CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

TESIS DE DOCTORADO

DISEÑO Y OPTIMIZACIÓN DE UNA INFRAESTRUCTURA COMPUTACIONAL BASADA EN LA NUBE

José Isaac Zablah Ávila

ESCUELA DE DOCTORADO INTERNACIONAL
PROGRAMA DE DOCTORADO EN INVESTIGACIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN

SANTIAGO DE COMPOSTELA
24 de Mayo del 2019

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

DECLARACIÓN DEL AUTOR DE LA TESIS

Diseño y optimización de una infraestructura computacional basada en la nube

Don José Isaac Zablah Ávila

Presento mi tesis, siguiendo el procedimiento adecuado al Reglamento, y declaro que:

1. La tesis abarca los resultados de la elaboración de mi trabajo.
2. En su caso, en la tesis se hace referencia a las colaboraciones que tuvo este trabajo.
3. La tesis es la versión definitiva presentada para su defensa y coincide con la versión enviada en formato electrónico.
4. Confirmo que la tesis no incurre en ningún tipo de plagio de otros autores ni de trabajos presentados por mí para la obtención de otros títulos.

En Santiago de Compostela, 24 de Mayo del 2019

Fdo. José Isaac Zablah Ávila

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

AUTORIZACIÓN DEL DIRECTOR/TUTOR DE LA TESIS

Diseño y optimización de una infraestructura computacional basada en la nube

Don Antonio García Loureiro, Profesor del Departamento de Electrónica y Computación de la Universidad Santiago de Compostela

INFORMA:

Que la presente tesis, corresponde con el trabajo realizado por **Don José Isaac Zablah Ávila** bajo mi dirección, y autorizo su presentación, considerando que reúne los requisitos exigidos en el Reglamento de Estudios de Doctorado de la USC, y que como director de ésta no incurre en las causas de abstención establecidas en Ley 40/2015.

En Santiago de Compostela, 24 de Mayo del 2019

Fdo. Antonio García Loureiro

Director/a tesis

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

José Antonio Loucel Monterroza, SDB

Mentor, por inculcarme disciplina, valores y carácter para emprender mis diferentes etapas formativas.

Oscar Antonio Sánchez

Tío, por brindarme su guía, apoyo y compromiso con las metas que he tenido a lo largo de mi vida adulta.

Fernando Gómez Folgar

Amigo, que me brindó su apoyo en las diferentes etapas de mi formación doctoral.

Antonio García Loureiro

Tutor, por haberme guiado, no sólo en la elaboración de esta tesis, sino a lo largo de mi doctorado, me brindó su apoyo incondicional para desarrollarme en mis actividades.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

El destino del Imperio está en vuestras manos...

Blas de Lezo y Olavarrieta
(Sitio de Cartagena, 13 de marzo de 1741)

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Resumen

Para los países en vías de desarrollo la ciencia y las actividades relacionadas a ella son aspectos descuidados, los programas de cooperación que se centran en estos países no poseen líneas prioritarias en este sentido. Es por ello, que las actividades de transferencia tecnológica junto con el consecuente desarrollo de soluciones que sean flexibles, escalables y sostenibles son una prioridad para la incipiente comunidad científica en estos países. En este documento se presenta la evaluación de diferentes infraestructuras físicas de ordenadores así como las basadas en el modelo de la computación en la nube, en las cuales se han ejecutado diversas aplicaciones y se han comparado para determinar las ventajas y desventajas que aportan los diferentes entornos de ejecución. Asi mismo, se han caracterizado las diferentes infraestructuras por medio de pruebas de rendimiento, con lo cual se han tenido variables y magnitudes aptas para ser relacionadas.

También se ha tomado en cuenta las funcionalidades del propio hardware, en los microprocesadores se consideró el soporte al *hiperthreading* y *turbo*, asi como diferentes implementaciones de almacenamiento, ya sea en unidades locales, de red y de estado sólido. Las ejecuciones de las aplicaciones evaluadas se han hecho en condiciones replicables y constantes. De forma que los resultados obtenidos, han mostrado la utilidad y beneficios para actividades de cálculo intensivo al emplear soluciones basadas en el paradigma de la computación en la nube. Con lo anterior dejo evidente los amplios beneficios de esta forma de operación para la educación superior a nivel administrativo, docente y de investigación científica.

Palabras clave: *Computación en la nube, hipervisores, virtualización, pruebas de rendimiento, computación de altas prestaciones*

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Índice general

1 Introducción **1**

2 Estado del arte **7**
2.1. Computación de altas prestaciones 7
2.2. Paradigma de la computación en la nube 12
2.3. Hipervisores 16
2.4. Gestores cloud de código abierto 17
2.5. Aplicaciones 22

3 Evaluación de los hipervisores **29**
3.1. Pruebas de referencia 29
3.2. Evaluación de hipervisores 34
3.3. Conclusiones 39

4 Casos de uso **41**
4.1. Astronomía y astrofísica 41
4.2. Multimedia 55
4.3. Medicina 70
4.4. Otras aplicaciones 77

Conclusiones **107**

Bibliografía **113**

Índice de figuras **127**

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Índice de tablas **131**

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Capítulo 1

Introducción

La ciencia es un aspecto a menudo descuidado en los países en vías de desarrollo y en los programas de cooperación que se centran en estos. Esta tesis se enfoca en demostrar los beneficios para ellos, aplicado concretamente al Gobierno de Honduras en caso que decida invertir en una infraestructura pública-académica de computación de alto rendimiento (HPC) basada en el paradigma de la computación en la nube. Se pretende demostrar que las aplicaciones actualmente en uso en investigación que se ejecutan sobre infraestructuras dedicadas tienen un comportamiento muy similar al ser portadas a la nube, permitiendo con ello que a través de un mismo hardware se puedan ejecutar múltiples aplicaciones, aprovechando así las tecnologías de virtualización, con disponibilidad elástica de recursos, ubicuidad de acceso y en general un menor costo de despliegue y operación. El objetivo principal es optimizar las actividades de procesamiento y manejo de datos que actualmente se llevan a cabo en el país, así mismo generar las condiciones para poder ejecutar otras más complejas. Para ello, se pretende transferir tecnología que permita hacer llegar los beneficios de la computación de alto rendimiento a todos los niveles de la sociedad por medio de proporcionar mejores servicios estatales y expandiendo las oportunidades de investigación científica, desarrollo e innovación.

No se ha adoptado una sola definición de HPC como un concepto tecnológico para justificar una arquitectura particular, sino que se ha propuesto como un modelo a implementar, de forma que se considera en su sentido más amplio, es pues una tecnología clave de tipo transversal que permitiría el desarrollo de Honduras y de otros países en similares circunstancias. Se han evaluado diversas aplicaciones para los fines de esta tesis, en los campos de las área de las ciencias médicas, astronomía y astrofísica, multimedia y difusión, bases de datos

relacionales, entre otras. Se pretende que la(s) infraestructura(s) aquí evaluadas sean consideradas como una tecnología catalizadora de soluciones a varios desafíos sociales; por ejemplo poder ser utilizadas para mejorar los indicadores de una baja productividad agrícola; mejorar las alertas tempranas antes y después de los desastres naturales; aumentar e implementar procesos de fiscalización por medio de mejores aplicaciones que sirvan de freno a la corrupción política y evasión fiscal, entre otros temas de importancia local.

Las universidades, centros de investigación y otras instituciones son las entidades que mejor pueden manejar los procesos de transferencia de tecnología; pueden trasladar descubrimientos científicos y técnicos de una organización a otra, con el fin de promover su desarrollo y comercialización. En este sentido, diferentes regiones geográficas integran estas oportunidades desde diversas perspectivas; algunos están más centrados en la ciencia y otros en el emprendimiento de empresas. En un país como Honduras, el Estado puede guiar este proceso en beneficio de toda la sociedad por medio de los servicios públicos y gubernamentales.

Objetivos

- Determinar las aplicaciones que actualmente están siendo usadas por los académicos en mi país de origen (Honduras) para reducción, procesamiento y modelado de datos.
- Medir el rendimiento de infraestructuras físicas en uso frente a una alternativa que emplee tecnología de virtualización y el paradigma de la computación en la nube.
- Portar y evaluar las aplicaciones que requieren altos recursos computacionales, determinando la magnitud del cambio del rendimiento en diferentes plataformas.
- Desarrollar y poner en marcha una infraestructura de altas prestaciones, con el fin de poder contar con la capacidad computacional para apoyar las tareas de manejo de Big-Data, por sus necesidades en cuanto a capacidad de procesamiento.
- Implementar y conjugar diversas tecnologías con el fin de tener una solución adaptable a las necesidades actuales y crecientes a futuro.
- Brindar a los usuarios la flexibilidad en cuanto al acceso y las prestaciones disponibles, de manera que se logren los resultados de las investigaciones hasta ahora inalcanzables en muchos casos, en otros en menos tiempo, acelerando la ciencia con todo ello.

Motivaciones

La principal razón de este trabajo es la carencia actual de capacidad de cálculo para el desarrollo de diversas tareas de procesado de datos y cálculo científico. Por lo tanto, se pretende reducir esta brecha a través de la planificación y puesta en marcha en mi país de una infraestructura de altas prestaciones como las aquí descritas, con el fin de impulsar el desarrollo de todas áreas del conocimiento, especialmente las que se listan a continuación:

- **Reducción, análisis y modelado de datos de origen médico:** Con el advenimiento de la medicina de precisión, se genera la necesidad de realizar análisis multiorigen (usando técnicas como *datamining* y *datawarehouse*), ejemplo de ello son los datos extraídos de equipos de imágenes médicas. Toda esta información se considera como *BigData*, debido a su complejidad, volumen y heterogeneidad; requiriendo para su análisis capacidad computacional elevada, elástica, ubicua y flexible; ya que la naturaleza de estos datos es de gran tamaño y cambiantes. El manejo oportuno de las imágenes médicas en la investigación científica, son determinantes para la obtención de resultados para conocer mejor padecimientos y su forma de tratarlos; para ello las capacidades de computación de altas prestaciones aceleraran el proceso de obtención de resultados a través del procesamiento y modelado computacional. Así mismo la combinación de conocimientos entre áreas médicas y de ingeniería van ligadas a conocimientos que están en incipiente desarrollo como son las ciencias biomédicas y la bioinformática; de manera que requieren de una solución computacional escalable como una herramienta valiosa si se implementa bajo el paradigma de infraestructura como servicio.

- **Creación de contenido multimedia de alta definición:** Los recursos multimedia son el resultado de un proceso computacional intensivo que busca obtener medios de alta calidad a través de millones de cálculos, donde se tiene en cuenta variables relacionadas a las tonalidades, sombras, texturas, refracciones, iluminación, profundidad de campo, movimientos, ambiente, efectos especiales, entre otros elementos, que se mezclan en las técnicas de edición y producción de medios en entornos de alta definición. El hecho de no contar con la capacidad computacional adecuada hacen ardua, lenta y muchas veces imposible esta tarea. Una infraestructura HPC reduciría el tiempo de obtención de contenido listo para difusión, siendo actualmente el mayor cuello de botella en el flujo de trabajo de edición y postproducción en la televisora de mi universidad de origen.

- **Modelado y animación en 2D y 3D:** La tecnología de gráficos en dos dimensiones (2D) y en tres dimensiones (3D) es aplicable en diversos campos. Dependiendo de la capacidad de los ordenadores empleados, la renderización puede requerir desde algunas horas hasta varias semanas para finalizar con un prototipo. Contar con servicios computacionales que permita el aprovechamiento de esta tecnología, impulsaría actividades académicas a nivel de ingeniería y de investigación. En el caso de los sistemas de producción televisiva, se requiere encarecidamente poder mejorar el tiempo de obtención de escenas para difusión en alta definición, las cuales sólo pueden ser obtenidas a través de infraestructuras de altas prestaciones, permitiendo poner a disposición de los animadores y diseñadores capacidad computacional que les permita poder dedicar más tiempo al producto y no al proceso. Esto también es aplicable a los campos de ingeniería en áreas mecánicas y civiles, arquitectura, física entre otras áreas.

- **Ciberseguridad, ciberguerra y técnicas de protección de datos:** El mundo enfrenta amenazas de una realidad distinta, invisibles y fuera de todo lo conocido de forma convencional. Por citar un evento, en 1989 un virus del tipo gusano llamado *WANK* creado por *hackers* australianos que se hacían identificar como *Electron* y *Phoenix*; atacaron sistemas DEC VMS en la red DECnet; que dió como resultado que las comunicaciones entre la NASA y el Departamento de Energía de los EEUU se comprometieran lo suficiente como para interrumpir el lanzamiento programado del transbordador espacial que llevaba la sonda Galileo a órbita. Por citar, situaciones más recientes en el contexto hondureño y de forma sostenida en los últimos años los sistemas de Gobierno que tienen portales o acceso a Internet han sido objeto de múltiples ataques, tanto de individuos de fuera como del interior de las mismas dependencias del estado, careciendo de todo tipo de recurso para analizar protocolos, bitácoras y otros elementos que pueden ser útiles en una respuesta rápida a estas amenazas. La infraestructura de altas prestaciones puede proveer instancias computacionales bajo demanda que sirvan para ayudar a determinar el origen de los ataques y encontrar la vía más eficiente para evitar sus daños.

- **Matemática y física computacional:** El desarrollo de modelos numéricos en los cuales se estudian fenómenos como las órbitas de planetas extrasolares y trayectorias de colisión de asteroides, entre otros, son actualmente una línea de investigación en mi institución. Se requiere para ello el manejo de altos volúmenes de datos que necesitan ser analizados utilizando múltiples modelos y técnicas, también se incluyen en estos, el análisis de datos provenientes de sensores espaciales y de bases de datos de múltiples relaciones

orientadas a la obtención de modelos astrofísicos. En el área matemática, esta infraestructura puede ser empleada para el cálculo en álgebra lineal, específicamente para resolver sistemas de gran tamaño y problemas vectoriales.

- **Sistemas de información geográfica:** Existe necesidad creciente del control territorial, tanto para conocer sus cambios como para hacer mejor uso de él. Estas tareas han recibido un impulso debido al empleo de técnicas de automatización y en especial de software orientado a la cartografía. Lo anterior, se ha unido con bases de datos relacionales obteniendo una cantidad enorme de aplicaciones. El problema en estos sistemas, es contar con la capacidad que permita manejar datos espaciales (imágenes provenientes de satélites, fotografía aérea, etc.), combinar metadatos (información concreta que le da valor añadido a los datos espaciales y de hecho los describen), hojas cartográficas y otros recursos. Los elementos mencionados al ser combinados a través del tiempo permiten realizar modelos para estimar el cambio en la superficie y de esa manera poder proponer caminos apropiados para su mejor uso. Para poder obtener un patrón, un mapa o un modelo, se requiere en gran medida poder utilizar varios procesos paralelos que necesitan de una infraestructura como las propuestas en este documento.

- **Predicción meteorológica:** El estudio del cambio climático y calentamiento global donde convergen varios factores como ser naturales y antropogénicos que han ocasionado cambios en el planeta, que requieren ser analizados hoy en día para predecir las causas y futuro de este fenómeno. También, es de mucha importancia estudiar el papel de la atmósfera solar y los efectos que esta puede tener sobre su contra parte terrestre, siendo esta según los especialistas la otra vertiente del calentamiento planetario. Debido a que hay una serie de variables que requieren ser modeladas, la necesidad de una infraestructura optimizada puede brindar un entorno para estudiar diversas condiciones que afecten el clima, dichos modelos requieren por su complejidad una gran capacidad de procesamiento y actualmente es una línea de investigación en mi país.

- **Gobierno electrónico:** Honduras necesita urgentemente que se le brinden soluciones basadas en sistemas informáticos que permitan la reducción de tiempos y costos en la gestión pública sobre todo en el relacionado a la atención del contribuyente y procesos electrónicos gubernamentales, una infraestructura resistente a fallos ofrecería los medios necesarios para poder desarrollar los sistemas útiles en estos sentidos y propiciar su adecuada ejecución.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Capítulo 2

Estado del arte

En esta sección se hace una recensión de los conceptos básicos indispensables para el desarrollo de esta investigación, haciendo énfasis en la evolución de los mismos y su estado actual de desarrollo. También se describen detalles de las retos actuales en los tópicos descritos y sus aplicaciones técnicas en la solución de problemas a diferentes niveles.

2.1. Computación de altas prestaciones

La humanidad reconoce al ENIAC (*Electronic Numerical Integrator and Computer*), construida por P. Eckert y J. Mauchly, como el primer gran ordenador. El ENIAC estaba compuesto por más de 19000 tubos de vacío, con un peso de 30 toneladas, ocupaba un espacio de 72 metros cuadrados y consumía al menos 140 KVA. Este primer ordenador disponía tan sólo de una velocidad de reloj de 100 KHz, con la capacidad de hacer 330 multiplicaciones por segundo. Pero no fue hasta 1961, que se instala en el Alamos National Laboratory el que ha sido considerado como el primer superordenador, el Cray-1. Entre sus características estaban la potencia de cálculo y la capacidad de memoria, podía realizar 160 millones de operaciones por segundo (160 megaflops), poseía 8 megabytes de RAM. En la década de los setenta y ochenta vieron el desarrollo de diversas tendencias en superordenadores, específicamente en procesamiento vectorial y paralelo. A mediados de la década de los noventa, la forma de un único procesador diseñado específicamente desapareció y fue reemplazada por el uso de varias unidades centrales de procesamiento, lo que permitió el desarrollo más acelerado y económico de la capacidad computacional. Al momento de redactar el presente documento, el superordenador más potente del mundo es el llamado *Summit*, instalado en el Laboratorio Na-

cional Oak Ridge en Estados Unidos de América, cuenta con una capacidad de procesamiento de 200.795 petaflops, suministrada por 2,397,824 núcleos [53].

La computación de altas prestaciones consiste en la utilización de súpercomputadores, sistemas multiprocesador y redes de ordenadores en la resolución de problemas con una carga computacional elevada y que son difícilmente abordables con otras plataformas. Es ampliamente utilizada en un gran abanico de campos de la ciencia, la ingeniería y la industria. Áreas como la nanotecnología, predicción climática, ingeniería aeroespacial y del automóvil, explotación del petróleo, diseño de fármacos, genética, predicción y evolución de la contaminación, diseño de dispositivos electrónicos, etc., todo ello necesita el apoyo de potentes ordenadores y el desarrollo de aplicaciones paralelas para seguir avanzando en su desarrollo. Además, existen diversas plataformas usadas en la computación de altas prestaciones, como las que usan procesadores con múltiples núcleos o (*manycore*) y otras que utilizan unidades gráficas de procesamiento (GPU). Dependiendo de las aplicaciones a ejecutar se debe determinar que plataforma de hardware es la que brindará el máximo rendimiento y se adaptará mejor al problema a resolver.

El incremento de la capacidad computacional (al concentrar un mayor número de recursos físicos) es directamente proporcional al poder de cálculo y con ello será mayor el número de problemas que se pueden solventar, aumentando su precisión y disminuyendo el tiempo de obtención de resultados. Las organizaciones que cuentan con altas capacidades poseen una ventaja competitiva que se convierte en un elemento diferenciador.

Las líneas de investigación abiertas en el ámbito de la computación de altas prestaciones (*High Performance Computing* - (HPC)) cubren tres áreas. La *primera* es el hardware para integrar soluciones a nivel físico lo que incluye redes, memoria, placas, procesadores entre otros. La *segunda* es el software que se utiliza para poner a disposición la capacidad de gestionar y monitorear la infraestructura en cuanto a su disponibilidad de recursos y su crecimiento. La *tercera* son las aplicaciones que requieren los usuarios finales, que van desde compiladores hasta simuladores los cuales deben estar adaptados para ejecutarse de forma optimizada, sacando el mayor provecho posible del entorno de alto rendimiento.

Cuando se realizan cálculos, se pueden considerar tres enfoques básicos para mejorar el rendimiento. El *primero*, es emplear un algoritmo más eficiente, en el cual se debe considerar detalladamente lo que se desea calcular y la manera de analizar como se puede sacar el mayor provecho al hardware, con ello se debe reescribir el código fuente a partir de una definición mejorada de los problemas evitando operaciones innecesarias. El *segundo* enfoque es utilizar

un equipo más rápido, se puede obtener una mejora en el procesamiento al reemplazar el equipo (ordenadores) empleados para cálculos, pero el costo usualmente se incrementa de forma exponencial mientras que el desempeño del mismo crece de forma lineal, de manera que es necesario encontrar el equilibrio en este aspecto para obtener un resultado apropiado. Y finalmente, el *tercer* enfoque es dividir el cálculo entre varios ordenadores, donde se puede considerar el empleo de otras técnicas como puede ser el paralelismo, en el que se busca la ejecución de varias instrucciones de forma simultánea. En HPC hay múltiples formas de lograr esto, una de ellas es el soporte nativo de esta modalidad de operación en la arquitectura interna de los CPU usados o bien, otra forma de lograrlo es utilizar varias computadoras interconectadas entre sí para distribuir las tareas a realizar entre ellas.

2.1.1. Arquitectura de ordenadores

Usualmente las computadoras se clasifican en base a su tamaño y desempeño, por decirlo así, existen las microcomputadoras, las estaciones de trabajo, las minicomputadoras, los llamados *mainframes* y las súpercomputadoras. Pero para los fines de este documento esa clasificación se queda muy corta, ya que el desarrollo tecnológico hace que fácilmente una microcomputadora supere en capacidades a un *mainframe* en poco tiempo; de modo que se necesita poder agrupar los sistemas de una manera más apropiada, especialmente cuando se introducen conceptos como la computación paralela y las arquitecturas HPC.

Para lograr una clasificación acorde a las actividades de altas prestaciones, es necesario traer a discusión el paradigma básico de la arquitectura moderna de computadoras, que ha sido atribuido al matemático de origen húngaro John Von Neumann. Él planteaba que la estructura básica de cualquier ordenador se basaba en una CPU conectada a una memoria por medio de un canal de comunicación o *BUS*, de manera que las instrucciones y datos se almacenan en la memoria y se mueven a la CPU por medio de este. El rendimiento global del ordenador depende tanto de la velocidad de la CPU para ejecutar instrucciones asi como del traslado de datos hacia la memoria (llamado también velocidad de *BUS*) conjugada con la capacidad de la memoria para realizar sus operaciones internas. A la fecha, se han desarrollado múltiples tecnologías para mejorar el rendimiento de las CPU, por ejemplo se encuentra la tecnología RISC (Instrucciones Reducidas de Computador) tiene como objetivo disminuir y uniformar las instrucciones internas de los procesadores, a través de la eliminación de ciclos de ciertas instrucciones, aumentando la velocidad del reloj de la CPU. También se han desarrollado las llamadas arquitecturas súperescalares utilizando unas estructuras conocidas

como segmentación o tuberías (*pipelining*), de manera que se pueden ejecutar dos o más instrucciones simultáneas por ciclo del reloj de la CPU, mejorando la velocidad de los mismos. Por otro lado, la memoria (específicamente la RAM) ha aumentado en cuanto a su ancho de banda o *bandwidth*, que básicamente se explica como el aumento de la cantidad de bits que se transfieren entre los chips de memoria hacia el *BUS* del sistema. De este modo, se reducen los cuellos de botella que predijo en su momento Von Neumann, ya que el mayor problema de los sistemas computacionales a nivel teórico es que una CPU quede con ciclos libres, debido a que no hay instrucciones a procesar porque la memoria es muy lenta como para mantener un flujo constante de datos en el *BUS*. Por lo tanto, los fabricantes se han esforzado a través del tiempo por mejorar esta situación. Inicialmente las súpercomputadoras contaban únicamente con un sólo CPU súperescalar o vectorial, diseñadas específicamente para ellas. Hoy en día, es posible crear arquitecturas HPC con cientos o miles de CPUs, usando para ello sistemas multiprocesador y multicomputador.

Los sistemas multiprocesador son clasificados en base a su arquitectura de manejo de memoria, existen de esta forma los sistemas llamados de memoria de acceso uniforme (UMA – *Uniform Memory Access*) y los de memoria de acceso no uniforme (NUMA – *Nonuniform Memory Access*). Las diferencias básicas radican en que los sistemas UMA (también llamados SMP por sus siglas en inglés *Symmetric MultiProcessors*) sólo posee una única memoria que es compartida; de forma que sólo existe un mapa de direcciones (en las mismas ubicaciones de la memoria física) sin tener en cuenta las CPU. Explicado de manera diferente, es que la memoria principal del sistema es igualmente accesible por todas las CPU que conforman el sistema, aclarando que cada CPU posee su propio caché que facilita este modo de operación. Por otra parte, los sistemas con arquitectura NUMA se diferencian con los primeros porque cada CPU posee un espacio propio reservado de memoria física, de forma que la memoria total se divide entre el número de procesadores del sistema.

La arquitectura de los sistemas multiprocesador poseen dos dificultades principales de implementación. La *primera* es la sincronización, lo cual corresponde a contar con los datos en los buses en el orden apropiado para su procesamiento individual (esta característica es mejor implementada en los NUMA, porque se mantienen fijas las asignaciones de memoria) y la *segunda* es lograr coherencia, considerada como la operación que asegura que todos los procesadores en conjunto tienen el valor actualizado de una misma variable. Lo anterior es importante, ya que en estos sistemas es necesario que los cálculos puedan ser granulados (esto es que puedan ser divididos en secciones) y así emplear la mayor cantidad de procesadores de

forma simultánea para lograr resultados en menos tiempo.

Los multicomputadores, es una configuración que se integra al unir un conglomerado de ordenadores de forma que trabajen en conjunto. Estos sistemas cuentan con tres elementos básicos, el *primero* son varios ordenadores. El *segundo*, es acceder a una red que los interconecte. El *tercero*, es el software que posibilita a todo el hardware para compartir el trabajo y llevarlo a cabo. Usualmente, estos sistemas pueden configurarse con equipos de tipo comercial (no especializado), de manera que el costo de construcción frente a los sistemas de multiprocesadores es mucho menor. Los sistemas multicomputador, pueden ser dedicados exclusivamente a una tarea específica o a varias a la vez. Poseen la facilidad de que los ordenadores que lo conforman pueden ser agregados o retirados conforme a las necesidades (llamados también nodos). Este tipo de sistemas se pueden clasificar conforme a la interconexión y funciones que desempeñen los nodos.

2.1.2. Distribución geográfica de los centros de computación de alto rendimiento

La introducción del primer ordenador personal se realizó a principios de la década de los ochenta. Hoy en día, el avance en todas las áreas de conocimiento requieren en sus diversos procesos contar con la interacción de sistemas informáticos y sus dispositivos asociados, es por ello que en la actualidad se le conoce como *la era de la información y del conocimiento*. Las necesidades computacionales específicas como ser la capacidad de cálculo, volúmen de almacenamiento, capacidad de memoria RAM y otros recursos, no eran suficientes hace tres décadas, en la actualidad las mismas necesidades persisten a pesar del desarrollo tecnológico alcanzado, aun así existe en el mundo un amplio número de infraestructuras dedicadas exclusivamente a la resolución de problemas tecnológicos que requieren cantidades altas de capacidad de cálculo y memoria.

