más elementos, y en medio el middleware que hace posible el funcionamiento de las aplicaciones paralelas. En la figura 2.1 se muestra la organización mencionada.

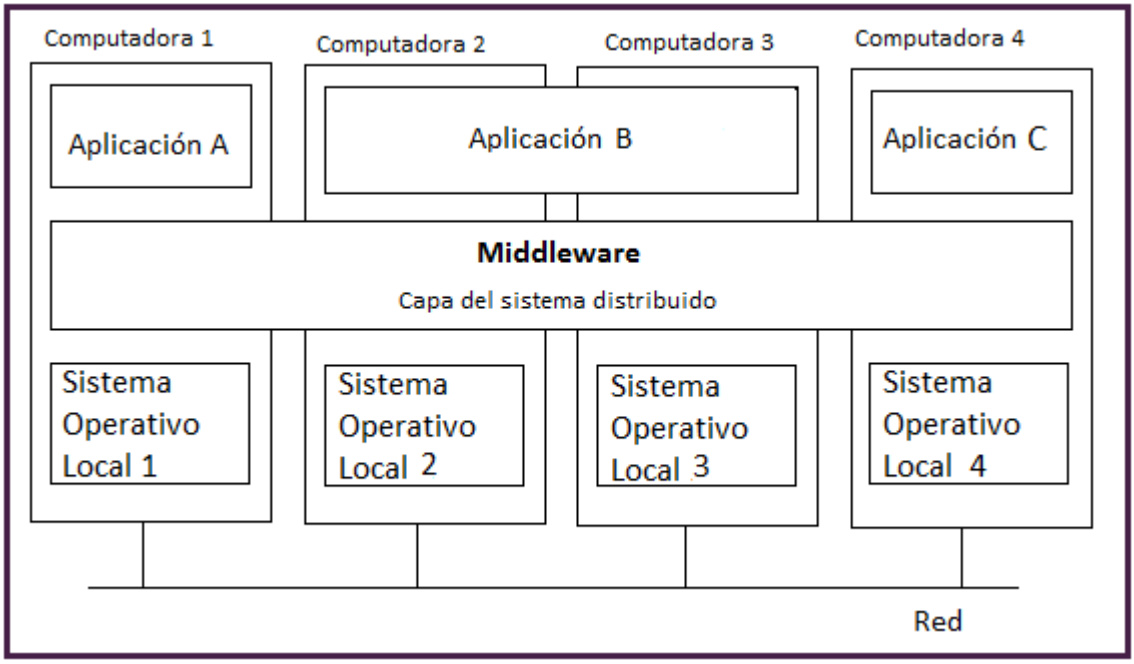

Figura 2.1: Sistema distribuido con base al software [6].

El tipo de recursos que se comparten dentro de un sistema distribuido son por ejemplo, impresoras, bases de datos, páginas web, entre otros. Además se comparten servicios, por ejemplo, los servicios bancarios que se encuentran alojados en diferentes servidores a los cuales se tiene acceso mediante algún equipo que se encuentre conectado a la misma red. En

este contexto el termino servidor se entiende como un programa especial en una computadora dentro de una red, ya que responde a las peticiones o solicitudes de programas que se ejecutan en otras computadoras de la red, a estas computadoras que realizan peticiones o solicitudes se les conoce como clientes. Las características que se describen a continuación, son parte de los sistemas distribuidos: [5].

- La *heterogeneidad* va ligada con la variedad entre los elementos que conforman la red, es decir, las divergencias entre el software y el hardware que es utilizado.

- La *extensibilidad* se refiere a la capacidad que tiene el sistema de poder extenderse tanto en equipo como en servicios, y reconfigurarse para que continúe trabajando de manera eficaz.

- La *seguridad* se refiere a mantener en el sistema y garantizar al usuario las siguientes características: confidencialidad, integridad y disponibilidad, es decir, no se debe permitir el acceso a información por usuarios no autorizados, protección para que no se altere la información y protección para evitar interferencia con los procesos de acceso, respectivamente. En términos de seguridad también se debe incluir la protección contra ataques de denegación del servicio y seguridad en código móvil.

- La *escalabilidad* tiene que ver con la adición en recursos, servicios y usuarios a una red, de manera que ésta conserve su efectividad.

- El *tratamiento de fallos* es una tarea que contempla la detección de fallos, tolerancia, recuperación y redundancia, considerando que cuando algún elemento de la red falla no significa que falle el sistema en su totalidad.

- La *concurrencia* es la ejecución de un proceso en más de un hilo de ejecución, de manera

que se atiendan múltiples peticiones de un mismo proceso, y esto debe sincronizarse de modo que los datos sean consistentes.

- La *transparencia* es la ocultación al usuario de que está en un sistema distribuido que se compone de diversos elementos, es decir, que el usuario perciba al sistema como un todo; los tipos de transparencia al usuario son: de acceso, ubicación, concurrencia, replicación, fallos, movilidad y escalado.

Como ya se hizo mención, Internet es un buen ejemplo de lo que es un sistema distribuido, el cual está compuesto por diversos tipos de computadoras y servidores que realizan procesos de comunicación mediante el empleo de protocolos y que además permite el acceso a sus usuarios desde casi cualquier lugar del mundo. Además del Internet, las intranets y el cómputo móvil forman parte de los sistemas distribuidos, dentro de éstos se pueden distinguir dos clases, los grids y los clústeres. Los grids se componen por conjuntos de sistemas heterogéneos tanto en hardware como en software, incluso pueden estar en distintos dominios de administración y geográficamente distantes. Los clústeres son conjuntos de equipos homogéneos ya que los nodos que los componen son similares (software y hardware), además estos nodos se conectan a través de una LAN de alta velocidad [6].

2.3. Clúster

Un clúster se define como "un grupo de computadoras interconectadas que trabajan conjuntamente en la solución de un problema. Estos sistemas constituyen una solución flexible, de bajo costo y gran escalabilidad para aplicaciones que requieren una elevada capacidad de cómputo y memoria" [7].

La construcción del primer clúster se remonta a 1994, en dicho momento fueron Thomas

Sterling y Don Becker científicos del CESDIS (Center of Excellence in Space Data and Information Sciences), quienes construyeron una super computadora a partir de 16 computadoras que contenían un procesador intel 486DX4. En dicho clúster se alcanzó una velocidad de 10 Mbps y el sistema operativo utilizado fue linux [8].

A cada elemento de un clúster se le denomina *nodo*, que se conectan a través de una red de área local. Se tienen dos tipos de nodos, el nodo maestro o *FrontEnd* y el esclavo. El primer tipo de nodo administra, todos los recursos y aplicaciones, mientras que el segundo tipo de nodo se dedica al procesamiento de operaciones.

Los clústeres trabajan bajo el paradigma del cómputo paralelo, el cual consiste en dividir un problema complejo en problemas más sencillos que se resuelven de manera independiente, es decir, para resolver un problema no se necesita de la solución del problema anterior. Algunas de las tareas que se resuelven con el uso de un clúster son: modelación de mercados financieros, predicción del clima, simulaciones petroleras y de biotecnología, animaciones y efectos para la producción de películas, entre otros.

Con todo lo anterior, se puede decir que un clúster *es un conjunto de computadoras independientes e interconectadas en el cual se trabaja para lograr un objetivo particular, distribuyendo las tareas entre los elementos que lo componen.*

2.3.1. Elementos de un clúster

Para construir un clúster es necesario contar con elementos de software y de hardware. Dichos elementos son: los nodos, el sistema operativo, el middleware, las conexiones de red, los proto-

colos de comunicación y la programación paralela. En seguida se describen las características de los nodos y de los sistemas operativos, los demás elementos se desarrollan más adelante y en la figura 2.2 se muestra un ejemplo de clúster, el cual está compuesto por cuatro nodos.

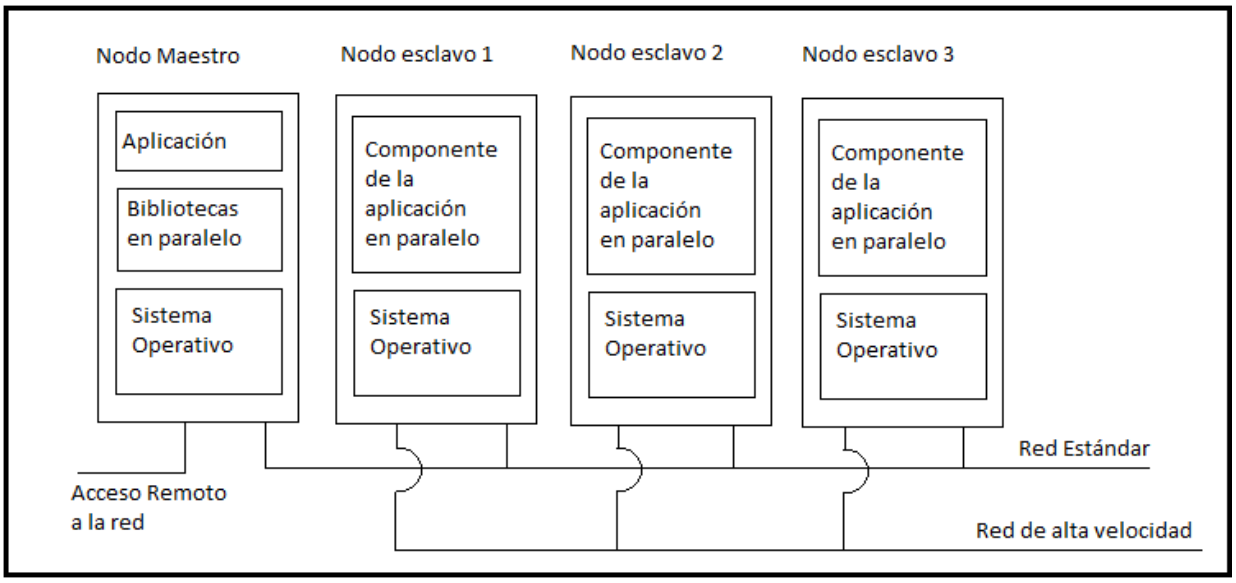

Figura 2.2: Ejemplo de clúster [6].

Nodos

Los nodos se pueden implementar desde computadoras personales hasta servidores. Hay dos tipos de nodos, los dedicados y los no dedicados, los primeros no tienen elementos de entrada-salida ya que solo se dedican al procesamiento de tareas, mientras que los nodos no dedicados sí contienen elementos de entrada-salida ya que además de realizar el procesamiento de tareas, estos nodos administran, controlan y monitorean el sistema, los primeros son conocidos como nodos esclavos y los segundos como nodos maestros o FrontEnd.

Sistemas operativos

Un sistema operativo es considerado como el conjunto de programas o software que hace posible el funcionamiento de una computadora. Estos sistemas se encargan de gestionar recursos (por ejemplo procesadores y memorias) entre los diferentes programas o procesos que se ejecutan en la computadora, además de realizar esta gestión se encarga de ofrecer al usuario de la computadora un entorno virtual en el que pueda desarrollar su trabajo sin necesidad de acceder al hardware de manera directa [9].

Existe diversidad de sistemas operativos, pero de ellos destacan los sistemas que se han desarrollado a partir de UNIX. Este sistema operativo fue desarrollado por los laboratorios Bell de AT&T en la década de los setenta se vendió a las instituciones educativas a un bajo costo [4], por lo que las universidades fueron las principales beneficiarias de este sistema, ya que pudieron tener acceso a un sistema operativo eficaz y de bajo costo, posteriormente se creó el sistema UNIX BSD (Berkeley Standard Distribution) desarrollado en la Universidad de California en Berkeley.

De UNIX se desprendió un sistema operativo llamado Minix, el cual fue una versión reducida del núcleo de UNIX, desarrollada con fines académicos por Andrew Tanenbaum. El sistema Minix a su vez fue la base del sistema operativo Linux creado en 1991 por Linus Torvalds quien se dio a la tarea desarrollar en lenguaje C una versión mejorada de Minix. En octubre de ese año Torvalds liberó la segunda versión de Linux (pero la primera pública) en la cual personas interesadas en este sistema operativo pudieron hacer a su vez correcciones y mejoras. Gracias al desarrollo en conjunto de Linux, en 1994 se logró generar la primera versión completa y sin errores [10].

Red Hat es una organización que se fundó en 1993 por un grupo de líderes de las tecnologías de la información -TI-, promotores y desarrolladores de código abierto cuyo objetivo es crear tecnología híbrida y abierta para brindar a las empresas con las que colabora, a través de código abierto, el control de las tendencias actuales y futuras en su área de desarrollo [11]. En 2004 Red Hat liberó el código abierto de CentOS para crear un sistema operativo de tipo empresarial gratuito, además con libertad de distribución, y que es desarrollado por una pequeña base de desarrolladores principales que son apoyados por la propia comunidad de usuarios de CentOS y voluntarios. Uno de los objetivos inmediatos de CentOS es establecerse como una plataforma líder para la comunidad de las nuevas tecnologías de código abierto procedentes de otros proyectos, por lo que se ha trabajado para que CentOS sea una plataforma estable, predecible, manejable y reproducible, lo que lo convierte en una herramienta fiable para el desarrollo de diversos tipos de proyectos [12].

Para implementar un sistema distribuido se puede hacer uso de diferentes tipos de sistemas operativos de distribución libre que lo hagan funcionar, por ejemplo, existen sistemas operativos como Pellican HPC que funcionan en vivo, es decir, que funcionan a partir de un CD que contenga a dicho sistema, el disco se inserta en la pc, y es suficiente con iniciar el equipo desde la unidad de CD en cada nodo para comenzar a trabajar en forma distribuida sin tener que realizar una instalación del sistema distribuido en el disco duro.
Existen otros sistemas que si necesitan instalarse en el disco duro, por ejemplo, los sistemas que trabajan a partir del sistema operativo CentOS, uno de estos sistemas es Rocks, que para ser instalado necesita como base a CentOS y durante su instalación se especifican claramente las herramientas con las que contará el sistema. Como se puede notar, a diferencia de Pellican HPC, Rocks no es un sistema en vivo ya que éste se tiene que instalar en cada uno de los

nodos del clúster. El software libre, como lo indica su nombre y como ya se ha explicado, es libre de ser utilizado por cualquier persona u organización, contiene un código abierto para poder ser modificado y mejorado, además de estar disponible en Internet. Estas son buenas razones para utilizar software libre en el desarrollo de proyectos, incluyendo sistemas distribuidos y particularmente los clústeres.

2.3.2. Interfaz de paso de mensajes (MPI)

Además del sistema operativo, hay otras herramientas útiles en los clústeres, como las bibliotecas de paso de mensajes, estas bibliotecas fueron necesarias para optimizar el rendimiento de los sistemas distribuidos en los que el hardware era heterogéneo.

Formalmente: La MPI asume que la comunicación ocurre dentro de un grupo conocido de procesos. A cada grupo se le asigna un identificador. Dentro de un grupo, a cada proceso también se le asigna un identificador local. Por lo tanto, un par (IDgrupo, IDproceso) únicamente identifica la fuente o el destino de un mensaje, y se utiliza en lugar de una dirección al nivel de transporte [5].

La MPI, está conformada por un conjunto de primitivas orientadas a mensajes que hacen posible crear aplicaciones eficientes. En la tabla 2.1 se muestran algunas de las primitivas de MPI.

Primitiva	Función
MPI_send	Envía mensaje y espera hasta que es copiado en un búfer
MPI_ssend	Envía mensaje hasta que comienza la recepción
MPI_recv	Recibe un mensaje y bloquea si no hay otro
MPI_irecv	Verifica si hay mensaje de entrada sin bloquearla

Tabla 2.1: Primitivas de MPI.

Una de las distribuciones más populares de Interfaz de paso de mensajes es MPICH, la cual se empezó a construir desde 1992 con base al sistema portable *Chameleon*, al cual debe su nombre. MPICH se distribuye de manera libre y trabaja en plataformas como Linux, Mac y Windows [13].

2.3.3. Organización lógica en capas

Los modelos de referencia fueron creados con el fin de generar la compatibilidad necesaria para lograr la comunicación entre dispositivos de diferentes fabricantes, estos modelos son conocidos también como arquitecturas de red.

Los modelos de referencia son formados por diferentes niveles o capas que tienen propósitos específicos. Estos modelos están compuestos por protocolos e interfaces, los protocolos determinan la comunicación entre entidades (por entidad se entiende a los elementos de una capa que se comunican con otros elementos del mismo nivel) de una misma capa en diferentes equipos, las interfaces determinan el intercambio de información entre entidades de diferentes capas de un dispositivo.

Dada la organización jerárquica de las capas en los modelos de referencia, los procesos de comunicación en las redes se dividen, por tanto, los procesos de comunicación son menos complejos, lo que permite que la tecnología sea interoperable, es decir, que las mejoras y avances que se realizan en cada capa no afectan a las otras capas. Hay varios modelos de referencia, los más populares son los modelos ISO/OSI y TCP/IP, ya que sirven como base para otros modelos [4]. En la figura 2.3 se muestra el esquema del modelo TCP/IP adaptado a los sistemas distribuidos y en seguida se describen las características generales de las capas que lo componen.

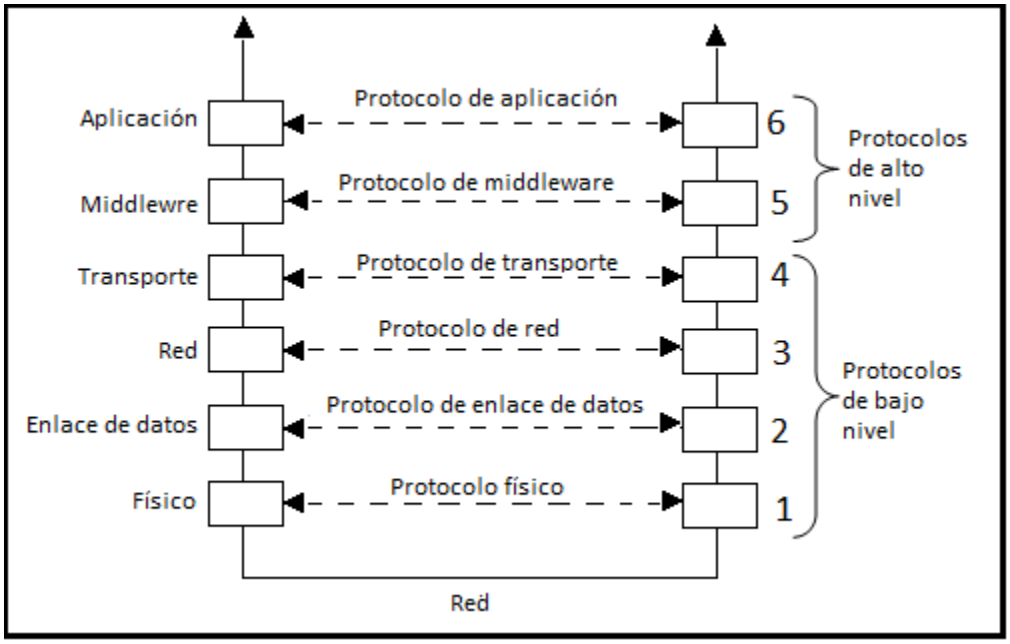

Figura 2.3: Organización en capas de un sistema distribuido [6].

- **Capa física**

  En orden ascendente la capa física es la primera, los protocolos de esta capa describen las especificaciones eléctricas, mecánicas, funcionales y procedimentales para la transmisión y recepción de secuencias de bits a través de un medio físico. Entre estas especificaciones se encuentran por ejemplo, niveles de voltaje, distancias de transmisión, tipos de conectores, velocidades y técnicas de transmisión.

- **Capa de enlace de datos**

  En esta capa se especifica cómo se da formato a los datos para que puedan ser transmitidos de un dispositivo a otro por medio de la capa anterior, es decir, controla el acceso al medio físico, y esto lo hace utilizando la dirección MAC. En esta capa se realizan acciones como la creación, detección, delimitación y corrección de errores en las tramas.