La mejor referencia para conocer la evolución tecnológica de las infraestructuras HPC se encuentra en la lista TOP500[53], esta se actualiza cada seis meses y muestra una relación de los quinientos súpercomputadores más potentes del mundo. Si analizamos la última publicación al momento de redactar este documento con fecha noviembre del 2018, se observan las tendencias en la computación de altas prestaciones. En cuanto a la arquitectura empleada, hay 442 sistemas utilizado *clusters* (multiordenadores) y sólo 58 están basados en sistemas multiprocesador. Hay una variedad de tecnologías de red usadas por estos sistemas, las cinco preferidas con *10G Ethernet* (23.8 %), *40G Ethernet* (12.6 %), *Infiniband FDR* (12.2 %),

25G Ethernet (11.8 %) e *Infiniband EDR* (9.6 %). Otro aspecto muy valorado en cuanto a la evolución de estos sistemas es la eficiencia energética, en la actualidad el mayor rendimiento se encuentra en 17.604 Gflops/W (mil millones de operaciones de punto flotante por watt de potencia eléctrica consumida), que se logró utilizando procesadores Intel Xeon D-1571 en el Shoubu System B - ZettaScaler-2.2 en el Centro Avanzado para la Computación y Comunicaciones del Instituto Riken en Japón; el cual ocupa el puesto 375 de la lista TOP500.

Este tipo de instalaciones están ubicadas de forma variada a nivel mundial, la región con la mayor capacidad computacional de altas prestaciones es Asia Oriental (53.2 %), seguido por la región de Norte América (23.6 %) y luego Europa Occidental (11.4 %). Se observa que en Latinoamérica sólo es considerado en esta lista un centro en la región suramericana (representando sólo un 0.2 % del total mundial), quedando Centro América y el Caribe sin participación por la ausencia de este tipo de infraestructuras. La distribución de estos centros se muestra de forma numérica en la tabla 2.1.

Se deduce que las regiones con alta capacidad computacional corresponden con las que poseen la mayor producción científica, de manera que es directamente proporcional con los resultados en investigación y el avance tecnológico. Esta situación pone de manifiesto que un país en vías de desarrollo necesita implementar infraestructuras de este tipo con el fin de impulsar el desarrollo del conocimiento y con ello el despegue económico tan deseado, reduciendo con ello las brechas existentes. Para facilitar el punto de vista anterior en la figura 2.1 se muestran los centros de supercomputación agrupados por región geográfica.

2.2. Paradigma de la computación en la nube

En la actualidad las infraestructuras de alto rendimiento dedicadas son usadas para actividades muy especializadas. Por motivos de costo se han diseñado soluciones basadas en el paradigma de computación en la nube o *cloud computing*. Esta puede ser definida como un modelo informático para permitir por medio de una red (Internet o intranet) el acceso ubicuo y conveniente a un conjunto compartido de recursos computacionales configurables (por ejemplo: redes, servidores, almacenamiento, aplicaciones, servicios, etc.) que se pueden provisionar y liberar rápidamente con un esfuerzo mínimo de gestión sin requerir la interacción de un proveedor de servicios [64]. La nube cuenta con diferentes niveles de implementación siendo el fundamental el *IaaS*, uno intermedio llamado *PaaS* y el superior el *SaaS*, que se detallan a continuación [52]:

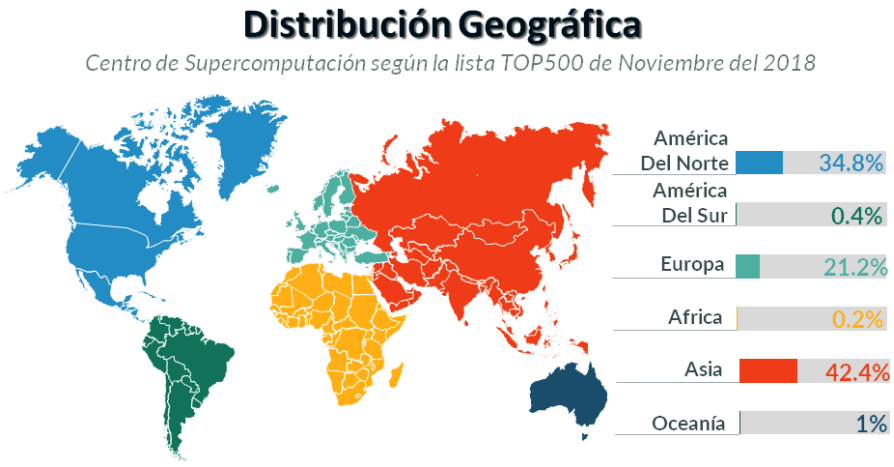

Figura 2.1: Mapa ilustrativo de la distribución geográfica de los centros de súper computación según la lista TOP500 de noviembre del 2018.

Región Geográfica	No. de Centros	% Mundial	Total de Núcleos
Norte América	118	23.60 %	16,538,000
Asia Oriental	266	53.20 %	33,345,336
Europa Occidental	43	8.60 %	4,148,026
Europa del Norte	38	7.60 %	2,529,300
Europa del Este	8	1.60 %	408,204
Europa del Sur	8	1.60 %	987,520
Asia Occidental	3	0.60 %	325,940
Australia y Nueva Zelandia	6	1.20 %	275,896
Sur América	1	0.20 %	38,400
Asia Centro-Sur	4	0.80 %	272,328
África del Sur	2	0.40 %	71,256
Sur Este Asiático	3	0.60 %	146,112
Total	**500**	**100.00 %**	**48,079,042**

Tabla 2.1: Distribución de los centros de supercomputación según la lista TOP500 de noviembre del 2018

- **Infraestructura como servicio (IaaS):** Se refiere al hardware que soporta las capas superiores, su configuración inicial tiende a perdurar a través del tiempo, ya que corresponde a los técnicos proporcionar los debidos ajustes a un hipervisor, a un gestor *cloud*, a los sistemas operativos anfitriones, la configuración de red y hardware en general.

- **Plataforma como servicio (PaaS):** Representan todas aquellas herramientas que permiten la ejecución de una o más aplicaciones, llámense base de datos, sistemas operativos, recursos de red configurables, entre otros. Requieren personal técnico para su configuración en base a la aplicación a ejecutar. Usualmente estos despliegues se hacen en base a plantillas pre-existentes minimizando el tiempo requerido para la puesta en marcha de instancias, de recurso complementarios y horas hombre para el despliegue de soluciones.

- **Software como servicio (SaaS):** Son aquellas aplicaciones listas para ser aprovechadas por parte del usuario, quien sólo interactúa con configuraciones propias del área de ejecución, no existe requerimiento alguno de manipular infraestructura, aplicaciones intermedias o sistemas operativos. La aplicación no requiere mayor soporte técnico, solamente es necesario el conocimiento del usuario en la operación de la misma. En la figura 2.2 se muestra la organización jerárquica de los diferentes niveles de implementación de la nube.

El paradigma permite el despliegue de aplicaciones, servicios e infraestructura en los siguientes modos de operación [52]:

- **Nube privada:** Es cuando toda la nube es provisionada para uso exclusivo de una sola organización en la que pueden participar múltiples unidades de negocio; puede ser operada, administrada y ser dueña una única entidad o bien puede ser provista por un tercero o una combinación; pueden existir recursos dentro y fuera de las instalaciones físicas de la entidad dueña.

- **Nube pública:** Es el tipo de nube que está abierta al público. Esta puede ser operada, administrada y ser dueña una entidad de gobierno, académica, una empresa o combinación de ellas; sus recursos operan del lado de un proveedor de servicios.

- **Nube híbrida:** Combina dos o más modelos de los mencionados. Se caracteriza por contar con los recursos estandarizados y con la tecnología apropiada para permitir la porta-

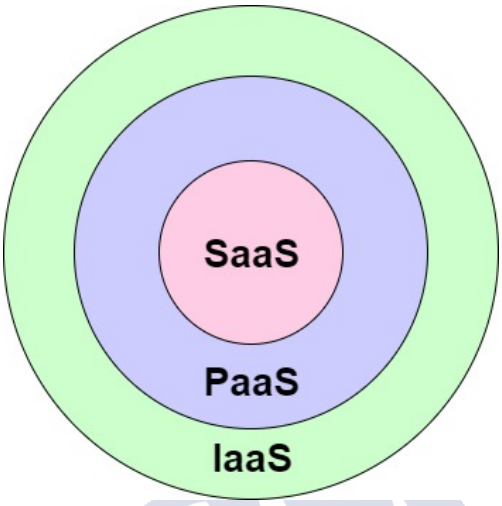

Figura 2.2: Niveles de servicio en la nube.

bilidad de datos entre las nubes. La figura 2.3 muestra la integración de este tipo de nube.

- **Nube comunitaria:** Es el tipo de nube que se desarrolla para el uso de un grupo de clientes o de unidades de varias organizaciones que tienen intereses comunes (llámense seguridad, políticas públicas, cumplimientos normativos, etc.). De la misma puede ser dueño, operador y administrador una o más instituciones que son miembros de la comunidad participante, pero existe la posibilidad que un tercero sea dueño pero la ofrezca bajo acuerdos de servicio, existiendo recursos físicos en el sitio de los miembros o fuera de ellos.

Las ventajas esenciales que aporta la nube a quienes deciden usar este paradigma se encuentra con: la capacidad computacional bajo demanda y auto gestionada, amplios acceso a redes de ordenadores, rápida elasticidad, alta disponibilidad de datos, funcionamiento en tiempo real, servicios medibles, se puede contar con una base común de recursos o *resource pooling*, entre otros beneficios. La operación de instancias en la nube, permite compartir sin importar su forma de ser implementada con facilidades de escalamiento masivo, homogeneidad de operación, uso de la tecnología de virtualización, menor costo del software, computación resilente (recuperación ante fallos), distribución geográfica, operación orientada a ser-

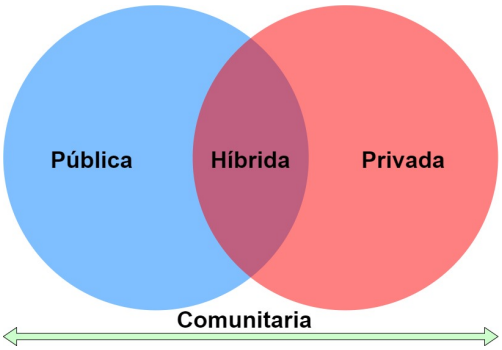

Figura 2.3: Diagrama de Venn que muestra la convergencia de los diferentes tipos de nubes.

vicios, esquemas de seguridad avanzada e información organizada de forma diferenciada y jerarquizada.

Adicionalmente la nube ha seguido un rápido y constante desarrollo, esto debido a su bajo costo y economía de escala. Lo anterior ha hecho que su capacidad de alta disponibilidad y rápida elasticidad la esté volviendo que sea el paradigma con mayor aceptación en la industria. La nube potenciará las nuevas tecnologías de telecomunicaciones y apoyará el desarrollo de tecnologías en múltiples áreas del conocimiento, su evolución brindará mayores opciones de crecimiento al sector de la informática.

2.3. Hipervisores

Los hipervisores son una serie de aplicaciones encargadas de gestionar los recursos de un sistema para permitir el despliegue de anfitriones virtuales (máquinas virtuales), permitiendo la ejecución de múltiples instancias de diversos sistemas operativos conforme a los recursos disponibles. Estas piezas de software representan el componente primordial de los gestores *cloud*. Los hipervisores considerados para las diferentes pruebas y evaluaciones hechas en este documento se describen a continuación:

- **VirtualBox:** Es una aplicación de virtualización multiplataforma, se instala en equipos con procesadores Intel o AMD ya sea que ejecuten sistemas operativos Windows, Mac, Linux o Solaris; logrando ampliar las capacidades de la computadora anfitriona para que pueda ejecutar múltiples sistemas operativos (a través de varias máquinas virtuales) de forma concurrente, limitándose únicamente por la disponibilidad de los recursos

de hardware. En VirtualBox se pueden ejecutar desde pequeños sistemas integrados o máquinas de clase de escritorio hasta llegar a implementaciones de centros de datos e incluso entornos de nube [94].

- **Xen:** Es una herramienta utilizada para implementar la tecnología de virtualización permitiendo la ejecución de máquinas virtuales de una gran variedad de sistemas operativos, logrando que la gestión sea eficiente en el manejo de altas cargas de trabajo, con baja latencia de red, soportando multimedia entre otras ventajas. Implementa de forma completa las instrucciones de extensión de vector avanzadas (*Advanced Vector eXtension*, AVX por sus siglas en inglés) incluídas en los microprocesadores. Xen se integra al sistema operativo anfitrión a través de un kernel modificado que brinda el soporte necesario para su ejecución, ya sea de forma paravirtualizada o en virtualización completa. No se requiere modificaciones del sistema anfitrión para su posterior ejecución [24].

- **KVM:** Es el llamado hipervisor de máquina virtual sobre kernel (*Kernel Virtual Machine*, KVM por sus siglas en inglés) se compone de módulos que se cargan durante el arranque del sistema operativo convirtiendo a todo el sistema en un gestor de virtualización sin necesidad como en otros hipervisores que requieren un kernel personalizado o modificado. Se ha buscado con estos módulos simplificar la gestión y aumentar el rendimiento de las máquinas virtuales al ejecutarse como un proceso dentro del sistema operativo. Para su implementación requiere del soporte en hardware de las extensiones de virtualización VTX [87].

2.4. Gestores cloud de código abierto

Para hacer un uso integrado de los recursos computacionales físicos y virtualizados, se han desarrollado herramientas para facilitar su gestión, logrando el despliegue de todo tipo de infraestructuras de forma efectiva. Existen múltiples soluciones privativas y de código abierto, para los fines del presente documento se enumeran y describen las de mayor interés.

2.4.1. OpenNebula

Es una plataforma de computación en la nube de código abierto para la gestión de infraestructuras heterogéneas en centros de datos distribuidos, permitiendo con ello concentrar en un entorno virtual la capacidad de conformar implementaciones privadas, públicas e híbridas

de una infraestructura como servicio (IaaS). OpenNebula organiza las tecnologías de almacenamiento, red, virtualización, monitoreo y seguridad para desplegar servicios con múltiples niveles de gestión y ejecución. Para ello hace uso de máquinas virtuales en infraestructuras distribuidas basados en políticas de asignación de recursos locales y remotos [66]. A grandes rasgos OpenNebula posee tres componentes principales [79] [70], el *primero* es la *capa de controladores* la cual se comunica directamente con el sistema operativo subyacente y encapsula la infraestructura como un servicio abstracto. Entre sus funcionalidades se encuentra la creación, inicio y cierre de máquinas virtuales (MV), la asignación de almacenamiento para MV y la supervisión del estado operativo de las máquinas físicas y MV. La *segunda* es la *capa principal*, su función es la de administrar los ciclos de vida completos de las MV, incluyendo la configuración dinámica de red y direcciones IP, la gestión del almacenamiento y la asignación de recursos para los discos virtuales de las VM, entre otros. La *tercera* es la *capa de herramientas*, esta proporciona interfaces para la comunicación con OpenNebula, entre ellas está la interfaz de línea de comandos (CLI), el navegador y la API *libvirt*; con esto los usuarios pueden administrar las MV. Existe la figura de un programador que administra las funcionalidad proporcionada por la capa principal. Con relación al almacenamiento, el repositorio de imágenes de MV se utiliza para almacenar y administrar aquellas que son registradas por los administradores y usuarios, permitiendo el despliegue rápido de MV y compartir la mismas. OpenNebula emplea un sistema de archivos de red siendo por omisión el NFS. Entre los hipervisores soportados se encuentran Xen, KVM y VMWare. Los diversos componentes pueden observarse en la figura 2.4.

2.4.2. OpenStack

Es una solución basada en software libre, es de código abierto y se utiliza para desplegar plataformas de computación en la nube bajo la modalidad de infraestructura como servicio (IaaS). Consta de componentes interrelacionados que controlan diversas agrupaciones de hardware pudiendo ser estos de varios proveedores de procesamiento, almacenamiento y recursos de red a través de un centro de datos. Los usuarios lo administran por medio de un panel de control basado en web, o bien mediante herramientas de línea de comandos y adicionalmente se tiene soporte de forma nativa para su control mediante servicios web [89]. La arquitectura de OpenStack esta conformada por siete partes principales [70][79] , la *primera* es *Nova* la cual es un controlador de la infraestructura en la nube, posee seis componentes como ser *Nova-API, Message-Queue, Nova-Compute, Nova-Network, Nova-Volume*

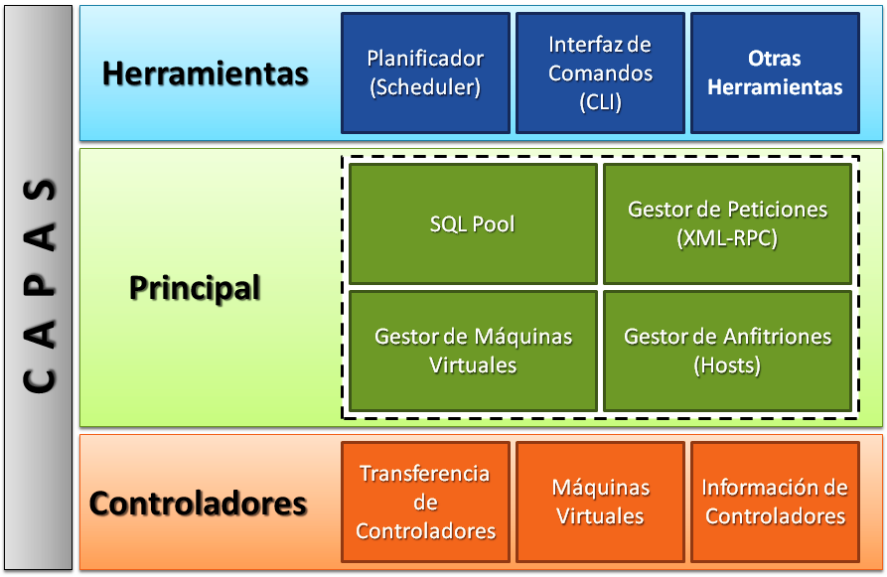

Figura 2.4: Diagrama que muestra los componentes de OpenNebula (66).

y *Nova-Scheduler*. Este componente brinda la funcionalidad del ciclo de vida completo de las instancias gestionadas por OpenStack, mostrándose como una plataforma administradora de recursos computacionales y de red, permitiendo la autorización junto con el escalado conforme a las necesidades de ejecución. Todos los componentes de la arquitectura siguen una política de operación como ser la de no compartición *shared-nothing* y la basada en la mensajería (*messaging-based*); el término *shared-nothing* hace referencia a que cada componente o cada grupo de componentes pueden ser instalados en cualquier anfitrión, ya sean todos en un mismo hardware o diferentes componentes en múltiples ordenadores. En cambio el *messaging-based* se aplica cuando el controlador de la nube se comunica con los diferentes elementos, intercambian información de operación y estado entre si; para ello existe otro componente llamado servidor de colas (*Queue Server*) el cual desarrolla esta política respecto a los elementos como ser *Nova-Volume, Nova-Network* y *Swift*. La comunicación entre los componentes es asíncrona, lo que ayuda a que las acciones de los usuarios no impactan por largos períodos la operación del gestor OpenStack. La *segunda* parte es *Swift*, la cual es una infraestructura de almacenamiento de objetos escalable que incluye un servidor *proxy*, un servidor de objetos, un servidor de cuentas, un servidor contenedor y el llamado anillo. Es

un sistema de almacenamiento a largo plazo para un tipo más permanente de datos estáticos que se pueden recuperar, aprovechar y actualizar [7]. Las características y funcionalidades principales son el almacenamiento seguro de una gran cantidad de objetos de diferente tamaño, proporcionar redundancia de datos, capacidad de archivado y transmisión de medios. La *tercera* parte es *Glance*, este componente ofrece las capacidades de gestión de imágenes de MV, está conformado por dos elementos *Glance-Registry* encargado de la creación y *Glance-Control* con la función de operación. La *cuarta* es *Cinder* que proporciona almacenamiento de bloques persistente para instancias de cálculo, este servicio es responsable de administrar el ciclo de vida de los dispositivos de bloques, desde la creación y el archivado de volúmenes, hasta la puesta en marcha de las instancias. Las consideraciones de seguridad para el almacenamiento en bloque son similares a las del almacenamiento de objetos. El *quinto* componente es *Neutron* el cual proporciona diversos servicios de red para usuarios de la nube, incluyendo la administración de direcciones IP, DNS, DHCP, balanceo de carga y grupos de seguridad (lo que incluye reglas de acceso a la red, como las políticas del cortafuegos). Adicionalmente proporciona un marco para redes definidas por software (*SDN*) que permite la integración con varias soluciones de red. Los aspectos de seguridad incluyen el aislamiento, la disponibilidad, la integridad y la confidencialidad del tráfico de red. El *sexto* componente es *Keystone* el cual actúa como un servicio compartido que proporciona mecanismos de autenticación y autorización en toda la infraestructura de la nube, también tiene soporte conectable para múltiples formas de autenticación. La elementos de seguridad con este componente incluyen la confianza en la autenticación, la administración de *tokens* de autorización y la comunicación segura. Finalmente, el *séptimo* componente es *Dashboard* el cual proporciona una interfaz basada en web para los administradores y usuarios de la nube. Al usar esta interfaz se puede aprovisionar, administrar y monitorear los recursos de la nube. En este panel se implementan de forma común y pública todos los aspectos de seguridad habituales de los portales web públicos. Los diversos componentes pueden observarse en la figura 2.5.

2.4.3. Apache CloudStack

Apache CloudStack es un software de código abierto diseñado para desplegar y administrar grandes redes de máquinas virtuales bajo la modalidad de infraestructura como servicio (IaaS) del paradigma de la computación en la nube. Se caracteriza por ser altamente escalable, por ello es utilizado por proveedores de servicios para ofrecer nubes públicas, privadas e híbridas [84]. La arquitectura de este gestor está compuesta por dos tipos elementos como ser

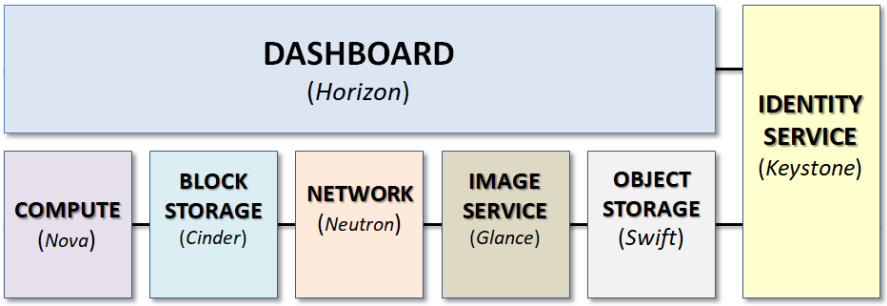

Figura 2.5: Diagrama que muestran los componentes de OpenStack (89).

el servidor de gestión y la infraestructura. El servidor de gestión administra por completo la nube y realiza la asignación de MV a los anfitriones (*hosts*). Una de las características notables en Apache CloudStack es la interfaz web que proporciona una administración completa de la nube de forma gráfica e intuitiva. También ofrece opciones muy interesantes, como la capacidad de definir máquinas virtuales altamente disponibles que Apache CloudStack mantiene operativas sin la intervención del usuario o del administrador y la instalación de un sistema operativo que utiliza una imagen ISO estándar. Su instalación puede realizarse a través de la interfaz web sin la necesidad de utilizar herramientas adicionales. La infraestructura está conformada por todo el hardware que es necesario para dar soporte a la nube.

Apache Cloudstack está compuesto en su operación interna por siete elementos, como ser: el servidor de administración de Apache CloudStack, la zona de disponibilidad, los *pods*, los clústers, los nodos de cálculo (también llamados nodos computacionales o CN) y los sistemas de almacenamiento primario secundario. El servidor de administración de Apache CloudStack controla la nube y la asignación de máquinas virtuales. Una zona de disponibilidad se puede ver como un sólo centro de datos compuesto por varios *pods* y al menos un sistema de almacenamiento secundario. Un clúster es una colección de CN que comparte el mismo hipervisor y el mismo sistema de almacenamiento primario. Los clústeres representan el segundo nivel de escalado. Un *pod* es una colección de clústeres y es una plataforma de *hardware* equivalente que incluye conmutadores (*switches*) de capa dos y representa el tercer nivel de escala física en la plataforma. Tanto los clústeres como los *pods* no son visibles para los usuarios finales. Los nodos de computación son sistemas habilitados por el hipervisor incluidos en la gestión de Apache CloudStack. El almacenamiento primario está asociado con el clúster y almacena el sistema de archivos raíz de máquinas virtuales invitadas. El sistema de almacenamiento

secundario está asociado con la zona de disponibilidad y almacena plantillas de MV, imágenes ISO e instantáneas de volúmenes del disco; convirtiéndose en el cuarto nivel de escalado físico. Gráficamente la operación de Apache CloudStack se puede apreciar en la figura 2.6.

Adicionalmente se puede decir sobre los nodos de computación (CN) que los mismos son en sí anfitriones habilitados para ser gestionados por el hipervisor posterior a la instalación y configuración del agente de Apache CloudStack. Estos anfitriones representan el bloque físico básico que nos permite escalar la plataforma de la nube. Se pueden agregar anfitriones adicionales en cualquier momento para aumentar la capacidad proporcionada para el despliegue y ejecución de máquinas virtuales (MV) invitadas. Los anfitriones no son visibles para los usuarios finales, por lo tanto estos no pueden determinar los anfitriones que se les han asignado para ejecutar sus máquinas virtuales invitadas.

Apache CloudStack admite tres roles de usuario: administrador raíz, administrador de dominio y usuarios sin privilegios. El administrador raíz, puede gestionar toda la nube sin restricción y es capaz de limitar a los demás usuarios. Los administradores de dominio pueden realizar las operaciones de gestión requeridas para que los usuarios que pertenecen a ese dominio puedan operar sin tener visibilidad de los CN físicos. El último, los usuarios no privilegiados, pueden administrar sus propias máquinas virtuales y desplegarlas a voluntad.

2.5. Aplicaciones

La mayor fortaleza de las tecnologías y paradigmas que han sido implementados a gran escala radica en la disminución de costos, velocidad para obtener resultados, integración y actualización entre otras. Las aplicaciones en la vida cotidiana toman ventaja de lo anterior para lograr nuevos productos y servicios de forma masiva que aprovechan la economía de escala para llegar a precios más competitivos al consumidor final. Las aplicaciones en la actualidad están en múltiples áreas, al punto que muchos sistema antiguos y que no eran migrados debido a su fiabilidad, hoy en día han sido portados a la nube. En los apartados siguientes se detalla el papel de la nube en diferentes contextos.