- **Capa de red**

  Es la capa encargada del encaminamiento de las tramas armadas en la capa de enlace de datos, es decir, controla la ruta de los datos desde el origen hasta el destino y este encaminamiento lo realiza por medio de un direccionamiento lógico.

- **Capa de transporte**

  En esta capa garantiza que la entrega de los mensajes se realiza sin errores. Algunas acciones que se efectúan son: segmentación de información y transmisión de los segmentos a la capa de red en el origen y ensamble de información en el destino, se confirma tanto en el origen como en el destino que la información se ha entregado correctamente, en caso de detectar errores esta capa los corrige antes de retransmitir la información.

- **Capa de servicios de middleware**

  En esta capa se encuentran protocolos que son similares a los de la capa de aplicación, la diferencia es que los protocolos de middleware tienen la finalidad de ejecutar aplicaciones distribuidas. Dos tipos de protocolos de esta capa son los de confirmación y de comunicación middleware, los primeros establecen que, en un grupo de procesos, o todos los procesos se realizan en una operación en particular o la operación no se realiza en absoluto [6], y en los protocolos de comunicación se pueden invocar elementos que se encuentran en equipos remotos.

- **Capa de aplicación**

  En esta capa se permite que los usuarios accedan a los servicios de internet, mismos que se apoyan en los protocolos de la capa de transporte. Algunos servicios de la capa de aplicación son los comandos y NFS.

2.3.4. Computadoras paralelas

Para comprender el funcionamiento de los sistemas distribuidos, incluidos los clústeres, es importante hacer mención de la *Taxonomía de Flynn* que hace una clasificación de computadoras paralelas que se basa en simples o múltiples instrucciones y datos, esta taxonomía es importante ya que la MPI está basada en computadoras MIMD -múltiples instrucciones, múltiples datos [14]. En la tabla 2.2 se muestra un esquema de la Taxonomía de Flynn y en seguida se hace una breve descripción de los diferentes tipos de computadoras según Flynn.

		Datos	
-	-	Simples	Múltiples
Instrucciones	Simples	SISD	SIMD
Instrucciones	Múltiples	MISD	MIMD

Tabla 2.2: Clasificación de computadoras según Flynn.

- SISD: Única Instrucción - Único Dato.

  Ejecuta una instrucción por ciclo de reloj y con un sólo dato, ver figura 2.4.

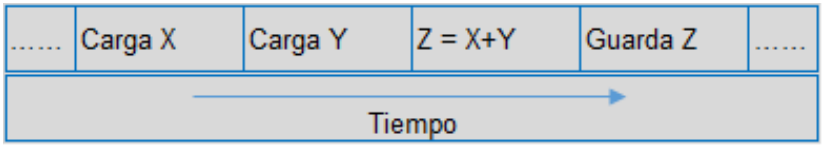

Figura 2.4: SISD - Flujo de una instrucción, un dato.

- SIMD: Única Instrucción - Múltiples Datos.

  Se ejecuta la misma instrucción sobre diferentes datos, ver figura 2.5.

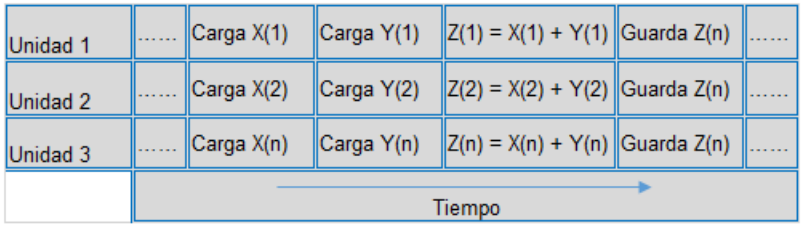

Figura 2.5: SIMD - Flujo de una instrucción, múltiples datos.

- MISD: Múltiples Instrucciones - Único Dato.

  Ejecuta múltiples instrucciones sobre el mismo dato, ver figura 2.6.

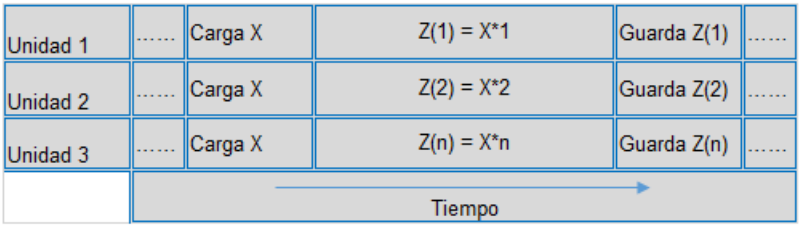

Figura 2.6: MISD - Flujo de múltiples instrucciones, un dato.

- MIMD: Múltiples Instrucciones - Múltiples Datos.

  Se ejecuta una instrucción distinta con un dato diferente, ver figura 2.7.

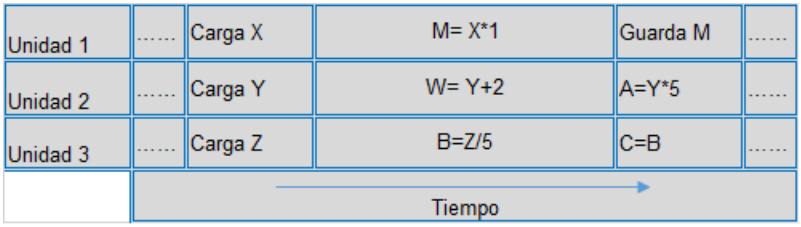

Figura 2.7: MIMD - Flujo de múltiples instrucciones, múltiples datos.

La taxonomía de Flynn se ha tomado como base para formular nuevas arquitecturas paralelas que contemplan otros factores aparte de las instrucciones y los datos. En el caso de los multiprocesadores que comparten un sistema de memoria, no se pueden ubicar en la taxonomía de Flynn ya que no toma en cuenta a la memoria.

Los multiprocesadores son conocidos también como *sistemas de memoria compartida*, y de ellos se derivan nuevas clasificaciones que se basan en el tiempo de acceso de los procesadores a la memoria, a continuación se describen estas clasificaciones [15].

UMA - Uniform Memory Access.

Los sistemas de acceso uniforme a la memoria son aquellos en los que los procesadores tienen el mismo tiempo de acceso a la memoria, pese a que cada procesador pueda contar con su propia memoria caché.

NUMA - Non Uniform Memory Access.

En los sistemas que no tienen un acceso uniforme a la memoria, los procesadores tienen diferentes tiempos de acceso, en este caso hay más de una memoria ya que cada procesador tiene una memoria local, si un procesador accede a su memoria local tendrá un tiempo de acceso menor, comparado con el tiempo de acceso a la memoria de otro procesador. En la figura 2.8 se puede ver el esquema de UMA y NUMA.

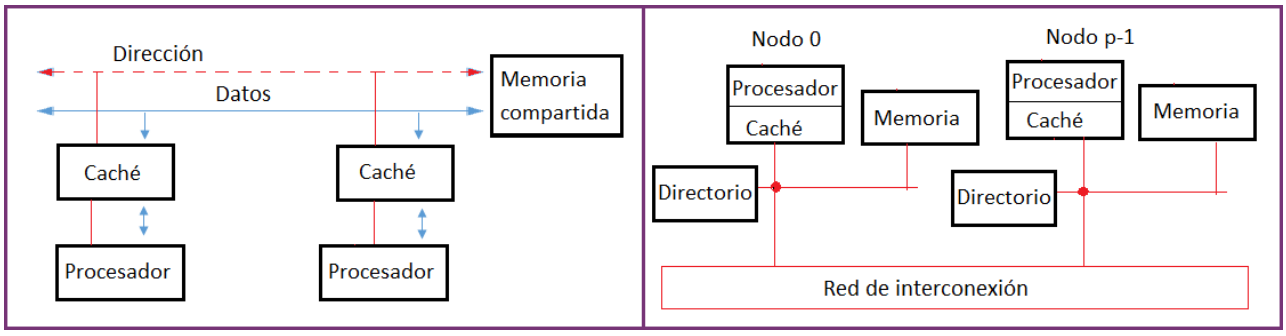

Figura 2.8: Modelo UMA (izquierda) y modelo NUMA (derecha) [16].

2.3.5. Clasificación de los clústeres

Debido a la eficiencia de los clústeres, éstos son usados en el ámbito empresarial y en el académico, por lo tanto no todos los clústeres tienen necesariamente el mismo fin en términos de aplicaciones, ya que dependiendo de los servicios que ofrecen se pueden clasificar en clústeres de alto rendimiento, clústeres de alta disponibilidad y clústeres de alta eficiencia.

Los clústeres de alto rendimiento se caracterizan por ejecutar tareas que demandan gran capacidad de cómputo y de memoria ya que para realizar las tareas solicitadas disponen de los recursos del clúster por largos tiempos. Los clústeres de alta disponibilidad, como su nombre lo indica ofrecen la máxima disponibilidad de sus capacidades, además de ofrecer confiabilidad que se refiere a la detección de fallos de manera que éstos se puedan corregir. Los clústeres de alta eficiencia ejecutan el mayor número de tareas en tiempos pequeños.

Además de esta clasificación los clústeres se pueden diferenciar como homogéneos y heterogéneos. En los clústeres homogéneos la arquitectura de los nodos es igual y el sistema operativo debe ser el mismo para cada uno de los nodos, en los clústeres heterogéneos puede existir variabilidad tanto en la arquitectura de los nodos como en el sistema operativo.

Clúster Beowulf

Los clústeres Beowulf tienen una arquitectura similar en sus nodos, y el sistema operativo además de ser el mismo debe pertenecer a una distribución de Linux, por lo que un clúster de este tipo se puede clasificar como homogéneo, estos clústeres surgieron por la necesidad en las universidades de tener sus propias supercomputadoras al menor costo posible, ya que se puede hacer uso de los recursos disponibles en lugar de adquirir equipos nuevos, además

los clústeres Beowulf se prestan a la escalabilidad, ya que su estructura permite la adición y cambio de elementos. Estos clústeres se conforman con un nodo maestro y varios nodos esclavos que son interconectados por medio de una red de área local, en esta red los usuarios del clúster inteactúan únicamente con el nodo maestro, el cual tiene al menos dos tarjetas de red, una para la red interna formada por los nodos del clúster y otra para el acceso al clúster desde una estación de trabajo externa. Debido a sus características, el uso de los clústeres Beowulf ha ganado popularidad y aceptación dentro de los grupos de investigación.

Un ejemplo del uso de este tipo de clúster ocurrió en el año 2003 en la Universidad de Toronto en el Departamento de Astronomía y Astrofísica donde se utilizó un clúster Beowulf con 512 procesadores para investigar entre otras cosas la formación de estructuras a gran escala en el universo, la interacción entre galaxias y la dinámica de flujos de acreción en agujeros negros [8].

2.3.6. Indicadores de desempeño de los clústeres

En los sistemas distribuidos se puede hacer uso de ciertos indicadores que sirven para conocer el rendimiento del sistema, algunos de estos indicadores son el factor de aceleración y la eficiencia.

Ley de Amdahl

La ley de Amdahl define el *factor de aceleración* como la relación entre el tiempo de ejecución sobre un procesador secuencial y el tiempo de ejecución en múltiples procesadores [17]. Este factor se obtiene a partir del cociente del tiempo de ejecución de un programa secuencial de referencia entre el tiempo de ejecución de un programa paralelo utilizando N procesadores.

Esta relación se puede ver en la ecuación 2.1.

$$S(N) = \frac{Tu}{TN} \tag{2.1}$$

donde:

S(N): Factor de aceleración

Tu: Tiempo de ejecución usando un procesador

TN: Tiempo de ejecución usando N procesadores

En la figura 2.9 se puede ver una gráfica del rendimiento de un sistema distribuido, en la cual se muestra el comportamiento ideal de un sistema, es decir, que la aceleración aumenta si crece el número de procesadores, pero también se encuentra el comportamiento real, en el que existe cierto número de procesadores a partir del cual la aceleración ya no aumenta, inclusive disminuye, a este punto se le conoce como punto de no regreso.

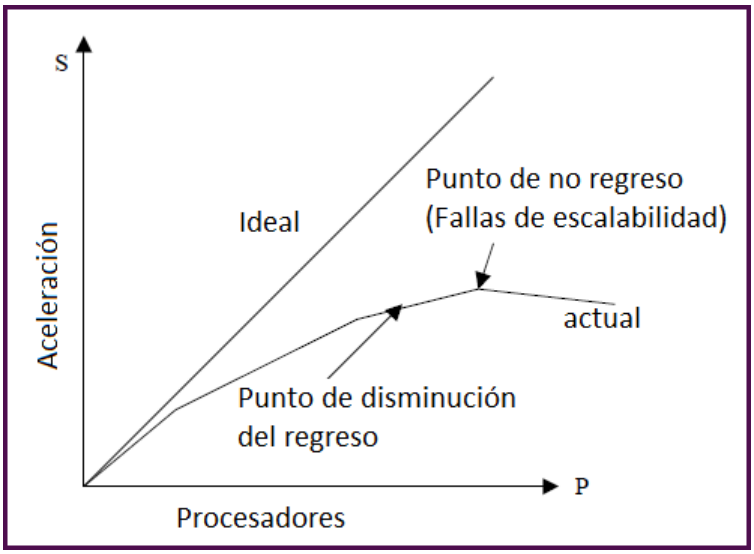

Figura 2.9: Gráfica del rendimiento en sistemas paralelos [17].

Eficiencia

La eficiencia es "la fracción del tiempo que el procesador consume haciendo trabajo útil" [17] y se deriva de la ley de Amdahl, ya que se obtiene con el cociente del factor de aceleración entre N, la cantidad de procesadores involucrados en la ejecución del proceso. La eficiencia generalmente es dada en porcentaje, cuando se tiene una eficiencia del 100 por ciento se obtiene la máxima eficiencia del sistema. La ecuación 2.2 describe a la eficiencia.

$$E = \frac{S(N)}{N} \tag{2.2}$$

donde:

E: Eficiencia

S(N): Factor de aceleración

N: Número de procesadores

2.3.7. Top de clústeres

La organización TOP500, se ha dado a la tarea desde 1993 de publicar un listado semestral en el que se encuentran los 500 clústers con mejor rendimiento a nivel mundial, dicho rendimiento es medido a través de una prueba de Linpank, esta prueba consiste en resolver sistemas densos de ecuaciones lineales, el rendimiento es medido en millones de operaciones de punto flotante por segundo, es decir, en Mflops [18].

Dentro de las publicaciones del TOP500 se incluyen también, estadísticas en las que se indican cuáles son las marcas más utilizadas para la construcción de clústers, los sistemas operativos más utilizados y la distribución de clústers por país.

La lista publicada en junio de 2016, indica que la distribución mundial de los clústeres del TOP500 es como se muestra en la tabla 2.3.

País	China	EUA	Japón	Alemania	Francia	UK	India	Rusia	Corea del Sur	Polonia	Otros
Número de Clústeres	168	165	29	26	18	11	9	7	7	6	54
Porcentaje	33.6%	33%	5.8%	5.2%	3.6%	2.2%	1.8%	1.4%	1.4%	1.2%	10.8%

Tabla 2.3: Distribución mundial de los clústers según el TOP500 de junio de 2016.

En esa ocasión, el primer lugar fue para un clúster ubicado en China, el último lugar también se encuentra en dicho país. Las características más importantes de este par de clústers se muestra en la tabla 2.4.

Característica	primer Lugar	Lugar 500
Nombre	Sunway TaihuLight	No especificado
País	China	China
Año de construcción	2016	2016
Segmento	Investigación	Industrial
Rendimiento	93,014,593.88 Gflops	286,100 Gflops
Procesador	Sunway SW26010 260C 1.45GHz	Intel Xeon E5-2650v2 8C 2.6GHz
Sistema operativo	Sunway RaiseOS 2.0.5-Linux	CentOS-Linux
Núcleos totales	1,064,9600	5,440

Tabla 2.4: Características del primer y último lugar del TOP500 de junio de 2016.

2.3.8. Clústeres en la Ciudad de México

En la Ciudad de México, las universidades públicas se han dado a la tarea de construir sus propios clústeres para que su uso contribuya al avance de investigaciones en diferentes áreas, el rendimiento de estos clústeres se ha medido con la prueba de Linpack, y debido a los resultados obtenidos han formado parte del TOP500 en más de una ocasión.

Un clúster que por su rendimiento apareció en la lista del TOP500 en junio de 2007 (en el lugar 391) fue construido por el sector privado, particularmente por el banco Azteca que construyo un clúster con sistema operativo HP Unix y rendimiento de 4,704 Gflops. En la tabla 2.5 se presentan algunas de las características de los clústers del sector académico de la Ciudad de México.

Institución	UNAM	UNAM	UAM	IPN
Año	junio, 2007	noviembre, 2012	noviembre, 2008	junio y noviembre 2015
Nombre	No especificado	No especificado	No especificado	ABACUS I
Lugar en el Top	309	348	225	225 y 370
Segmento	Académico	Académico	Académico	Investigación
Rendimiento (Gflops)	5,090	92,282.1	18,480	277,504
Procesador	AMD x86_64 dual core	xeon E5-2670	Intel Xeon	Xeon E5- 2697
Sistema operativo	Linux	Linux	Linux	Suse-Linux
Año de construcción	2006	2012	2008	2015
Núcleos	No especificado	5616	2120	8820

Tabla 2.5: Características de los clústeres de la CDMX en el TOP500.

En 2012 se anunció en la Ciudad de México la construcción de un grid, el proyecto fue denominado *Laboratorio Nacional de Cómputo de Alto Rendimiento*, el propósito del proyecto es resolver problemas científicos. Este grid se constituirá por nodos robustos, es decir, cada nodo será un clúster, justamente los de la UNAM, UAM y CINVESTAV. El grid se conectará a través de una red de fibra óptica que se encuentra tendida sobre la Red de Transporte Colectivo Metro.

2.4. Modelo WRF

El modelo WRF (Weather Research & Forecasting Model), es un modelo mesoescalar (la mesoescala se describe como parte del conjunto de las escalas meteorológicas, definidas en

función de las dimensiones espaciales horizontales y de la duración media de los sistemas atmosféricos, según lo cual, la mesoescala va de los 3 km a los 100 km [19]) de pronóstico e investigación meteorológica que surgió en la década de los noventa.

WRF está diseñado para trabajar sobre sistemas de alto rendimiento por las siguientes razones: las resoluciones son más finas, los dominios son más grandes, las escalas de tiempo son más largas, entre otras [20]. Actualmente está en uso y actualización gracias a NCEP (National Centers for Environmental Prediction), AFWA (Air Force Weather Agency), otros centros de pronóstico y una comunidad de usuarios [21], en seguida se listan algunas de las entidades que participan en su desarrollo.

- NCAR: National Center for Atmospheric Research
- FSL: Forecast Systems Laboratory
- FAA: Federal Aviation Administration
- Naval Research Laboratory
- University of Oklahoma

La figura 2.10 ilustra las mallas de trabajo en los modelos de pronóstico meteorológico, donde el color negro indica la malla de un modelo global; el color azul, un modelo regional y el color rojo, la malla de un modelo mesoescalar.

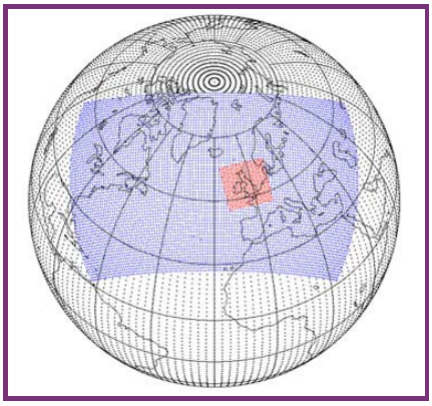

Figura 2.10: Mallas de trabajo para modelos meteorológicos [20].