2.5.1. Negocios

La computación en la nube es reconocida como una tecnología que evoluciona cualquier proceso o actividad que se ejecuta sobre ella, a pesar de esto la idea de poseer negocios que operen completamente en la nube no es nueva y en la actualidad su potencial para condu-

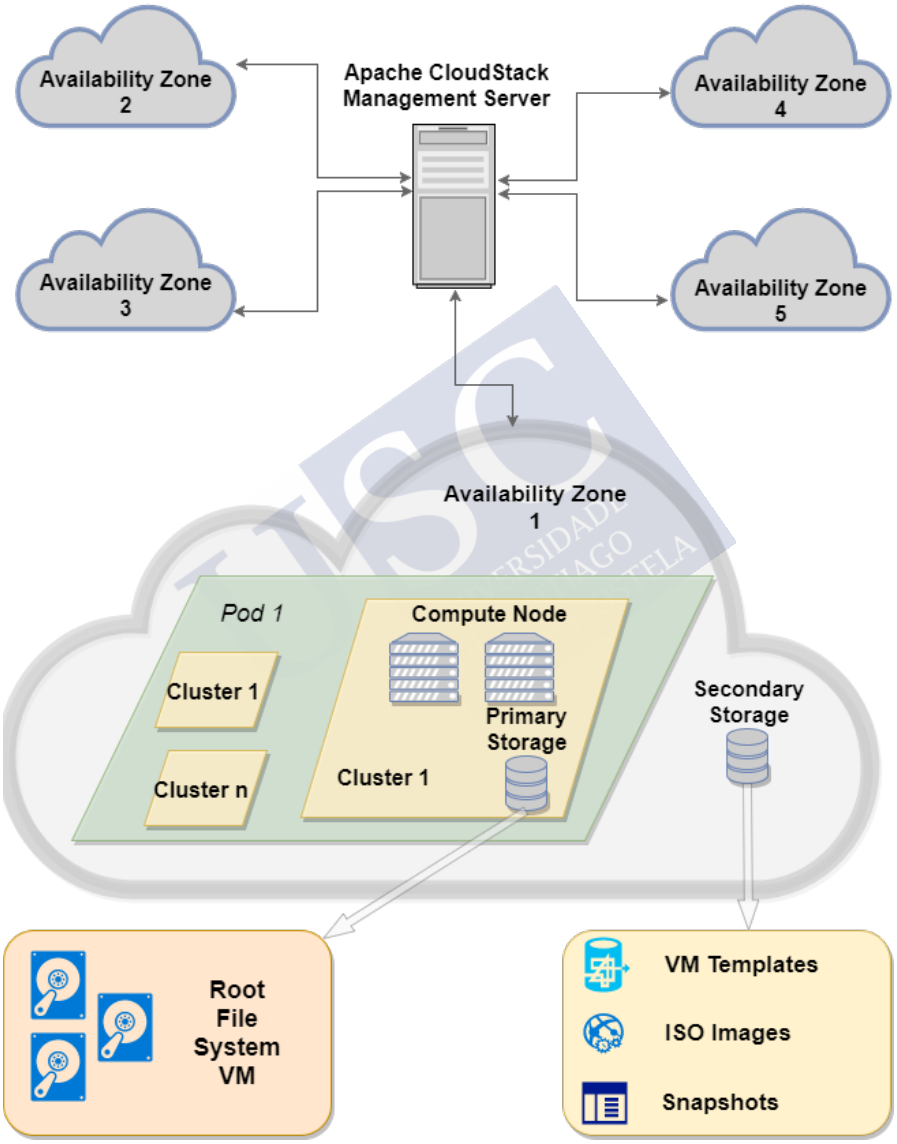

Figura 2.6: Diagrama de operación del gestor Apache CloudStack.

cirlos e innovarlos sigue aumentando. La nube ofrece la capacidad de cambiar los esquemas competitivos al ofrecer una plataforma para crear valor en los negocios. Se necesita que se diseñen varios modelos de negocios que aprovechen y mejoren las cadenas de valor entre el consumidor final y la industria junto con los proveedores de servicios; de manera que se promuevan ventajas competitivas de forma sostenible [31] [46]. En este aspecto los elementos a considerar son:

- **Flexibilidad en los costos de operación:** La nube ayuda a las organizaciones a reducir costos fijos por servicios informáticos al prescindir del despliegue en el sitio de todos los recursos necesarios para operar, sino que solamente se contrata lo que se necesita en el momento justo.

- **Escalabilidad de la operación de negocio:** Es la capacidad que permite a una organización escalar fácilmente sus operaciones comerciales a través de una rápida expansión de los recursos informáticas sin inversión exhaustiva de capital, beneficiándose de las economías a escala que definen al sector tecnológico.

- **Adaptabilidad al mercado:** En el contexto actual las empresas definen su futuro en el mercado conforme se adapten a las necesidades cambiantes de los clientes, volviéndose este un factor de diferenciación frente a la competencia. La nube ofrece las herramientas tecnológicas necesarias para ayudar en este sentido a través de procesos de prototipado, innovación y acelerando el tiempo de comercialización.

- **Complejidad enmascarada:** La nube ofrece a las empresas la capacidad de ocultar la complejidad de sus productos y servicios, de forma que el consumidor no se da cuenta y obtiene lo que busca de una forma sencilla y transparente. Al incrementarse la complejidad con el tiempo, se pueden realizar tareas de mantenimiento y actualización sin que el usuario note que esto ocurre, evolucionando constantemente.

- **Variabilidad basada en contexto:** Los servicios en la nube permiten la capacidad de almacenar información sobre las preferencias del consumidor, permitiendo personalizar productos o servicios a las necesidades del cliente. Las experiencias que se adaptan al cambio sutil del usuario y las necesidades fragmentadas de ellos resultan un factor diferenciador en los negocios.

- **Interconexión del ecosistema:** Es la facilidad de la colaboración externa con socios y clientes, lo que conducen a mejoras en la productividad y el aumento de la innova-

ción. Las plataformas basadas en la nube pueden reunir grupos dispares de personas que pueden colaborar y compartir recursos, información y procesos; logrando entre ellas productos y servicios de un nivel diferente.

En los próximos años se observará el desarrollo y evolución de la manufactura basada en la nube, que es considerado como un modelo de fabricación centrado en el cliente que explora el acceso bajo demanda a una colección compartida de recursos de producción diversificados y distribuidos para formar líneas reconfigurables. Lo anterior, incluye líneas temporales que mejoran la eficiencia, disminuyan los costos del ciclo de vida del producto; permitiendo emplear los recursos óptimos en respuesta a la demanda variable de las tareas generadas por el cliente [97].

La manufactura en la nube aprovecha el modelo de implementación de software como servicio a diferentes niveles de la cadena de producción [98]. En la etapa de diseño e ingeniería brinda recursos computacionales accesibles, mejora la eficiencia y el acceso a parámetros de diseño desde diferentes grupos colaboradores en múltiples regiones. Para el proceso de manufactura la nube ofrece las bondades de reducir costos, rápido prototipado y una mejora en la compartimento de recursos a través de una producción distribuida y en cadenas de suministros justo a tiempo. Para la etapa final de comercialización la nube ofrece mejorar la interacción con el cliente a través de evacuar las necesidades de este, porque impulsa la calidad y reduce el tiempo para que un producto o servicio sea comercializado.

La nube ofrece atractivamente la oportunidad de evitar los costos de capital e incurrir en gastos previsibles de forma escalable conforme se modifican las necesidades de una empresa, siendo los más beneficiados los negocios con uso ocasional de recursos o que consumen en ráfagas, ya que sólo pagan por los recursos cuando los están usando. Los negocios con patrones de uso estables también se benefician debido al menor costo al comprar servicios en la nube que desarrollarlos, implementarlos y mantenerlos. Lo anterior representa el mayor aporte a cualquier modelo de negocio que requiera de recursos de las tecnologías de la información [59].

2.5.2. Gobierno electrónico

En la actualidad el ciudadano se beneficia del menor costo de los ordenadores y dispositivos móviles, además cada día estos se vuelven más potentes en sus capacidades. Los gobiernos pueden interactuar mejor con los ciudadanos utilizando los medios digitales, ellos desarrollan

individualmente habilidades en lo relacionado al uso y acceso a tecnología simplificando el proceso de implementación de soluciones gubernamentales. La computación en la nube y específicamente la arquitectura orientada a servicios (SOA) representan las tecnologías puente hacia los ciudadanos en la actualidad, ya que estas permiten cubrir todo un país con soluciones de software accesibles en tiempo real. Se debe destacar que además en este caso las actividades de gobierno electrónico no se ven afectadas si las dependencias gubernamentales cuentan o no con los recursos y la preparación necesaria para este tipo de procesos.

La disminución de los costos, el tiempo para trámites y la disminución de los obstáculos burocráticos junto con la simplificación de los procesos en las instituciones estatales representan indicadores medibles que son altamente valorados por parte de los contribuyentes. Los servicios al ciudadano que pueden ser ofrecidos mediante las tecnologías mencionadas se conocen como *servicios integrados* que de forma típica son brindados por las diferentes unidades de la administración pública que reemplazan a aquellos ofrecidos en ventanillas y requiriendo al interesado estar de forma presencial; por otro lado se encuentran los *servicios de valor agregado* que es la serie de negocios electrónicos (*e-bussiness*) que se pueden ofrecer e integrar tomando como base todos las capacidades del gobierno electrónico y finalmente se identifican los *servicios varios* que se caracterizan por reunir todos los servicios brindados por unidades administrativas y de negocios del gobierno en conjunto, permitiendo de ser requerido interactuar con terceros para ofrecer una alternativa conjunta al usuario [12].

La computación en la nube proporciona acceso a recursos informáticos escalables que se encuentran remotos de una manera muy eficiente y rápida, este potencial ofrece cambiar drásticamente la forma como las personas interactúan con el gobierno y sus servicios. Esta es una oportunidad que permitiría a países en donde se carece de los medios económicos para el despliegue de soluciones basadas en tecnologías de la información, utilizar de forma más económica los recursos informáticos compartidos y accesibles bajo demanda en otros sitios geográficos con una mejor relación de costos comparado a si estos se ofreciesen de forma local. Lo anterior al unirse con las tecnologías móviles de la actualidad, hace posible brindar servicios de forma ubicua sin importar la hora o el sitio donde se encuentre el usuario final [31].

Lo que hace potente a la computación en la nube para los servicios de gobierno electrónico es la capacidad que tiene este paradigma de dividir los problemas y replicar soluciones a gran escala, haciéndolos accesibles mediante redes de área amplia como el Internet. La nube cambiará como los gobiernos toman decisiones, interactúan y responden a las necesidades

de los ciudadanos en áreas como salud, seguridad, saneamiento, servicios administrativos, procesos democráticos y en una finalidad casi infinita de servicios.

2.5.3. Televisión

Con el advenimiento de la nube, los sistemas dedicados para distribución de contenido televisivo están evolucionando a utilizar infraestructuras basadas en Internet, lo que implica un cambio en tanto la forma en que vemos la televisión, cómo se distribuye y llega a nosotros. La entrega y la producción del contenido actual está en la cúspide de la innovación y evolución; de forma que están haciendo cambiar desde sus bases a todo el sector. La entrega vía aire, cable y satélite gradualmente se ve reemplazada por Internet. Los responsables de entregar el contenido como ser los programadores (estaciones de TV) y los distribuidores de contenidos (productoras) están evolucionando sus operaciones y modelos de negocio a un entorno más ágil y flexible que ofrece la nube. Lo anterior significa tomar muchas de las funciones actualmente atendidas por hardware dedicado y software localizado, para moverlos a entornos de almacenamiento y computación distribuidos.

En los últimos años no se han visto cambios a gran escala en el servicio de televisión semejantes a los que se están dando en la actualidad, esto es semejante a lo que significó para esta industria el reemplazó de las ruedas mecánicas, el advenimiento del color, el uso de los transistores y la televisión digital en alta definición. La nube está ocasionando una rápida transformación, siendo una de las más destacadas el ahorro en los costos para los programadores y distribuidores; ya que desaparecen los medios físicos (discos compactos y cintas) reemplazados por medios digitales de menor costo, debido a que emplean el Internet para transporte, almacenamiento y diseminación de los contenido. El paradigma de la nube ofrece también eficiencia, capacidad para optimizar las operaciones y facilidad de escalabilidad, mejor monetización por reproducción y audiencia, entre otros. Todo ello ha hecho evolucionar la producción y distribución tradicionales de los flujos de trabajo (por ejemplo, desaparece la necesidad de codificación y transcodificación, la conversión de archivos entre formatos y el empaquetado de secuencias donde se prepara el contenido para la entrega multipantalla) ahora todas esas actividades pasan a la nube sin la necesidad de distribuidores, programadores, radiodifusores o grupos de estaciones locales para acceder hardware localizado, hasta el punto de la personalización del contenido para el usuario final.

En resumen, la migración de la televisión a la nube no sólo ofrece a los programadores ahorro y eficiencia de los distribuidores, pero también innovación que podría cambiar la te-

levisión tal como la conocemos hoy, sino que se ha vuelto más dinámica, personalizada y direccionable. Además se ha potenciado nuevas formas para obtener métricas verificables, caracterización y segmentación de la audiencia; ofreciendo una mejor experiencia a los televidentes. Este nuevo modelo de operación ofrece una mayor cantidad de contenido de calidad y permite el acceso a nuevos segmentos de televidentes, monetización eficiente y una mayor disponibilidad de contenidos a nivel global [71].

2.5.4. Ciencia

La nube aporta cambios a los servicios de computación científica, en cuanto a la arquitectura de hardware esta evoluciona de requerir un procesador con más hilos a explotar las técnicas de paralelismo para aprovechar múltiples unidades de cálculo. De la misma manera ocurre con los discos rígidos con partes móviles migran a los de tipo *flash* que ofrecen mejor rendimiento y una mayor confiabilidad. En cuanto a la administración de datos se migra de los repositorios centralizados a versiones distribuidas de los mismos, además con mayores recursos se desarrolla un paradigma conocido como *BigData* [64]. También la nube ofrece mejoras en cuanto a seguridad y la interacción entre grupos de investigadores [17].

Una de las ventajas que los investigadores disfrutan en la actualidad, es la posibilidad de contar con grupos de científicos diseminados alrededor del mundo, pudiendo colaborar con infraestructuras compartidas disponibles a través de internet. La ejecución de aplicaciones científicas sobre la nube sufren en muchos casos de pérdida de rendimiento, es por ello que las pruebas que comparen las ejecuciones locales con la nube son de utilidad para saber si el nuevo paradigma es aceptable a las necesidades de investigación en cada caso. En cuanto a costos, la nube ofrece la posibilidad de utilizar los recursos computaciones bajo demanda permitiendo a científicos contar con el recurso necesario para aplicaciones específicas y dentro del presupuesto que muchos proyectos puedan disponer, esta alternativa es de utilidad para las operaciones que demandan capacidad de procesamiento y de memoria intensiva. En cuanto al almacenamiento, se deben utilizar soluciones de agrupamiento para incrementar capacidad y resistencia a fallos [35].

CAPÍTULO 3

EVALUACIÓN DE LOS HIPERVISORES

Durante los últimos años se ha desarrollado de forma acelerada las infraestructuras en la nube (*cloud*), de la misma forma que los fundamentos tecnológicos que permiten su desarrollo e implementación, entre ellos se destaca la arquitectura orientada a servicios (*Service Oriented Architecture - SOA*, por sus siglas en inglés) y la llamada virtualización por hardware [106]. Esta última es uno de los principales fundamentos del paradigma de la computación en la nube, sobre todo a nivel de la implementación de infraestructuras como servicio (IaaS), que está basado en la gestión de los recursos físicos de hardware.

Para el diseño de una infraestructura de alto rendimiento para manejo de datos que plantea la adopción del modelo de computación en la nube basada en IaaS, es indispensable conocer el comportamiento de los diferentes hipervisores empleados por los gestores *cloud*. Esto se debe a que se plantea utilizar las tecnologías de virtualización para poder sacar provecho a los ordenadores que integren la solución computacional de alto rendimiento, ya que esta contará con máquinas virtuales que se ejecutarán bajo demanda de forma remota, en ellas se requerirá la implementación de una arquitectura distribuida para ser utilizada en aplicaciones cálculo intensivo. Esta infraestructura estará disponible para diversos fines, entre ellos el simulaciones, multimedios, aplicaciones científicas, entre otras.

3.1. Pruebas de referencia

Las comparativas de evaluación de referencia (también llamados *benchmarks*) tienen como finalidad medir el desempeño de un elemento informático concreto bajo condiciones controladas y su resultado o calificación puedan ser comparables con otros sistemas similares. Los

mismos pueden ser de tipo sintéticos o de aplicación, dependiendo lo que se desee evaluar en el sistema. En este documento se ha tenido el cuidado de evitar utilizar el término *rendimiento*, ya que se requiere tener la precaución de no usar de forma indiscriminada herramientas para evaluar el desempeño global de un sistema informático, de manera que el desempeño es algo muy complejo de determinar basado en algoritmos, problemas, implementaciones, entre otros. En este caso en particular, se han utilizado las pruebas siguientes descritas en las entradas siguientes.

3.1.1. Sintéticos

Este tipo de pruebas de referencia tienen como finalidad evaluar un componente determinado de un sistema computacional, típicamente se somete a carga de trabajo de múltiples formas y se miden en unidades apropiadas. A continuación se describen las pruebas utilizadas en las infraestructuras donde se evaluaron aplicaciones.

3.1.2. Linpack

Empleado para medir la capacidad de cálculo de la unidad central de procesamiento (CPU). Es una prueba de referencia de tipo sintético, siendo uno de los más ampliamente usados. Fue desarrollado en el *Argone National Laboratory*, su característica principal es hacer uso intensivo de las operaciones de punto flotante, sus resultados dependen de la capacidad de la unidad de punto flotante (FPU) de los sistemas evaluados [13].

Está compuesto por una colección de rutinas desarrolladas en el lenguaje FORTRAN para solventar distintos sistemas de ecuaciones lineales. De forma que dado un problema que involucra una matriz llamada A, esta se descompone en factores volviéndolos un producto de matrices simples y bien estructuradas que pueden manipularse fácilmente para resolver el problema original, esta prueba de referencia tiene la capacidad de manejar muchos tipos diferentes de matrices y tipos de datos, con ello se genera una gama amplia de opciones [18].

Linpack está basado en otros componentes llamados como Subrutinas Básicas de Algebra Lineal (*Basic Linear Algebra Subroutines - BLAS* [43]), estas abordan operaciones de vectores simples, como agregar múltiples vectores a otro vector (SAXPY) o formar un producto interno (SDOT). La mayor parte del trabajo de las operaciones de punto flotante dentro de los algoritmos Linpack es llevado a cabo por el BLAS, aprovechando el hardware informático ya sea común o especial, sin tener que modificar el algoritmo subyacente. Este enfoque otorga la

transportabilidad y la claridad del software sin sacrificar la confiabilidad de operación y sus resultados.

3.1.3. IOZone

Es utilizado para conocer una referencia del comportamiento de los procesos de entrada/salida (E/S) del sistema de almacenamiento secundario (disco rígido), midiendo la tasa de transferencia de datos en lectura/escritura hacia la unidad evaluada. Esta prueba es del tipo sintético y ha sido ideado específicamente para generar un punto de referencia que mide una variedad de operaciones de archivos [65]. IOzone ha sido portado a múltiples plataformas de ordenadores y se ejecuta bajo muchos sistemas operativos, es muy útil para realizar un análisis del sistema de archivos de diferentes plataformas de fabricantes y proveedores de ordenadores. IOZone permite hacer pruebas de desempeño de entrada y salida (E/S) en archivos en operaciones de *read*, *write*, *re-read*, *read backwards*, *read strided*, *fread*, *fwrite*, *random read*, *mmap*, *aio_read* y *aio_write*.

- **Read:** Es una prueba que mide el desempeño del sistema de almacenamiento al leer un archivo existente.

- **Write:** Consiste en medir el desempeño obtenido al escribir un nuevo archivo. En esta se considera que el proceso de escribir un archivo nuevo no sólo implica que se deben almacenar los datos, sino que también la información general para realizar un seguimiento de dónde se encuentran los mismos en el medio de almacenamiento, estos elementos descriptivos se les denomina *metadata*. Esta consiste en la información del directorio, la asignación del espacio y cualquier otro elemento asociado con un archivo que no es parte de los datos almacenados en él. Es normal, que el desempeño de escritura inicial sea inferior al compararse al momento de reescribir un archivo debido a esta información adicional.

- **Re-read:** Esta prueba mide el desempeño de leer un archivo que se leyó recientemente. Se considera normal que el rendimiento sea más alto ya que el sistema operativo generalmente mantiene un caché de los datos para los archivos que se leyeron recientemente, el cual se puede usar para satisfacer lecturas repetidas y mejorar la respuesta del sistema de archivos.

- **Read backwards:** Esta prueba evalúa el proceso de leer un archivo al revés, ya que hay aplicaciones realizan este tipo de lecturas, muchos sistemas operativos tienen características especiales que les permiten leer un archivo hacia adelante más rápidamente, pero muy pocos tienen la habilidad para detectar y mejorar el rendimiento de leer un archivo al revés.

- **Read strided:** Esta prueba mide los valores de referencia del sistema evaluado al momento de leer un archivo con un comportamiento de acceso restringido, que suele ser un comportamiento típico de aplicación que tienen estructuras de datos contenidas dentro de un archivo y que están accediendo a una región particular de la estructura de datos. La mayoría de los sistemas operativos no detectan este comportamiento, ni implementan ninguna técnica para mejorar sus tasas de lectura bajo este tipo de comportamiento de acceso.

- **Fread:** Mide el desempeño de leer un archivo usando la función *fread()* incorporada en múltiples sistemas operativos. Esta es una rutina que realiza operaciones de lectura bloqueadas con memoria intermedia (*buffered*). Tomando el hecho que el búfer está dentro del espacio de direcciones del usuario, una aplicación fuese a leer en transferencias de tamaño muy pequeño, la función *fread()* mejoraría las funciones de E/S leyendo bloques mayores y logrando mejorar el rendimiento de la aplicaciones reduciendo el número real de llamadas que el sistema operativo deba realizar. *Fread*, realiza lo mismo que la prueba *Fread* en operaciones de escritura, considerando que esta función mejoraría notablemente el desempeño ya que no hay necesidad de escribir el metadato debido a que el archivo ya existe.

- **Random read:** Mide el comportamiento al momento de leer un archivo con accesos a ubicaciones aleatorias dentro de él. Un sistema bajo este tipo de actividad puede verse afectado por varios factores como ser el tamaño de la memoria caché del sistema operativo, número de discos, latencias de búsqueda y otros.

- **Mmap:** Es una prueba diseñada debido a que muchos sistemas operativos admiten el uso de la función *mmap()* para asignar un archivo al espacio de direcciones de un usuario. Una vez que esta asignación esté en su lugar, almacenar en esta ubicación en la memoria hará que los datos almacenados vayan a un archivo, esto es útil si una aplicación desea tratar los archivos como trozos de memoria; de manera que mide la velocidad del uso de la función *mmap()* como mecanismo de E/S.

- **Aio_read y aio_write:** Son una serie de pruebas especiales útiles para evaluar el comportamiento de los sistemas que dan soporte a la interfaz portable de sistema operativo UNIX (POSIX) en actividades de E/S llamadas *async I/O*.

Las computadoras generalmente se compran para utilizarse en una aplicación específica, también es probable que con el tiempo la combinación de aplicaciones cambie, esto ocurre frecuentemente; es por ello que los sistemas operativos han hecho mejoras para tener un buen rendimiento en un entorno mixto. Aunque las mejoras usualmente aceleran las operaciones de E/S para unas pocas aplicaciones de uso más frecuente, por lo que es probable que el sistema no funcione muy bien con otras aplicaciones que no fueron diseñadas específicamente para ser ejecutadas sobre ese sistema. IOZone es útil para obtener valores de referencia del comportamiento del sistema de archivos para que pueden ser comparados con otras plataformas y conocer esa manera cual es el más apropiado para la aplicación que se desea utilizar.

3.1.4. Pruebas de referencia usando aplicaciones

En este tipo de pruebas de referencia se ha considerado la simulación de nanodispositivos la cual es una aplicación de cálculo científico para realizar simulaciones [38][78] de componentes electrónicos conocidos como transistor de efecto de campo metal-óxido-semiconductor (*Metal Oxide Semiconductor Field Effect Transistor* o MOSFET por sus siglas en inglés), empleados para amplificar señales electrónicas y ampliamente usado en la actualidad, sobre todo en cuanto a la producción de microprocesadores.

Estos dispositivos se reducen progresivamente con el fin de mejorar sus prestaciones, sin embargo cuando la longitud de la compuerta (*gate length*) de los dispositivos semiconductores se reduce a dimensiones nanométricas, varias fuentes de fluctuaciones pueden afectar significativamente su rendimiento y fiabilidad. En estas escalas, la inestabilidad de los parámetros son introducidas no sólo por el proceso de fabricación en sí, sino también por la discretización fundamental de la materia [5]. Estas fluctuaciones microscópicas afectan el rendimiento del dispositivo independientemente de la calidad del proceso tecnológico.

Este tipo de simulaciones en la que es necesario ejecutar cientos o miles de veces, es requerido por los estudios de las fluctuaciones de parámetros de dispositivos semiconductores 3D y es en cuanto a recursos computacionales muy exigente. Las operaciones de memoria y de punto flotante necesarias para este tipo de simulaciones hace necesario el uso de técnicas de

paralelismo, incluyendo el uso de algoritmos apropiados para obtener el máximo rendimiento y reducir el tiempo de simulación [73].

Esta prueba de referencia simula de forma paralela un dispositivo 3D usando el modelo deriva-difusión (D-D), desarrollado para estudiar las fluctuaciones intrínsecas de los parámetros en semiconductores basados en heteroestructuras como transistores de alta movilidad de electrones (HEMT) o heteroestructuras libre de implantes (MOSFET)[80] [82].

Las ecuaciones D-D se discretizan utilizando el método de elementos finitos (FEM)[49] en una malla tetraédrica no estructurada. El conjunto de ecuaciones obtenido se resuelve en paralelo en un número de procesadores utilizando la biblioteca estándar de la interfaz de paso de mensajes (MPI) [60]. El esquema de solución se basa en el desacoplamiento y la linearización de las ecuaciones D-D usando iteraciones de Gummel. Finalmente, los métodos de descomposición de dominio, implementados por la biblioteca PSPARSLIB [74], se han empleado para resolver la fracción local de los sistemas lineales asignados a cada procesador.

3.2. Evaluación de hipervisores

En este apartado se describen los pasos seguidos para evaluar la infraestructura física en conjunto con los hipervisores, empleando herramientas diseñadas para medir las variables de referencias de los diferentes componentes de hardware, obteniendo de esta forma valores comparables con otros entornos [101][105].

3.2.1. Metodología

Esta se enfocó en la ejecución de las pruebas de referencia y la comparación de sus resultados sobre la máquina física y posterior una máquina virtual gestionada por cada hipervisor, se ejecutó en las mismas condiciones para evitar la influencia de factores espurios que alteraran las mediciones obtenidas. Las consideraciones con IOZone, fue utilizar una partición con una capacidad de 20 GB ubicada al final del disco duro. En ella se han grabado los datos temporales durante la ejecución, evitando de esta forma que los mismos se dispersen por diversas partes del disco e introduzcan condiciones de variabilidad y que vuelvan irreales los resultados. Respecto a Linpack se utilizó la versión 10.3.3 que es parte de las herramientas *Math Kernel* de Intel. Las máquinas virtuales y los sistemas anfitriones reunían las características de ejecución siguientes:

- En todos ellos se desactivó la memoria de intercambio (SWAP).

- Se empleó el sistema de archivos a nivel de partición EXT2.
- Se utilizó la misma cantidad de memoria RAM para la ejecución de pruebas en todos los sistemas (1 GB).
- Se desactivó la funcionalidad de *hyperthreading* del microprocesador y además sólo se ha utilizado un núcleo físico de procesador en todas las pruebas.
- En el caso específico del IOZone, se utilizó una partición al final del disco, ejecutándose bajo las mismas condiciones y en la misma región.
- Se desactivó la utilidad *CPUSpeed*, que viene integrada en varias distribuciones de Linux.
- En KVM se empleó la utilidad *virt-manager* para gestionar las máquinas virtuales y además en ellas se emplearon los controladores VirtIO [41].
- Los sistemas estaban siendo usados de forma exclusiva para las pruebas de rendimiento, no contaban con protector de pantalla ni otro tipo de aplicaciones.

3.2.2. Infraestructura empleada

Se utilizó el hipervisor Xen 4.0.2-rc3 con Linux kernel 2.6.32.37 sobre CentOS Linux 5.6 (Xen4), Xen 3.1.2-238.9.1.el5 con Linux kernel 2.6.18-238.9.1.el5xen sobre CentOS 5.5 (Xen3) y KVM sobre Ubuntu 10.04.2 LTS con Linux kernel 2.6.32-31-server en combinación con la versión de qemu-kvm 0.12.3(KVM). Todo este software es de código abierto y se encuentran entre los más ampliamente usados en la computación de alto rendimiento.

En cuanto a los recursos de hardware se empleó un ordenador del fabricante Toshiba modelo Qosmio X500 con un procesador Intel Core i7 Q720 @ 1.60 GHz, con 4 GB de RAM y disco rígido de 500 GB de 7200 RPM interfaz SATAII para las pruebas del hipervisor KVM. Para Xen, se utilizó un ordenador de escritorio con procesador Intel Xeon E5520 @ 2.27 GHz, con 16 GB de RAM y disco rígido de 500 GB de 7200 RPM; a este conjunto de equipos las llamaremos "*Equipamento 1*".

Por otro lado, se utilizó otra plataforma basada en hardware básico (*commodity hardware*), que consistía en un ordenador con un procesador Intel Core 2 Duo E7500 @ 2.93 GHz, con 4 GB de RAM y disco duro de 500 GB también con una velocidad de rotación de 7200

	Equipamento 1		*Equipamento 2*	
Sistema	**GFlops**	**Tiempo(s)**	**GFlops**	**Tiempo (s)**
Máquina física	8.36	59.49	10.31	49.15
Anfitrión KVM	8.61	57.30	10.31	49.25
MV KVM	5.54	85.07	5.24	88.26
Anfitrión Xen3	8.44	59.59	10.27	49.07
MV Xen3	6.76	74.47	9.93	51.38
Anfitrión Xen4	8.16	60.95	9.02	55.38
MV Xen4	8.12	61.13	8.73	62.01

Tabla 3.1: Resultados con Linpack, detalla capacidad de cálculo y tiempo requerido de ejecución.

RPM y conectado mediante una interfaz SATAII. Lo anterior es con el fin de conocer el comportamiento en este tipo de equipos, ya que es la configuración más común en los laboratorios de mi universidad de origen al momento de realizar las pruebas aquí descritas, este conjunto lo denominaremos "*Equipamento 2*". Se utilizaron las mismas versiones del sistema operativo y de los hipervisores que en el *Equipamento 1*, se han seguido los mismo criterios como los usados con la primera infraestructura tal como está descrito anteriormente en este apartado.

3.2.3. Resultados

Las diversas pruebas de referencia realizadas de tipo sintéticas y de aplicación se describen en los apartados siguientes e incluyen los resultados obtenidos:

3.2.4. Linpack

Se utilizó un tamaño de problema con un valor de 5000 en lo referente a la dimensión y con valores de alineación de 4 KBytes. El resultado es dado en unidades de GigaFLOPS (GFlops) y se pueden observar en la tabla 3.1:

Para el *Equipamento 1*, específicamente la máquina virtual (MV) en KVM tiene una disminución en la capacidad de cálculo de 3.07 GFlops respecto al anfitrión; requiriendo la MV de 27.77 segundos adicionales para completar la prueba. En Xen3, la MV reportó una medición inferior que el sistema anfitrión, perdiendo 1.68 GFlops y requiriendo 14.88 segundos adicionales para completar la prueba. En Xen4, los resultados fueron muy parecidos entre la MV y el anfitrión, donde se presentó una mínima pérdida de 0.04 GFlops en el sistema invitado y un tiempo similar. Es importante mencionar que el anfitrión Xen3 obtuvo resultados más

altos en esta prueba de rendimientos que en el anfitrión sin el kernel modificado del hipervisor, superando a éste por 0.08 GFlops, en cambio el anfitrión Xen4 reportó valores inferiores al ser comparado con el sistema con kernel original perdiendo 0.20 GFlops.

Para el *Equipamento 2*, tomando como referencia la máquina física comparada con el anfitrión KVM se obtuvo una medición igual pero completó la evaluación un décimo de segundo más tarde. Por otro lado la MV KVM perdió 5.07 GFlops y necesitó 39.01 segundos más que el anfitrión KVM. El anfitrión Xen3 redujo su medida en capacidad cálculo en 0.04 GFlops al ser comparado con el ordenador físico y frente a la MV Xen3 esta última perdió 0.34 GFlops y utilizó adicionalmente 2.31 segundos. El anfitrión Xen4, perdió 1.29 GFlops frente al computador físico y al compararlo con la MV Xen4, esta restó 0.29 GFlops necesitando 6.63 segundos adicionales para completar la prueba.

Las diferencias entre las mediciones obtenidas con los anfitriones KVM y sus MV en ambos equipamentos pudo ser ocasionado por un mal mapeo de las instrucciones SSE3 del microprocesador hacia los anfitriones. Los mejores resultados mostrados en las pruebas del *Equipamento 2* se debe a que el CPU usado tiene una frecuencia mayor que la del *Equipamento 1*. La MV KVM del *Equipamento 2* fue la que requirió más tiempo para completar la prueba de todos los sistemas invitados, pero en cambio su anfitrión fue el más eficiente de todos.

3.2.5. IOZone

En esta prueba, se evaluó la tasa de transferencia para completar un proceso de escritura de un archivo de 4 GB. En la tabla 3.2, se muestran los resultados por cada uno de los sistemas y entornos evaluados:

En el conjunto llamado *Equipamento 1*, los mejores resultados se obtuvieron en la tasa de escritura al emplear el sistema con el kernel sin modificar, en la tasa de re-escritura el mejor resultado fue el del anfitrión con Xen4. En KVM, la MV obtuvo en la tasa de escritura una transferencia 1779 Kbytes/s inferior al ser comparada con su anfitrión. En la tasa de re-escritura ocurrió lo mismo pero con una diferencia de 3510 Kbytes/s. En el caso de Xen3, la MV presentó una diferencia de 45718 Kbytes/s en la tasa de escritura y de 34806 Kbytes/s en la tasa de re-escritura, siendo esta inferior a la del anfitrión. En Xen4, los valores obtenidos de la MV fueron inferiores con una diferencia de 37030 Kbytes/s en tasa de escritura y de 33907 Kbytes/s en la tasa de re-escritura frente al sistema anfitrión. En estas pruebas, todos

Sistema	*Equipamento 1*		*Equipamento 2*	
	Tasa Escritura	**Tasa Re-Escritura**	**Tasa Escritura**	**Tasa Re-Escritura**
Máquina física	95605	67530	93775	94236
Anfitrión KVM	47273	50416	93845	95551
MV KVM	45494	46906	84753	77308
Anfitrión Xen3	95744	66445	95638	95465
MV Xen3	47026	31639	63383	66749
Anfitrión Xen4	86836	95387	93277	91259
MV Xen4	49797	61480	64526	68255

Tabla 3.2: Tasas de transferencia en Kbytes/s de escritura y re-escritura en IOZone.

los anfitriones reportaron valores inferiores que la máquina física con el kernel sin modificar que trae por defecto la distribución del sistema operativo usado.

Con el *Equipamento 2*, las pruebas realizadas en cuanto a escritura, el anfitrión Xen3 con una tasa de 95638 Kbytes/s tuvo el mejor resultado y la mejor tasa de re-escritura fue del anfitrión KVM con una medida de 95551 Kbytes/s. En KVM, la MV reportó una tasa de escritura de 9092 Kbytes/s inferior al anfitrión y en la tasa de re-escritura se mantuvo la tendencia con una diferencia de 18243 Kbytes/s. En el caso de Xen3, la MV presentó una diferencia en los resultados de 32255 Kbytes/s en tasa de escritura y de 28716 Kbytes/s en la tasa de re-escritura, siendo ambas inferiores al anfitrión. En Xen4 los valores reportados de la MV fueron inferiores con una diferencia de 33907 Kbytes/s en tasa de escritura y de 23004 Kbytes/s en la tasa de re-escritura frente al sistema anfitrión. Se mantiene un comportamiento de los anfitriones superando a sus sistemas invitados.

Entre ambos conjuntos de *hardware*, las pruebas utilizaron los controladores VirtIO sobre KVM que emplean por defecto caché para la gestión del almacenamiento, en modo *WriteThrough* y ello repercute en los resultados obtenidos. Xen3 presentó mejores valores en sus anfitriones, pero Xen4 fue más eficiente a nivel de sistemas huésped.

3.2.6. Simulación de nanodispositivos

El resultado de la ejecución de simulación es dada en unidades de segundos y representa la medición del tiempo requerido para completarse. Los mejores resultados son los que cuentan con cifras de menor valor, los mismos se pueden comparar en la tabla 3.3.

En el conjunto del *Equipamento 1*, al comparar las MV con sus respectivos anfitriones,

	Equipamento 1
Sistema	**Tiempo (s)**
Máquina Física	4406.59
Anfitrión KVM	3600.89
MV KVM	4477.36
Anfitrión Xen3	4901.35
MV Xen3	5341.71
Anfitrión Xen4	4486.63
MV Xen4	4859.74

Tabla 3.3: Resultados de la simulación de nanodispositivos, valores en segundos.

encontramos que la MV en KVM requirió 876.47 segundos adicionales para completar la simulación. En el caso de la MV en Xen3, necesitó de 440.35 segundos más para completar la prueba y en la MV en Xen4 requirió de 373.11 segundos más que el anfitrión. La MV y anfitrión más eficientees fueron los de KVM, llegando a superarse a la misma máquina física con el kernel original y sin modificar del sistema operativo empleado, esto denota que el kernel con KVM posee mejoras en el manejo del almacenamiento en disco.

Sólo se ha evaluado el *Equipamento 1*, debido a que es el hardware con prestaciones semejantes a la de los servidores y estaciones de trabajo, siendo esta la configuración requerida para la ejecución de la prueba de referencia de nanodispositivos; el *Equipamento 2* no reúne las condiciones mínimas de capacidad y por eso no se consideró.

3.3. Conclusiones

Para el *Equipamento 1*, la MV que se ejecutó con el hipervisor Xen4 presentó los mejores resultados en Linpack; ya que la diferencia con el anfitrión fue menor del 1 %. En la prueba de IOZone se evaluó la tasa de transferencia en modo de escritura en la cual la MV bajo KVM presentó la menor pérdida respecto al anfitrión, siendo un 3.76 %. En cambio, los resultados obtenidos con la MV en Xen3 y Xen4 en la misma prueba, mostraban una pérdida mucho más pronunciada. Cuando se evaluó la tasa de transferencia en modo re-escritura, la MV en KVM sólo perdió un 6.96 % frente al anfitrión, siendo mucho más eficiente que la MV en Xen3 y Xen4.

En la prueba de cálculo científico, presentó una mayor eficiencia la MV en Xen4 (con poca diferencia con el Xen3), ya que requirió únicamente un 8 % más tiempo que el anfitrión

para completar la simulación; lo cual representa mejores resultados que la MV en KVM. Se puede concluir que las MV que se ejecutan con el hipervisor Xen requirieron menos tiempo para completar las tareas evaluadas que estaban relacionadas con el cálculo y procesamiento intensivo. En cambio, la situación es distinta en cuanto a la tasa de transferencia de escritura en disco, en la cual la MV sobre KVM mostró un mejor desempeño que la MV de Xen.

En los resultados obtenidos se observa una diferencia considerable entre las capacidades de las MV dependiendo del uso que se haga del disco o del procesador, consideramos que los controladores VirtIO pueden ser los causantes de las diferencias entre los hipervisores evaluados.

En el caso del *Equipamento 2*, las pruebas realizadas con Linpack los anfitriones de KVM y Xen3 no mostraron cambios significativos en los resultados cuando son comparados con la máquina física, pero el anfitrión Xen4 perdió un 12.51 % en la misma operación. Las MV presentaron resultados heterogéneos, de forma que la MV KVM obtuvo la pérdida menos pronunciada comparada con su anfitrión con una proporción de 9.88 %, pero la más penalizada fue la MV Xen4 con una caída del 72.90 %. La MV Xen3 y la MV Xen4 sólo perdieron un 3 % de eficiencia frente a sus anfitriones, pero la KVM MV perdió un 49.18 %.

En IOZone, la tasa de escritura del anfitrión Xen3 fue 2 % superior que la de sus contrapartes; pero en general no hubo diferencias significativas frente al ordenador base. En la tasa de re-escritura los anfitriones mantuvieron un comportamiento muy similar al ordenador anfitrión, las diferencias más pronunciadas fueron la del anfitrión KVM que obtuvo un rendimiento mayor en 1.4 % mientras que el anfitrión Xen4 fue menos eficiente por un 2.87 %. En cuanto a los sistemas invitados, la MV KVM obtuvo los mejores resultados en la tasa de escritura y en la tasa de re-escritura, perdiendo un 9.69 % y 19.09 % respectivamente frente al anfitrión. La MV Xen3 perdió 33.73 % en la tasa de escritura y un 30.8 % en la de re-escritura.

De los resultados obtenidos se deduce que el hipervisor KVM maneja a nivel de anfitriones un rendimiento muy similar con el ordenador físico pero sus MV pierden capacidad de cálculo lo que las convierte en una mala opción para aplicaciones que requieran virtualizar aplicaciones de procesamiento, pero para las que requieran acceso al sistema de archivos, su rendimiento será el mejor. Por otro lado, Xen tiene mejores capacidades si se requiere procesamiento, pero presenta un desempeño reducido en el acceso al sistema de archivos, perdiendo hasta un terció de la capacidad física en sus MV.

Capítulo 4

Casos de uso

La informática científica implica el diseño de modelos matemáticos generalmente caracterizados por requerir alta capacidad computacional para abordar los nuevos desafíos [93]. Debido a que la complejidad de los cálculos y modelos crece con el tiempo, las aplicaciones usadas generalmente requieren recursos informáticos con mayor capacidad. Las infraestructuras de altas prestaciones proporcionan a los usuarios la potencia informática necesaria para abordar estos problemas, pero su costo, complejidad y el mantenimiento dificulta su uso. Por otra parte, en los últimos años surgió una nueva alternativa conocida como computación en la nube [95]. La principal ventaja de la nube es la posibilidad de escalar según la demanda y la potencia informática requerida, generalmente en forma de máquinas virtuales (VM) que ejecutan el software del usuario, sobre una cierta infraestructura física.

En este apartado se detallan las aplicaciones analizadas y evaluadas, que se encuentran en uso por los investigadores de mi universidad al momento de la redacción del presente documento. Se han agrupado de acuerdo a las áreas de estudio que mayor trabajo científico y técnico realizan.

4.1. Astronomía y astrofísica

La astronomía se define como la ciencia del universo, estudia el movimiento, estructura, origen y desarrollo de los cuerpos celestes asi como de los sistemas que estos conforman [72]. Lo anterior lo hace por medio de tres tareas fundamentales: la *primera*, es el estudio de las posiciones, movimientos reales y aparentes de las estrellas. La *segunda*, es el estudio de la estructura física y de la materia que compone los cuerpos celestes. La *tercera*, es el

desarrollo y resolución de los problemas de la posible evolución y destino de los cuerpos celestes, incluyendo los sistemas que estos conforman.

Dado lo anterior, esta ciencia se divide en astrometría o astronomía posicional, astrodinámica o mecánica celeste y astrofísica [6]. Este último campo, está dividido en cosmología, astrofísica, radioastronomía, astronomía de rayos X, rayos gamma, infrarrojos, ultravioleta y estadísticas estelares. Siendo más específico, la radioastronomía estudia las estrellas y estructuras que emiten radiaciones en la longitud de onda que corresponde al segmento de radio del espectro electromagnético.

4.1.1. GADGET2

GADGET2 es un conjunto de programas que su código fuente está libremente disponible para la realización de simulaciones cosmológicas de N-cuerpos considerando la hidrodinámica de partículas suavizadas (SPH - *Smoothed-particle hydrodynamics*) [8][54] para usarse en infraestructuras paralelas de memoria distribuida. GADGET2 utiliza un modelo de comunicación implementado con la interfaz MPI estandarizada [88].

El código fuente se puede ejecutar en prácticamente todos los sistemas de súper ordenadores en uso actualmente, incluyendo clústeres conformados por estaciones de trabajo y ordenadores personales individuales. GADGET2 realiza los cálculos de las fuerzas gravitacionales usando un algoritmo de árbol jerárquico (opcionalmente puede utilizar de manera combinada con un esquema de malla de partículas para considerar las fuerzas gravitacionales de largo alcance) y permite representar los fluídos por medio de hidrodinámica de partículas suavizadas (SPH). El código puede utilizarse para estudios de sistemas aislados o para simulaciones que incluyan la expansión cosmológica del espacio, con o sin condiciones de frontera periódicas. En todos estos tipos de simulaciones, GADGET2 sigue la evolución de un sistema de N-cuerpos auto-gravitante sin colisión y permite opcionalmente incluir la dinámica de gases estelares. Tanto el cálculo de la fuerza como el paso del tiempo de GADGET2 son totalmente adaptativos, con un rango dinámico que en principio es ilimitado [104].

Esta herramienta de software se puede utilizar para abordar una amplia gama de problemas astrofísicamente interesantes, que van desde la colisión y la fusión de galaxias hasta la formación de una estructura a gran escala en el Universo. Con la inclusión de procesos físicos adicionales como el enfriamiento y calentamiento radiativo, GADGET2 también puede usarse para estudiar la dinámica del medio intergaláctico gaseoso o bien para abordar la formación estelar y su regulación mediante procesos de retroalimentación [83].

Infraestructura

GADGET2 fue ejecutado sobre tres infraestructuras como ser una máquina virtual, un clúster virtual y en un ordenador físico anfitrión que los contenía. El objetivo es determinar cuál de estas plataformas resultaba más eficiente al momento de tomar ventaja de la tecnología de virtualización. Se ha considerado el efecto de la sobre demanda en la ejecución, de manera que se midió el tiempo de cálculo con el *hyperthreading* habilitado y deshabilitado. Estas plataformas se detallan a continuación:

- **Anfitrión físico:** El anfitrión físico cuenta con un microprocesador Intel Core i7-2600 con una velocidad de reloj de 3.40 GHz, con 8 GB de memoria RAM y conexión de área local tipo Gigabit Ethernet ejecutando el sistema operativo (SO) CentOS Linux 6.2 x86 de 64 bit. Las pruebas se realizaron empleando 1, 2, 4, 6 y 8 hilos de proceso.

- **Máquina virtual:** Se ha desplegado sobre el anfitrión físico, se compone de una única MV gestionada por el hipervisor KVM, contando con 1 GB de RAM, con un número de *cores* variable, que se ha ido incrementando de 1 a 8, según las necesidades de procesamiento de GADGET2. De esta forma se puede evaluar el *overhead* que introduce la infraestructura virtualizada con KVM sobre el desempeño de la aplicación. El sistema operativo de la MV es CentOS 6.2 x86 de 64 bit, de forma idéntica al ordenador físico.

- **Clúster virtual:** Esta infraestructura también fue desplegada sobre el anfitrión físico descrito, este clúster está conformado por ocho MV, que se inician de manera progresiva en función de las necesidades de ejecución. Cada una de estas MV se le asignó 1 GB de RAM y un núcleo físico para poder operar. El clúster utiliza una MV para hacer la función de *Head* y el resto hacen el papel de *Nodes*. Se diferencian entre sí, porque las segundas carecen de las herramientas capaces de compilar aplicaciones, de manera que sólo se configuran como nodos de procesamiento. Es importante mencionar que los *Nodes* se despliegan de forma paulatina en función de los requerimientos computacionales, de esta manera podemos evaluar el rendimiento de GADGET2 empleando MPI [60] para comunicarse con los diferentes procesos que ahora estarán distribuidos en múltiples MV usando el clúster.

Resultados

Para las mediciones, se empleó el ejemplo de formación cosmológica de un clúster de galaxias que se incluye en la distribución de GADGET2, en la que se ejecuta una dinámica sin colisiones en un universo en expansión. Se desarrolla una simulación de un pequeño clúster estelar en el que se han colocado las condiciones iniciales definidas por Bepi Tormen, que establecen alrededores vacíos y el empleo de técnicas multimasa. La simulación consta de un total 276498 partículas y al concentrarse en la zona de mayor resolución se llegan a contabilizar un total de 140005 partículas, contiguo a ella se encuentran dos capas, la interior con 39616 y la exterior con 96877 partículas respectivamente. Para los fines de las pruebas de esta aplicación se ha acotado el tiempo de simulación ajustando el parámetro *TimeMax* de 1.0 a 0.5, en el archivo de condiciones de ejecución. Los resultados obtenidos se describen a continuación:

Con hyperthreading deshabilitado

Los resultados se muestran en la figura 4.1, donde se puede comparar todas las mediciones a partir del número de núcleos empleado. A continuación se muestras los detalles:

- **Anfitrión físico:** El tiempo requerido entre ejecuciones fue similar hasta que en la simulación se utilizó un número de procesos igual a la cantidad de *cores* disponibles en el procesador. De forma que si utilizan más recursos el rendimiento se ve afectado de forma negativa más allá de esta configuración identificada. El menor tiempo requerido para completar la simulación fue al utilizar cuatro núcleos, que al compararse con un núcleo se necesitaron 1114 segundos adicionales, con dos núcleos se requerió adicionalmente de 392 segundos, con seis se utilizaron 166 segundos más y finalmente con ocho el tiempo aumentó 75 segundos.

- **Máquina virtual:** El menor tiempo requerido para completar la simulación fue al utilizar cuatro núcleos, que al compararse con un núcleo se necesitaron 1146 segundos adicionales, con dos núcleos se requerió adicionalmente de 394 segundos, con seis se utilizaron 905 segundos más y finalmente con ocho el tiempo aumentó 1157 segundos para finalizar la simulación. Al igual que en la infraestructura anterior, el menor tiempo se obtiene cuando el número de núcleos iguala a la disponibilidad de *cores* físicos del procesador del ordenador anfitrión. La degradación entre la MV fue superior al compararse al ordenador físico cuando se empleaban más recursos, se necesitó 76 % de tiempo

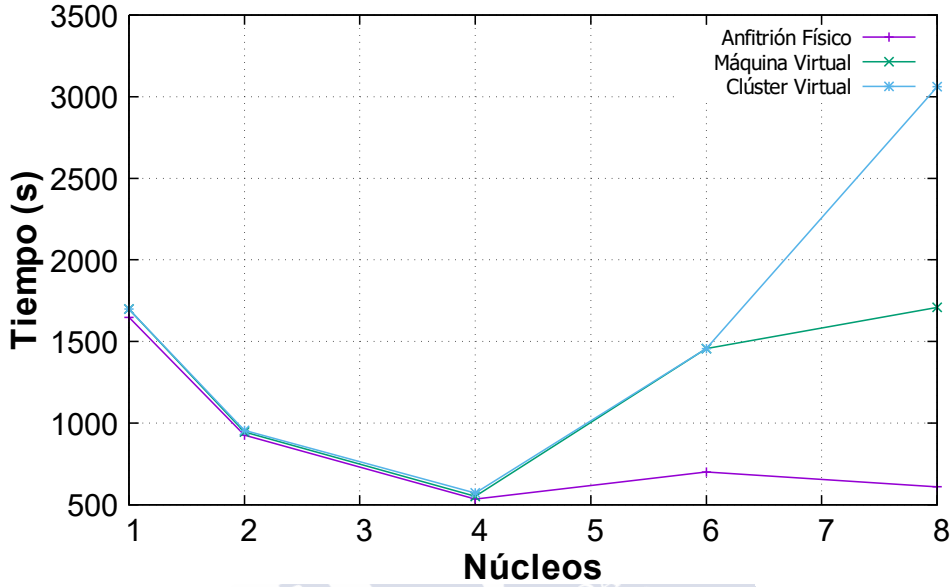

Figura 4.1: Comparación del tiempo requerido (en segundos) para completar las simulación en las diferentes infraestructuras cuando estas no poseen el soporte del *hyperthreading*.

adicional con cuatro *cores* y aumentando hasta requerir de un 180 % más de tiempo al emplear ocho *cores*.

- **Clúster virtual:** El tiempo requerido sólo ejecutando la simulación con el *Head* fue de 1698 segundos, siendo similar al obtenido con la MV de un sólo núcleo. Al agregar un *Node*, se redujo el tiempo en 742 segundos, con cuatro *Nodes* disminuyó en 1126 segundos, con seis *Nodes* el tiempo fue inferior por 241 segundos y con ocho *Nodes* el tiempo se incrementó, siendo superior por 1362 segundos; siendo esta la peor medición de todas las pruebas en esta plataforma. El comportamiento resultante fue similar a los observados en los casos de uso de las máquinas virtuales y del anfitrión físico previamente evaluados, ya que al aumentar el número de *cores* físicos se degrada la ejecución, ya que se requirió hasta de un 179 % de tiempo adicional para completar la simulación, si tomamos como referencia a una MV con ocho núcleos.

Con hyperthreading habilitado

Como se indicó en los párrafos anteriores es de interés conocer si la funcionalidad del *hyperthreading* ofrece alguna ventaja en la ejecución de GADGET2, en este apartado se detallan los hallazgos usando dicha funcionalidad activa. Los resultados se muestran en la figura 4.2, donde se puede comparar todas las mediciones considerando el número de núcleos empleado.

- **Anfitrión físico:** Al emplear un sólo núcleo se requirió de 1648 segundos, se redujo en 720 segundos con dos núcleos, la tendencia se mantuvo con cuatro núcleos reduciéndose 1112 segundos, con seis núcleos el tiempo aumentó, ya que la diferencia fue de 1046 segundos y con ocho la diferencia es de 1149 segundos; siendo este el menor tiempo de ejecución en esta serie de pruebas. Al emplear para la ejecución entre uno y cuatro núcleos se obtuvo como resultado tiempos similares a los reportados por el anfitrión físico sin el uso de *hyperthreading*, valores muy diferentes se midieron al usar seis u ocho núcleos donde se obtienen mejorías del 14 % y 18 % respectivamente. Los datos obtenidos indican que los mejores resultados se dan cuando el número de procesos a utilizar igualan al número máximo de núcleos lógicos disponibles en la unidad central de procesamiento.