Este modelo puede trabajar con datos atmosféricos reales o bajo condiciones idealizadas, por lo que ofrece diversas aplicaciones meteorológicas, por ejemplo: investigación meteorológica, simulaciones atmosféricas idealizadas, estudios de asimilación y desarrollo de datos, entre otros. WRF está diseñado para trabajar sobre dos bases, la primera es un sistema de asimilación de datos (WPS) y la segunda una arquitectura de software que trabaja en paralelo, con el objetivo de reducir el tiempo de procesamiento del programa $wrf.exe$, que se produce con base al programa $real.exe$ que a su vez forma parte de un sistema de pre-procesamiento, por lo que este programa es el que tiene mayores requerimientos de cómputo [20].

2.4.1. Sistema de pre-procesamiento: WPS

El modelo WRF necesita un sistema de pre-procesamiento de datos llamado WPS, que se puede describir con base a los programas: *geogrid.exe, ungrib.exe y metgrid.exe*. La función en conjunto de estos programas, es preparar la entrada al programa real para simulaciones de datos reales en WRF. El diagrama del modelo WPS se muestra en la figura 2.11.

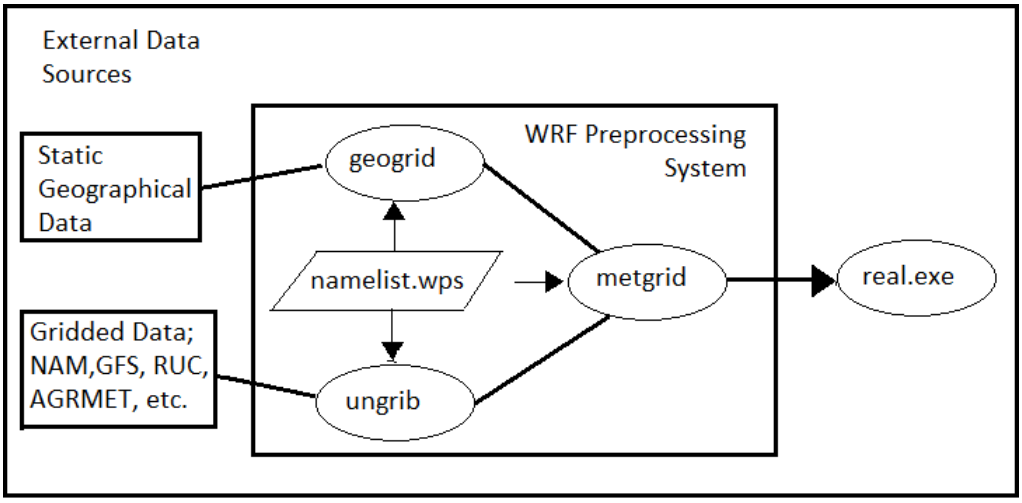

Figura 2.11: Componentes del modelo WRF [22].

Con el programa **geogrid** se puede definir el dominio en el que se realizará la predicción (latitud y longitud), además de interpolar diferentes tipos de datos terrestres, por ejemplo, el tipo de uso del suelo, y las características de la zona elegida en función de la época del año. El programa **ungrib** no depende de ningún dominio del modelo WRF y su función es descomprimir y leer los archivos de formato grib, que contienen datos meteorológicos y los escribe en un formato que el programa metgrid pueda leer. El programa **metgrib** interpola horizontalmente los datos del programa ungrid dentro del dominio definido por geogrid. La salida metgrid interpolada es la entrada del programa real WRF.

Con base a los fundamentos teóricos presentados, en el siguiente capítulo se describe cómo se llevó a cabo la construcción y configuración de un clúster a partir de una red de cuatro servidores y software libre.

Capítulo 3

Desarrollo e implementación del clúster Xexelo0

En este capítulo se describen las características de la infraestructura y el arreglo para la implementación del clúster Xexelo0, se describe a detalle la configuración del software utilizado, como el sistema operativo, el middleware, el firewall y las bibliotecas necesarias para poder ejecutar el modelo de predicción climática WRF.

3.1. Componentes del clúster

En términos de hardware, el clúster Xexelo0 está compuesto por cuatro servidores que son administrados de manera remota. Los servidores son homogéneos, es decir, presentan las mismas características físicas. Dichos servidores fueron adquiridos gracias a los recursos otorgados por la SECITI [3]. En la Tabla 3.1 se describen las características de los servidores.
Respecto al software, el clúster también es homogéneo ya que se instaló el mismo software en cada nodo. Los programas utilizados son de distribución libre, aunque también se usan algunas bibliotecas de Intel, que a pesar de ser una marca que ofrece soluciones privativas,

[3]Convenio UACM/SECITI 060/2013.

permite descargar de manera gratuita algunas de sus soluciones a miembros del sector académico como estudiantes y profesores.

Marca	Supermicro
Modelo	X9DRD-iF/LD
Procesadores	2
Tipo de Procesador	Intel Xeon E5-2620 a 2.1 GHZ
Hyper-threding	Sí, 12 núcleos por procesador
Memoria RAM	32 GB
Disco Duro	1 TB
NIC de procesamiento	10 Gbps
NIC de monitoreo	1 Gbps

Tabla 3.1: Características de los servidores.

Además de los servidores que son los agentes más relevantes del clúster se utilizan elementos como switches, cable y el equipo de administración remota. En la siguiente lista se menciona el equipo adicional y sus características.

- 1 Gabinete SBE-TECH, con 20 Unidades de rack, puertas laterales desmontables y puertas frontal y trasera con llave.
- 1 switch Netgear ProSafe XS708E con 8 puertos 10 GBASE-T.
- 1 computadora Dell Optiplex-960, con procesador Intel Core 2 Duo a 2.99 Ghz y tarjeta de red a 1 Gbps
- Cable UTP categoría 6
- 1 switch Extrem Networks X440-24T
- Fibra óptica (5 m aprox) 62.5/125 um

3.2. Diseño y construcción

Los cuatro servidores que componen la base del clúster, los switches Extreme Networks y Netgear se colocaron dentro del gabinete mencionado con anterioridad, estos elementos se ubicaron en una zona apropiada dentro del laboratorio B-404 de manera que tuvieran cercanía tanto a las tomas de corriente eléctrica, como a las conexiones de red. La computadora Dell Optiplex-960 se utilizó como equipo de administración remota y se colocó en el aula B-207. Los servidores se conectaron al switch Netgear, con el propósito de formar una red con capacidad de transferencia de 10 Gbps. Esta red, a su vez se integró a la red del laboratorio B-404 a través del switch Extreme Networks. En la figura 3.1 se muestra un esquema general de la topología del clúster.

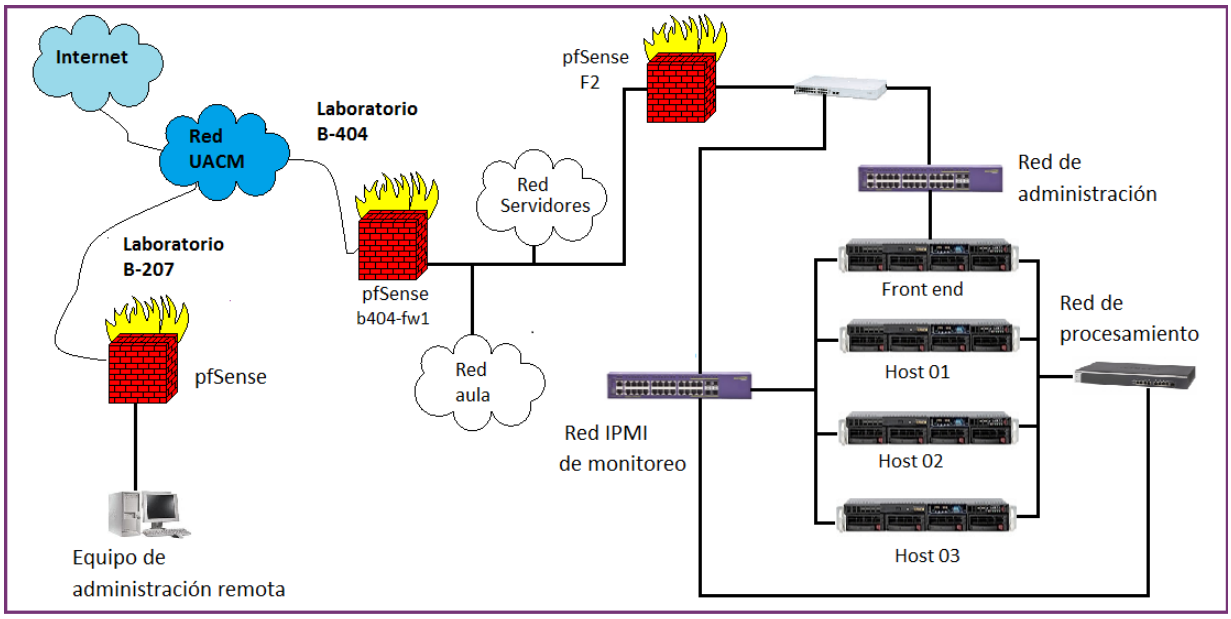

Figura 3.1: Esquema general de la red del clúster Xexelo0.

IPMI es una red de monitoreo, su software está incluido en los servidores por decisión del fabricante, y pese a que se muestra dentro del esquema de la figura 3.1 no se habla de ella, ya que el monitoreo de los servidores no forma parte de los objetivos en este trabajo.

En modo general, el cableado se realizó con base a las siguientes consideraciones: dejar un exceso de cable UTP no mayor a treinta metros, evitar las curvas en el tendido del cable, agrupar los cables con cinta de velcro, que la distancia entre el tendido de la red y el tendido eléctrico tuviera al menos cincuenta centímetros de distancia [23].

Para identificar el clúster dentro del laboratorio, se etiquetó conforme al estándar EIA/TIA 606-A, estándar para la infraestructura de telecomunicaciones para centros de datos. En la figura 3.2 se muestra el equipo de administración remota, en las figuras 3.3 y 3.4 se muestran imágenes del clúster.

Figura 3.2: Equipo de administración remota-Dell Optiplex-960.

Figura 3.3: Xexelo0 en laboratorio B-404. Del lado izquierdo se aprecia el rack, del lado derecho se aprecia un acercamiento a los servidores

Figura 3.4: Etiquetado del rack en el que se encuentra el clúster Xexelo0.

3.3. Configuración

En esta sección se describen las características de la configuración de los switches, el firewall y el software utilizado, incluyendo el sistema operativo, el sistema NFS (Sistema de Archivos compartidos) y las bibliotecas utilizadas para la ejecución del modelo WRF. Con los elementos mencionados en la sección anterior, se propone que los cuatro servidores formen una red a 10 Gbps a través del switch Netgear.

3.3.1. Configuración de los switches y firewall

- Switches

  El switch Netgear tiene una transmisión de datos de 10 Gbps y es el equipo a través del cual se lleva a cabo la comunicación de los nodos del clúster. Se configuraron dos vlan en este switch, una de administración y otra para la comunicación entre los nodos del clúster.

  A través del Switch Extreme Networks, se estableció la salida del clúster al laboratorio B-404 con una velocidad de 1 Gbps. En este caso, se configuraron dos vlan, una para la red de administración IPMI, y otra dedicada al clúster.

- Firewall

  Los firewalls son una herramienta para brindar seguridad a los sistemas de redes. pfSense es parte del software libre que tiene funciones de firewall y router que se puede administrar a través de una interfaz web [24], debido a esto, se eligió pfSence como el firewall adecuado para el laboratorio B-404.

Figura 3.5: Firewalls del laboratorio B-404.

El pfSense se instaló en equipos Compaq Evo con procesador Intel pentium 4. En la figura 3.5 se muestra el equipo utilizado para los firewalls, a la izquierda está el firewall del laboratorio y a la derecha el firewall del clúster.

En el primer firewall, denominado $b404 - fw1$ y dedicado al laboratorio B-404, se configuraron seis interfaces de red, una de ellas para la conexión con la red de la UACM y las otras interfaces se configuraron con base a las necesidades particulares del laboratorio, entre ellas, una red dedicada a los equipos del aula, una más para servidores web y otra para el clúster. En la figura 3.6 se muestran las interfaces de este firewall.

En el segundo firewall denominado $pfsense$ y dedicado al clúster, se configuraron 2 interfaces de red, una de ellas para establecer comunicación con la red del laboratorio, y la otra para la red del clúster. En la figura 3.7 se muestran las interfaces de este firewall.

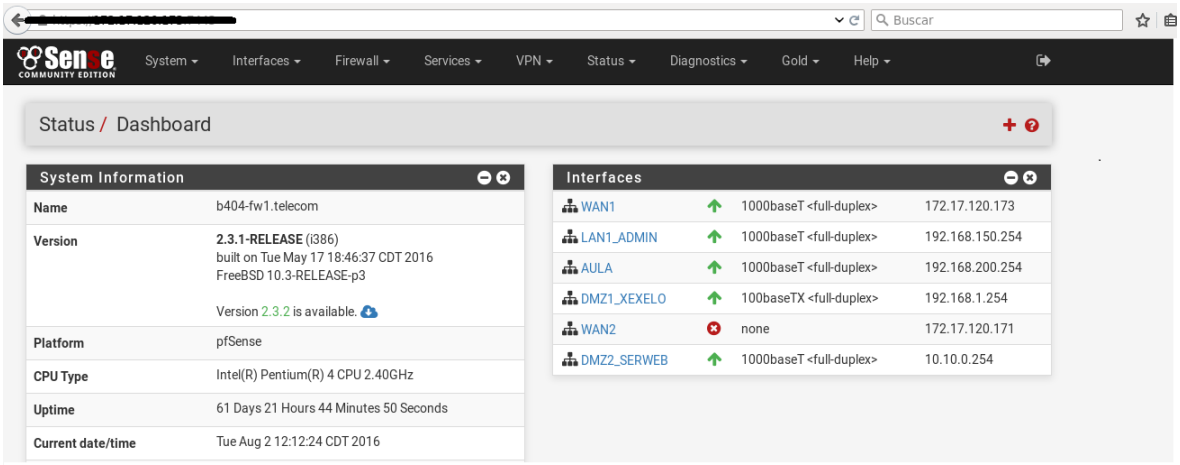

Figura 3.6: Administración de pfSense para la red del laboratorio B-404.

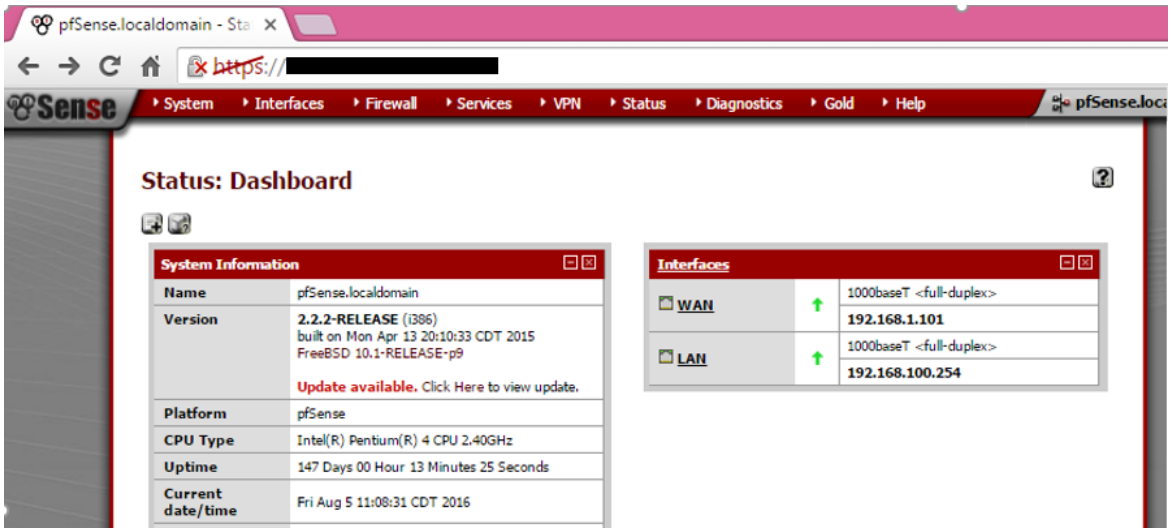

Figura 3.7: Administración de pfSense para la red del clúster.

De modo general, en los firewall se configuraron diferentes interfaces de red y se modificaron algunos parámetros para brindar seguridad a la red de datos, permitiendo la salida de las peticiones de los equipos del laboratorio, pero restringiendo las peticiones entrantes. Los cambios que se realizaron son los siguientes: cambio del protocolo http por el protocolo https y cambio del puerto SSH, se activaron los servicios dhcpd, sshd, ntpd, dnsmasq y dpinger. Los cambios mencionados se realizaron desde la pestaña *system* e *interfaces* y se pueden corroborar en la pestaña *status*.

3.3.2. Configuración del sistema operativo

El clúster fue configurado con software de distribución libre, el sistema operativo utilizado tanto en los nodos como en el equipo de administración remota es CentOS 6.6.
La instalación del sistema operativo se realizó *in situ* y de modo escalonado.
Las opciones elegidas para la instalación de CentOS fueron las siguientes:

- Idioma del sistema: Español
- Idioma del teclado: Latinoamericano
- Tipo del dispositivo: Almacenamiento básico
- Nombre del host: frontend, host01, host02 y host03
- Zona Horaria: América, Ciudad de México
- Espacio en Disco Duro: Usar todo el espacio
- Instalación: Mínima

3.3.3. Configuración para la comunicación

Al terminar la instalación del sistema operativo, se configuraron las tarjetas de red de cada nodo. En el *frontend* se configuraron dos tarjetas, la primera con capacidad de transferencia de 10 Gbps para la red interna, es decir, la red de procesamiento, la otra tarjeta de 1 Gbps se configuró para tener comunicación con la red del laboratorio B-404 y tener acceso a internet. En los nodos *esclavos* sólo se configuró la tarjeta de 10 Gbps. En la tabla 3.2 se muestra la configuración de las interfaces de red.

Nodo	Interfaz	IP	Máscara	Gateway
frontend	eth0	10.10.10.1	24	10.10.10.254
frontend	eth3	27.27.27.2	8	No hay
host01	eth3	27.27.27.3	8	27.27.27.2
host02	eth3	27.27.27.4	8	27.27.27.2
host03	eth3	27.27.27.5	8	27.27.27.2

Tabla 3.2: Configuración de las interfaces de red.

La configuración se guardó con la instrucción:

service network restart

Además de configurar las tarjetas de red, se realizó una regla NAT en el frontend para que los nodos esclavos resolvieran a través de él sus peticiones a internet. Esta regla se realizó en modo root, las instrucciones que describen a la regla NAT se encuentran en el apéndice A. Posteriormente en cada nodo se realizaron las siguientes acciones:

- Se deshabilitó el firewall nativo de CentOS, esto con el fin de abrir los puertos que por seguridad estaban bloqueados, por ejemplo ftp o ssh. Se utilizó la siguiente instrucción para deshabilitar el firewall a través de una interfaz gráfica

system-config-firewall-tui

En la figura 3.8. se muestra el entorno para activar/desactivar el firewall de CentOS.

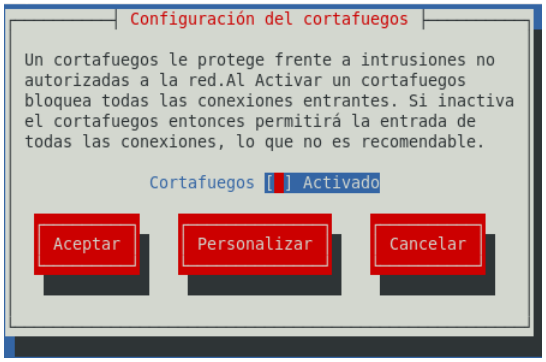

Figura 3.8: Configuración del firewall nativo de CentOS.