- **Máquina virtual:** En la ejecución en esta instancia se requirió de 1676 segundos al emplear un núcleo, el tiempo se reduce en 722 segundos con dos núcleos, disminuye en 1113 segundos con cuatro núcleos, siendo este el menor tiempo requerido de esta serie de pruebas. El tiempo aumentó con seis núcleos, ya que la diferencia fue 1062 segundos y volvió a disminuir con ocho núcleos, ya que la diferencia fue de 1108 segundos. El menor tiempo de ejecución se obtiene cuando la MV iguala al número de núcleos físicos del sistema anfitrión y al activar el *hyperthreading* mejora el rendimiento en un 48 % con seis núcleos y un 66 % con ocho núcleos; estas mediciones comparadas con las pruebas realizadas con el *hyperthreading* deshabilitado en la misma instancia.

- **Clúster virtual:** Con una MV se requirió para completar las operaciones de 1676 segundos (estaba únicamente ejecutándose en el *Head*). Al agregar un *Node*, el tiempo disminuyó en 729 segundos y al colocar cuatro MV decreció en 1129 segundos para completar la prueba. El tiempo se incrementó al usar seis MV, ya que la diferencia fue de 1062 segundos y con ocho MV el tiempo se redujo en 1155 segundos, siendo este el mejor tiempo obtenido. Al comparar el clúster virtual con *hyperthreading* este obtuvo mejoras, reduciendo el tiempo de ejecución en un 42 % usando seis MV y de un 82 % con

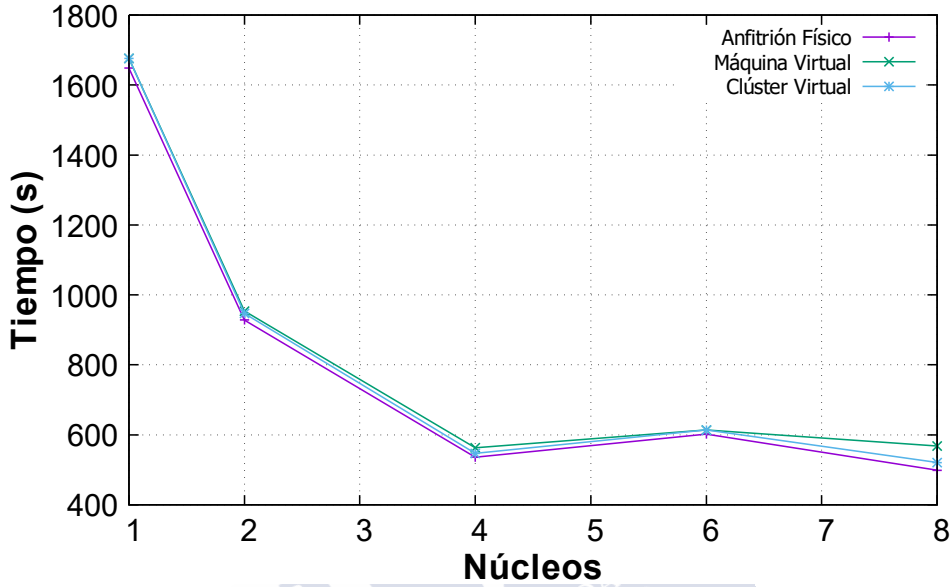

Figura 4.2: Comparación del tiempo requerido (en segundos) para completar las simulación en las diferentes infraestructuras cuando estas poseen el soporte del *hyperthreading* activado.

ocho MV, al ser comparado con el mismo clúster sin la sobre demanda activada. Los datos obtenidos ponen de manifiesto que no se produce degradación considerable en el tiempo requerido, porque hace uso de todos los núcleos lógicos disponibles, sino que la optimización del tiempo de ejecución llega a un punto estable sin mejoría posible posteriormente. El *hyperthreading* sólo aporta beneficios cuando el número de de procesos supera a la cantidad de núcleos físicos.

Conclusiones

El *hyperthreading* aporta mejoras al tiempo de ejecución cuando se requiere emplear más procesos que el número de núcleos físicos disponibles de forma que si este soporte a la sobre demanda está inactivo no aportará beneficios si sólo emplean los núcleos físicos. En caso que se presente una sobre suscripción de recursos en cuanto a núcleos requeridos, el tiempo de ejecución de la aplicación GADGET2 empeora notablemente al usarse seis u ocho núcleos lógicos, en este sentido el *hyperthreading* si aporta beneficios considerables al estar activo.

Existe una mínima pérdida de tiempo de ejecución en el clúster virtual cuando se emplean ocho MV con un núcleo cada una, esto si se compara con el tiempo obtenido en el anfitrión físico utilizando ocho núcleos lógicos. Conforme a los resultados es viable que la aplicación cosmológica GADGET2 se pueda portar a una infraestructura que emplee virtualización, como ser la nube sin que esta presente pérdidas considerables de tiempo para completar sus simulaciones, siempre que se evite que se produzcan sobre suscripción de recursos en el número de núcleos a emplear, en cambio esto puede ser minimizado al utilizar el soporte del *hyperthreading* en el ordenador físico o *host*, donde se ejecuten las MV.

4.1.2. CASA

Las aplicaciones científicas se caracterizan por requerir alto poder de computación y muchas veces esto ha demostrado ser un factor limitante en algunas investigaciones. Para proporcionar la capacidad informática requerida varias soluciones se han desarrollado a través del tiempo, como la computación de altas prestaciones, computación en malla, la nube y entre otros. Con el surgimiento específico de la computación en la nube y el desarrollo del paradigma IaaS, se ha logrado que la capacidad computacional se puede escalar bajo demanda incorporando máquinas virtuales (MV) a las operaciones. En esta sección, se evalúa una infraestructura en la nube para optimizar la reducción y análisis de datos radioastronómicos utilizando un conjunto de aplicaciones de software llamado *CASA* [39]. Esta aplicación se emplea actualmente para el análisis de datos provenientes de observaciones astronómicas en longitudes de onda del segmento de radio.

CASA [39] es un paquete de herramientas para calibrar, analizar y reducir datos radioastronómicos obtenidos de instrumentos especializados como radiotelescopios e interferómetros, CASA permite marcar y descartar datos espurios, calibrar parámetros de los instrumentos, corregir y ajustar a los valores ideales. También permite incluir los efectos de la altitud, temperatura, ganancia de antena y otros elementos necesarios para procesar los datos correctamente. También incluye soporte para mostrar gráficos de los datos en análisis, las últimas versiones pueden simular observaciones astronómicas con instrumentos como el EVLA (*Expanded Very Large Array*), ALMA (*Atacama Large Millimeter / submillimeter Array*) y SKA (*Square Kilometer Array*) [69]. CASA funciona bajo el sistema operativo Linux y MacOS. Al momento de escribir este documento, la última versión diponible es la 4.1.0 y fue la utilizada en los procedimientos y pruebas realizadas sobre CASA.

La realización de simulaciones astronómicas usando CASA fue implementada en máqui-

nas virtuales como las que se usarían en un entorno en la nube real, se evaluó como los procesos altamente exigentes ejecutados de forma simultánea generaron carga en la CPU en un mismo nodo de cálculo. También se estudió la influencia del *hyperthreading* en el rendimiento de las máquinas virtuales empleadas con CASA. Nuestro objetivo es evaluar el rendimiento de la reducción de datos en CASA cuando varias MV coexisten en el mismo nodo de cómputo, este tipo de operación representa la peor de las condiciones a las que puede enfrentarse una implementación de MV en la nube.

Para nuestros propósitos, ejecutamos tres ejemplos que están disponibles en el sitio web de CASA, que contienen datos originados en los cuerpos estelares NGC4826, 3C129, NGC2403. Adicionalmente y como referencia ejecutamos la prueba de rendimiento sintética Intel Linpack. Nótese que pretendemos medir el efecto de la sobre demanda de recursos, esta condición genera la forma más compleja de trabajo en la nube [102].

La primera prueba se realizó con datos de la galaxia NGC4826, también conocida como M64 en el catálogo Messier o «Galaxia del Ojo Negro», que se encuentra a 16 millones de años luz de la Tierra y tiene forma de espiral [72]. Estos datos provienen del estudio de la línea de carbono de 3 milímetros obtenida a través del proyecto denominado como BIMA SONG (*Berkeley Illinois Maryland Association Survey of Nearby Galaxies*) [34].

La segunda prueba se realizó con datos tomados de la radio galaxia 3C129. Durante años, este cuerpo celeste no fue entendido hasta el desarrollo de instrumentos de radio más sensibles que permitieron a los astrónomos descubrir que este cuerpo celeste en sí es un cúmulo de galaxias. Después de esto, cambiaron el nombre de sus estructuras más grandes con los identificadores C3C29 y 3C129.1. El comportamiento de este cúmulo de galaxias se comprendió como una interacción entre dos grandes cuerpos celestes, como lo indica el estudio de la polarización de ondas de radio emitida por las estructuras [11] [56].

La tercera prueba se realizó con datos tomados de la galaxia NGC2403. El astrónomo William Hershel descubrió este objeto celestial en 1788. Está a 12 millones de años luz de la Tierra y la característica más importante es que cuenta con un núcleo de galaxia activa [90]. Los datos provienen del estudio del hidrógeno molecular obtenido por VLA (*Very Large Array*). La última prueba se realizó con un continuo de 6 cm de longitud de onda, tomada con el VLA en el año 1999 como parte del proceso de calibración y el programa de prueba de este instrumento, ya que NGC4203 es un objeto que genera emisiones de radio a gran escala.

Infraestructura

La infraestructura en la nube de tipo privada que fue empleada estaba gestionada con Apache CloudStack 4.0 y se utilizó KVM como el hipervisor, todo ello administrado en los nodos computacions (CN). Los CN utilizan un procesador Intel Core i7-2600@3.40 GHz con 8 GB de RAM y corriendo el sistema operativo CentOS 6.3 de 64 bits. El procesador Intel Core i7-2600 tiene cuatro núcleos y ocho subprocesos, un caché L2 de 4x256 Kbytes, un caché compartido L3 de 8 MB, *hyperthreading*, tecnología de virtualización Intel (VT-x) y virtualización para entrada - salida (E/S) dirigida. La red de interconexión es de tipo Gigabit Ethernet.

Resultados

Para satisfacer nuestros propósitos, se implementaron dos tipos de máquinas virtuales (MV): CASA MV y Linpack MV. Las características de CASA MV son las siguientes: emplea 1 CPU virtual, 1 GB de RAM y 20 GB de disco duro. Su sistema operativo es Ubuntu 12.04 64 bits. Esta MV proporciona el software CASA y también los experimentos NGC4826, 3C129, NGC2403 y Linpack. Las características de Linpack MV son las siguientes: emplea 1 CPU virtual, 1 GB de RAM y 5 GB de disco duro con CentOS 5.5 de 64 bit. Para evaluar el impacto de la cantidad de MV implementadas por el sistema anfitrión en el rendimiento de las simulaciones CASA, consideramos n MV, donde n = 1, 2, 4, 6 y 8. La primera máquina virtual ejecuta el software CASA y mide el tiempo de simulación. Las máquinas virtuales n-1 restantes ejecutan Linpack en un modo de bucle infinito.

NGC4826

Los resultados se muestran en la figura 4.3, al compararse las simulaciones para este cuerpo celeste se requirió de mayor tiempo cuando se aumentó el número de MV por anfitrión. Si el *hyperthreading* está desactivado, la cantidad de tiempo requerido es mayor. En este caso, cuando se escala el número de MV hasta usar cuatro de ellas, el tiempo de simulación se incrementa en un 13 % a que si sólo se emplease una de manera exclusiva. Para un mayor número de MV por anfitrión, se presenta un deterioro importante en el tiempo requerido, se observó que al utilizar 6 u 8 MV los tiempos de simulación se incrementan en un 68 % y 90 % respectivamente, comparándolos a los resultados obtenidos con una sola MV. Al habilitar el soporte del *hyperthreading* y compararlo a las mediciones sin este, se encontró que el tiempo

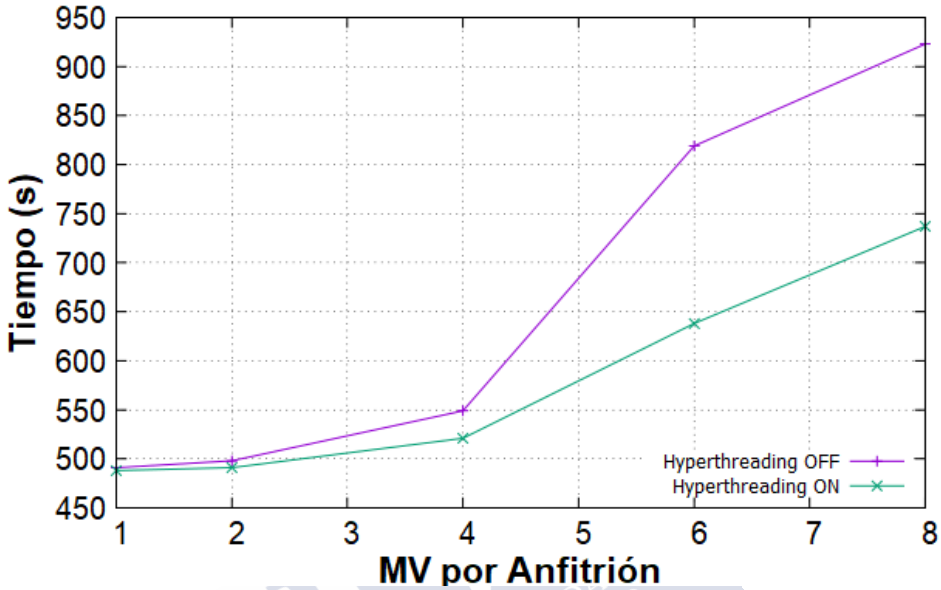

Figura 4.3: Tiempo requerido por las máquina virtual (MV) para completar la simulación de la evolución de NGC4826, se muestra el efecto del soporte del *hyperthreading*.

requerido para completar la simulación disminuyó un máximo de un 7 % al utilizarse cuatro o menos MV. En cambio la eficiencia aumenta alcanzando un 31 % y un 51 % para 6 y 8 MV respectivamente, aportando mejores tiempos en la simulación al presentarse sobre demanda de recursos lo que difiere sino se emplease el *hyperthreading*.

3C129

Los resultados se pueden observar en la figura 4.4. Se mantiene la tendencia del incremento en el tiempo de simulación cuando se aumenta el número de MV por anfitrión. Sin embargo, si el *hyperthreading* está desactivado el aumento del tiempo requerido es un poco más notable. En este caso, cuando se implementan cuatro o menos máquinas virtuales, el aumento en el lapso de simulación es inferior al 4 %. Para un número mayor de MV por anfitrión, hay un aumento en el tiempo, requiriendo un 14 % más al usarse 6 MV y un 24 % más si se emplean 8 MV; si estos resultados son comparados con los obtenidos con una MV.

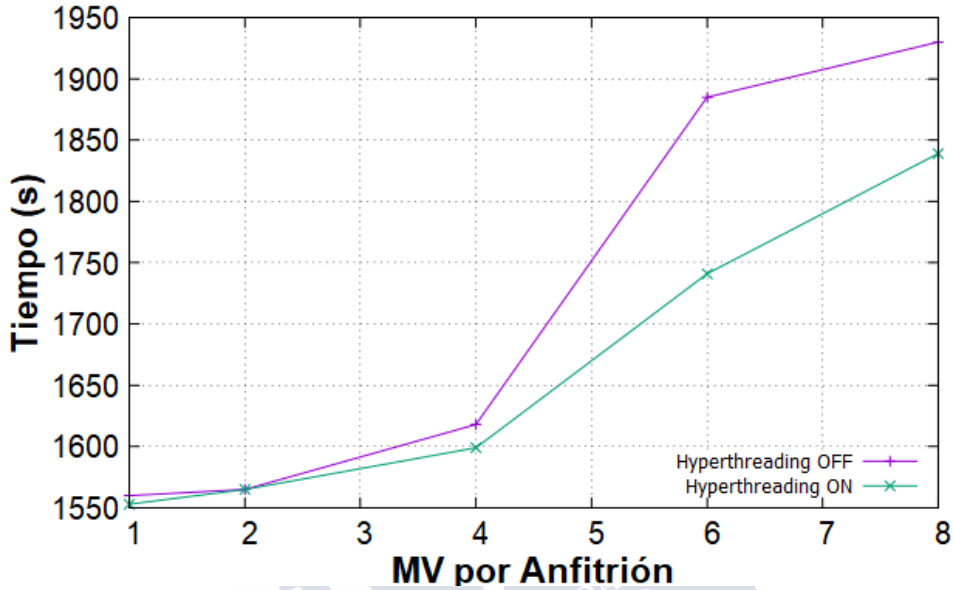

Figura 4.4: Tiempo requerido por las máquinas virtuales (MV) para completar la simulación de la evolución de 3C129, se muestra el efecto del soporte del *hyperthreading*.

NGC2403

Los resultados se muestran en la figura 4.5. En esta prueba de ejecución, se mantiene la tendencia como en los casos previos donde existe un aumento en el tiempo de simulación cuando se incrementa el número de MV por ordenador. Si el *hyperthreading* está deshabilitado, la cantidad de tiempo requerido es más notable cuando se ejecutan más de cuatro máquinas virtuales. En este caso, cuando se implementan cuatro o menos MV, el aumento en el tiempo de simulación es inferior al 4 %. Para un mayor número de MV por ordenador, hay un deterioro importante en el lapso de ejecución, observándose que con 6 y 8 máquinas virtuales los tiempos de simulación se incrementan en torno a un 22 % y un 30 % respectivamente, comparado con los obtenidos con una MV.

Linpack

Se ha realizado un estudio similar a las simulaciones, con el fin de conocer la carga que tiene sobre un mismo anfitrión el incremento de MV hasta el máximo de núcleos lógicos

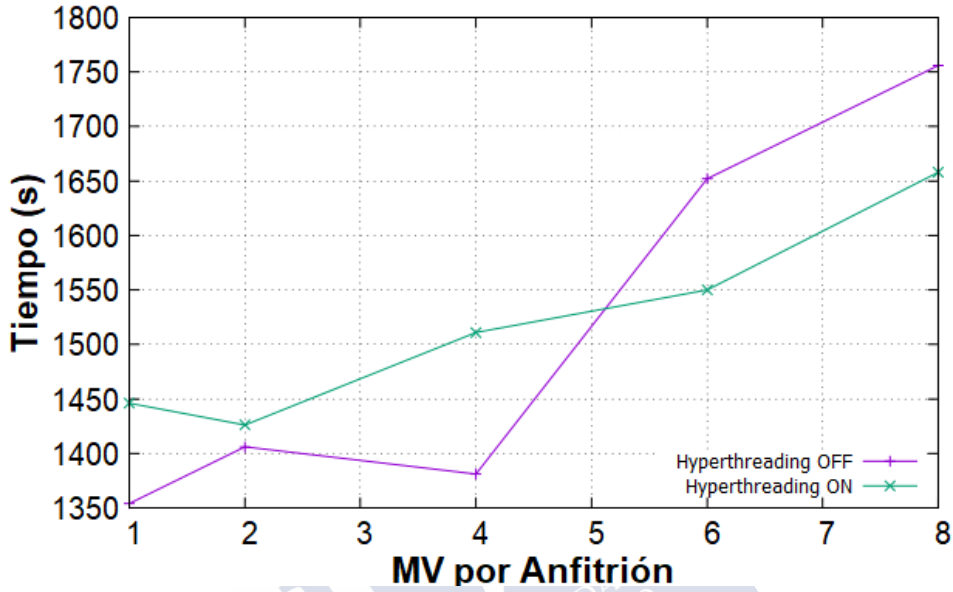

Figura 4.5: Tiempo requerido por las máquinas virtuales (MV) para completar la simulación de la evolución de NGC2403, se muestra el efecto del soporte del *hyperthreading*.

disponibles, pero en este caso utilizando la prueba de rendimiento Intel Linpack, los resultados se muestran en la figura 4.6. Para ello, hemos utilizado un tamaño de problema de 5000, con valores de alineación de 4 KB. Intel Linpack proporciona el rendimiento promedio en GFlops. Se mantiene la tendencia de la disminución en los GFlops cuando se aumenta el número de MV por anfitrión. Al habilitar el *hyperthreading*, se puede apreciar un incremento en las operaciones de coma flotante hasta que el número de MV no superen los núcleos físicos del sistema, reportando resultados con una magnitud de hasta un 1 GFlop superiores respecto cuando este soporte está desactivado. En este caso, cuando se implementan seis u ocho MV se presenta una caída de alrededor de 6 GFlops en las operaciones al contar con el *hyperthreading* activado. Sin embargo, si el *hyperthreading* está deshabilitado y el número de MV supera a los núcleos físicos, se obtienen mejores resultados en la prueba de Linpack.

Conclusiones

Se evaluaron simulaciones astronómicas sobre Apache CloudStack, empleando KVM como el hipervisor que administra múltiples MV, todas ellas con un alto nivel de exigencia de

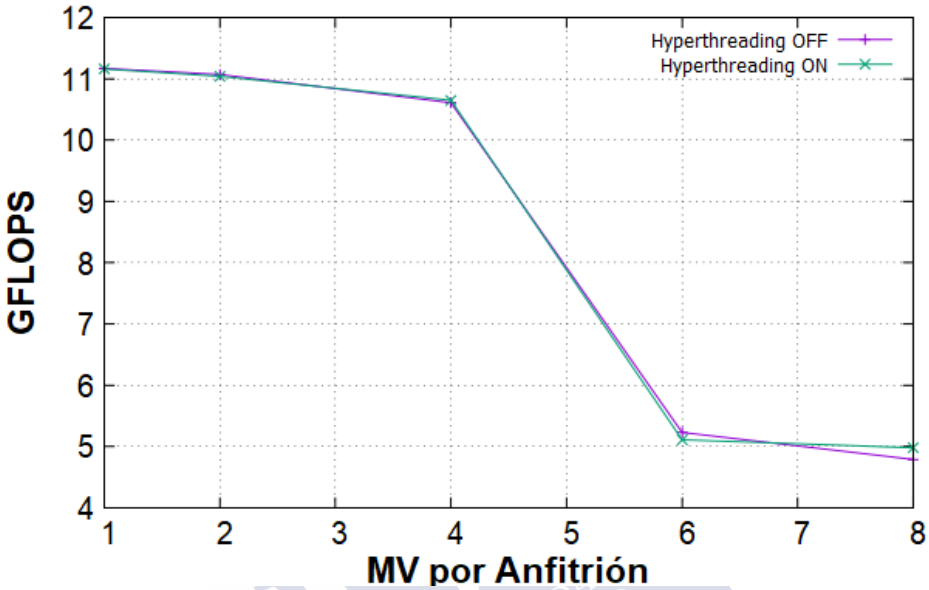

Figura 4.6: La gráfica muestra la capacidad de procesamiento en GFlops obtenido por las máquinas virtuales (MV) y el efecto del soporte del *hyperthreading* en esas mediciones.

CPU. Para ello se ejecutan de forma concurrente en un mismo nodo de computación, tareas de simulación de computación intensiva, creando las condiciones del peor caso posible de implementación de MV en una infraestructura en la nube, es decir con sobre demanda de recursos de procesamiento.

Los resultados obtenidos al simular la evolución estelar de los cuerpos celestes conocidos como NGC4826, 3C129 y NGC2403 mostraron un comportamiento similar con la degradación en el rendimiento cuando el *hyperthreading* estaba deshabilitado, esta condición aumenta al utilizar más de cuatro MV en un mismo anfitrión, ya que el procesador del nodo físico cuenta con cuatro núcleos. En caso de habilitar el *hyperthreading* la disminución del tiempo requerido se mantuvo al utilizar más de cuatro MV, pero en una proporción menor que sin esta característica habilitada. Para el caso de referencia en el que se usó Linpack, se determinó que no hay un efecto notable en las operaciones de coma flotante cuando el *hyperthreading* está habilitado.

Con los datos obtenidos se concluye que la realización de simulaciones astronómicas en

la nube es adecuada, ya que en todas las simulaciones realizadas se notó una reducción del tiempo requerido. En cambio para simulaciones numéricas como las que realiza Linpack el *hyperthreading* no aportó beneficios de ejecución notables.

4.2. Multimedia

En los últimos años se incorporaron tecnologías [56] como el correo electrónico, el chat, las teleconferencias y actividades en línea, para respaldar los procesos de enseñanza y aprendizaje. Hoy vivimos la mayor revolución en informática, multimedia y la televisión, desde la invención de la emisión en color a principios de la década de los cincuenta [2]. Actualmente los profesores y los estudiantes pueden ver vídeos en una variedad de medios, desde teléfonos móviles, computadoras y hasta en pantallas de alta definición. Estos cambios van de la mano con la evolución constante de la tecnología informática y de las herramientas computacionales que evolucionan constantemente para crear vídeos de calidad superior y de alta complejidad desde diferentes fuentes de contenido.

Renderización es el proceso de conformar medios animados y escenas. Esta es una actividad que consume mucho tiempo y altos recursos computacionales. Por lo general, las empresas y los centros de capacitación utilizan recursos costosos para reducir el tiempo necesario de edición para preparar vídeos. Con el fin de aprovechar estas tecnologías dentro de entornos docentes y de investigación, y con ello poder crear contenido audiovisual de alta calidad, pero con la condición de no incurrir en la adquisición de licencias costosas ni requerir recursos de hardware dedicados, esta sección tiene como objetivo mostrar un método para reutilizar el hardware existente en escuelas, facultades, laboratorios y otras instituciones. Todo ello con el fin de integrar vídeos en alta definición mediante la implementación de tecnología en la nube y virtualización de forma local [9].

4.2.1. Handbrake

HandBrake es un transcodificador de vídeo de código abierto disponible para Linux, Mac y Windows. Esta aplicación toma los vídeos en diferentes formatos y codificaciones. Posteriormente a través de un proceso de conversión los adecua para ser vistos en múltiples dispositivos que van desde teléfonos móviles, tabletas, ordenadores, hasta dispositivos profesionales de difusión, permitiendo así poder utilizar cualquier vídeo en casi todo dispositivo que admita formatos modernos, posteriormente al ser procesados por esta aplicación [86].

HandBrake funciona con archivos y formatos de vídeo más comunes, la mayoría de ellos creados o capturados por cámaras profesionales y para consumidor, dispositivos móviles como ser teléfonos y tabletas, grabaciones originadas en juegos y pantallas de computadora y también soporta vídeos desde discos DVD y Blu-ray, entre otros. HandBrake aprovecha herramientas como ser *Libav*, *x264* y *x265* disponibles de código abierto para crear nuevos archivos transcodificados a MP4 o MKV [23, 50], dependiendo de lo que seleccione el usuario como preferencia.