- Se deshabilitó SELinux para no tener inconvenientes en la gestión de los servicios, particularmente en su ejecución ya que SELinux es el módulo de seguridad. La modificación se hizo en el archivo */etc/selinux/config* con la instrucción:
  SELINUX= disabled

- Se generó un usuario llamado *usuario*, al cual se le asignó contraseña. Dicho usuario no goza de privilegios de root. Este se generó con la línea:
  adduser usuario
  y la contraseña se asignó con:
  passwd usuario

- En la ubicación de *usuario* se creó un directorio llamado *aplicacion* con el fin de tener un directorio que se pueda compartir.

- Se actualizó el sistema operativo para tener las versiones más actuales de la paquetería de CentOS, dicha actualización se realizó con las instrucciones:

yum -y update

yum -y upgrade

- Se agregaron repositorios al sistema operativo ya que la instalación de CentOS fue mínima, esto se logró con las instrucciones que se encuentran en el apéndice B.

- Se instaló un grupo de herramientas, dentro del cual se encuentran *gcc* y *gfortran*, dos compiladores necesarios para la ejecución del modelo WRF. El grupo de herramientas se instaló con la instrucción:
  yum groupinstall 'Development Tools'

- Se instalaron los paquetes wget y htop, para descargar archivos en línea y poder monitorear el estado de los procesadores de cada nodo. Las instrucciones para estas instalaciones son respectivamente:
  yum install -y wget
  yum -y install htop

3.3.4. Configuración de la llave RSA

RSA es un criptosistema de llave pública, en el cual, las claves se componen de números primos con cientos de dígitos. Computacionalmente es más complejo y tardado que otros sistemas criptográficos, por tanto, es un sistema adecuado para implementarse en el clúster.

Una vez listo el sistema operativo y la configuración de la red de Xexelo0 se configuró una llave RSA con la que el *usuario* puede acceder sin restricciones desde el frontend a los otros nodos. La llave se configuró de la siguiente manera:
En todos los nodos, en el *usuario* se creó un directorio oculto llamado ssh con la instrucción

mkdir .ssh

Desde el frontend se ejecutaron las siguientes instrucciones para generar la llave RSA, dar permisos al directorio *.ssh* y renombrar la llave.

$ssh - keygen - trsa$

$chmod \sqcup 700 \sim /.ssh$

$chmod \sqcup 600 \sim /.ssh/id\_rsa$

$cd \sqcup .ssh$

$cat \sqcup id\_rsa.pub >> \sim /.ssh/authorized\_keys$

Las siguientes instrucciones se ejecutaron para copiar la llave desde el frontend hacia el directorio *.ssh* de los nodos esclavos del clúster.

$scp \sqcup authorized\_keys \sqcup usuario@27.27.27.3 : /home/usuario/.ssh$

$scp \sqcup authorized\_keys \sqcup usuario@27.27.27.4 : /home/usuario/.ssh$

$scp \sqcup authorized\_keys \sqcup usuario@27.27.27.5 : /home/usuario/.ssh$

Después, en los nodos host01, host02 y host03 se ejecutaron las siguientes instrucciones para permitir la lectura, escritura y ejecución en el directorio *.ssh* al usuario *usuario*.

$chmod \sqcup 700 \sim /.ssh$

$chmod \sqcup 600 \sim /.ssh/authorized\_keys$

3.3.5. Configuración del sistema de archivos compartidos - NFS

Una vez que se tiene acceso desde el frontend hacia los demás nodos, se continuó con la configuración desde la máquina de administración remota. El siguiente paso fue la configuración del sistema de archivos compartidos, para lo cual se instaló NFS, por tanto, en todos los

nodos se creó el directorio *aplicacion*, después, en el frontend y en modo root se ejecutaron las siguientes instrucciones para dar permisos al directorio *aplicacion*, instalar y poner en marcha el NFS.

$chmod \sqcup 777 \sqcup /home/usuario/aplicacion$

$yum \sqcup install \sqcup nfs - utils \sqcup nfs - utils - lib$

$chk \sqcup config \sqcup nfs \sqcup on$

$service \sqcup rpcbind \sqcup start$

$service \sqcup nfs \sqcup start$

Se editó el archivo */etc/exports*, para indicar el directorio a compartir, para lo que agregó la siguiente instrucción:

$/home/usuario/aplicacion \sqcup 27.27.27.0/8(rw, sync, no\_root\_squash, no\_subtree\_check)$

La primer parte de la instrucción, indica la ruta del directorio a compartir, seguido del conjunto de direcciones IP del clúster, incluyendo la máscara de subred e indicando entre paréntesis la configuración de NFS, donde:

- *rw*: indica que se puede leer y escribir en el directorio.
- *sync*: coordina las peticiones de los clientes al servidor de manera que el servidor responderá a un cliente cuando se haya completado su trabajo actual.
- *no_root_squash*: permite el acceso de los clientes root como root local, para que los cambios realizados por parte de los clientes en modo root sean aceptados.
- *no_subtree_check*: hace que la transferencia de archivos sea más rápida ya que no verifica que los directorios pertenezcan al mismo nivel.

Se ejecutó *exports -a* para exportar el directorio compartido, posteriormente en cada uno de los nodos esclavos y en modo root se ejecutaron las siguientes instrucciones para instalar NFS y montar la carpeta compartida.

$yum \sqcup install \sqcup nfs - utils \sqcup nfs - utils - lib$

$mount \sqcup 27.27.27.2 : /home/usuario/aplicacion \sqcup /home/usuario/aplicacion$

Se editó el archivo */etc/fstab* de cada nodo, agregando la siguiente línea con el fin de que el directorio *aplicacion* quedara compartido de manera permanente, evitando el tener que montar el directorio compartido en cada inicio de sesión en los nodos del clúster.

$27.27.27.2 : /home/usuario/aplicacion \sqcup /home/usuario/aplicacion \sqcup nfs \sqcup defaults \sqcup 0 \sqcup 0$

3.3.6. Instalación de las bibliotecas Intel

Los servidores que conforman el clúster están equipados con procesadores Intel, esta marca además de desarrollar productos de hardware también desarrolla software especializado, de modo que se optimice el funcionamiento de sus plataformas. Intel ha desarrollado el programa *Intel Parallel Studio XE*[4] que ofrece diversas herramientas para el cómputo paralelo y además puede operar sobre sistemas Linux. Para la instalación de las bibliotecas de Intel se realizaron las siguientes acciones:

- Se creó en cada nodo el directorio *LIB-INT* en la ruta */home/usuario/aplicacion*, para almacenar en este directorio las bibliotecas de intel
- Desde la máquina de administración se descargó el archivo *parallel_studio_xe_2016.tgz* [25].

[4]Este software puede ser descargado de manera gratuita por integrantes de la comunidad académica, el único requisito es proporcionar una cuenta de correo institucional vigente.

- El archivo descargado se copió al frontend con la instrucción:

  scp -P xxxx /home/descargas/parallel_studio_xe_2016.tgz //

  usuario@10.10.10.1:/home/usuario/aplicacion/LIB-INT

  El parámetro -P xxxx, indica el puerto ssh a través del cual se tiene acceso al clúster, ya que por motivos de seguridad el puerto definido por defecto ha sido modificado.

- Desde el frontend, en el directorio LIB-INT se descomprimió el archivo con la línea:

  tar -xvzf parallel_studio_xe_2016.tgz

- Al descomprimir se genera el directorio *parallel_studio_xe_2016*, dentro del cual se ejecutó en modo root la instrucción:

  sh install.sh

  Después de esta instrucción, se generó un entorno de instalación como el que se muestra en la figura 3.9. La instalación consta de siete pasos que se describen a continuación.

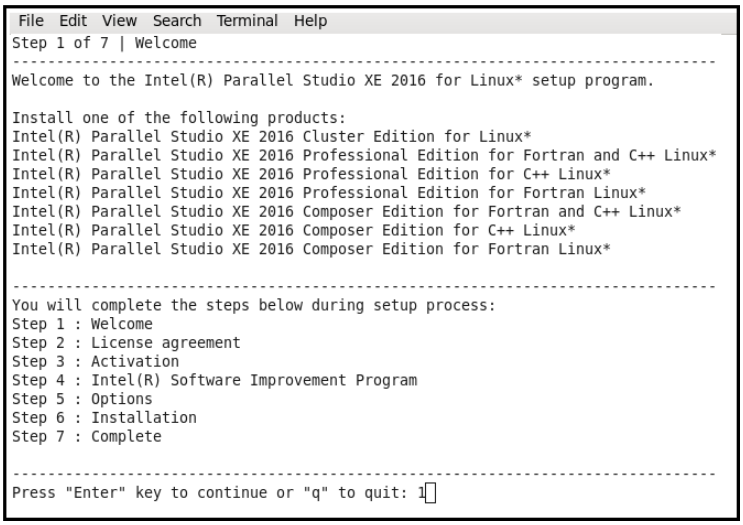

Figura 3.9: Bienvenida al entorno de configuración de las bibliotecas de Intel.

- Paso 1: Bienvenida al entorno de instalación y revisión de prerrequisitos del sistema.

- Paso 2: Acuerdo de licencia, se describen los términos y condiciones para el uso del software, para continuar hay que aceptarlos escribiendo la palabra *accept*.

- Paso 3: Activación, se especifica qué tipo de activación se desea, en este caso se elige la activación del producto a través de un número de serie que es proporcionado por Intel al registrarse como usuario académico.

- Paso 4: Programa de mejora de software, se indica si se desea o no compartir las experiencias con el software.

- Paso 5: Opciones, se indica el tipo de configuración, en este caso se eligió la configuración por defecto.

- Paso 6: Instalación, se indica el directorio de instalación. Se elige el directorio */opt*, que es el directorio por defecto para la instalación.

- Paso 7: Instalación, se instala el software y el sistema envía mensaje de *completo*, como el que se ve en la figura 3.10.

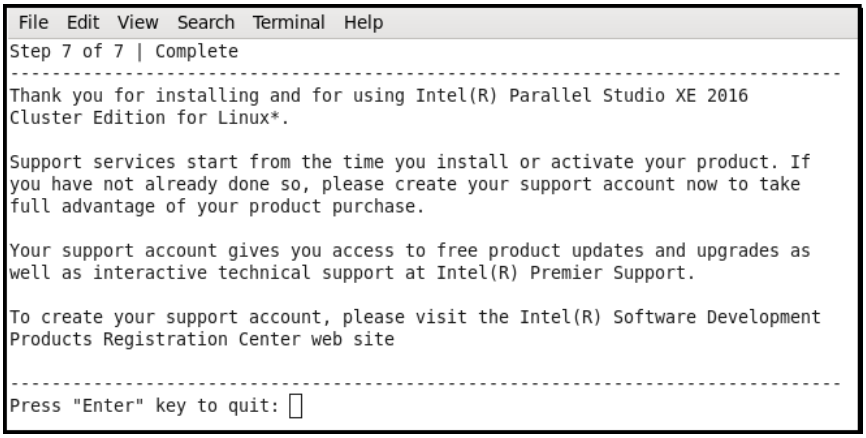

Figura 3.10: Último paso para la instalación de las bibliotecas de Intel.

Para la instalación de las bibliotecas en los demás nodos, se accedió a la carpeta compartida */aplicacion/LIB-INT/parallel_studio_xe_2016*, se ejecutó la instrucción *sh install.sh* y se siguieron los siete pasos mencionados anteriormente.

Después de instalar el software, en cada nodo se editó el archivo *.bash_profile* (las líneas agregadas a este archivo son importantes, ya que si otro usuario desea utilizar las bibliotecas de Intel no tiene que realizar toda la instalación, bastará con que modifique su archivo *.bash_profile*), en el que se agregaron las siguientes líneas para exportar la ruta en la que se instalaron las bibliotecas de Intel, ésto con el fin de que con cada inicio de sesión las bibliotecas estén activas.

$\#aditional \sqcup path$

$source \sqcup /opt/intel/bin/compilervars.sh \sqcup intel64$

$export \sqcup PATH = \$PATH : /opt/intel/bin$

$export \sqcup LD\_LIBRARY\_PATH = \$LD\_LIBRARY\_PATH :$

$/opt/intel/composer\_xe\_2016/Lib/intel64$

Al terminar de modificar y guardar los cambios en el archivo *.bash_profile* se reinició cada uno de los servidores.

3.3.7. Configuración de mpich, netcdf, zlib, libpng, y jasper

El modelo de predicción climática WRF necesita de ciertas bibliotecas, con versiones específicas que ya han sido probadas para proporcionar al usuario cierta estabilidad. A su vez, para que éstas bibliotecas funcionen, el sistema debe contar con los compiladores gcc, cpp y gfortran (en [26] se encuentra la guía que se tomó como referencia en este trabajo para la

instalación de bibliotecas, la configuración de WPS y WRF, además de las pruebas de los compiladores y bibliotecas). Para hacer las pruebas de los compiladores se creó el directorio *TESTS* en *usuario*, dentro de dicho directorio se verificó la versión de los compiladores obtenidos del grupo de herramientas $'DevelopmentTools'$, en la tabla 3.3 se muestran los comandos utilizados y sus respectivas respuestas.

Instrucción	Respuesta
which gfortran	/usr/bin/gfortran
which gcc	/usr/bin/gcc
which cpp	/usr/bin/cpp
gcc –version	gcc (GCC) 4.4.7 20120313 (Red Hat 4.4.7-11)

Tabla 3.3: Ubicación y versión de los compiladores de C.

En el directorio *TESTS* se descargó el archivo comprimido *Fortran_C_tests.tar* para hacer más pruebas, después de descomprimirlo con *tar -xf Fortran_C_tests.tar* se realizaron los tests a los compiladores, en los que se ejecutan programas sencillos en lenguaje C y fortran, los resultados de los tests se muestran en el apéndice C, Después de realizar los test con éxito, en el directorio */opt* se creó el directorio *bibliotecas* en el cuál se descargaron los archivos comprimidos de netcdf-4.1.3, mpich-3.0.4, zlib-1.2.7, libpng-1.2.50, y jasper-1.900.1. y se realizó la declaración de las siguientes variables[5] que facilitan el proceso de instalación de mpich, netcdf, zlib, libpng y jasper.

[5]En las guías de WRF generalmente se utiliza el comando *setenv* debido a que se trabaja con el shell *tcsh*, pero en éste caso se trabaja con el shell *bash* por tanto el comando usado para declarar variables es *export*.

$export \sqcup DIR =$ "$/opt$"

$export \sqcup CC =$ "gcc"

$export \sqcup CXX =$ "$g++$"

$export \sqcup FC =$ "$gfortran$"

$export \sqcup FCFLAGS =$ "$-m64$"

$export \sqcup F77 =$ "$gfortran$"

$export \sqcup FFLAGS =$ "$-m64$"

Se descomprimieron las bibliotecas descargadas con la instrucción:

$tar \sqcup xzvf \sqcup archivo$

donde *archivo* es cada uno de los archivos comprimidos, es decir, netcdf-4.1.3.tar.gz, mpich-3.0.4.tar.gz, zlib-1.2.7.tar.gz, libpng-1.2.50.tar.gz y jasper-1.900.1.tar.gz.

A continuación, se describen las instrucciones que se utilizaron para instalar netcdf, mpich, zlib, libpng y jasper. Al finalizar, se agregaron todas las variables generadas durante la instalación, en el archivo *.bashrc*, el cual se puede encontrar en el apéndice D.

- **Instrucciones para la instalación de NetCDF**

  $cd \sqcup netcdf - 4.1.3$

  $./configure \sqcup --prefix = \$DIR/netcdf \backslash$

  $--disable-dap --disable-netcdf-4 --disable-shared$

  $make$

  $make \sqcup install$

  $export \sqcup PATH =$ "$\$DIR/netcdf/bin : \$PATH$"

  $export \sqcup NETCDF =$ "$\$DIR/netcdf$"

  $cd \sqcup ..$

- **Instrucciones para la instalación de Mpich**

  $cd \sqcup mpich - 3.0.4$

  $./configure - -prefix = \$DIR/mpich$

  $make$

  $make \sqcup install$

  $export \sqcup PATH = "\$DIR/mpich/bin : \$PATH"$

  $cd \sqcup ..$

- **Instrucciones para la instalación de Zlib**

  $export \sqcup LDFLAGS = " - L\$DIR/grib2/lib"$

  $export \sqcup CPPFLAGS = " - I\$DIR/grib2/include"$

  $cd \sqcup zlib - 1.2.7$

  $./configure - -prefix = \$DIR/grib2$

  $make$

  $make \sqcup install$

  $cd \sqcup ..$

- **Instrucciones para la instalación de Libpng**

  $cd \sqcup libpng - 1.2.50$

  $./configure - -prefix = \$DIR/grib2$

  $make$

  $make \sqcup install$

  $cd \sqcup ..$

- **Instrucciones para la instalación de Jasper**

  $cd \sqcup jasper - 1.900.1$

  $./configure - -prefix = \$DIR/grib2$

  $make$

  $make \sqcup install$

  $cd \sqcup ..$

Pruebas de funcionamiento de las bibliotecas

Una vez instaladas las bibliotecas, se realizaron un par de pruebas para comprobar su funcionamiento y compatibilidad. En el directorio *TESTS* se descargó un archivo comprimido para realizar dichas pruebas. El archivo se descomprimió con la línea:

tar -xf Fortran_C_NETCDF_MPI_tests.tar

Se hizo una copia del archivo netcdf.inc en el directorio *TESTS* con la instrucción:

$cp \sqcup \${NETCDF}/include/netcdf.inc \sqcup .$

```
[usuario@host02 TESTS]$ gfortran -c 01_fortran+c+netcdf_f.f
[usuario@host02 TESTS]$ gcc -c 01_fortran+c+netcdf_c.c
[usuario@host02 TESTS]$ gfortran 01_fortran+c+netcdf_f.o 01_fortran+c+netcdf_c.o -L${NETCDF}/lib
-lnetcdff -lnetcdf
[usuario@host02 TESTS]$ ./a.out
   C function called by Fortran
   Values are xx =  2.00 and ii = 1
 SUCCESS test 1 fortran + c + netcdf
[usuario@host02 TESTS]$ mpif90 -c 02_fortran+c+netcdf+mpi_f.f
[usuario@host02 TESTS]$ mpicc -c 02_fortran+c+netcdf+mpi_c.c
[usuario@host02 TESTS]$ mpif90 02_fortran+c+netcdf+mpi_f.o 02_fortran+c+netcdf+mpi_c.o -L${NETCDF
}/lib -lnetcdff -lnetcdf
[usuario@host02 TESTS]$ mpirun ./a.out
   C function called by Fortran
   Values are xx =  2.00 and ii = 1
 status =            2
 SUCCESS test 2 fortran + c + netcdf + mpi
[usuario@host02 TESTS]$
```

Figura 3.11: Pruebas para comprobar la compatibilidad de las bibliotecas.

En la figura 3.11 se muestran las pruebas realizadas con los resultados esperados, en los que se muestra la ejecución de programas que integran a Fortran, C, netcdf y mpich. Las pruebas consisten en compilar y ejecutar programas sencillos escritos en C y Fortran, la primera prueba fue para el funcionamiento de Fortran, C y NetCDF, la segunda prueba fue para Fortran, C, NetCDF y MPI. Estos programas de prueba se encuentran en [26].

3.3.8. Configuración de WRF

Ya que se hicieron las pruebas de compatibilidad de las bibliotecas se configuró WRF en *usuario* sin privilegios, para lo cual se creó el directorio de trabajo *Build_WRF* en la ruta *home/usuario/aplicacion*. En *Build_WRF* se descargó el archivo comprimido de WRF y se descomprimió con las instrucciones:

gunzip WRFV3.8.TAR.gz

tar -xf WRFV3.7.TAR

Al descomprimir el archivo se generó el directorio WRFV3 al cual se accedió, y para configurar se ejecutó la instrucción:

./configure

La instrucción anterior imprime una lista de opciones de los compiladores y la forma de construir WRF. Para este trabajo se eligió la **opción 15** como se puede ver en la figura 3.12, con esta opción se eligen los compiladores de **Intel (ifort/icc)** y un tipo de construcción **dmpar** que significa Distributed-memory Parallelism que utiliza MPI para la ejecución en uno y en varios nodos del clúster. Después, se compiló WRF como se muestra en la siguiente línea, cabe mencionar que el tiempo que tardó esta compilación fue de 42 minutos.

./compile ⊔ em_real⊔ > & ⊔ log.compile

```
------------------------------------------------------------------------
Please select from among the following Linux x86_64 options:

  1. (serial)   2. (smpar)   3. (dmpar)   4. (dm+sm)   PGI (pgf90/gcc)
  5. (serial)   6. (smpar)   7. (dmpar)   8. (dm+sm)   PGI (pgf90/pgcc): SGI MPT
  9. (serial)  10. (smpar)  11. (dmpar)  12. (dm+sm)   PGI (pgf90/gcc): PGI accelerator
 13. (serial)  14. (smpar)  15. (dmpar)  16. (dm+sm)   INTEL (ifort/icc)
                                         17. (dm+sm)   INTEL (ifort/icc): Xeon Phi (MIC architec
ture)
 18. (serial)  19. (smpar)  20. (dmpar)  21. (dm+sm)   INTEL (ifort/icc): Xeon (SNB with AVX mod
s)
 22. (serial)  23. (smpar)  24. (dmpar)  25. (dm+sm)   INTEL (ifort/icc): SGI MPT
 26. (serial)  27. (smpar)  28. (dmpar)  29. (dm+sm)   INTEL (ifort/icc): IBM POE
 30. (serial)                31. (dmpar)                PATHSCALE (pathf90/pathcc)
 32. (serial)  33. (smpar)  34. (dmpar)  35. (dm+sm)   GNU (gfortran/gcc)
 36. (serial)  37. (smpar)  38. (dmpar)  39. (dm+sm)   IBM (xlf90_r/cc_r)
 40. (serial)  41. (smpar)  42. (dmpar)  43. (dm+sm)   PGI (ftn/gcc): Cray XC CLE
 44. (serial)  45. (smpar)  46. (dmpar)  47. (dm+sm)   CRAY CCE (ftn/gcc): Cray XE and XC
 48. (serial)  49. (smpar)  50. (dmpar)  51. (dm+sm)   INTEL (ftn/icc): Cray XC
 52. (serial)  53. (smpar)  54. (dmpar)  55. (dm+sm)   PGI (pgf90/pgcc)
 56. (serial)  57. (smpar)  58. (dmpar)  59. (dm+sm)   PGI (pgf90/gcc): -f90=pgf90
 60. (serial)  61. (smpar)  62. (dmpar)  63. (dm+sm)   PGI (pgf90/pgcc): -f90=pgf90

Enter selection [1-63] : 15
```

Figura 3.12: Elección de compilador para WRF.

Para comprobar que la compilación se realizó adecuadamente, se verificó que existieran los archivos ejecutables *wrf.exe*, *real.exe*, *ndown.exe* y *tc.exe* en los directorios *WRFV3/main*, *WRFV3/run* y *WRFV3/test/em_real*, con la instrucción: $ls \sqcup -ls \sqcup main/*.exe$, en la figura 3.13 se muestran los archivos ejecutables en los tres directorios.

```
[usuario@frontend WRFV3]$ ls -ls main/*.exe
42932 -rwxrwxr-x 1 usuario usuario 43960362 dic 11 05:51 main/ndown.exe
43028 -rwxrwxr-x 1 usuario usuario 44060637 dic 11 05:51 main/real.exe
42368 -rwxrwxr-x 1 usuario usuario 43381660 dic 11 05:51 main/tc.exe
48724 -rwxrwxr-x 1 usuario usuario 49889808 dic 11 05:49 main/wrf.exe
[usuario@frontend WRFV3]$ ls -ls run/*.exe
0 lrwxrwxrwx 1 usuario usuario 17 dic 11 05:51 run/ndown.exe -> ../main/ndown.exe
0 lrwxrwxrwx 1 usuario usuario 16 dic 11 05:51 run/real.exe -> ../main/real.exe
0 lrwxrwxrwx 1 usuario usuario 14 dic 11 05:51 run/tc.exe -> ../main/tc.exe
0 lrwxrwxrwx 1 usuario usuario 15 dic 11 05:49 run/wrf.exe -> ../main/wrf.exe
[usuario@frontend WRFV3]$ ls -ls test/em_real/*.exe
0 lrwxrwxrwx 1 usuario usuario 20 dic 11 05:51 test/em_real/ndown.exe -> ../../main/ndown.exe
0 lrwxrwxrwx 1 usuario usuario 19 dic 11 05:51 test/em_real/real.exe -> ../../main/real.exe
0 lrwxrwxrwx 1 usuario usuario 17 dic 11 05:51 test/em_real/tc.exe -> ../../main/tc.exe
0 lrwxrwxrwx 1 usuario usuario 18 dic 11 05:51 test/em_real/wrf.exe -> ../../main/wrf.exe
[usuario@frontend WRFV3]$
```

Figura 3.13: Archivos ejecutables generados tras la compilación de WRF.

3.3.9. Configuración de WPS

En el directorio *home/usuario/aplicacion/Build_WRF* se descargó el archivo de WPS y descomprimió con las instrucciones:

gunzip WPSV3.7.TAR.gz

tar -xf WPSV3.7.TAR

Al descomprimir se generó el directorio WPS al cual se accedió y se declararon las siguientes variables (estas variables, al igual que las anteriores se guardaron en el *.bashrc*):

$export \sqcup JASPERLIB =$ "$\$DIR/grib2/lib$"

$export \sqcup JASPERINC =$ "$\$DIR/grib2/include$"

Se inició la configuración ejecutando *./configure*, lo que arrojó una serie de opciones para elegir el tipo de compilador, como se puede ver en la siguiente figura, en éste trabajo se eligió la **opción 20** que contiene el compilador de ***Intel-dmpar*** y ***NO_GRIB2*** que es el formato compatible con los datos geográficos con los que se trabajará.

Antes de compilar WPS, se modificó el archivo *configure.wps* que se encuentra en el directorio *home/usuario/aplicacion/Build_WRF/WPS*, en dicho archivo se agregó la ruta del directorio WRFV3. La siguiente línea muestra cómo se configuró la ruta y en el apéndice E se muestra el archivo completo.

$WRF\_DIR = /home/usuario/aplicacion/Build\_WRF/WRFV3$

La compilación se realizó ejecutando la instrucción:

$./compile \sqcup > \& \sqcup log.compile$

En esta ocasión la compilación no tardó más de cinco minutos y para comprobar que se hizo de manera adecuada, se verificó que en el directorio *WPS* se hubieran generado los ejecutables *geogrid.exe*, *ungrib.exe* y *metgrid.exe*. Con la línea *ls -ls *.exe* se listaron los archivos ejecutables que se generaron en la compilación, en la tabla 3.4 se muestran los archivos ejecutables y la ruta en las que se generaron.

Archivo	Ruta
geogrid.exe	/home/usuario/aplicacion/Build_WRF/WPS/
metgrid.exe	/home/usuario/aplicacion/Build_WRF/WPS/
ungrid.exe	/home/usuario/aplicacion/Build_WRF/WPS/

Tabla 3.4: Archivos ejecutables de WPS y su ubicación.

3.4. Datos

El modelo WRF necesita de dos tipos de datos, datos geográficos y datos meteorológicos que son utilizados durante el pre-procesamiento llamado WPS y que es formado por los programas metgrid, geogrid y ungrib. A continuació,n se describe cómo se obtuvieron estos datos.

3.4.1. Datos geográficos

Para que el modelo WRF pueda trabajar con casos reales se necesita la ubicación física de una región del mundo y sus datos estáticos, para lo cual se requieren datos topográficos y sobre las categorías del uso de la tierra, además, estos datos son necesarios para el programa *geogrid*. Los datos se encuentran en la referencia [27] y su descarga duró alrededor de tres a cuatro horas, y se requirió de suficiente espacio en el disco para los datos, ya que al des-

comprimirse tienen un tamaño aproximado de 51 GB. La descarga de los datos requeridos se hizo en el directorio *Build_WRF* y se descomprimieron con la instrucción:
$-xjvf \sqcup geog\_complete.tar.bz2$

Al descomprimir los datos se generó el directorio *erod* el cual se renombró como WPS_GEOG, y a él se movieron los demás archivos que se descomprimieron.

En el archivo *namelist.wps* del directorio *WPS* se modificó la línea que indica la ruta de los datos geográficos. La siguiente línea muestra como quedó modificada y en el apéndice F se encuentra el archivo completo.
$geog\_data\_path =' /home/usuario/aplicacion/Build\_WRF/WPS\_GEOG'$

3.4.2. Datos en tiempo real

Los datos meteorológicos necesarios para el programa *ungrib* se descargaron en un nuevo directorio llamado *DATA*, ubicado en *Build_WRF*. Estos datos se obtienen a través de The National Centers for Environmental Prediction (NCEP), que ejecuta el Global Forecast System (GFS) a las 00, 06, 12 y 18 hrs de cada día. La descarga de estos datos se realizó accediendo al servidor mediante línea de comando y con usuario anónimo. En la siguiente línea se muestra el formato para descargar los datos del 06 de diciembre a las 00 hrs.

$curl \sqcup -s \sqcup -- disable - epsv \sqcup -- connect - timeout \sqcup 30 \sqcup -m \setminus$
$\sqcup 60 \sqcup -u \sqcup anonymous : USER\_ID@INSTITUTION \sqcup -o \sqcup GFS\_\mathbf{00}h \setminus$
$\sqcup ftp : //ftpprd.ncep.noaa.gov/pub/data/nccf/com/gfs/prod \setminus$
$/gfs.20151206\mathbf{06}00/gfs.t00z.pgrb2.0p50.f0\mathbf{00}$

Para los demás datos se modificaron las partes en negritas de la instrucción anterior, el **06** se modificó por 07, 08, 09 10, 11 y 12, que representan los días, Los **00** se cambiaron por 06, 12 y 18 que representan las horas, y el nombre del archivo descargado, identificado como GFS_00h se modificó por $GFS\_00h\_dd-mm-aa$, donde $dd-mm-aa$ representa la fecha en formato día-mes-año, esto con el objeto de diferenciar con claridad cada dato.
Para este trabajo, en concreto se descargaron los datos GFS de una semana. Los días y las horas de los datos descargados se muestran en la tabla 3.5.

Día	00hrs	06hrs	12hrs	18hrs
06 de diciembre de 2015	sí	sí	sí	sí
07 de diciembre de 2015	sí	sí	sí	sí
08 de diciembre de 2015	sí	sí	sí	sí
09 de diciembre de 2015	sí	sí	sí	sí
10 de diciembre de 2015	sí	sí	sí	sí
11 de diciembre de 2015	sí	sí	sí	sí
12 de diciembre de 2015	sí	sí	sí	sí

Tabla 3.5: Fechas y horas de los datos GFS.

3.5. Ejecución de WPS

Antes de iniciar con la ejecución de WPS y WRF, se deben tener los datos de los días a procesar además de las coordenadas de la zona geográfica que se quiere trabajar y el número de anidados en dicha zona. En la figura 3.14 se muestra la zona sobre la que se ejecutará el modelo WRF y sus anidados.

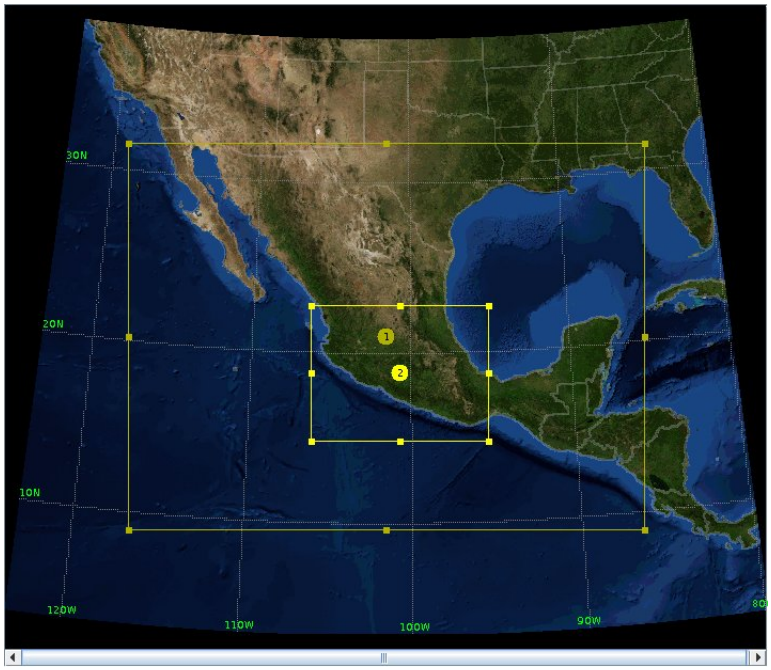

Figura 3.14: Zona geográfica.

- Se modificó el archivo *namelist.wps* (este archivo ya había sido modificado en un paso anterior, en el apéndice F se encuentra el archivo con todas las modificaciones necesarias) del directorio *WPS* con base a las fechas de los datos descargados para este trabajo y con las coordenadas apropiadas para que el cálculo del modelo WRF se realizara sobre territorio mexicano.

- Se ejecutó el archivo *geogrid* con la línea:

  $./geogrid.exe \sqcup > \& \sqcup log.geogrid$

  El resultado se pudo visualizar con la instrucción:

  cat log.geogrid

  y al final del archivo se mostraba el siguiente mensaje:

  !!!Successful completion of geogrid. !!!

Este proceso creo archivos con prefijo *geo_em**.

Los primeros intentos de ejecución del archivo *geogrid.exe* no envió mensaje de éxito ya que faltaban algunos datos geográficos, pero aparecía un mensaje de error indicando los archivos faltantes, los cuales se descargaron y descomprimieron de manera individual en el directorio WPS_GEOG.

- Se realizó una liga a los archivos GFS con la instrucción:

  $./link\_grib.csh \sqcup /home/usuario/aplicacion/Build\_WRF/DATA/$

  y otra liga al archivo Vtable con:

  $ln \sqcup -sf \sqcup ungrib/Variable\_Tables/Vtable.GFS \sqcup Vtable$

- Se ejecutó el archivo *ungrib* con la instrucción:

  $./ungrib.exe \sqcup > \& \sqcup log.ungrib$

  El resultado se pudo visualizar con la instrucción:

  cat log.ungrib

  Al final del archivo se mostraba el siguiente mensaje:

  !!!Successful completion of ungrib. !!!

  Además, se crearon archivos con el prefijo *FILE*.

- Se ejecutó el archivo *metgrid* con la instrucción:

  $./metgrid.exe \sqcup > \& \sqcup log.metgrid$

  El resultado se pudo visualizar con la línea:

  cat log.metgrid

  Al final del archivo se mostraba el siguiente mensaje:

  !!!Successful completion of metgrid. !!!

  Esto generó archivos con el prefijo *met_em**.

3.6. Ejecución de WRF

- WRF se puede ejecutar desde el directorio */usuario/aplicacion/Build_WRF/WRFV3/run* o desde*/usuario/aplicacion/Build_WRF/WRFV3/test/em_real*, lo que es indistinto, pero en este trabajo se eligió el directorio *run*.

- Se modificó el archivo *namelist.input* con la información particular de este trabajo, es decir, se modificaron las fechas y las coordenadas. El archivo completo se encuentra en el apéndice G.

- Se hizo una copia de los archivos *met_em∗* al directorio *run*, con la instrucción:
  $cp \sqcup /home/usuario/aplicacion/Build\_WRF/WPS/met\_em*.$

- Se ejecutó el programa real con la línea:
  $mpirun \sqcup -np \sqcup 1 \sqcup ./real.exe$
  lo que generó dos archivos con prefijo *rsl*.

- Se verificó el contenido del archivo *rsl.error.0000* usando la instrucción:
  tail rsl.error.0000
  El archivo contenía el mensaje:
  "SUCCESS COMPLETE REAL_EM INIT"

- Finalmente se ejecutó WRF con base a la siguiente línea:
  time mpirun -np ***xx*** *-machinefile machines./wrf.exe*
  Donde:

- El comando *time* se utilizó para medir el tiempo de ejecución de cada prueba.

- *xx* representa el número de procesos a realizar en cada prueba.

- El comando *-machinefile*, indica que en la ejecución se usará el archivo *machines* en el que se incluyen los hostnames de los cuatro nodos.
- *./wrf.exe* es el archivo ejecutable para procesar WRF.

Con la ejecución de WRF se terminó la configuración del clúster, a partir de ello, se llevaron a cabo diversas pruebas, tanto de Linpack como del modelo WRF, el resultado de dichas pruebas se analiza en el siguiente capítulo.

Capítulo 4

Análisis de resultados

En el presente capítulo se muestran los resultados obtenidos de las pruebas realizadas en el clúster Xexelo0, en el que se ejecutó el modelo de predicción climática WRF y la prueba de rendimiento Linpack.

4.1. Resultados del modelo WRF

Con la configuración del software lista, se realizaron múltiples pruebas del modelo WRF, en la figura 4.2 se muestra la gráfica del tiempo que tardó cada proceso, se tomaron los tiempos de ejecución desde 2 hasta 100 procesos y cada proceso se repitió 5 veces para observar las variaciones de tiempo, al terminar cada ejecución el modelo arrojó como respuesta un par de archivos con prefijo $wrfout$ que corresponden a los dominios 1 y 2, señalados en el capítulo anterior.

Dado que los servidores de Xexelo0 sólo se manejan a través de línea de comandos, en la máquina de administración remota se instaló el programa *ncview* para visualizar gráficamente los archivos de salida *wrfout*, los cuales se copiaron del clúster a la máquina de administración remota. En la figura 4.1 se muestra el ambiente de *ncview*.

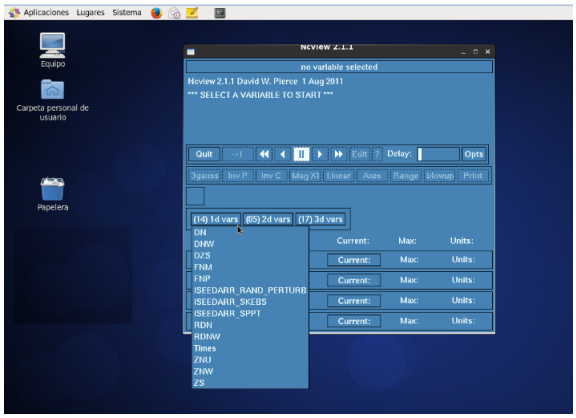

Figura 4.1: Ambiente ncview.

En la gráfica de la figura 4.2, las líneas de las series 1, 2, 3, 4 y 5, representan los tiempos registrados en cinco corridas diferentes, la línea de la serie 6 es el promedio de las cinco corridas, como se puede ver no existieron diferencias notorias en el tiempo de ejecución.