Esta aplicación en cuanto al manejo de vídeos permite multiples operaciones como ser la de recortar y cambiar el tamaño, restaurar fuentes antiguas y de baja calidad, retirar los efectos causados por entrelazado y telecine, asi mismo poder manejar ciertos tipos de audio sin necesidad de conversión. Ofrece la capacidad de mezclar el sonido envolvente discreto con el sonido envolvente matricial o estéreo, hacer ajustes a los niveles de volumen de audio y el manejo del rango dinámico para ciertos tipos de audio, permite conservar los subtítulos existentes o bien puede agregar o quitar subtítulos suaves (estos son los que se almacenan como texto). HandBrake también puede crear vídeos que sean más pequeños (ocupando menos espacio de almacenamiento en su dispositivo) que los originales; ya que los convierte a los estándares MP4 y MKV.

Dentro de las ventajas complementarias que ofrece HandBrake es la de respetar la protección de copia. No funciona con archivos de vídeo que emplean gestión de derechos digitales (DRM). Esto incluye, pero no se limita a contenido protegido contra copia en iTunes, Amazon Video, Netflix u otros proveedores en línea así como muchos discos comerciales de DVD y Blu-ray [4, 3, 47, 62].

Los profesionales de la multimedia en sus actividades diarias transcodifican formatos, generalmente desde fuentes crudas hacia archivos de formatos comprimidos, durante este proceso aprovechan las herramientas de software y de edición no lineal, pueden hasta cierto punto agregar efectos, recortar fuentes, colocar traducciones, subtítulos, etc. El resultado puede ser ingestado en sistemas de automatización televisiva o de otra índole de difusión; estos sistemas requieren parámetros de vídeo bien definidos y estandarizados. Es por ello, que se requiere del proceso de transcodificación con el fin de unificar a un formato estándar. Lastimosamente este proceso toma mucho tiempo y requiere equipo que esté dedicado y no siempre es desatendido. En este caso de uso se propone una solución que pueda ejecutarse en segundo plano y sin la intervención ni supervisión del usuario, requiriendo únicamente interacción al momento del envío del archivo.

Infraestructura

Al momento de realizar las pruebas no había en los repositorios oficiales para el sistema operativo usado ningún paquete precompilado disponible de HandBrake, ya que pretendemos utilizar el mismo que en los otros casos de uso realizados hasta el momento y descrito en los diferentes capítulos. Por lo tanto sólo se tenían dos opciones: la *primera* era descargar las archivos de código fuente desde del sitio web del desarrollador para posteriormente compilarlo e instalarlo utilizando la consola del sistema, o bien una *segunda* opción es la de agregar manualmente un repositorio de terceros. Para nuestro caso particular utilizaremos el repositorio, para así evitar la necesidad de instalar paquetes adicionales y herramientas de desarrollo en el sistema donde realizaríamos las mediciones, ya que no podremos identificar el efecto de ellas en el rendimiento general.

Típicamente, el proceso de transcodificación es dedicado y se pretende lograr cumplir con normas de operación con los archivos resultantes, ya que en nuestro caso de uso se probará en el flujo de trabajo de una estación de televisión. Por lo tanto, la infraestructura descrita en esta sección presenta una solución desatendida y de ejecución en segundo plano. Gráficamente se muestra en la figura 4.7.

El planificador de tareas recibe las peticiones de procesamiento por medio de un *script* que de forma periódica verifica por nuevos archivos de medios en la carpeta compartida de red en el sistema de archivos NFS. Al ser detectado es movido a una carpeta particular que sólo es accesible por el planificador y las máquinas virtuales que realizan la transcodificación, estas se están ejecutando de forma distribuida en la intranet en diferentes anfitriones; cada MV recibe un archivo completo para transcodaje a la vez y al finalizar el producto de este proceso se almacena en otra carpeta donde el planificador copia a una tercera carpeta de salida en el sistema de almacenamiento de red para uso y manipulación por parte de los usuarios, los cuales si se desea pueden ser enviados de manera automática al sistema de automatización de difusión para su uso a través de la plataforma de distribución de contenidos preferida, en caso de ser necesario.

Se aclara que no se hace una división en fragmentos de los archivos fuente de vídeo para ser transcodificado de forma concurrente por múltiples MV. Lo anterior no se lleva a cabo a pesar que existen utilidades que se podrían emplear para ello, como ser *ffmpeg, videosplit o mencoder*; pero es aquí que surge un problema técnico complejo como es la de identificar la ubicación precisa de los fotogramas clave al momento de ensamblar un sólo archivo de salida, debido a esta falencia se genera un error que es apreciable al momento de visualizar el archivo

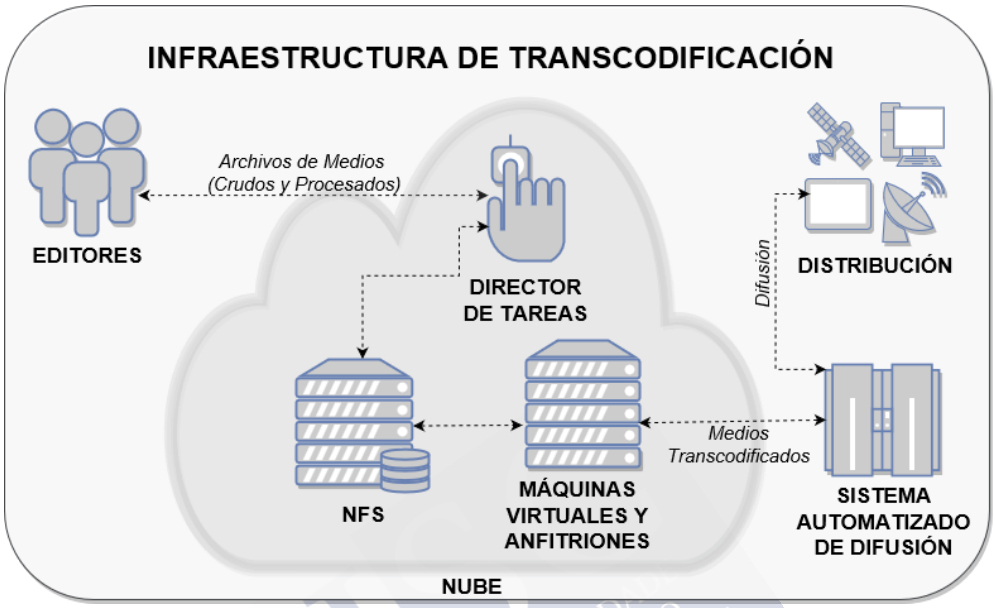

Figura 4.7: Flujo de trabajo del transcodaje. Handbrake se encuentra virtualizado y su uso es transparente al usuario.

transcodificado y que no fue posible encontrarle una solución práctica. Por la razón descrita se decidió enviar un archivo por nodo de forma secuencial.

Es importante aclarar que para las mediciones realizadas por la técnica descrita, fue requerido sincronizar el reloj de todos los nodos virtuales y anfitriones por medio del servicio basado en el protocolo de tiempo de red (NTP). Así mismo se ha usado el hipervisor Oracle VirtualBox 4.1.14 [94]. Este fue ejecutado sobre ordenadores equipados con un microprocesador Intel i5-3570 el cual cuenta con cuatro núcleos con una frecuencia básica de 3.4 GHz y usando el soporte para el llamado modo *turbo*, con el cual este CPU puede entregar hasta 3.8 GHz de velocidad, el mismo cuenta con soporte para extensiones *VT-x, vPro* y con una arquitectura de 64 bits y no posee soporte para *hyperthreading*.

Para las mediciones se utilizó un ordenador anfitrión físico que cuenta con 16 GB de memoria RAM del cual se tomaron 4 GB para las MV. El sistema operativo usado ha sido Ubuntu Linux 13.01 64 bits. Las pruebas realizadas evalúan el uso de diferente número de núcleos combinados con el soporte *turbo* activo o inactivo, ya que nos interesa conocer si esta funcionalidad aporta beneficios en la ejecución. Se han realizado una serie de diez ejecucio-

nes sucesivas donde se ha medido el tiempo requerido para transcodificar un archivo origen en formato MTS con una duración de treinta minutos exactos y con una resolución de alta definición completa en 1080i. El formato resultante es MPEG4 con la misma resolución y duración.

En cuanto a la adecuación de la disponibilidad de la RAM en los diferentes sistemas al momento de completar las medidas sucesivas fue necesario hacer ajustes manuales a los bancos de memoria física instalados. En el caso del anfitrión se dejaron únicamente 4 GB de RAM al momento de las pruebas, con ello se evitó afectar las mediciones por las diferencias en RAM al ser comparadas a la capacidades que tienen las máquinas virtuales. En el gestor de las MV, se configuró el uso de 4 GB de RAM por sistema invitado, pero el anfitrión se le colocaron 16 GB de RAM físicos. Con ello se pretende evitar situaciones de inanición y sobre demanda de recursos, minimizando sus efectos nocivos en el desempeño.

Resultados

En el anfitrión los resultados fueron mejorando conforme se incrementaban los núcleos disponibles para transcodificar de forma concurrente y esto disminuía el tiempo requerido directamente. Al habilitar el soporte del *turbo* en el microprocesador fue beneficioso, porque ayudaba a disminuir el tiempo requerido para completar las tareas. Tomando el promedio de las múltiples ejecuciones llevadas a cabo, se obtuvo con cuatro núcleos y con el soporte *turbo* habilitado, los mejores resultados, los cuales requirieron un total de 969.91 segundos para completar las tareas, mientras tanto que si se deshabilitaba el soporte del *turbo* el período de ejecución necesitaba a 1021.83 segundos, siendo el segundo mejor resultado; de la misma manera al emplear un sólo núcleo se requirió de 3842.68 segundos para completar la transcodificación, siendo el peor resultado de toda la serie. Los resultados generales se muestran en la tabla 4.1.

En la máquina virtual, los resultados mejoraron conforme se aumentaban los núcleos disponibles para transcodificar de forma concurrente como se observa en la tabla 4.2, pero al contar con el soporte del *turbo* en el microprocesador anfitrión el tiempo requerido aumentó. Tomando el promedio de las múltiples ejecuciones llevadas a cabo, se obtuvo con cuatro núcleos y sin el soporte *turbo* habilitado, el menor tiempo de ejecución, donde se requirió un total de 1112.02 segundos para completar la tarea ejecutada, mientras que si se contaba con el soporte *turbo* el tiempo requerido aumenta a 1213.97 segundos, este último valor representa el segundo mejor resultado. De la misma forma, al emplear un sólo núcleo se requirió

		Pruebas Sucesivas Realizadas (s)										
T	**N**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Prom.**
ON	4	969	967	968	967	972	970	971	971	972	972	969.91
ON	2	1806	1803	1808	1813	1806	1807	1810	1808	1806	1808	1807.43
ON	1	3466	3466	3473	3475	3469	3465	3479	3470	3469	3474	3470.68
OFF	4	1020	1018	1021	1019	1019	1027	1021	1023	1023	1027	1021.83
OFF	2	1996	2019	2007	2030	1997	1991	1994	1991	1994	1996	2001.70
OFF	1	3822	3860	3867	3841	3837	3831	3843	3847	3842	3836	3842.68

Tabla 4.1: Transcodaje del archivo fuente sobre anfitrión físico. Medidas en segundos (s), menor tiempo indica mejores resultados. De la primera columna de izquierda a derecha, corresponden a la funcionalidad de *Turbo* (T) y la segunda el número de núcleos empleados (N), la última es la media aritmética de las medidas (Prom.)

		Pruebas Sucesivas Realizadas (s)										
T	**N**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Prom.**
ON	4	1223	1223	1219	1207	1208	1208	1207	1207	1210	1228	1213.97
ON	2	2315	2310	2313	2310	2328	2314	2317	2315	2318	2315	2315.39
ON	1	4435	4393	4400	4411	4412	4405	4418	4443	4412	4409	4413.84
OFF	4	1128	1086	1092	1097	1107	1122	1122	1122	1122	1124	1112.02
OFF	2	2139	2137	2142	2109	2107	2144	2160	2105	2098	2095	2123.31
OFF	1	4009	3938	4005	3999	4017	4014	4008	4025	4008	4032	4010.10

Tabla 4.2: Transcodaje del archivo fuente sobre máquinas virtuales. Medidas en segundos (s), menor tiempo indica resultados más eficientes. La primera columna de izquierda a derecha, corresponde a la funcionalidad de *Turbo* (T) y la segunda el número de núcleos empleados (N), la última es la media aritmética de las medidas (Prom.)

de 4010.10 segundos para completar la transcodificación, siendo el peor resultado de toda la serie sobre la MV.

Tomando los mejores resultados promedio en la MV como en el anfitrión físico, se aprecia que al incrementar núcleos adicionales se reduce de forma consistente el tiempo requerido para completar las tareas de transcodaje. La mayor proporción fue de 1663 segundos al pasar de uno a dos núcleos en el anfitrión y en la MV este cambio fue de 2098 segundos. Al pasar de dos a cuatro núcleos en el anfitrión el tiempo se redujo en 837 segundos, mientras que la máquina virtual este tiempo fue de 1101 segundos. Nótese que la MV tiene valores mayores debido a que está sometida a los efectos de la ejecución anidada dentro del sistema operativo anfitrión.

Conclusiones

Al realizar las pruebas considerando la influencia del número de núcleos y el soporte a la funcionalidad de *Turbo* en las diferentes ejecuciones, se encontró que este soporte beneficia al anfitrión físico cuando está activa, disminuye entre un 5-10 % el tiempo de ejecución de las pruebas realizadas. En cambio este soporte reduce la capacidad de ejecución en la máquina virtual en un 8 % de forma sostenida al compararse cuando esta funcionalidad está desactivada en el anfitrión.

Si comparamos los mejores resultados obtenidos, siendo los del anfitrión con el soporte del *Turbo* y de la máquina virtual sin el, encontramos que la MV necesitó para completar las pruebas de 142.11 segundos adicionales usando cuatro núcleos, 315.88 segundos con dos núcleos y 539.42 segundos con un sólo núcleo. Con los resultados anteriores en promedio se requiere de 13.70 % de tiempo adicional para completar el transcodaje en la mejor configuración posible en la MV al compararse con las mejores condiciones de ejecución sobre el anfitrión físico.

Conforme a los resultados obtenidos, un entorno virtualizado en la nube provee una alternativa fiable y viable para las tareas de transcodaje. La principal ventaja es que se puede utilizar capacidad computacional que no esté en uso o que esté distribuida y con ello lograr tiempos similares a lo que se pueden obtener sobre ordenadores físicos. Adicionalmente, se puede contar con una infraestructura flexible para múltiples tareas en las que se involucre el manejo y procesamiento de medios.

Handbrake es una opción viable para emplearse en entornos virtuales, pero a nuestro criterio se debe estandarizar la instalación y configuración de esta aplicación, de igual manera se debe tener especial cuidado en los soportes de las extensiones de microprocesador que puedan beneficiar o afectar la ejecución. Por otro lado el sistema operativo a usar debe ser seteado con los servicios y aplicaciones mínimas necesarias para no agregar carga de trabajo en las máquinas virtuales como en los anfitriones.

4.2.2. Cinelerra

Con la revolución de la televisión de alta definición, los creadores de contenidos requieren más eficiencia en las tareas de producción y postproducción. Existen diversas aplicaciones para ordenadores disponibles para este propósito, incluidas las soluciones comerciales que a menudo son muy costosas y están algunas de código abierto como es el caso de Cinelerra.

En esta sección se propone una infraestructura en la nube para usar Cinelerra, que es una aplicación desarrollada para edición de vídeo no lineal, de forma que es una solución idónea para ser usada en instituciones educativas, de investigación y canales de televisión públicos. El objetivo principal de esta sección, es reutilizar el poder computacional que provee la nube para editar o crear vídeos sin la necesidad de adquirir una infraestructura de hardware dedicada. De forma que al utilizarse recursos compartidos, como los que se encuentran disponibles en laboratorios de computación o los ordenadores multinúcleo de uso típico en las oficinas, se logró desarrollar una configuración para reducir el tiempo necesario para renderizar a través de una distribución de las tareas involucradas para ese fin. En esta sección se describe la infraestructura utilizada y los resultados obtenidos en las pruebas realizadas [99].

Infraestructura

La infraestructura utilizada para ejecutar Cinelerra está compuesta por varios tipos de componentes como ser nodos físicos de cómputo, un nodo maestro virtual, nodos de renderización virtual y un sistema de archivos de red (NFS) todos interconectados entre si por medio de un clúster virtual.

- **Los nodos físicos de cómputo:** Cuentan con un procesador Intel Core i7-2820QM con una velocidad de reloj de 2.30 GHz y 8 GB de RAM DDR3 y ejecutan el hipervisor VirtualBox 4.1.14. Este procesador cuenta con ocho núcleos lógicos al tener el *hyperthreading* habilitado.

- **El nodo maestro y los nodos de renderización:** Son MV de VirtualBox, configurados con dos núcleos por MV, con 1 GB de RAM y se han aprovechado las extensiones de virtualización de hardware (VT-x).

- **La infraestructura del clúster virtual:** Compuesto por el nodo maestro y los nodos de cómputo, utilizan como sistema operativo Ubuntu 12.04 LTS 64 bits, todos los nodos están interconectados a través de una interfaz Ethernet Gigabit.

- **NFS:** Almacena los recursos de entrada y salida de vídeo, el cual es también una máquina virtual en un anfitrión dedicado para evitar que la concurrencia de entrada y salida afecte las mediciones de rendimiento hechas para determinar si la portabilidad de esta solución es de beneficio para los procesos de integración de vídeo.

Se diseñaron dos casos de estudio como pruebas de uso de la infraestructura descrita anteriormente. El *primero* es la implementación de Cinelerra sobre la infraestructura disponible en la Universidad Nacional Autónoma de Honduras (UNAH), especificamente en las instalaciones de la televisión universitaria en el llamado Sistema de Difusión de Radio y Televisión (Utv) [76]. El *segundo* caso es usando Cinelerra en un centro de entrenamiento local, donde se presenta el escenario típico de los laboratorios de informática utilizados para la formación de personal que operará herramientas multimedia.

Televisión universitaria

La infraestructura descrita se implementó en un canal de televisión que transmite al aire de forma ininterrumpida todo el día y que tiene requerimientos altos de renderización por la producción continua de noticieros, con el fin de volver eficiente este proceso de emisión se requiere cumplir con normas y parámetros propios de las estaciones de difusión audiovisual, como se muestran gráficamente en la figura 4.8. Se desplegó el clúster virtual en la infraestructura física a modo de una IaaS, lo que permitió reutilizar el hardware disponible en la televisora para las aplicaciones de vídeo con Cinelerra.

Actualmente en la estación de televisión existe implementado un flujo de trabajo que debe seguirse con el objetivo principal de incorporar y adaptar cualquier solución de software que aporte las ventajas de renderizar en la nube a todo ese proceso. Toda la producción de vídeo y TV comienza con la introducción del contenido multimedia en el sistema de transmisión a través de un punto común llamado ingesta. Luego, mediante una serie de procedimientos con parámetros predefinidos, los vídeos de una cámara, estudio, satélite, DVD, cinta, etc. se convierten en archivos de vídeo para ordenador y usualmente estos son acompañados de archivos de metadatos estandarizados que describen su contenido y propiedades. Ya existen parámetros de formatos y codificación que deben cumplirse para que los vídeos puedan ser utilizados en las diferentes etapas del proceso de edición, producción y postproducción.

Como resultado del proceso de ingesta, los archivos de vídeo se almacenan en una videoteca que está compuesta por un grupo de servidores NFS configurados en uno o más arreglos redundantes de unidades de disco. Posteriormente los metadatos se almacenan en un servidor de base de datos relacional. El almacenamiento de vídeo puede ser organizado en niveles jerárquicos para distinguir contenidos de diferentes orígenes como ser archivos fuentes crudos (RAW), recursos (fotografías, diapositivas,etc.), soporte de medios (audios, fragmentos, transiciones, etc.) y vídeos editados listos para transmitir.

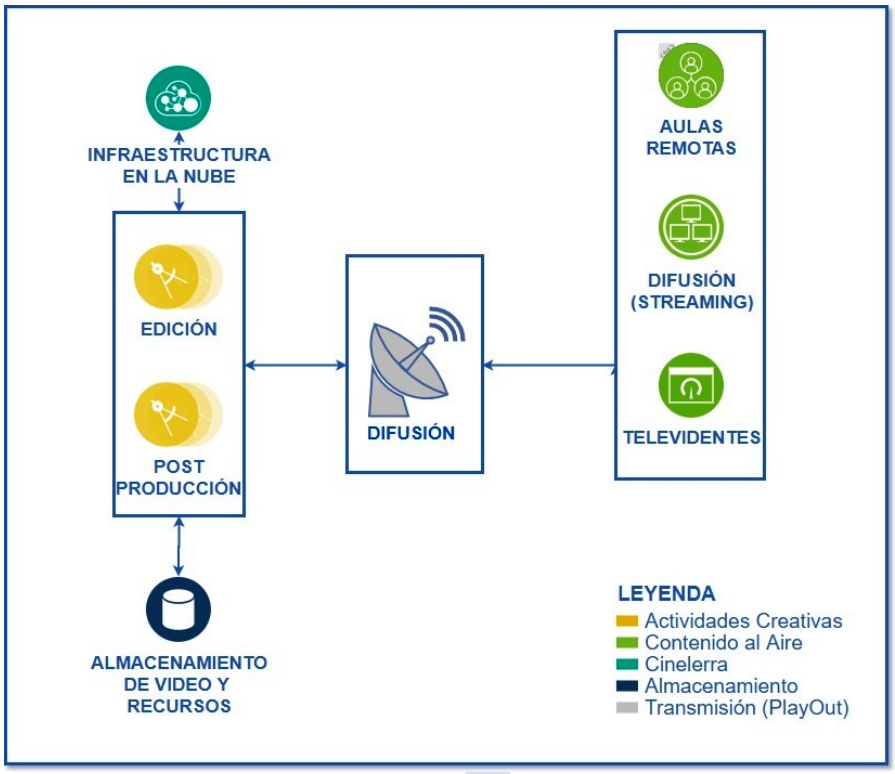

Figura 4.8: Flujo de trabajo con las diferentes etapas involucradas en la operación televisiva.

Existe en la televisora una unidad organizativa que realiza la edición y producción. Los editores cuentan con potentes estaciones de trabajo, desde ellas acceden al almacenamiento de vídeo jerárquico en red para tomar los recursos necesarios que se incorporarán al proyecto de vídeo en Cinelerra. Lo anterior incluye y no se limita a sonidos, vídeos, imágenes y escenarios de producción. Una vez que se edita el proyecto de vídeo, se encuentra listo para comenzar la etapa de renderizado empleando la potencia computacional disponible en la infraestructura de la nube que está compuesta por recursos de hardware no dedicados. La etapa de renderizado finaliza con la creación de una versión final del programa de TV listo para transmitir. Los multimedios resultantes se almacenan en la videoteca, para que los mismos pueden ser tomados por los operadores de transmisión (*Play Out*) para ser enviados al aire y ponerlos a disposición de los usuarios finales por todos los métodos de difusión que actualmente se emplean. Esos

contenidos, también pueden ser accedidos por el personal académico desde el salón de clases usando la red universitaria existente o bien el público en general empleando la transmisión por Internet y distribución de contenido a través de diferentes plataformas dedicadas a ello.

Centro de entrenamiento

Este escenario propone el uso de Cinelerra como herramienta en entornos educativos que emplean una nube compuesta por recursos de hardware no dedicados. En este caso, la nube se emplea como una cola de renderización donde los proyectos se procesarán como un sistema por lotes para finalmente obtener los vídeos listos para distribuirse.

El flujo de trabajo inicia cuando los estudiantes crean proyectos de edición no lineal utilizando Cinelerra en sus estaciones. Los vídeos de origen, los efectos de transición y los medios adicionales necesarios conforman esos proyectos que adicionalmente se integran a la actividad creativa del propio editor. Cuando finaliza el proceso de edición, el proyecto se almacena en el nodo maestro. Este elemento comparte el directorio de medios utilizando el protocolo de sistema de archivos de red (NFS) y también se ha instalado Cinelerra como una cola de procesamiento que gestiona los trabajos pendientes de ser renderizados. Estos trabajos se envían a los nodos de renderización virtual en los que se crea el vídeo listo para distribuirse. Los nodos de procesamiento virtual son máquinas virtuales habilitadas con Cinelerra que montan los directorios NFS compartidos por el nodo maestro. Esto de forma gráfica se puede apreciar en la figura 4.9.

Los nodos virtuales de renderización pueden ser desplegados en los laboratorios informáticos de las instituciones educativas. El vídeo creado por el proceso de renderizado finalmente se almacena en el directorio compartido del servicio NFS en el nodo maestro. Hay que hacer notar que las estaciones de trabajo del alumno, el nodo maestro y los nodos virtuales deben estar accesibles e interconectados por una red de comunicaciones fiable con capacidad de manejar la transferencia de archivos de gran tamaño. El objetivo principal de este escenario es la posibilidad de reutilizar el poder computacional inactivo disponible en los laboratorios de forma elástica, ya que están disponibles para reducir el tiempo necesario para terminar el proceso de renderizado.

Resultados

Cinelerra se usó para editar y renderizar vídeo en resoluciones de 720p y 1080p, con flujos de entrada/salida usados para televisión de alta definición (HD). Como estándar común para

Figura 4.9: Flujo de trabajo en un centro de capacitación, nótese que los estudiantes no saben donde se encuentran realmente los nodos físicos, pueden los mismos estar distribuidos en múltiples sitios siempre que estén conectados a través de una red.

audio/vídeo, se utilizó el códec MPEG4 [23]. Actualmente este ha sido el estándar preferido a nivel mundial para la edición de vídeo desde la década de los noventa.

Para evaluar si la infraestructura propuesta en la nube podría considerarse como una buena opción para el procesamiento, preparamos una serie de pruebas usando archivos fuentes definidos como recursos de un proyecto de edición. La *primera* fue con un vídeo de salida de 40 segundos de duración con una resolución de 1080/60p. La *segunda* empleó un vídeo de 30 minutos a 720p. La *tercera* prueba usó un vídeo de 30 minutos a 720p, compuesto por treinta fragmentos de un minuto sin transiciones ni efectos. Ejecutamos las pruebas de renderizado inicialmente sólo en el nodo maestro utilizando dos núcleos, luego se ejecutó con uno y posteriormente con dos nodos virtuales, empleando dos núcleos cada uno. El objetivo de las pruebas es saber cuánto tiempo se necesita en cada configuración y cómo el proceso de renderizado puede ser beneficiado o penalizado con el uso de *hyperthreading*. Las primeras dos pruebas se llevaron a cabo en la infraestructura de la televisora y la última en el centro de entrenamiento.