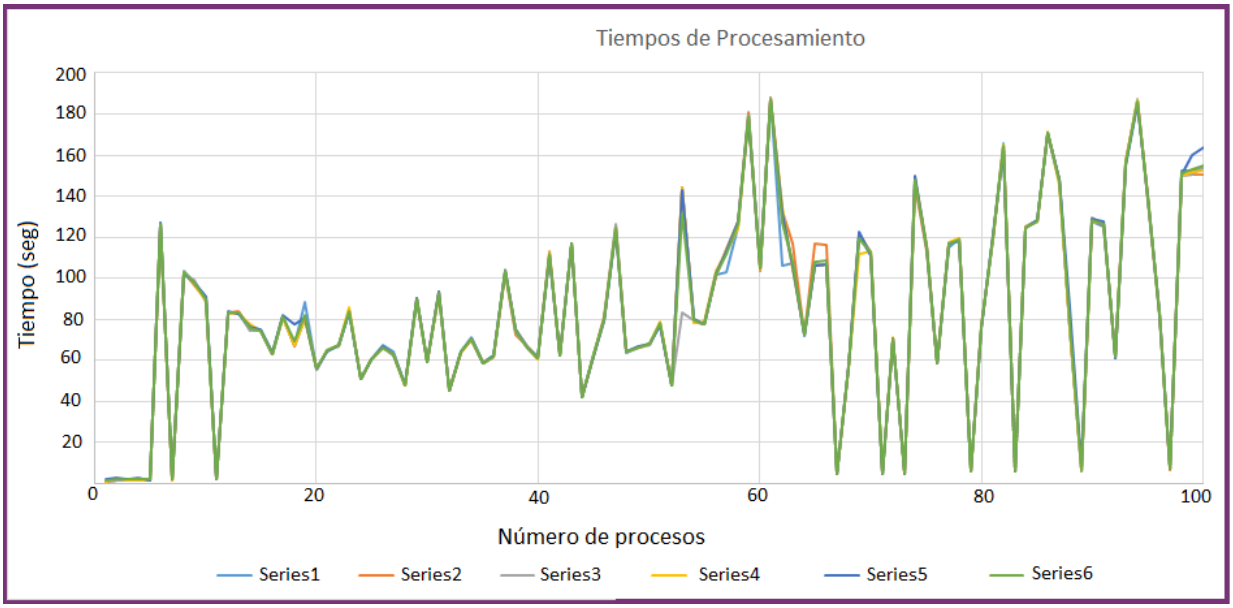

Figura 4.2: Gráfica de los tiempos de ejecución del modelo WRF.

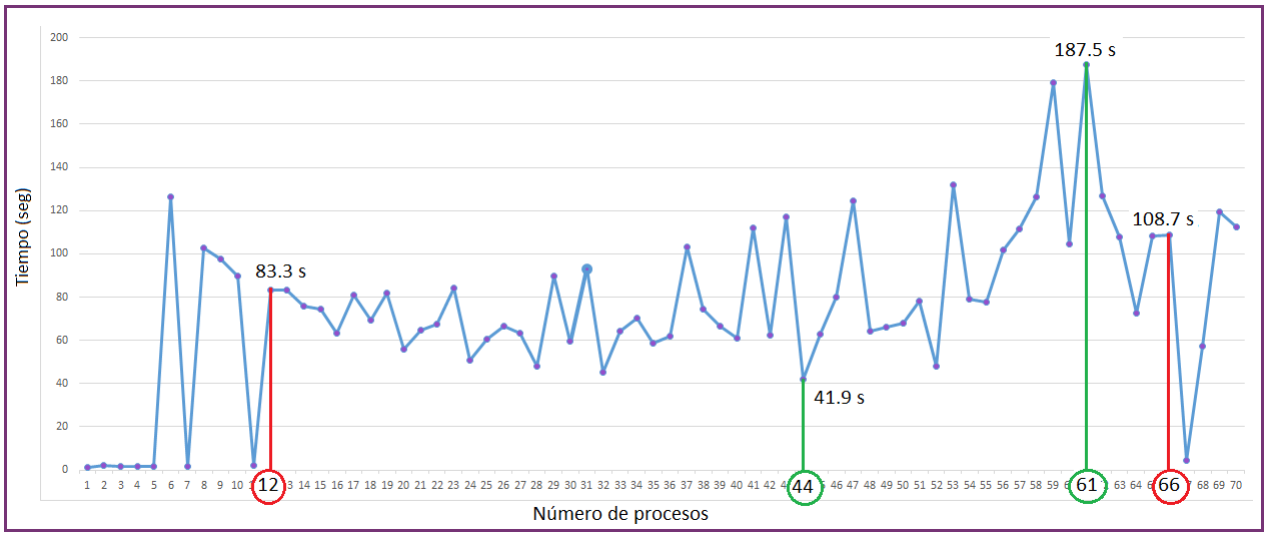

Figura 4.3: Gráfica de los tiempos promedio de la ejecución del modelo WRF.

En la figura 4.3 se muestra la gráfica de los tiempos promedio, con esta referencia el valor de tiempo máximo fue de 187.5 segundos y el tiempo mínimo fue de 41.9 segundos, es decir, que en este clúster conviene ejecutar el modelo WRF dividido en 44 procesos ya que con esa cantidad de procesos el clúster realiza el procesamiento más rápido. En la gráfica también se puede observar que hay valores muy pequeños en el tiempo de procesamiento, antes de los 12 procesos y después de los 66 procesos, estos valores no son tomados como válidos ya que al terminar la ejecución en los tiempos mínimos antes de los 12 procesos el archivo de salida fue de 11 MB y en los tiempos mínimos después de los 66 procesos, no hay archivo de salida, mientras que en los demás procesos el archivo de salida $wrfout$ fue de 190 MB. También se compararon los archivos de salida con el ncview, y se observó que los archivos de 11 MB generan la predicción de algunas variables climáticas, pero falta el cálculo de la mayor parte de ellas, lo cual implica que para este clúster conviene dividir el procesamiento de WRF mínimo en 12 procesos y máximo en 66 procesos.

Tomando en cuenta que en un nodo se pueden ejecutar hasta 24 tareas paralelas, el procesa-

miento de WRF necesitaría sólo dos nodos de Xexelo0.

Con las gráficas de las figuras 4.2 y 4.3 se muestra que en esta aplicación el comportamiento en tiempo no es el esperado, ya que idealmente el tiempo de ejecución tiende a disminuir conforme al aumento del número de procesos, este comportamiento se puede atribuir a las características propias de la aplicación (este tipo de resultados ya fueron registrados en un trabajo anterior [28]), dado que WRF no es un programa lineal e implica en esta configuración particular, el cálculo de 260 variables climáticas.

WRF presenta sus cálculos por medio de mapas, los resultados muestran de lado derecho el mapa de la región geográfica sobre la cual se realizó el cálculo climático y de lado izquierdo se muestra la escala a colores que corresponden al mapa. En las figuras 4.4 y 4.5 se encuentra el cálculo de WRF para la variable *temperatura* para el primer y segundo anidado respectivamente. Cabe señalar que el primer anidado corresponde a prácticamente todo el territorio nacional, mientras que el segundo anidado hace un acercamiento a la zona central del país.

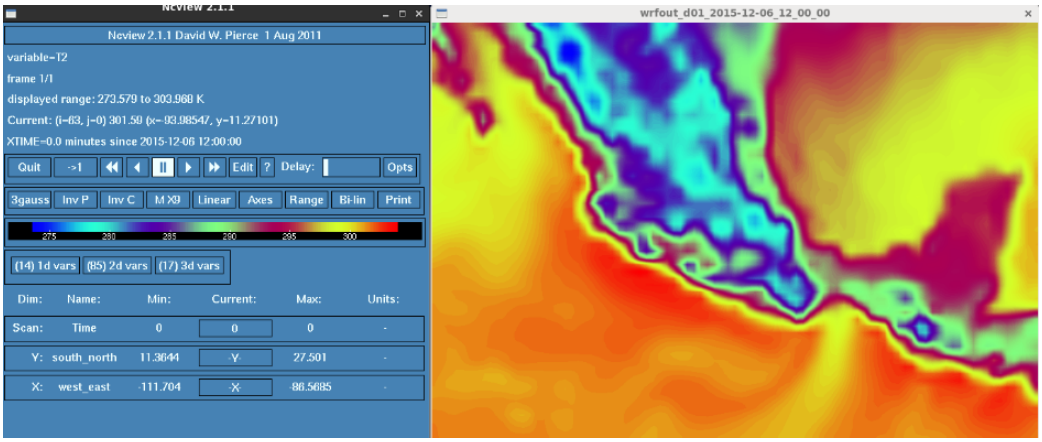

Figura 4.4: Cálculo de temperatura para el primer anidado.

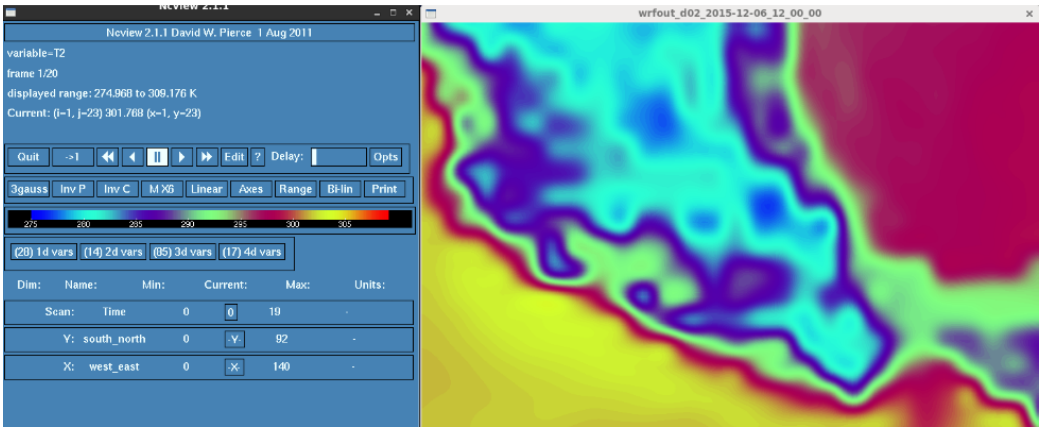

Figura 4.5: Cálculo de temperatura para el segundo anidado

En las figuras 4.6 y 4.7 se encuentra el cálculo de WRF para la variable *vapor de agua* para el primer y segundo anidado respectivamente. Aunque estos resultados se parecen a los de la variable *temperatura*, es importante notar las diferencias de los colores.

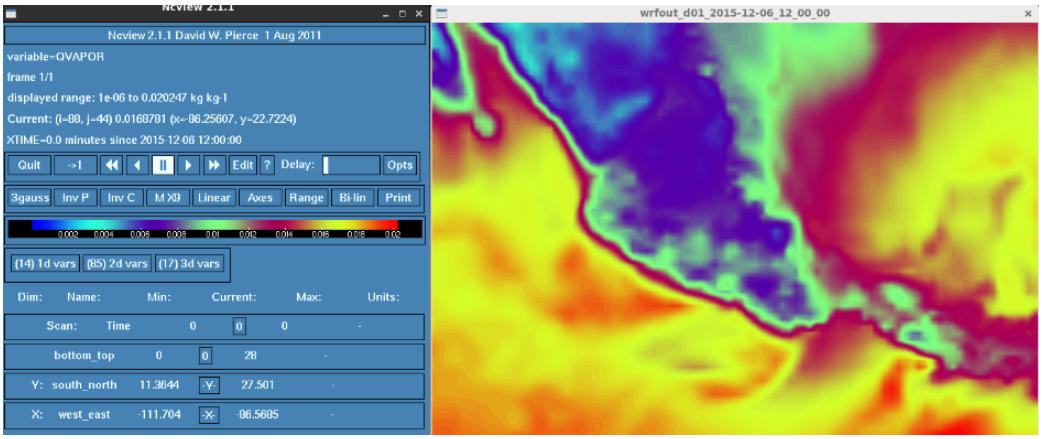

Figura 4.6: Cálculo de vapor de agua para el primer anidado.

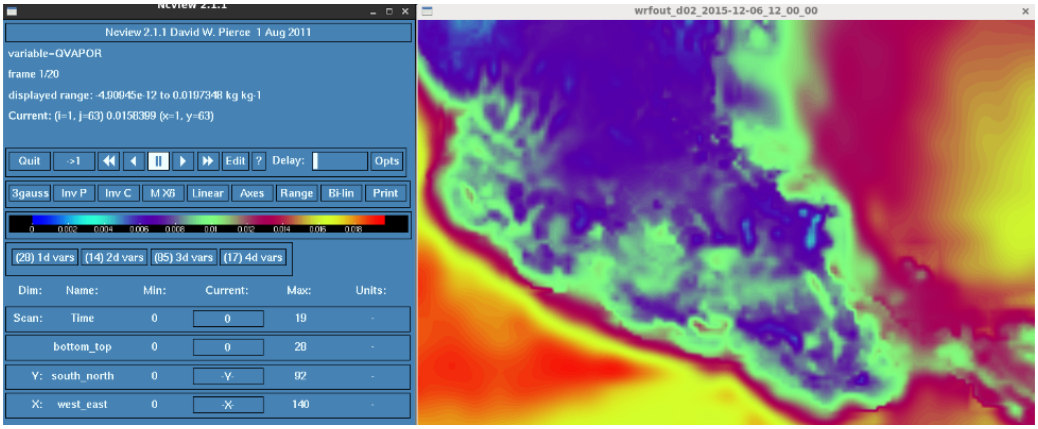

Figura 4.7: Cálculo de vapor de agua para el segundo anidado.

Al ejecutar el modelo WRF, en la terminal o consola se despliega una lista de los procesos en los que se ha dividido la ejecución actual, en la figura 4.8 se muestra parte de una lista en la que se enumeran los 96 procesos de una ejecución del modelo WRF.

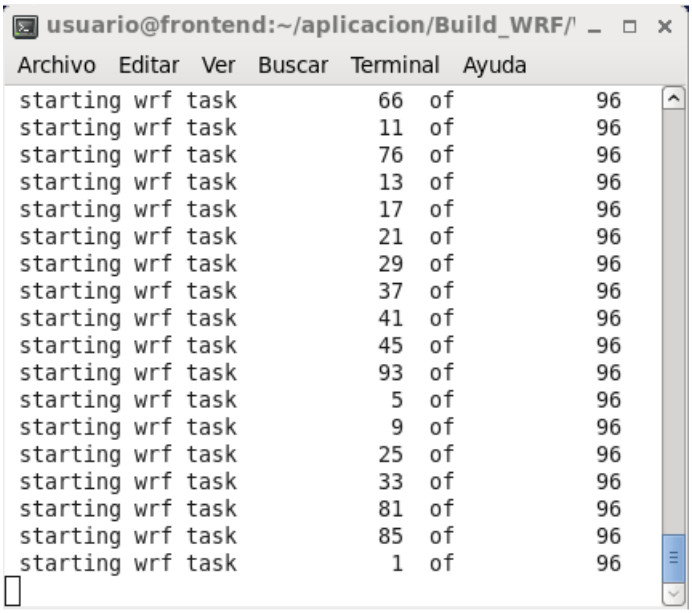

Figura 4.8: Visualización del número de procesos en los que se divide una ejecución.

Para comprobar que el clúster utilizaba los nodos indicados en cada ejecución, se usó el comando *htop* para visualizar la saturación de los procesadores. En las figuras 4.9 y 4.10 se muestra el estado de los procesadores del frontend y el host01 respectivamente, en la parte superior de las figuras se ve a los 24 procesadores activos y saturados al 100 %, en la parte inferior se muestran los números que identifican a cada proceso y el programa que se está ejecutando, en este caso el programa *wrf.exe*.

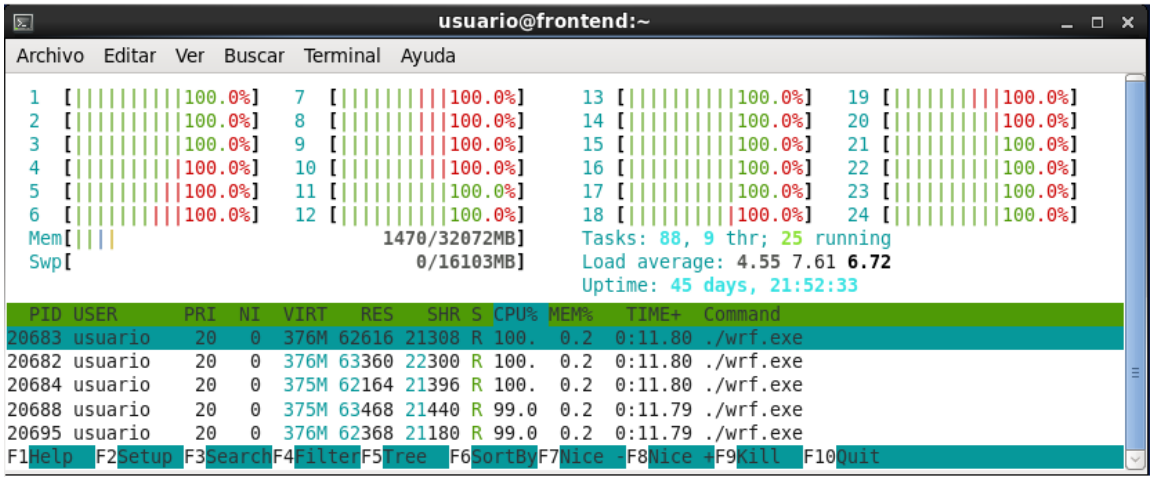

Figura 4.9: Visualización de los 24 procesadores del frontend saturados al 100 %.

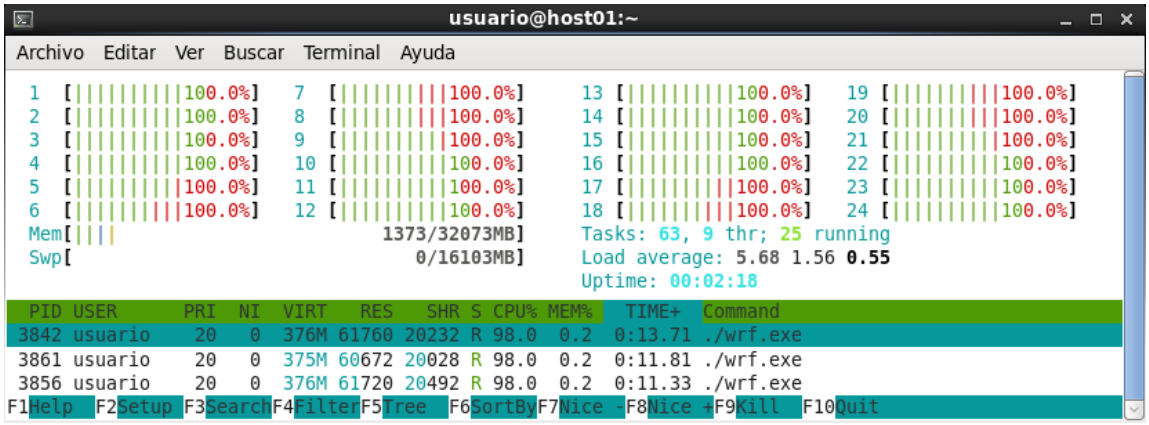

Figura 4.10: Visualización de los 24 procesadores del host01 saturados al 100 %.

4.2. Prueba de rendimiento Linpack

Con la prueba de Linpack se calculó el rendimiento del clúster Xexelo0, esta prueba se realizó con ayuda de las herramientas incluidas en el software de Intel, en seguida se describe el proceso con el que se llevó a cabo la prueba de Linpack.

En el usuario *usuario* se accedió a la ruta $/opt/intel/compilers\_and\_libraries\_2016/linux/mkl/benchmarks/mp\_linpack$ y se ejecutó la instrucción $make \sqcup arch = intel64$ para generar los archivos $HPL.dat$ y $xhpl$ en la ruta $/opt/intel/compilers\_and\_libraries\_2016/linux/mkl/benchmarks/mp\_linpack/bin/intel64$

Los archivos $HPL.dat$ y $xhpl$ son importantes porque con el primero se puede adecuar la prueba de Linpack con base en las características de Xexelo0 y el segundo es el archivo ejecutable para realizar la prueba de Linpack.

El archivo $HPL.dat$ se configuró para indicar la cantidad de nodos, los núcleos por nodo y la cantidad de memoria RAM (ver referencia [29]). Una vez modificado el archivo se ejecutó la instrucción:

$mpiexec.hydra - np \sqcup 96 - machinefile \sqcup machines./xhpl$

El número 96 que indica el número de procesos, es igual a la cantidad de procesadores totales del clúster; $-machinefile \sqcup machines$ se usa para indicar que se usará el archivo; $machines$ que contiene las direcciones ip de los cuatro nodos y $./xhpl$ es el archivo ejecutable para llevar a cabo la prueba de linpack.

El resultado de la prueba arrojó un documento en el que se indican los tiempos de inicio y fin de la prueba, además del rendimiento en Gflops. Dado que los tiempos de ejecución pueden variar, la prueba de Linpack se realizó cinco veces, los rendimientos obtenidos se muestran

en la figura 4.11, en el apéndice H se muestra el archivo *HPL.dat* configurado con las características del clúster Xexelo0 y uno de los archivos de salida que indican el rendimiento.

A partir de los rendimientos registrados, se calculó un rendimiento promedio de 657.377 Gflops, con una desviación estándar de 0.495 Gflops. Cabe señalar que esta prueba tardo de 26 a 27 minutos en ejecutarse.

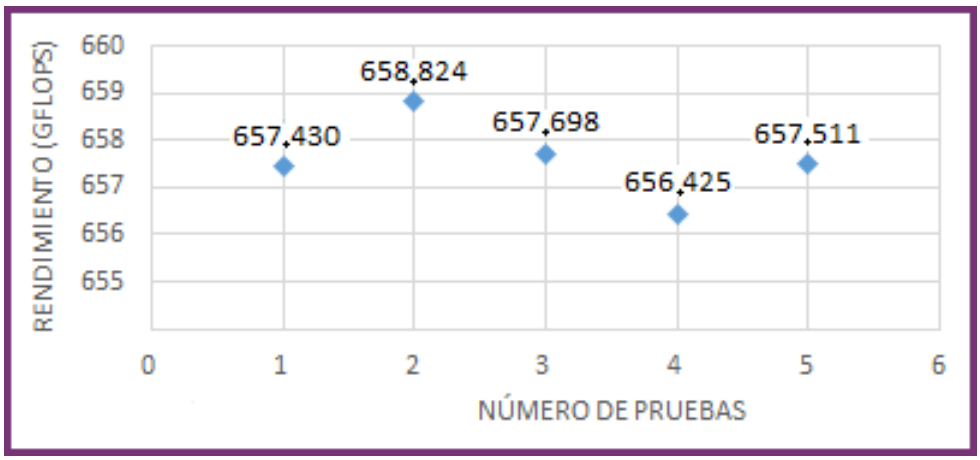

Figura 4.11: Rendimiento en Gflops del clúster con base a la prueba de linpack.

4.2.1. Otras pruebas

Además de ejecutar el modelo WRF, se ejecutó la multiplicación de dos matrices cuadradas de 6000 por 6000 números [30], esto se hizo con la finalidad de observar los tiempos de ejecución de otro programa y así poder calcular el factor de aceleración. En la tabla 4.1 se muestran los tiempos de ejecución para diferentes procesos y su correspondiente factor de aceleración, el cual se calculó con la ecuación 2.1, en dicha tabla se muestra que el menor tiempo de ejecución y mayor factor de aceleración se presentó al dividir la ejecución de la multiplicación de dos matrices en 85 procesos. En la figura 4.12 se muestra la gráfica de tiempo vs número de procesos que se muestran en la tabla.

Procesos	Tiempo [s]	Factor de aceleración	Procesos	Tiempo [s]	Factor de aceleración
1	1824.144	1	55	347.090	5.255
2	1956.594	0.932	60	339.729	5.369
3	1075.651	1.695	65	332.809	5.481
5	721.105	2.529	80	301.449	6.051
10	468.204	3.896	**85**	**290.922**	**6.270**
25	404.427	4.510	90	297.880	6.123
30	375.513	4.857	95	417.351	4.370
35	360.164	5.064	96	435.577	4.187
40	351.276	5.192	100	676.991	2.694

Tabla 4.1: Número de procesos vs tiempo vs factor de aceleración al ejecutar la multiplicación de dos matrices.

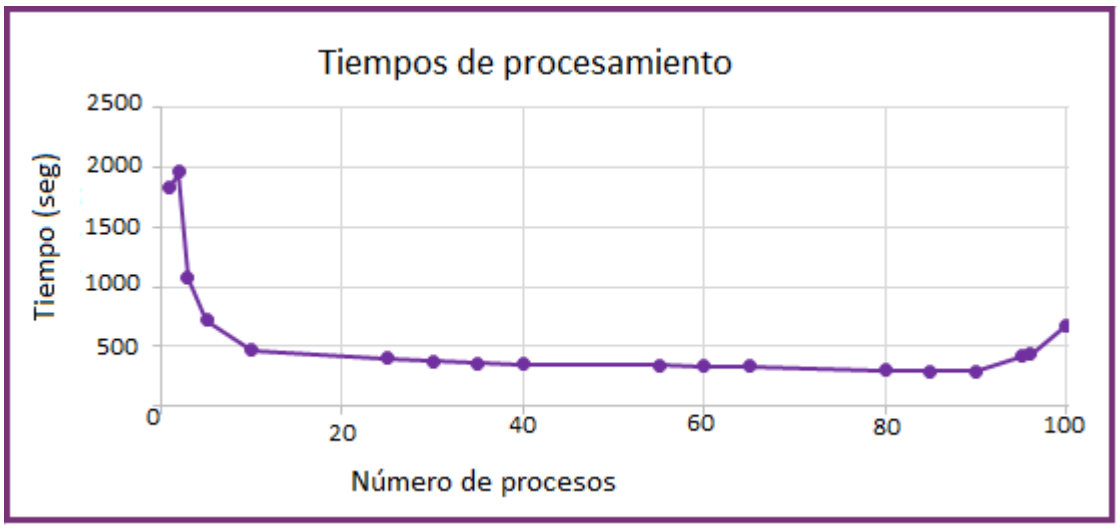

Figura 4.12: Número de procesos vs tiempo de ejecución en multiplicación de dos matrices.

Con base en los resultados presentados y a las experiencias que se presentaron durante la fase de desarrollo y pruebas, en el siguiente capítulo se ofrecen las conclusiones de este trabajo.

Capítulo 5

Conclusiones y propuestas de trabajo futuro

En este trabajo, se implementó un clúster Beowulf con cuatro servidores Supermicro en el cual una tarea se puede dividir en procesos paralelos, cada uno de los servidores cuenta con dos procesadores Intel Xeon y la red de procesamiento se conecta a través de un switch Netgear con capacidad de transferir datos a 10 Gbps.
El clúster fue denominado *Xexelo0*, y forma parte del proyecto *Laboratorio de sistemas distribuidos y de redes de alto desempeño.*

Respecto al software, en los servidores se instaló el sistema operativo de distribución libre *CentOS*, el cual les permite un funcionamiento estable, además se instaló software de Intel que permite mejorar el rendimiento de sus procesadores en aplicaciones paralelas. El firewall también es parte del software libre y brinda seguridad al sistema, ya que *pfSense* fue configurado para restringir el acceso remoto a la red del clúster, de modo que sólo ciertos usuarios pueden tener acceso.

El clúster Xexelo0, ejecuta códigos paralelos escritos en lenguaje C, C++, java y fortran ya que en él se ejecuta sin inconvenientes el programa de predicción climática WRF que integra bibliotecas escritas en los lenguajes mencionados, los cuales trabajan con base a la lectura de datos binarios y código ASCII.

Dado lo anterior, se obtienen las siguientes conclusiones:

- La gráfica de los tiempos obtenidos en el procesamiento de WRF, muestra que no siempre que se utilicen más procesos para dividir una tarea se obtienen necesariamente mejores tiempos de procesamiento.
- Dependiendo de la aplicación, el número de procesos en los que se debe dividir un problema para obtener un mejor rendimiento es variable y queda en función del usuario.
- El rendimiento de Xexelo0 comparado con el rendimiento del último lugar del TOP500 de junio de 2016 es muy pequeño, ya que se necesitarían 2 mil clústers tipo Xexelo0 para alcanzar el rendimiento del lugar 500 de dicho top, aun así, es suficiente para ejecutar tareas y/o aplicaciones que requieren ser paralelizadas de modo que puede ofrecer sus capacidades a la comunidad académica de la UACM.

Con estas conclusiones se da por finalizado el presente trabajo, en el que se diseñó e implementó un clúster en el que se realiza cómputo distribuido y paralelo utilizando software libre, y se propone el siguiente trabajo futuro:

- Instalar el servicio de información de red (Network Information Server - NIS) para volver eficiente el sistema de creación y administración de usuarios.
- Realizar el monitoreo de los servidores de Xexelo0 a través de la red IPMI.

- Continuar el trabajo con WRF en Xexelo0, agregando más de dos anidados para saturar el clúster y registrar nuevos tiempos de ejecución.

- Averiguar y justificar por qué después de dividir una tarea en 60 procesos, en ciertas ejecuciones el clúster no realiza el procesamiento del modelo WRF.

- Instalar en Xexelo0 otra aplicación que requiera procesamiento en paralelo para comparar los tiempos del rendimiento.

Referencias

[1] Román. G. (2006). Conceptos básicos [Material de clase]. Programación Distribuida, Universidad Autónoma Metropolitana, Ciudad de México.

[2] RICM. (). *Dec Vax 11/780* [En línea]. <http://www.ricomputermuseum.org/Home /equipment/dec-vax-11-780>. [Consulta: mayo, 2014].

[3] Aguilar. L. (1992). *El problema de N-Cuerpos en la Astronomía.* Revista Mexicana de Física. Instituto de astronimía. UNAM. [En línea]. <http://www.am.ub.edu/∼sroca/Nbody/documents/Aguilar92_RevMexFis38_701 _Nbody.pdf>.[Consulta: abril, 2014].

[4] Castillo. J. (2016). *Redes de datos: Contexto y evolución.* Ciudad de México. Segunda Ed. Samsara. 196p.

[5] Colouris. G., Dollimore. J., Kindberg. T. (2001). *Sistemas Distribuidos. Conceptos y diseño.* Tercera ed. Madrid. Pearson Educación. 744p.

[6] Tanenbaum. A., Vann Steen. M.(2008). *Sistemas Distribuidos, principios y paradigmas.* Segunda ed. México. Prentice Hall.

[7] Milone. D., Azar. A., Rufiner. L. (2002). *Supercomputadoras basadas en "clusters" de PCs.* [En línea]. <http://fich.unl.edu.ar/sinc/sinc-publications/2002/MAR02/sinc_MAR02.pdf>. [Consulta: marzo, 2014].

[8] Hernández. L., Santillán. A., Caballero. R. (2003). *Breve historia del origen de los clusters beowulf.* [En línea]. <http://www.revista.unam.mx/vol.4/num2/art3/clustb.htm>. [Consulta: mayo, 2014].

[9] Zamareño. J.M., Alvarez. M.T., Acebes. L.F., García. M.A., Tadeo. F.J. (2002). *Fundamentos de Informática y Programación Científica. Resolución en C y Matlab.* Ed. Minor Network.

[10] Velázquez. A., Saynez. J. C. ().*Historia de Linux.* [En Línea]. <http://estigia.fib.unam.mx/chontalpa/unix/CursoUnix.pdf>. [Consulta: Marzo, 2015].

[11] Red Hat. (2016). *Plataformas Linux. Red Hat Enterprise Linux.* [En línea]. <https://www.redhat.com/es/technologies/linux-platforms/enterprise-linux>. [Consulta: mayo, 2015].

[12] CentOS, (2014). *The CentOS Project.* [En línea]. <http://www.centos.org/>. [Consulta: marzo, 2014].

[13] Mpich. (2016). *MPICH.* [En línea]. <https://www.mpich.org/>. [Consulta: noviembre, 2015].

[14] Miguel. J. (1997). *Programación de aplicaciones paralelas con MPI (Message Passing Interface)* [En línea]. <http://www.sc.ehu.es/acwmialj/edumat/mpi.pdf>. [Consulta: mayo, 2014].

[15] Pardo. F. (2002). *Arquitecturas avanzadas.* Universidad de Valencia. España.

[16] Jiménez. D. (). *Multiprocesadores y multicomputadores.* [En línea]. <https://www.exabyteinformatica.com/uoc/Informatica/Arquitecturas_de_computadores_avanzadas/Arquitecturas_de_computadores_avanzadas_%28Modulo_2%29.pdf>. [Consulta: octubre, 2016].

[17] Aguilar. J., Leiss. E. (2004). *Introducción a la computación paralela.* Primera ed. Universidad de Los Andes, Venezuela. Graficas Quinteto. 246p.

[18] Top 500. (2016). *Top 500 the list.* [En línea]. <http://www.top500.org/>. [Consulta: julio, 2016].

[19] Oviedo. B.(2011). *Caracterización de fenómenos meteorológicos.* [En línea]. <http://datateca.unad.edu.co/contenidos/358026/358026/CFM_CORE/modulo_fenomenos_meteorologicos.pdf>. [Consulta: noviembre, 2015].

[20] Solano. L. (2014). *Modelo de pronóstico meteorológico mesoescalar.* [En línea]. <http://www.tdx.cat/bitstream/handle/10803/6836/07Ojc07de12.pdf?sequence=7>. [Consulta: febrero. 2016].

[21] WRF. (2016). *The Weather Research & Forecasting Model.* [En línea]. <http://www.wrf-model.org/index.php>. [Consulta: noviembre, 2015].

[22] Developmental Testbed Center. (2016). *WRF Preprocessing System Features.* [En línea]. <http://www.dtcenter.org/wrf-nmm/users/overview/wps_overview.php>. [Consulta: febrero, 2016].

[23] Hubbell (). *Hubbell Wiring Systems - Curso Mission Critical Warranty.*

[24] Electric Sheep Fencing. (2016). *Take a tour, Getting Started.* [En línea]. <https://www.pfsense.org/getting-started/>. [Consulta: junio, 2016].

[25] Intel (). *Zona para desarrolladores de Intel. Intel Parallel Studio XE 2016.* [En Línea]. <https://software.intel.com/es-es/intel-parallel-studio-xe>. [Consulta: noviembre, 2015].

[26] UCAR. (2016). *How to Compile WRF: The Complete Process.* [En línea]. <http://www2.mmm.ucar.edu/wrf/OnLineTutorial/compilation_tutorial.php>. [Consulta: octubre, 2015].

[27] WRF. (2016). *WRF Source Codes and Graphics Software Download Page.* [En línea]. <http://www2.mmm.ucar.edu/wrf/users/download/get_sources_wps_geog.html>. [Consulta: noviembre, 2015].

[28] Gualán. R.M., Solano. L.D. (2014). *Análisis de rendimiento y profiling del modelo WRF en un clúster HPC.* [En línea]. <http://dspace.ucuenca.edu.ec/bitstream/123456789/21399/1/TIC.EC_13_Gual%C3%A1n%20%26%20Solano.pdf>. [Consulta: mayo, 2016].

[29] Advanced Clustering Technologies (2015). *How do I tune my HPL.dat file?.* [En línea]. <http://www.advancedclustering.com/act-kb/tune-hpl-dat-file/>. [Consulta: noviembre, 2014].

[30] DaniWeb (2011). *Matrix Multiplication using MPI Parallel Programming Approach.* [En línea]. <https://www.daniweb.com/programming/software-development/code/334470/matrix-multiplication-using-mpi-parallel-programming-approach>. [Consulta: marzo, 2015].

[31] Dell. (). *¿Cuál es el valor de la informática de alto rendimiento (HPC)?.* [En línea]. <http://www.dell.com/learn/es/es/rc1081323/hpcc-what-is-it>. [Consulta: marzo, 2014].

[32] Gamboa. M., Oliver. H., Velázquez. R. (2014, marzo). *Xiuhcoatl: clúster híbrido de supercómputo del cinvestav.* Conversus, 107, 4-5.

Apéndice A

Reglas Nat

Las siguientes instrucciones muestran cómo se configuraron las reglas NAT (reglas de conversion se direcciones de red) en el frontend, con la finalidad de que los nodos esclavos se conectaran a internet a través del frontend. Los parámetros entre corchetes son direcciones ip.
==========

/sbin/iptables -P INPUT ACCEPT
/sbin/iptables -F INPUT
/sbin/iptables -P OUTPUT ACCEPT
/sbin/iptables -F OUTPUT
/sbin/iptables -P FORWARD DROP
/sbin/iptables -F FORWARD
/sbin/iptables -t nat -F
/sbin/iptables -A FORWARD -i [interfaz de red local] -o [interfaz de internet] -m state –state ESTABLISHED, RELATED -j ACCEPT
/sbin/iptables -A FORWARD -i [interfaz de internet] -o [interfaz de rel local] -j ACCEPT
/sbin/iptables -t nat -A POSTROUTING -s [red local] -o [interfaz de internet] -j MASQUERADE
iptables -A FORWARD -i [interfaz de internet] -j ACCEPT
iptables -A FORWARD -o [interfaz de internet] -j ACCEPT
/sbin/iptables-save >/etc/sysconfig/iptables
/sbin/service iptables restart

Apéndice B
Repositorios

Dado que se hizo una instalación mínima de CentOS, se habilitaron los repositorios del sistema operativo que se encuentran el formato RPM.

==========

rpm –import /etc/pki/rpm-gpg/rpm-gpg-key*

rpm –import http://dag.wieers.com/rpm/packages/RPM-GPG-KEY_dag.txt

cd tmp

wget http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.2-2.2l6.rf.x86_64

rpm -ivh rpmforge-release-0.5.2-2.el6.rf.86_64

rpm –import https://fedoraproject.org/static/0608B895.txt

wget http://dl.fedoraproject.org.pub/epel/6/x8664/epel-release-6-8.noarch.rpm

rpm -ivh epel[6]-release-6-8.noarch.rpm

yum install yum-priorities

[6]Se modificó la prioridad de EPEL en el archivo *epel.repo*, ubicado en /etc/yum.reposd. En dicho archivo se agregó la línea $priority = 10$

Apéndice C

Test a los compiladores

A continuación se muestra una serie de pruebas realizadas a los compiladores de C y Fortran. Las instrucciones llevan al inicio un signo de $, lo que las distingue de los resultados obtenidos.

==========

$which gfortran
/usr/bin/gfortran

$which cpp
/usr/bin/cpp

$which gcc
/usr/bin/gcc

$gcc –version
gcc (GCC) 4.4.7 20120313 (Red Hat 4.4.7-11) Copyright (C) 2010 Free Software Foundation, INC. Esto es software libre; vea el código para las condiciones de copia. No hay garantía; ni siquiera para MERCANTIBILIDAD o IDONEIDAD PARA UN PROPOSITO EN PARTICULAR

$gfortran TEST_1_fortran_only_fixed.f
$./a.out
SUCCESS test 1 fortran only fixed format

$gfortran TEST_2_fortran_only_free.f90
$./a.out
Assume Fortran 2003: has FLUSH, ALLOCATABLE,
derived type, and ISO C Binding SUCCESS
test 2 fortran only free format

```
$gcc TEST_3_c_only.c
$./a.out
SUCCESS test 3 c only

$gcc -c -m64 TEST_4_fortran+c_c.c
$gfortran -c -m64 TEST_4_fortran+c_f.f90
$gfortran -m64 TEST_4_fortran+c_f.o TEST_4_fortran+c_c.o
$./a.out
C function called by Fortran
Values are xx = 2.00 and ii = 1
SUCCESS test 4 fortran calling c

$./TEST_csh.csh
SUCCESS csh test


$./TEST_perl.pl
SUCCESS perl test

$./TEST_sh.sh
SUCCESS sh test
```

Apéndice D

Archivo .bashrc

En el archivo *.bashrc* se añadieron las variables generadas en la instalación de los programas con los que trabaja wrf y la ruta de las bibliotecas de intel.
==========

```
# .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
. /etc/bashrc
fi

# User specific aliases and functions
export DIR=" /opt"
export CC="gcc"
export CXX="g++"
export FC="gfortran"
export FCFLAGS="–m64"
export F77="gfortran"
export FFLAGS="–m64"
export PATH="$DIR/netcdf/bin:$PATH"
export NETCDF="$DIR/netcdf"
export PATH="$DIR/mpich/bin:$PATH"
export LDFLAGS="–L$DIR/grib2/lib"
export CPPFLAGS="–I$DIR/grib2/include"
export JASPERLIB="$DIR/grib2/lib"
export JASPERINC="$DIR/grib2/include"
#additonal path:
 source /opt/intel/bin/compilervars.sh intel64
```

```
export PATH=$PATH:/opt/intel/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:
/opt/intel/composer_xe_2016/lib/intel64
```

Apéndice E

Archivo configure.wps

En el archivo *configure.wps* se modificó la ruta del directorio en el que se ubica la aplicación wrf. Se encuentra indicado con letras negritas.