Cuando se utilizan varios nodos, se configuró la opción en Cinelerra que permite dividir

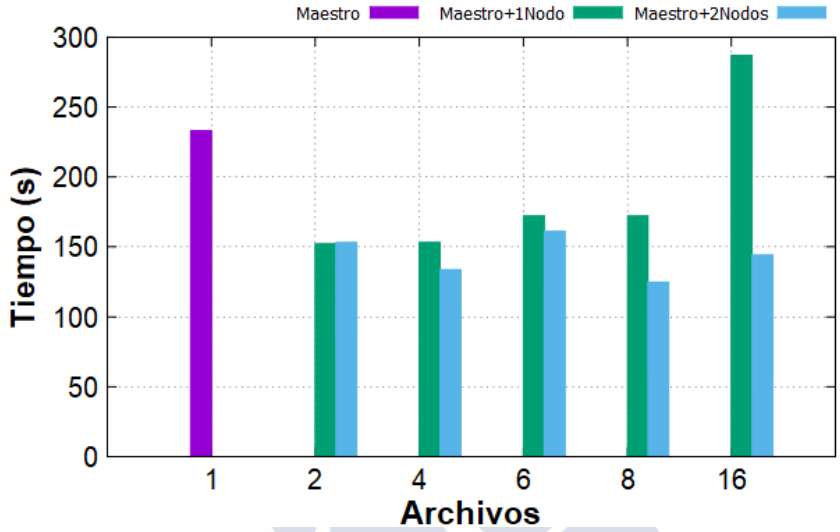

Figura 4.10: Desempeño con 40 segundos de vídeo a una resolución de 1080/60p. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse.

automáticamente el trabajo en varias partes. Cinelerra en sí mismo divide los trabajos tratando de dar el mismo número de partes a todas las máquinas participantes (incluído el nodo maestro). De esta forma, el número de archivos generados escalará con la cantidad de nodos empleados (siendo uno cuando sólo se realiza el renderizado en el nodo maestro, dos, cuatro, seis, ocho hasta dieciséis al contar con múltiples nodos). Este método evita que existan nodos sin trabajo en las diferentes pruebas realizadas.

En la primera prueba se realizó con el vídeo de 40 segundos a 1080/60p. El peor resultado fue obtenido cuando usamos el nodo maestro en combinación con un nodo virtual y especificando una salida de dieciséis archivos que requirieron en total 287 segundos, donde el nodo maestro necesitó 233 segundos para completar la tarea. El mejor tiempo se obtuvo empleando el nodo maestro y dos nodos virtuales especificando ocho archivos, la prueba se completó en 125 segundos. Empleando la misma combinación con cuatro archivos, el tiempo requerido fue 134 segundos. Usando el nodo maestro y un nodo virtual, se obtuvo los mejores resultados con dos y cuatro archivos de salida; utilizando únicamente 153 segundos en cada uno, esto es 28 segundos más que el mejor tiempo de toda la presente prueba. Los resultados pueden observarse en la figura 4.10.

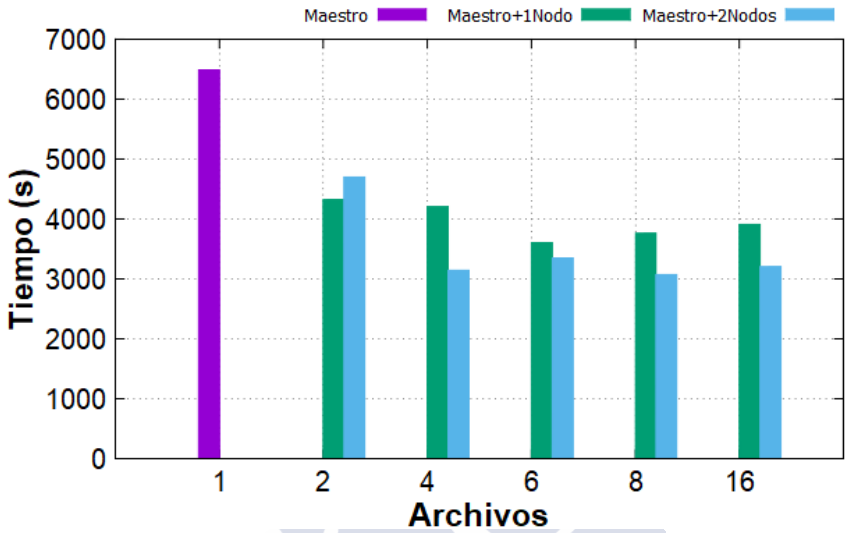

Figura 4.11: Desempeño con 30 minutos de vídeo a una resolución de 720p. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse

La segunda prueba es un proyecto que se requiere renderizar en un archivo de 30 minutos con una resolución de 720p, se pueden ver los resultados en la figura 4.11. En este caso se combinaron varias fuentes de vídeo y se agregaron transiciones para integrarlos y se agregaron efectos básicos al conjunto. El tiempo de renderizado para el nodo maestro fue de 6487 segundos, siendo el peor resultado. Al agregar un nodo virtual y permitir fraccionar el proceso en seis archivos se obtuvo la mayor eficiencia posible en esta configuración utilizando sólo 3605 segundos. Al agregar un segundo nodo virtual se logró el mejor resultado con el presente proyecto, requiriendo 3072 segundos, permitiendo el uso de ocho fragmentos. Debido a que el proyecto esta formado por fuentes de vídeo heterogéneo combinada con efectos y transiciones que hacen que el conjunto difiera en complejidad en diferentes secciones, siendo unas partes la renderización mas sencilla que otras. Es por ellos que se realizó una tercera prueba con orígenes de vídeo homogénea, permitiendo a cada nodo realizar actividades computacionales similares.

La tercera prueba se preparó con la ingesta de 30 archivos de vídeo de duración de 60 segundos cada uno, vale aclarar que es un archivo que se le realizó múltiples copias de sí mismo, los resultados se pueden apreciar en la figura 4.12. La resolución es de 1080i y no se

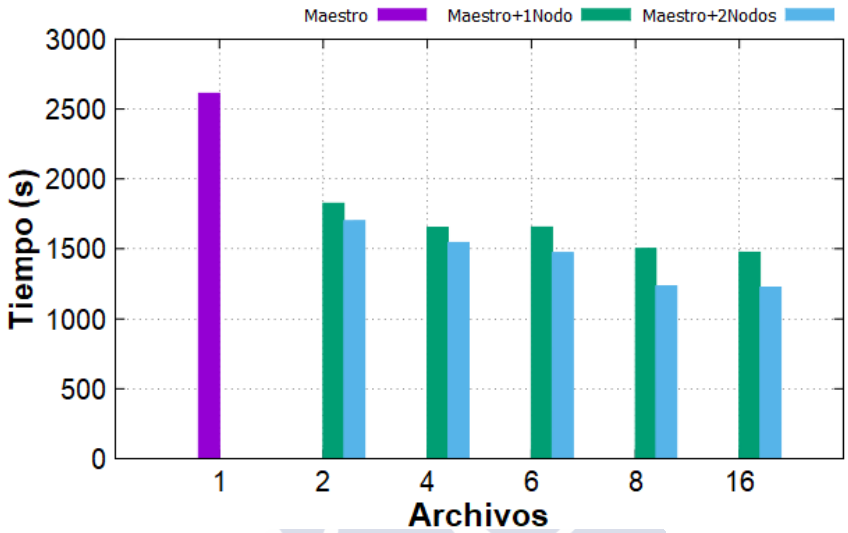

Figura 4.12: Desempeño con 30 minutos de vídeo a una resolución de 1080/60i. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse

utilizaron transiciones ni efectos. El mejor tiempo obtenido fue de 1226 segundos usando el nodo maestro y dos nodos virtuales y especificando dieciséis archivos de salida, un resultado similar se obtuvo con la misma configuración pero con ocho archivos. Empleando únicamente el nodo maestro se necesitó de 2614 segundos, este es el peor resultado de la presente prueba. El menor tiempo usando el nodo maestro y un nodo virtual fue de 1477 segundos con dieciséis archivos de salida. Se obtuvo resultados similares con esta configuración al especificar ocho archivos de salida, utilizando sólo 24 segundos adicionales.

Conclusiones

El uso de las tecnologías de la computación en la nube para integrar vídeos para entornos educativos y de difusión es factible sin la necesidad de incurrir en costos de licenciamiento ni de equipo especializado o dedicado. Se analizó una infraestructura en la nube para Cinelerra, demostrando su viabilidad para entornos académicos y televisivos, se logró demostrar que de esta forma se puede ampliar el número de usuarios desarrollando de forma concurrente diferentes proyectos y renderizar de manera eficiente; empleando recursos disponibles de forma

típica y gestionados a través de un entorno virtual sacando provecho de las tecnologías para estos fines.

Con las diferentes pruebas llevadas a cabo en los dos casos de uso, se puede deducir que para obtener el mayor provecho de los recursos es necesario identificar el número idóneo de fragmentos en los cuales hay que dividir el proyecto de Cinelerra que se enviará a procesar en los nodos de trabajo disponibles. En los resultados mostrados, el hecho de contar con ocho fragmentos del proyecto cuando se tiene un nodo maestro más dos nodos de renderización entrega los mejores tiempos en todas las pruebas, con lo que se afianza la necesidad de identificar la carga idónea para que la infraestructura sea viable.

Es importante que al momento de planificar este tipo de soluciones se debe evitar que en los nodos virtuales se den lugar situaciones de sobre demanda de recursos que puedan ocasionar pérdida de rendimiento debido a la competencia que puede existir entre las máquinas virtuales y el anfitrión. Es por ello que se indicó que es necesario utilizar capacidad computacional que esté disponible, de manera que si se utilizan estaciones de trabajo compartidas se debe organizar bien la cantidad de recursos con los que el hipervisor va a disponer.

Los resultados indican que la infraestructura y configuración propuesta resultan ser eficientes con proyectos grandes, utilizando tiempos razonables para concluir proyectos. Para el caso de fragmentos cortos es más viable utilizar la estación de trabajo de los editores, ya que de esta forma se evita el atraso que pueda presentarse al fragmentar y distribuir archivos demasiados pequeños.

4.3. Medicina

En la actualidad los recursos de computación de altas prestaciones son requeridos para las actividades de investigación médica, debido al volumen de datos provenientes de equipos y sensores utilizados para análisis de pacientes que ofrecen grandes cantidades de datos que requieren ser analizados e interpretados. La medicina de precisión es una nueva área del conocimiento de participación multidisciplinaria donde las ciencias de la computación ofrecen las herramientas puente entre los diferentes profesionales que combinan esfuerzos con personal sanitario. En esta sección, se analiza una aplicación relacionada con neurociencias e imágenes biomédicas llamada FreeSurfer, esta se caracteriza por ser una herramienta de software que actualmente está enfocada en medicina personalizada y de precisión. El acceso a equipos radiológicos digitales junto con la capacidad de realizar análisis de diversa índole hacen nece-

sario contar con la capacidad de sistematizar el manejo de datos para investigación y acciones clínicas [1][100][91]

4.3.1. FreeSurfer

Las enfermedades neurológicas son causa importante de muerte e incapacidad permanente en la población mundial [22, 58, 61]. Por ello, la neuroimagen ha avanzado rápidamente en las últimas dos décadas. Las técnicas avanzadas de neuroimagen no invasiva, como la imagen de resonancia magnética (IRM), han permitido la visualización y el análisis de la función y la estructura cerebral, con un nivel de detalle sin precedentes, transformando la forma en que estudiamos el sistema nervioso en condiciones normales y patológicas [48]. Estos avances van de la mano con nuevos recursos computacionales y/o aplicaciones para poder realizar análisis útiles de las imágenes procesadas [51][103].

La nube debido a su flexibilidad de gestión, hace que sea una solución mucho más eficiente, escalable y adaptable para usarse en neurociencias. Los esquemas de implementación de la nube ofrecen la facilidad de gestionar los recursos mediante una plataforma centralizada, permitiendo escalar los recursos computacionales bajo demanda o bien dependiendo de los perfiles de los usuarios [44]. El desarrollo de la nube ha permitido compartir recursos, lo que ha facilitado el manejo de mayor cantidad de información heterogénea, conglomerada en la actualidad bajo el término BigData [96]. En esta sección se evalúa la idoneidad de las infraestructuras en la nube aplicadas al análisis de imágenes de resonancia magnética [63].

Una de las aplicaciones utilizadas en medicina es FreeSurfer [25], con la cual se pueden realizar múltiples análisis, reconstrucción y segmentación de la corteza cerebral, permitiendo el manejo de imagen estructural y funcional por medio de visualización. Permite realizar un registro de imágenes, tractografía y otros análisis. Este software requiere amplios recursos de CPU para el procesamiento de resonancias magnéticas, lo que la convierte en una aplicación de cálculo intensivo, que usualmente se ejecuta sobre estaciones de trabajo dedicadas y haciendo uso de clústeres para disminuir el tiempo de ejecución. Se ha evaluado el empleo de infraestructuras *cloud* para su ejecución. Para ello, se realizó la comparación de las diferencias del rendimiento que existen entre un ordenador físico e instancias virtualizadas en la nube, con el fin de determinar si estas últimas representan una alternativa viable para la reducción y procesamiento de datos para el estudio de enfermedades cerebrales a través de neuroimagen.

FreeSurfer ha sido desarrollado en el *Athinoula A. Martinos Center for Biomedical Ima-*

ging [55], centrándose en tres propósitos principales: I) la creación de modelos computarizados del cerebro a partir de imágenes de resonancia magnética, II) medir diversas propiedades morfométricas del cerebro, en especial el espesor cortical, la características de curvatura, volúmenes regionales corticales y subcorticales, y III) la normalización espacial cortical entre sujetos sobre la base de la alineación de patrones de plegado de un individuo con las de una población promedio de patrón de plegamiento cortical para establecer correspondencia entre regiones anatómicas homólogas [21].

Existen trabajos relacionados para evaluar el rendimiento de FreeSurfer enfocados principalmente en la evaluación del rendimiento en computación híbrida donde Delgado J. et al. [16] propone un nuevo *pipeline* con la intención de optimizar el uso de recursos CPU y GPU para el procesamiento en un modelo de computación híbrida. Este esquema combina la versión GPU existente de FreeSurfer con un esquema paralelo de gestión de tareas. Por otra parte Gronenschild, E. et al. [32], evalúan los efectos sobre el rendimiento de FreeSurfer derivados de la utilización de diferentes tipos de estaciones de trabajo y diferentes sistemas operativos.

Infraestructura

El flujo de trabajo o *workflow* de FreeSurfer para la reconstrucción cortical y la segmentación volumétrica de imágenes de resonancia magnética es usualmente empleado como herramienta de investigación médica del cerebro y comprende dos flujos: I) el estudio de las superficies, que calcula para cada punto del cortex, el grosor cortical, el volumen, el área de superficie y la curvatura y, II) los estudios volumétricos, que cuantifican los volúmenes de las estructuras subcorticales [55].

El *workflow* se encuentra estructurado como flujos de procesos de computación cuya ejecución se encuentra controlada por medio del script *recon-all* [15]. Típicamente, este flujo de trabajo puede tardar decenas de horas de computación intensiva, dependiendo principalmente de la plataforma de hardware y software empleada, el tiempo requerido se traduce en un atraso en el diagnóstico de los pacientes y/o en una demora en la obtención de resultados por parte de los investigadores [26, 42]. En este estudio se ha empleado FreeSurfer v5.3. Las diferentes infraestructuras de procesamiento empleadas se detallan a continuación:

- **Plataforma cloud:** Basada en una infraestructura configurada con Apache CloudStack 4.4 [84] empleando el hipervisor KVM [87], que ha sido desplegada en los nodos de computo (CN ó *compute node*). Los CN emplean procesadores Intel Core i7-2600@3.4

GHz con 8 GB de RAM y sistema operativo (SO) CentOS 6.8 64 bits [24]. El procesador empleado tiene cuatro núcleos y ocho hilos, una caché L1 de 4x32 Kbytes de capacidad para instrucciones y 4x32 Kbytes para datos, una caché L2 de 4x256 KBytes, y una cache compartida L3 de 8 MB. También dispone de *Intel Virtualization Technology* (VT-x) y virtualización de I/O (VT-d). Un NAS de 12 TB proporciona el almacenamiento compartido, mediante NFS v4, para la infraestructura cloud. La red de interconexión de esta infraestructura está basada en RTL8111/8168/8411 *PCI Express Gigabit Ethernet Controller*.

- **Infraestructura física:** Tiene procesadores Intel Core i7-2600@3.4 GHz con 8 GB de RAM y sistema operativo (SO) CentOS 6.8 64 bits. El procesador empleado tiene cuatro núcleos y ocho hilos, una caché L1 de 4x32 Kbytes de capacidad para instrucciones y 4x32 Kbytes para datos, una caché L2 de 4x256 Kbytes, y una cache compartida L3 de 8 MB. También dispone de *Intel Virtualization Technology* (VTx) y virtualización de I/O (VT-d). Un NAS de 12 TB proporciona el almacenamiento compartido, mediante NFS v4, para la infraestructura cloud. La red de interconexión de esta infraestructura está basada en RTL8111/8168/8411 *PCI Express Gigabit Ethernet Controller*.

- **Anfitrión físico:** Este despliega una máquina virtual KVM, que dispone de 4 GB de RAM y un disco duro de 30 GB. La red de interconexión proporcionada por el hipervisor emplea controladores Virtio. El sistema operativo huésped es CentOS 6.8 64 bits.

- **Máquina VirtualBox:** Esta MV desplegada sobre VirtualBox v5.1.3, cuenta con 4 GB de RAM. Emplea el *chipset* PIIX3, I/O APIC y reloj hardware empleando tiempo UTC. El sistema de almacenamiento virtualizado es de tipo AHCI, empleando la cache de anfitrión de entrada/salida. Como en el caso anterior, se ha empleado CentOS 6.8 64 bits como sistema operativo (SO) huésped.

Resultados

En este trabajo hemos considerado tres escenarios de pruebas: el primero la evaluación de FreeSurfer en una plataforma cloud empleando KVM como hipervisor. Los otros dos escenarios se han diseñado con la finalidad de comparar los resultados obtenidos con otras plataformas. De esta manera, el segundo de los escenarios está orientado a evaluar a FreeSurfer ejecutado sobre la plataforma física, mientras que el tercero, y último, está orientado a ejecutar el FreeSurfer sobre VirtualBox.

En cada uno de los escenarios se lleva a cabo la misma metodología, que consiste en la instalación de FreeSurfer en un directorio del NAS que proporciona el almacenamiento de red compartido por medio de una red Gigabit Ethernet mediante NFS v4 y que se monta en la misma ruta en cada uno de los sistemas empleados. Además, también se efectúa la misma configuración del entorno en los diferentes escenarios considerados.

Una vez desplegados los escenarios anteriormente descritos, se ejecutan las pruebas de FreeSurfer utilizando los datos de resonancia magnética del individuo *Bert* que se incluye en la distribución de esta aplicación. En cada uno de los escenarios descritos, se han realizado dos tipos de mediciones, la *primera* de ellas evalúa el tiempo para completar el análisis de *autorecon 1* para el individuo *Bert* en una serie de doce ejecuciones continuas, de las cuales se han considerado las diez últimas del análisis con la finalidad de eliminar el posible efecto de *warm-up* de los sistemas, lo cual podría afectar a las dos primeras mediciones. Las etapas procesadas por *autorecon 1* se corresponden a:

I) *motion correction and conform*, II) *NU (Non-Uniform intensity normalization)*, III) *talairach transform computation*, IV) *intensity normalization* y, V) *skull strip*

El *segundo* tipo de medidas está destinado a extraer el perfil de utilización de la CPU en el análisis de este individuo, de tal forma que durante la ejecución de FreeSurfer se mide la utilización de la CPU de forma periódica en intervalos de 10 segundos.

En la figura 4.13 se representa el tiempo empleado por FreeSurfer para ejecutar el *workflow* correspondiente al *autorecon 1* en las plataformas anteriormente descritas. La infraestructura anfitriona (*host*) es la más eficiente empleando un tiempo promedio de 428.2 segundos, para procesar el *workflow* de *autorecon 1*. Si se ejecuta lo anterior sobre la máquina virtual KVM, el tiempo promedio empleado es de 441.6 segundos, siendo un 3.2 % más lento que el resultado obtenido sobre el anfitrión. El resultado más desfavorable se obtiene cuando se ejecuta el *workflow* de *autorecon 1* sobre la máquina virtual de VirtualBox, obteniéndose un resultado promedio de 446.6 segundos, siendo un 4.3 % más lento el procesamiento si se toma como referencia el resultado obtenido en el anfitrión.

Uno de los desafíos más importantes de las infraestructuras cloud es mejorar el rendimiento de las operaciones de entrada/salida, que constituyen uno de los cuellos de botella en este tipo de plataformas. Por el contrario, las infraestructuras cloud efectúan de forma muy eficiente la asignación compartida de la CPU a diversas máquinas virtuales, lo que hace que este tipo de infraestructuras sean especialmente interesantes a la hora de ejecutar aplicaciones *CPU-bound*. Como se ha visto, el perfil de procesamiento de FreeSurfer se corresponde con

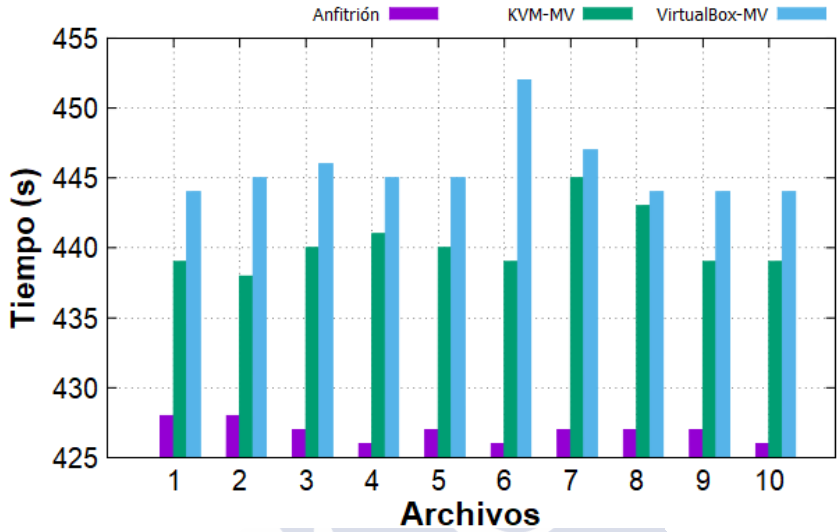

Figura 4.13: Tiempo de ejecución de *autorecon 1* para el sujeto *Bert* para cada una de las series de medidas obtenidas sobre la plataforma física (anfitrión), sobre la máquina virtual KVM y sobre la máquina virtual de VirtualBox.

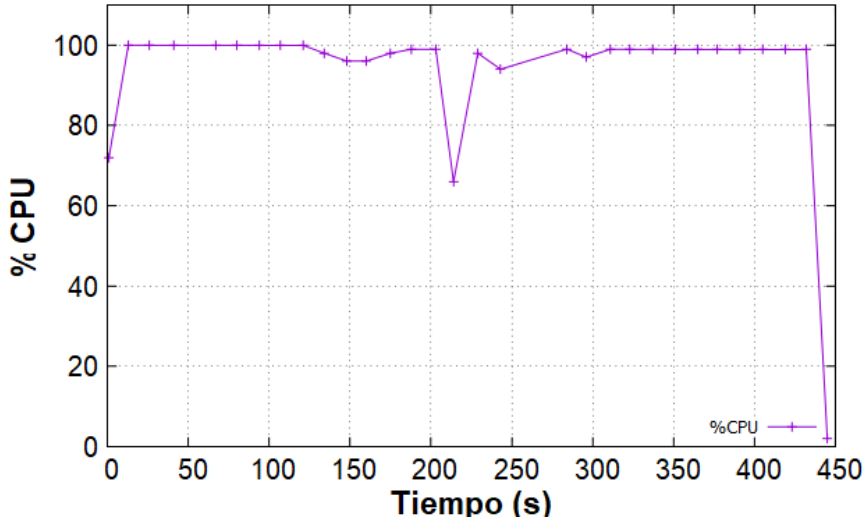

Figura 4.14: Perfil de utilización de la CPU durante la ejecución de *autorecon 1* para el individuo *Bert* frente al tiempo.

una aplicación de este tipo, haciendo factible su ejecución en entornos cloud. En la figura 4.14 se muestra el perfil de utilización de la CPU durante la ejecución del *workflow* del *autorecon 1* en FreeSurfer. Como se puede observar, el *workflow* efectúa un uso intensivo del procesador durante prácticamente el 88 % del tiempo empleado en la ejecución, lo que supone que el *workflow* ejecutado podría incluirse en la categoría de aplicación *CPU-bound*.

A pesar de la pequeña sobrecarga en el rendimiento que sufriría FreeSurfer al ser desplegado en el entorno cloud (con una sobrecarga entorno al 3.2 %), el despliegue de esta aplicación en este tipo de plataformas proporcionaría importantes beneficios a la hora de diagnosticar las enfermedades cerebrovasculares ya que sería factible escalar los recursos computaciones bajo demanda, de forma elástica y en el momento en el que se necesiten, contribuyendo a ofrecer un diagnóstico más rápido a los pacientes afectados por este tipo de dolencias, sin requerir de plataformas dedicadas.

Conclusiones

En este estudio se ha evaluado la idoneidad de las infraestructuras cloud virtualizadas para ejecutar *workflows* de FreeSurfer, que es un conjunto de herramientas de software para el análisis y la visualización de imagen estructural y funcional del cerebro. Los análisis han sido efectuados sobre las infraestructuras siguientes: I) instancia cloud virtualizada mediante KVM, II) plataforma física y, III) entorno virtualizado empleando VirtualBox.

Los resultados obtenidos en la ejecución del *workflow* de *autorecon 1*, que incluye las etapas *Motion Correction and Conform*, *on-Uniform intensity normalization*, *Talairach transform computation*, *Intensity Normalization y Skull Strip*, muestran que: I) la plataforma física es la más eficiente a la hora de ejecutar el *workflow* previamente descrito, habiéndose obtenido un tiempo de ejecución de 428.2 segundos, II) KVM es el entorno virtualizado más eficiente, habiendo sido sólo un 3.2 % más lento que la plataforma física, mientras que VirtualBox es un 4.3 % más lento y, III) FreeSurfer hace un uso intensivo de la CPU, durante el 88 % del tiempo de ejecución, siendo una aplicación CPU-bound.

Como se ha visto, la penalización por el uso de las infraestructuras virtualizadas es menor que el 4.5 %, por lo que se concluye que FreeSurfer es una aplicación que puede ser desplegada en entornos cloud con apenas una pequeña sobrecarga. Esto supone, que a pesar de la leve penalización en el tiempo de ejecución de las infraestructuras cloud, por si solas representan una interesante alternativa para el despliegue de FreeSurfer, ya que permiten escalar los recursos computacionales de forma elástica, bajo demanda, y cuando se necesitan, lo que

permitiría acelerar el procesamiento de los datos de los pacientes. Todo esto contribuiría a proporcionar unos tiempos de respuesta de cara a brindar diagnósticos oportunos de enfermedades cerebro-vasculares.

4.4. Otras aplicaciones

La optimización de una infraestructura de alto rendimiento debe considerarse de naturaleza flexible para poder emplearse en actividades específicas más allá de aplicaciones científicas y educativas; este tipo de soluciones debe de poderse utilizar en operaciones diarias en áreas muy específicas como ser en administración y operaciones de gobierno. En esta sección se describe un caso de uso que involucra bases de datos estructuradas.