==========

```
# configure.wps
#
# This file was automatically generated by the configure script in the
# top level directory. You may make changes to the settings in this
# file but be aware they will be overwritten each time you run configure.
# Ordinarily, it is necessary to run configure once, when the code is
# first installed.
#
# To permanently change options, change the settings for your platform
# in the file arch/configure.defaults, the preamble, and the postamble -
# then rerun configure.
#

.SUFFIXES: .F .f .c .o

SHELL = /bin/sh

NCARG_LIBS = -L$(NCARG_ROOT)/lib -lncarg -lncarg_gks -lncarg_c
-L/usr/X11R6/lib -lX11

NCARG_LIBS2 = # May be overridden by architecture specific value below

FDEFS = -DUSE_JPEG2000 -DUSE_PNG

# Listing of options that are usually independent of machine type.
```

When necessary, these are over-ridden by each architecture.

ARFLAGS =

PERL = perl

RANLIB = echo

WRF_DIR = /home/usuario/aplicacion/Build_WRF/WRFV3

WRF_INCLUDE = -I$(WRF_DIR)/external/io_netcdf
-I$(WRF_DIR)/external/io_grib_share
-I$(WRF_DIR)/external/io_grib1
-I$(WRF_DIR)/external/io_int
-I$(WRF_DIR)/inc
-I$(NETCDF)/include
WRF_LIB = -L$(WRF_DIR)/external/io_grib1 -lio_grib1
-L$(WRF_DIR)/external/io_grib_share -lio_grib_share
-L$(WRF_DIR)/external/io_int -lwrfio_int
-L$(WRF_DIR)/external/io_netcdf -lwrfio_nf
-L$(NETCDF)/lib -lnetcdff -lnetcdf

Architecture specific settings
COMPRESSION_LIBS = # intentionally left blank, fill in COMPRESSION_LIBS below

COMPRESSION_INC = # intentionally left blank, fill in COMPRESSION_INC below

#
Settings for Linux x86_64, Intel compiler (serial)
#
#
COMPRESSION_LIBS = -L/opt/grib2/lib -ljasper -lpng -lz

COMPRESSION_INC = -I/opt/grib2/include

FDEFS = -DUSE_JPEG2000 -DUSE_PNG

SFC = ifort

SCC = icc

```
DM_FC = mpif90 -f90=ifort

DM_CC = mpicc -cc=icc

FC = $(SFC)

CC = $(SCC)

LD = $(FC)

FFLAGS = -FR -convert big_endian

F77FLAGS = -FI -convert big_endian

FCSUFFIX =

FNGFLAGS = $(FFLAGS)

LDFLAGS =

CFLAGS = -w

CPP = /lib/cpp -P -traditional

CPPFLAGS = -D_UNDERSCORE -DBYTESWAP -DLINUX -DIO_NETCDF -DIO_BINARY
-DIO_GRIB1 -DBIT32

ARFLAGS =

CC_TOOLS =
 ##########################


# Macros, these should be generic for all machines

LN = ln -sf

MAKE = make -i -r
```

```
RM = /bin/rm -f

CP = /bin/cp

AR = ar ru

.IGNORE:

.SUFFIXES: .c .f .F .o

# There is probably no reason to modify these rules

.c.o:

$(RM) $@

$(CC) $(CPPFLAGS) $(CFLAGS) -c $<

.f.o:

$(RM) $@ $*.mod

$(FC) $(F77FLAGS) -c $< $(WRF_INCLUDE)

.F.o:

$(RM) $@ $*.mod

$(CPP) $(CPPFLAGS) $(FDEFS) $(WRF_INCLUDE) $< > $*.f90

$(FC) $(FFLAGS) -c $*.f90 $(WRF_INCLUDE)

# $(RM) $*.f90
```

Apéndice F

Archivo namelist.wps

En el archivo *namelist.wps* seconfiguró el intervalo de tiempo para el pre-procesamiento de wrf, la cantidad de dominios a procesar, las coordenadas geográficas, y el directorio WPS_GEOG en el que se ubican los datos geográficos estáticos.

==========

```
&share
wrf_core = 'ARW',
max_dom = 2,
start_date = '2015-12-10_12:00:00','2015-12-10_12:00:00',
end_date = '2015-12-10_18:00:00','2015-12-10_18:00:00',
interval_seconds = 21600
io_form_geogrid = 2,
/


&geogrid
parent_id = 1, 1,
parent_grid_ratio = 1, 3,
i_parent_start = 1, 31,
j_parent_start = 1, 17,
e_we = 74, 112,
e_sn = 61, 97,
geog_data_res = '10m','2m',
dx = 30000,
dy = 30000,
map_proj = 'lambert',
ref_lat = 34.83,
ref_lon = -81.03,
truelat1 = 30.0,
```

truelat2 = 60.0,
stand_lon = -98.0,
geog_data_path = '/home/usuario/aplicacion/Build_WRF/WPS_GEOG/'
/

&ungrib
out_format = 'WPS',
prefix = 'FILE',
/

&metgrid
fg_name = 'FILE'
io_form_metgrid = 2,
/

Apéndice G

Archivo namelist.input

El archivo *namelist.input* se configuró para indicar a wrf el intervalo de tiempo y las coordenadas geográficas en las que se realizaría el procesamiento climático.

==========

&time_control
run_days = 0,
run_hours = 54,
run_minutes = 0,
run_seconds = 0,
start_year = 2015, 2015, 2015,
start_month = 12, 12, 12,
start_day = 06, 06, 06,
start_hour = 12, 12, 12,
start_minute = 00, 00, 00,
start_second = 00, 00, 00,
end_year = 2015, 2015, 2015,
end_month = 12, 12, 12,
end_day = 08, 08, 08,
end_hour = 18, 18, 18,
end_minute = 00, 00, 00,
end_second = 00, 00, 00,
interval_seconds = 21600
input_from_file = .true.,.true.,.true.,
history_interval = 180, 60, 60,
frames_per_outfile = 1000, 1000, 1000,
restart = .false.,
restart_interval = 5000,
io_form_history = 2
io_form_restart = 2

```
io_form_input = 2
io_form_boundary = 2
debug_level = 0
/

&domains
time_step = 180,
time_step_fract_num = 0,
time_step_fract_den = 1,
max_dom = 1,
e_we = 74, 112, 94,
e_sn = 61, 97, 91,
e_vert = 30, 30, 30,
p_top_requested = 5000,
num_metgrid_levels = 27,
num_metgrid_soil_levels = 4,
dx = 30000, 10000, 3333.33,
dy = 30000, 10000, 3333.33,
grid_id = 1, 2, 3,
parent_id = 0, 1, 2,
i_parent_start = 1, 31, 30,
j_parent_start = 1, 17, 30,
parent_grid_ratio = 1, 3, 3,
parent_time_step_ratio = 1, 3, 3,
feedback = 1,
smooth_option = 0
/

&physics
mp_physics = 3, 3, 3,
ra_lw_physics = 1, 1, 1,
ra_sw_physics = 1, 1, 1,
radt = 30, 30, 30,
sf_sfclay_physics = 1, 1, 1,
sf_surface_physics = 2, 2, 2,
bl_pbl_physics = 1, 1, 1,
bldt = 0, 0, 0,
cu_physics = 1, 1, 0,
cudt = 5, 5, 5,
isfflx = 1,
```

```
ifsnow = 1,
icloud = 1,
surface_input_source = 1,
num_soil_layers = 4,
sf_urban_physics = 0, 0, 0,
/

&fdda
/

&dynamics
w_damping = 0,
diff_opt = 1, 1, 1,
km_opt = 4, 4, 4,
diff_6th_opt = 0, 0, 0,
diff_6th_factor = 0.12, 0.12, 0.12,
base_temp = 290.
damp_opt = 0,
zdamp = 5000., 5000., 5000.,
dampcoef = 0.2, 0.2, 0.2
khdif = 0, 0, 0,
kvdif = 0, 0, 0,
non_hydrostatic = .true., .true., .true.,
moist_adv_opt = 1, 1, 1,
scalar_adv_opt = 1, 1, 1,
/

&bdy_control
spec_bdy_width = 5,
spec_zone = 1,
relax_zone = 4,
specified = .true., .false.,.false.,
nested = .false., .true., .true.,
/
&grib2
/
&namelist_quilt
nio_tasks_per_group = 0,
nio_groups = 1,
/
```

Apéndice H

Archivo HPL.dat y prueba de Linpack

H.1. Archivo HPL.dat

El Archivo HPL.dat se genera al ejecutar la instrucción $make \sqcup arch = intel64$ dentro del directorio mp_linpack y se configuró con las características particulares del clúster Xexelo0.

==========

```
HPL.out         output file name (if any)
6               device out (6=stdout,7=stderr,file)
1               # of problems sizes (N)
115584          Ns
1               # of NBs
192             NBs
0               PMAP process mapping (0=Row-,1=Column-major)
1               # of process grids (P x Q)
8               Ps
12              Qs
16.0            threshold
1               # of panel fact
2               PFACTs (0=left, 1=Crout, 2=Right)
1               # of recursive stopping criterium
4               NBMINs (>= 1)
1               # of panels in recursion
2               NDIVs
1               # of recursive panel fact.
1               RFACTs (0=left, 1=Crout, 2=Right)
1               # of broadcast
1               BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)
1               # of lookahead depth
1               DEPTHs (>=0)
```

```
2                SWAP (0=bin-exch,1=long,2=mix)
64                swapping threshold
0                L1 in (0=transposed,1=no-transposed) form
0                U in (0=transposed,1=no-transposed) form
1                Equilibration (0=no,1=yes)
8                memory alignment in double (> 0)
##### This line (no. 32) is ignored (it serves as a separator). ######
0                                Number of additional problem sizes for PTRANS
1200 10000 30000                 values of N
0                                number of additional blocking sizes for PTRANS
40 9 8 13 13 20 16 32 64          values of NB
```

H.2. Resultados de la prueba de Linpack

El siguiente archivo, es el resultado de una de las pruebas de linpack que se realizaron en el clúster. Cada prueba tardó aproximadamente 40 minutos y el rendimiento se indica en la parte final del archivo con letras en negritas.

```
================================================================================

HPLinpack 2.1  --  High-Performance Linpack benchmark  --  October 26, 2012
Written by A. Petitet and R. Clint Whaley, Innovative Computing Laboratory, UTK
Modified by Piotr Luszczek, Innovative Computing Laboratory, UTK
Modified by Julien Langou, University of Colorado Denver
================================================================================

An explanation of the input/output parameters follows:
T/V    : Wall time / encoded variant.
N      : The order of the coefficient matrix A.
NB     : The partitioning blocking factor.
P      : The number of process rows.
Q      : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N      : 115584
NB     : 192
```

PMAP : Row-major process mapping
P : 8
Q : 12
PFACT : Right
NBMIN : 4
NDIV : 2
RFACT : Crout
BCAST : 1ringM
DEPTH : 1
SWAP : Mix (threshold = 64)
L1 : transposed form
U : transposed form
EQUIL : yes
ALIGN : 8 double precision words

- The matrix A is randomly generated for each test.
- The following scaled residual check will be computed:
||Ax-b||_oo / (eps * (|| x ||_oo * || A ||_oo + || b ||_oo) * N)
- The relative machine precision (eps) is taken to be 1.110223e-16
- Computational tests pass if scaled residuals are less than 16.0

Column=000768 Fraction=0.005 Mflops=1011889.58
Column=001344 Fraction=0.010 Mflops=873348.08
Column=001920 Fraction=0.015 Mflops=805522.61
Column=002496 Fraction=0.020 Mflops=729610.48
Column=003072 Fraction=0.025 Mflops=739611.44
Column=003648 Fraction=0.030 Mflops=735719.55
Column=004224 Fraction=0.035 Mflops=727932.58
Column=004800 Fraction=0.040 Mflops=701194.65
Column=005376 Fraction=0.045 Mflops=710109.52
Column=005952 Fraction=0.050 Mflops=709694.14
Column=006528 Fraction=0.055 Mflops=706849.28
Column=007104 Fraction=0.060 Mflops=691472.38
Column=007680 Fraction=0.065 Mflops=697063.51
Column=008256 Fraction=0.070 Mflops=697549.13
Column=008832 Fraction=0.075 Mflops=696333.36
Column=009408 Fraction=0.080 Mflops=686299.63
Column=009984 Fraction=0.085 Mflops=691066.11

Column=010560 Fraction=0.090 Mflops=692417.68
Column=011136 Fraction=0.095 Mflops=691865.21
Column=011712 Fraction=0.100 Mflops=683618.09
Column=012288 Fraction=0.105 Mflops=687247.70
Column=012864 Fraction=0.110 Mflops=689152.25
Column=013440 Fraction=0.115 Mflops=689451.17
Column=014016 Fraction=0.120 Mflops=680956.37
Column=014592 Fraction=0.125 Mflops=684336.28
Column=015168 Fraction=0.130 Mflops=686098.48
Column=015744 Fraction=0.135 Mflops=687136.50
Column=016320 Fraction=0.140 Mflops=683800.64
Column=016896 Fraction=0.145 Mflops=683748.50
Column=017472 Fraction=0.150 Mflops=685327.54
Column=018048 Fraction=0.155 Mflops=685624.05
Column=018624 Fraction=0.160 Mflops=679582.60
Column=019200 Fraction=0.165 Mflops=681867.12
Column=019776 Fraction=0.170 Mflops=683166.90
Column=020352 Fraction=0.175 Mflops=680644.12
Column=020928 Fraction=0.180 Mflops=677746.99
Column=021504 Fraction=0.185 Mflops=679745.84
Column=022080 Fraction=0.190 Mflops=681043.04
Column=022656 Fraction=0.195 Mflops=682203.55
Column=023232 Fraction=0.200 Mflops=677802.22
Column=023808 Fraction=0.205 Mflops=679517.68
Column=024384 Fraction=0.210 Mflops=680486.07
Column=024960 Fraction=0.215 Mflops=681025.95
Column=025536 Fraction=0.220 Mflops=677073.19
Column=026112 Fraction=0.225 Mflops=678425.23
Column=026688 Fraction=0.230 Mflops=678817.81
Column=027264 Fraction=0.235 Mflops=677268.65
Column=027840 Fraction=0.240 Mflops=675583.33
Column=028416 Fraction=0.245 Mflops=676992.97
Column=028992 Fraction=0.250 Mflops=677899.34
Column=029568 Fraction=0.255 Mflops=678534.76
Column=030144 Fraction=0.260 Mflops=675610.65
Column=030720 Fraction=0.265 Mflops=676675.07
Column=031296 Fraction=0.270 Mflops=677342.25
Column=031872 Fraction=0.275 Mflops=677340.63
Column=032448 Fraction=0.280 Mflops=676357.14
Column=033024 Fraction=0.285 Mflops=676070.13

Column=033600 Fraction=0.290 Mflops=676667.71
Column=034176 Fraction=0.295 Mflops=676723.61
Column=034752 Fraction=0.300 Mflops=675172.77
Column=035328 Fraction=0.305 Mflops=675126.23
Column=035904 Fraction=0.310 Mflops=675734.55
Column=036480 Fraction=0.315 Mflops=676058.59
Column=037056 Fraction=0.320 Mflops=673634.24
Column=037632 Fraction=0.325 Mflops=674376.27
Column=038208 Fraction=0.330 Mflops=674871.44
Column=038784 Fraction=0.335 Mflops=673366.97
Column=039360 Fraction=0.340 Mflops=672609.15
Column=039936 Fraction=0.345 Mflops=673423.39
Column=040512 Fraction=0.350 Mflops=673252.70
Column=041088 Fraction=0.355 Mflops=672153.89
Column=041664 Fraction=0.360 Mflops=672482.01
Column=042240 Fraction=0.365 Mflops=672920.51
Column=042816 Fraction=0.370 Mflops=673315.59
Column=043392 Fraction=0.375 Mflops=673517.96
Column=043968 Fraction=0.380 Mflops=671947.85
Column=044544 Fraction=0.385 Mflops=672270.99
Column=045120 Fraction=0.390 Mflops=672719.28
Column=045696 Fraction=0.395 Mflops=672830.94
Column=046272 Fraction=0.400 Mflops=672347.76
Column=046848 Fraction=0.405 Mflops=671884.04
Column=047424 Fraction=0.410 Mflops=672179.99
Column=048000 Fraction=0.415 Mflops=672211.21
Column=048576 Fraction=0.420 Mflops=671253.41
Column=049152 Fraction=0.425 Mflops=671236.78
Column=049728 Fraction=0.430 Mflops=671542.37
Column=050304 Fraction=0.435 Mflops=671625.48
Column=050880 Fraction=0.440 Mflops=670284.75
Column=051456 Fraction=0.445 Mflops=670571.21
Column=052032 Fraction=0.450 Mflops=670502.67
Column=052608 Fraction=0.455 Mflops=670114.94
Column=053184 Fraction=0.460 Mflops=669489.61
Column=053760 Fraction=0.465 Mflops=669844.40
Column=054336 Fraction=0.470 Mflops=670145.86
Column=054912 Fraction=0.475 Mflops=669991.49
Column=055488 Fraction=0.480 Mflops=669087.51
Column=056064 Fraction=0.485 Mflops=669331.06

Column=056640 Fraction=0.490 Mflops=669491.47
Column=057216 Fraction=0.495 Mflops=669403.38
Column=059712 Fraction=0.515 Mflops=668599.67
Column=062016 Fraction=0.535 Mflops=668091.28
Column=064320 Fraction=0.555 Mflops=667272.67
Column=066624 Fraction=0.575 Mflops=666885.87
Column=068928 Fraction=0.595 Mflops=666284.89
Column=071232 Fraction=0.615 Mflops=665596.10
Column=073536 Fraction=0.635 Mflops=664885.67
Column=075840 Fraction=0.655 Mflops=664280.18
Column=078144 Fraction=0.675 Mflops=663796.88
Column=080448 Fraction=0.695 Mflops=663242.96
Column=091968 Fraction=0.795 Mflops=660742.87
Column=103488 Fraction=0.895 Mflops=659101.96
Column=115008 Fraction=0.995 Mflops=658184.67
==========

T/V N NB P Q Time **Gflops**

WR11C2R4 115584 192 8 12 1564.95 **6.57824e+02**

HPL_pdgesv() start time Thu Mar 31 13:36:33 2016

HPL_pdgesv() end time Thu Mar 31 14:02:38 2016

||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)= 0.0020452 ..... PASSED
==========

Finished 1 tests with the following results:
1 tests completed and passed residual checks,
0 tests completed and failed residual checks,
0 tests skipped because of illegal input values.

End of Tests.
==========