4.4.1. Reducción de datos no estructurados

El paradigma de la computación en la nube provee la facilidad de reutilizar infraestructuras de forma transparente al usuario, para ello se puede contar con capacidad computacional a un menor costo al ser comparado con soluciones dedicadas. De esta manera cualquier requerimiento de carácter temporal o de ejecución planificada podría enviar trabajos que aprovechasen la nube para proveer una infraestructura como servicio bajo demanda, donde esta podría ejecutar las aplicaciones requeridas, este tipo de necesidades es más común en instituciones educativas y administrativas [20].

En este apartado se evalúa el rendimiento que pueden tener los procesos de análisis y reducción de datos no estructurados contenidos en bases de datos relacionales, específicamente el manejo del contenido de las cadenas de texto almacenadas como atributos en tablas manejadas por el gestor relacional de bases de datos (RDBMS) llamado MySQL. Se ha hecho una comparación de la ejecución en un ordenador anfitrión físico respecto a instancias en la nube e instancias virtual; todos ellos empleando los mismos recursos computacionales. La finalidad es conocer los tiempos de ejecución de cada plataforma y estimar de esa manera la conveniencia de cada una en este de tipo de aplicaciones. Es por ello que las secuencias de códigos o *scripts* utilizados se han desarrollado como funciones y procedimientos almacenados, con la finalidad de procesar los datos, desde la normalización, clasificación, búsqueda de palabras y raíces semánticas previamente definidas.

Para este caso de uso, se emplearon expresiones regulares sobre cadenas de texto que se originan a partir de procesos de captura de datos sin estructura conteniendo oraciones en prosa,

para este caso se utilizó datos provenientes de un instrumento tipo encuesta. Los *scripts* son ejecutados dentro del entorno RDBMS, teniendo como especial característica que los datos usados se integran en un esquema a partir de un volcado o *dump*.

Es importante determinar el valor agregado de las diferentes configuraciones en esta aplicación de manejo de cadenas de texto que requieren concurrencia; de forma que medir sus diferencias en el procesamiento entre ambas plataformas (física y virtual) resulta ser el objetivo más importante de este caso de uso. Es por ello, que se ha considerado el uso de discos duros con tecnología de estado sólido (*Solid State Drive* o SSD), para conocer si ofrecen mejores prestaciones en el acceso al sistema de ficheros al ser comparado con un sistema de almacenamiento de red (NFS); este último se utilizó para la infraestructura *cloud* donde se realizaron las pruebas, sobre todo porque ofrece flexibilidad y reusabilidad del hardware.

Detalles de la aplicación

En la Universidad Nacional Autónoma de Honduras (UNAH) se lleva a cabo una Valoración del Desempeño Docente (VDD) que es accesible por medio de una interfaz web, donde los estudiantes cada período evalúan a los docentes que les imparten los diferentes cursos. Este aplicativo está compuesto de dos partes, la *primera* consta de una evaluación cuantitativa a través de preguntas cerradas diseñadas en una escala positiva, mientras que la *segunda* está conformada por preguntas abiertas donde el estudiante puede libremente detallar su experiencia con sus docentes.

Este instrumento genera millones de registros al finalizar cada período académico, para procesar estos datos se emplean los *scripts* mencionados, ejecutados en el gestor de bases de datos MySQL. Estos hacen, entre otras cosas, un conteo de muestras por docentes y una búsqueda de palabras y raíces semánticas agrupadas en patrones de hábitos (previamente definidos) que se espera que se encuentren en el texto libre introducido como respuestas del instrumento VDD.

Entre los *scripts* evaluados se encuentra uno que se denomina muestreo, este crea una tabla en la que se introducen los registros únicos de los docentes que recibieron una evaluación y calcula el valor global de respuestas que recibió. Posteriormente, se ejecuta el *script* llamado patrón que se encarga de buscar los aciertos de las palabras y raíces. El método utilizado es la búsqueda de cadenas de caracteres (*strings*) mediante expresiones regulares. Para evaluar el rendimiento se utilizaron cinco patrones con una longitud media que oscila entre siete y ocho caracteres por palabra. Para fines prácticos un patrón debe entenderse como un conjunto de

palabras y raíces semánticas agrupadas y relacionadas entre sí, que en conjunto pueden ser usadas para describir comportamientos de los docentes que son evaluados.

Se ha descartado el uso de técnicas de búsqueda de texto completa (*full text search*) [67] porque requieren de la creación de índices optimizados que permitan hacer consultas basadas en el significado de las palabras utilizadas, complementados con el sentido que agregan al texto cuando se hace uso de adjetivos, adverbios, verbos y sus conjugaciones; esto permite hacer búsquedas extendidas utilizando los resultados como nuevos parámetros recursivos de las subsiguientes consultas. El despliegue de este tipo de tecnología difiere en cada gestor de bases de datos, limitando así su portabilidad a otras plataformas. Por lo tanto se ha preferido el uso de expresiones regulares para facilitar la implementación y migración entre plataformas de diversos fabricantes de los gestores de base de datos basados en el estándar SQL.

El RDBMS utilizados ha sido el MySQL Community Edition v5.6 64bit [68], con el cual se creó una base de datos con el cotejamiento UTF-8 y del tipo MyISAM. Una única tabla sirvió como fuente de datos para análisis. La misma es resultado de restaurar un volcado de datos de otro sistema, que contiene 902,051 registros para analizar el rendimiento. Los *scripts* evaluados procesan los datos para normalizarlos, clasificarlos y para hacer una búsqueda de palabras y raíces semánticas definidas a través de expresiones regulares sobre cadenas de texto con el contenido de interés, el resultado es una tabla en la que se resume el número de aciertos por patrón de comportamiento y la muestra global de las respuestas por docente.

Infraestructura

Los experimentos de ejecución se han realizado empleando dos infraestructuras, una física y una cloud. El anfitrión físico es un ordenador que cuenta con un microprocesador Intel Core i7 Q740 con una velocidad de reloj de 1.73 Ghz, 4 GB de RAM, disco duro de estado sólido SSD de interfaz SATA. Las pruebas del anfitrión físico se realizaron con dos núcleos activos, esta configuración se cambió en el BIOS. Debido al *hyperthreading* el sistema operativo puede acceder a cuatro núcleos. La instancia cloud, es una infraestructura computacional virtualizada que se desplegó con las mismas características de hardware que el anfitrión físico descrito. Esta instancia se ejecutó siempre sobre el mismo nodo, ya que el cloud utilizado cuenta con varios de ellos, es gestionado mediante Apache Cloudstack y accede a un sistema de archivos en red tipo NFS. Con el fin de tener una medición del tiempo de ejecución lo más precisa posible, se ha hecho una sincronización del reloj del ordenador anfitrión y de la instancia cloud, por medio del uso del protocolo NTP [57], implementado al usar el comando *ntpdate*.

Finalmente, el sistema operativo común para todos las instancias y equipos fue Linux en su distribución CentOS v6.6 64 bit [85].

Como sistema de almacenamiento primario del cloud se empleó un almacenamiento de red dedicado que implementa el protocolo NFS, su hardware fue fabricado por Synology y el modelo DS411, configurado con dos discos duros de 2 TB de 7500 RPM e interfaz SATA2; ambos configurados en un arreglo RAID 1. Todos los dispositivos que se utilizan, están interconectados a través de una red Ethernet Gigabit que provee un *switch* del fabricante Cisco modelo Catalyst 500 Series. La instancia virtual, se ha desplegado con 4 GB de memoria RAM, un disco duro virtual de 10 GB y se le ha deshabilitado la interfaz gráfica.

La instancia virtualizada dentro de un entorno de nube privada es gestionada mediante Apache CloudStack v4.0.1 [84], implementando el hipervisor KVM [87]. Tanto la plataforma física como la virtual se ejecutaron empleando recursos computacionales similares, con el fin de evitar introducir sesgo en los resultados. La administración de la nube se realizó con los procedimientos típicos y con los componentes que conforman la gestión bajo Apache CloudStack, descrito anteriormente en este documento.

Metodología

Con la finalidad de conocer de forma unificada el rendimiento de las diferentes plataformas en lo que respecta al sistema de archivos, se hizo uso de IOzone v3.424 64bit [65], que es un *benchmark* sintético que se utiliza para conocer la medida del rendimiento del sistema de archivos. IOzone fue ejecutado evitando la influencia de la cache en el acceso al sistema de ficheros. Para esto, se usó un tamaño de archivo que fuese equivalente al doble de la memoria RAM disponible por cada plataforma, evaluando de esta manera la velocidad de escritura, re-escritura, lectura, re-lectura del tipo secuencial y aleatoria.

Posteriormente, se ejecutaron los *scripts* para analizar los datos de la VDD, cada iteración de ejecución requirió tener el cuidado de borrar los resultados y reiniciar las instancias de ejecución al finalizar cada ciclo. Cada ciclo se componía de un total de doce mediciones de búsqueda por cada patrón por plataforma, pero se descartaron las dos primeras para limitar el efecto de calentamiento o *warm up* del gestor de base de datos. Entre cada ciclo de mediciones, se eliminó la estructura de datos y se restauraba de nuevo a partir de un respaldo que contiene los datos crudos y se reiniciaban las medidas. También se cuantificó el uso de la memoria RAM y de la memoria de intercambio SWAP previo al inicio y posterior a la ejecución, con ello se busca considerar la variación del efecto del uso de la memoria y en qué medida esta

interferiría con los resultados obtenidos.

Resultados

Los resultados están agrupados por plataforma en la que MySQL se ejecutó como ser el *cloud* y la máquina física descritos previamente, sobre los que se ha realizado la comparación de los diferentes resultados. Luego se describen y comparan los resultados obtenidos para IOzone en el *cloud*, en la máquina física con discos SSD y también en la máquina física sobre NFS.

- **MySQL en cloud:** La evaluación inició con el muestreo, que en el caso de la instancia en la nube, requirió en promedio 6919.34 segundos para completar y reducir los datos en una tabla con los registros y frecuencias de respuestas, los resultados se representan en la figura 4.15. Las mediciones presentaron una desviación estándar de 502.06 segundos. La ejecución del *script* patrón, tuvo un comportamiento heterogéneo entre las diferentes series en la instancia *cloud*. El cálculo medio del tiempo requerido por los aciertos del patrón(1) fue de 6212.21 segundos, el patrón(2) utilizó una media de 6023.78 segundos, siendo el más eficiente. El patrón(3) utilizó 6314.30 segundos, el patrón(4) 6290.50 segundos y finalmente el patrón(5) utilizó 6635.32 segundos el cual representó el mayor tiempo de ejecución.

  En cuanto a los tiempos medios de ejecución por palabra, el patrón(3) requirió una media de 92.86 segundos por cada una, siendo el que usó menos tiempo de todos; en cambio el patrón(4) fue el que más tiempo empleó con una media de 232.98 segundos. Se hace notar que el patrón(1) y el patrón(5) tienen el mismo número de palabras, pero este último conjunto requirió 423.11 segundos más para completarse; muy probablemente este sea efecto de la moda estadística de las palabras que conforman al patrón(1), que tienden a ser encontradas en cadenas de mayor tamaño. El promedio general requerido por palabra fue de 135.67 segundos entre todos los patrones.

  Los tiempos de ejecución más dispersos fueron los del patrón(1) con una desviación de 237.46 segundos, el patrón(2) tuvo la menor desviación estándar entre sus medidas con un valor de 91.18 segundos; el patrón(3) presentó 155.52 segundos, el patrón(4) de 183.24 segundos y el patrón(5) de 118.53 segundos de desviación estándar respectivamente.

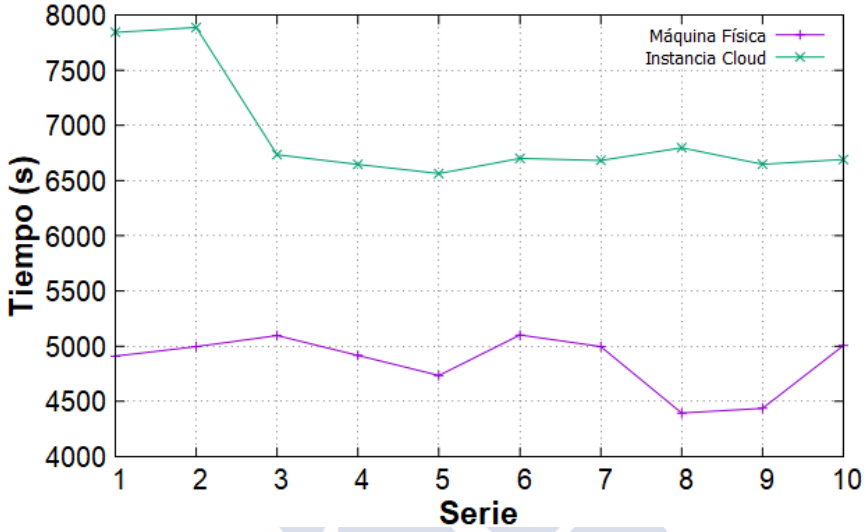

Figura 4.15: Tiempo (en segundos) utilizado por las infraestructuras para completar el script muestreo, en una serie de diez pruebas consecutivas de ejecución.

El uso de memoria RAM en la instancia *cloud* antes de la ejecución fue en promedio 676.30 MB, con una desviación estándar de 1.95 MB, con un uso máximo de 680 MB y el mínimo de 674 MB en toda la serie. Posterior a la ejecución, el uso promedio fue 1341.5 MB, con una desviación estándar de 20.87 MB, con un uso máximo de 1381 MB y un mínimo de 1327 MB. Entre antes y después de la ejecución la diferencia de uso oscila en 665.20 MB en promedio entre todas las mediciones realizadas.

- **MySQL en máquina física SSD:** El muestreo en la máquina física requirió en promedio 4859.08 segundos para completar y reducir los datos, pero presentó una desviación estándar de 255.82 segundos entre todas las mediciones, los resultados se muestran en la tabla 4.3 y en la tabla 4.4. En esta plataforma se mantuvo una menor desviación entre las medidas del tiempo de ejecución de cada patrón, pero entre ellos no fue completamente homogéneo. El patrón(1) utilizó una media de 4159.53 segundos para completarse, siendo el más eficiente. El patrón(2) usó 4299.23 segundos, el patrón(3) empleó 4324.69 segundos, el patrón(4) requirió 4369.01. El patrón(5) usó 4878.44 segundos el cual representó el mayor tiempo de ejecución. Es importante mencionar que el patrón(5), tiene el mismo número de palabras que el patrón(1), pero presentaron una

		Máquina Física		
Patrón	**Palabras**	**Tiempo Medio (s)**	**Desviación Estándard (s)**	**Tiempo Medio por Palabra(s)**
1	41	4159.53	106.21	101.45
2	55	4299.23	126.17	78.17
3	68	4324.69	165.42	63.60
4	27	4369.01	70.34	161.81
5	41	4878.44	267.67	118.99

Tabla 4.3: Tiempo de ejecución por patrón en la máquina física. A menor tiempo mayor eficiencia.

diferencia significativa de ejecución de 718.91 segundos.

El tiempo medio requerido por palabra por el patrón(1) fue de 101.45 segundos, el patrón(2) usó 78.17 segundos; el patrón(3) sólo requirió una media de 63.60 segundos, siendo el más eficiente. En cambio, el patrón(4) fue el que más tiempo empleó con una media de 161.81 segundos y, finalmente, el patrón(5) requirió 118.99 segundos.

En cuanto a la dispersión de las mediciones, el patrón(4) presentó la menor desviación estándar entre sus medidas, con un valor de 70.34 segundos. En cambio, los más dispersos fueron los del patrón(5), con una desviación de 267.67 segundos. El promedio general requerido por palabra fue de 94.96 segundos entre todos los patrones.

El uso de memoria RAM en la máquina física antes de la ejecución fue en promedio 1193.4 MB, con una desviación estándar de 23.56 MB, con un uso máximo de 1240 MB y el mínimo de 1170 MB en toda la serie. Posterior a la ejecución el uso promedio fue 1910.3 MB, con una desviación estándar de 43.28 MB, con un uso máximo de 2003 MB y mínimo de 1858 MB. Entre antes y después de la ejecución la diferencia de uso oscila en 716.90 MB en promedio en todas las mediciones realizadas.

- **Comparación de resultados MySQL:** La instancia en la nube requirió 42.4 % más tiempo de ejecución que la máquina física para completar el muestreo. Ello equivale a 2060.27 segundos de diferencia promedio para finalizar el procedimiento almacenado. A grandes rasgos, ambas infraestructuras mantuvieron un comportamiento homogéneo entre si durante las diversas mediciones en el conjunto de pruebas efectuadas.

  En cuanto a los patrones, el comportamiento entre infraestructuras fue similar entre todas las series. En todos los casos, la instancia en la nube requirió más tiempo para

		Instancia Cloud		
Patrón	**Palabras**	**Tiempo Medio (s)**	**Desviación Estándard (s)**	**Tiempo Medio por Palabra (s)**
1	41	6212.21	237.46	151.52
2	55	6023.78	91.18	109.52
3	68	6314.30	155.52	92.86
4	27	6290.50	183.24	232.98
5	41	6635.32	118.53	161.84

Tabla 4.4: Tiempo de ejecución por patrón de la instancia cloud. A menor tiempo mayor eficiencia.

completar las tareas de ejecución frente a la instancia física. La mayor diferencia fue en el patrón(1), que requirió un 49 % de tiempo adicional, en cambio el patrón(5) sólo necesitó un 36 % más. El patrón(2) necesitó 40 %, el patrón(3) un 46 % y el patrón(4) un 44 % de tiempo adicional respectivamente. En general, la instancia cloud requirió en promedio de un 43 % más tiempo para completar todos los patrones. En cuanto al uso de memoria, ésta se mantuvo sin variaciones importantes entre cada ciclo de ejecución. Es importante mencionar y hacer notar que la memoria de intercambio no fue utilizada en ninguna plataforma, lo que indica que el gestor de base de datos no escaló de forma que requiriera esos recursos.

En cuanto al uso de memoria RAM, antes de la ejecución el ordenador físico usaba en promedio 1193 MB y la instancia 677 MB con una diferencia de 43.25 % menos de uso inicial de memoria por parte del *cloud*. Esto se debe principalmente, a que el sistema operativo en la instancia carece de interfaz gráfica y se está ejecutando en modo consola, condición que se mantiene en todo momento. Posterior a la ejecución el uso del ordenador físico fue de 1931 MB y la instancia empleó 1340 MB, siendo un 30.60 % menos. Por otro lado, el uso promedio de la memoria RAM previo a la ejecución por parte de la instancia *cloud* fue de 665.20 MB siendo menor la infraestructura física que usó 716.90 MB; esto es un incremento de un 7.77 % por parte de la última. Lo anterior se representa en la figura 4.16.

- **IOZone en *cloud*:** En la ejecución de IOzone en esta plataforma, en cuanto a su velocidad promedio en escritura reportó 11352 Kbytes/s, en re-escritura presentó 11241 Kbytes/s, lectura 11686 Kbytes/s, re-lectura 12964 Kbytes/s, lectura aleatoria 9282 Kbytes/s y, finalmente, en escritura aleatoria 11190 Kbytes/s. En general, las mediciones mantu-

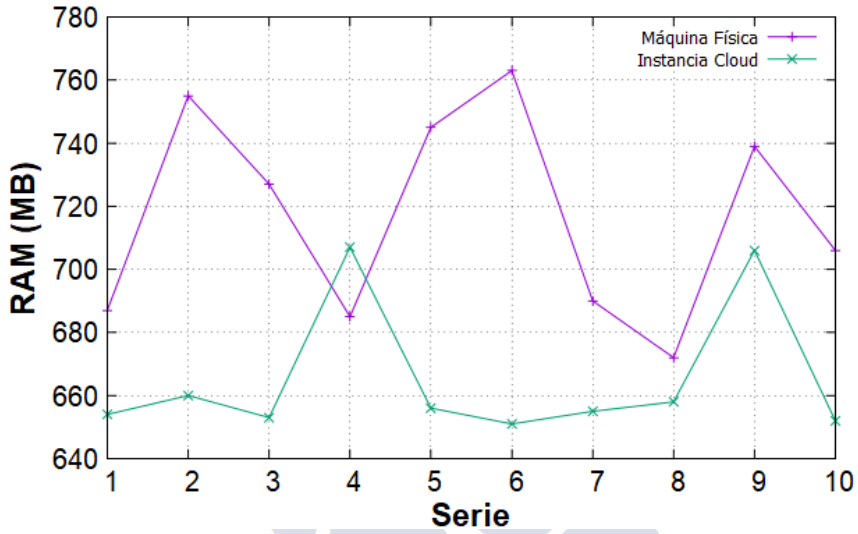

Figura 4.16: Utilización de memoria RAM en las diferentes ejecuciones en cada infraestructura.

vieron un comportamiento similar en todas las ejecuciones, excepto en la lectura que presentó una desviación estándar de 1317.69 segundos.

De igual manera hubo dispersión en las medidas de re-lectura con una desviación estándar de 1622.45 segundos y en la lectura aleatoria con 669.11 segundos respectivamente. Esta dispersión en las medidas se origina por la tecnología de plato rotatorio y cabezales que compone al arreglo de discos que provee el NAS, donde se realizan las pruebas del *cloud*. Adicionalmente al ser un sistema virtualizado es afectado por el controlador de disco que provee el hipervisor.

- **IOzone en máquina física SSD:** En esta serie de mediciones se obtuvo los mejores resultados. La velocidad promedio en escritura reportó 150630 Kbytes/s, en re-escritura presentó 143570 Kbytes/s, lectura 202387 Kbytes/s, re-lectura 203521 Kbytes/s, lectura aleatoria 164737 Kbytes/s y finalmente en escritura aleatoria 54095 Kbytes/s. En esta serie de mediciones se obtuvo una dispersión muy baja, siendo la de mayor proporción el proceso de escritura que presentó una desviación estándar de 1263.40 Kbytes/s. Las otras mediciones no presentaron dispersión de importancia.

- **IOzone en máquina física sobre NFS:** En la ejecución de IOzone en esta plataforma, en

cuanto a su velocidad promedio en escritura reportó 10467 Kbytes/s, en re-escritura presentó 10614 Kbytes/s, en lectura 11331 Kbytes/s, en re-lectura 11330 Kbytes/s, en lectura aleatoria 6648 Kbytes/s y finalmente en escritura aleatoria 10529 Kbytes/s. En general, las mediciones mantuvieron un comportamiento similar en todas las ejecuciones sin presentar una dispersión de importancia.

- **Comparación de resultados IOzone:** Los mejores resultados los presentó la máquina física, la cual contaba con una unidad de almacenamiento secundario del tipo SSD. En cambio, la instancia física con almacenamiento NFS presentó el peor rendimiento; pero muy similar a la instancia *cloud*, siendo esta mejor en re-lectura en un 14 % y lectura aleatoria en un 39 %. El sistema de almacenamiento en red (NAS) fue accedido sobre una red Ethernet Gigabit con el fin de limitar el efecto de la red, pero la misma no fue causa de los bajos resultados; sino que la tecnología que utilizan los discos que lo conforma es la clásica de platos rotatorios y cabezales de lectura. El sistema de archivo que aprovechó el almacenamiento SSD, presentó su mayor eficiencia en las tareas de lectura y re-lectura siendo hasta diez y siete veces superior al NFS.

Conclusiones

El primer tipo de *script* desarrollado, denominado muestreo, se encarga de crear una tabla que contiene un registro único con los datos del docente y la frecuencia de las respuestas asociadas a él. El segundo tipo de *script*, llamado patrón, calcula el número de aciertos de las palabras y raíces semánticas encontradas en la muestra específica por docente. Se repitió la ejecución de los *scripts* en diferentes ocasiones, teniendo el cuidado de borrar los resultados y reiniciar las plataformas al finalizar cada ciclo.

Al ejecutar las pruebas en ambas plataformas, el ordenador físico mostró medidas superiores respecto a la instancia *cloud*, ya que esta última emplea un sistema de almacenamiento basado en disco de platos rotatorios siendo menos eficiente. La tecnología SSD provee una interesante alternativa para ser considerada como sistema de almacenamiento en una infraestructura *cloud*, siendo la única limitante de esto la relación de precio y capacidad. En cambio, al evaluar el tiempo de ejecución de los *scripts* dentro del gestor de base de datos, los resultados se ven sacrificados en un tercio en la instancia *cloud* frente a la alternativa física.

Con el fin de identificar que ocasiona la pérdida de eficiencia, se ha ejecutado IOzone bajo tres condiciones diferentes como ser sobre la plataforma *cloud*, sobre la máquina física empleando como almacenamiento el SSD, y sobre la máquina física empleando como alma-

cenamiento un directorio exportado desde el sistema de almacenamiento primario del *cloud* exportado por medio de NFS. Los resultados han mostrado que el cuello de botella de la infraestructura *cloud* proviene del sistema de almacenamiento primario empleado en el *cloud*, el sistema NAS en RAID 1, que es el que provee los discos raíz de las máquinas virtuales. El cambio del sistema de almacenamiento a uno de estado sólido puede aportar mucho beneficio en el tiempo de ejecución de aplicaciones en una instancia *cloud* o física, este deberá ser un punto de estudio a futuro para conocer las ventajas y poderlas cuantificar.

4.4.2. Simulaciones numéricas en la nube

Este caso de uso se estudia la influencia del número de máquinas virtuales empleadas por anfitrión y las operaciones de entrada y salida (E/S) que se producen, específicamente en cuanto al desempeño de dos pruebas de referencia, la *primera* es una aplicación científica basada en una simulación de dispositivos semiconductores y la *segunda* es el *benchmark* Linpack. Ambos se ejecutaron con y sin el soporte al *hyperthreading* en el anfitrión físico. Las máquinas virtuales son administradas utilizando el hipervisor KVM incluido en la plataforma de gestión de Apache CloudStack [81].

Infraestructura

En la actualidad se ha tenido un importante progreso en la tecnología de virtualización, que ha permitido el desarrollo de varios hipervisores de código abierto. Específicamente, KVM [87] es un hipervisor potente pero relativamente simple que ha encontrado su lugar al integrarse en el kernel de Linux desde la versión 2.6.20, dando las capacidades de virtualización nativa. Utiliza la virtualización completa asistida por hardware y no requiere sistemas operativos invitados modificados. KVM puede ejecutarse en cualquier plataforma Linux, siempre que se implemente en un microprocesador compatible con la virtualización asistida por hardware [41]. Para KVM, cada máquina virtual (MV) se considera un proceso regular de Linux, el cual tiene tres modos de ejecución: usuario, núcleo e invitado. El modelo de usuario se usa como predeterminado para las aplicaciones, el modo kernel se usa cuando una aplicación necesita algún servicio del kernel (por ejemplo, escribir en un disco) y el modo invitado se usa por procesos que se ejecutan desde la máquina virtual [33].

Para proporcionar una abstracción básica de la CPU subyacente a las máquinas virtuales, se requiere el uso de una plataforma de administración de la nube, para ello se utilizó Apache CloudStack [84]. Este gestor ha sido diseñado para implementar y gestionar grandes redes de

máquinas virtuales, como una plataforma de computación en la nube escalable y altamente disponible.

En este caso de uso se ha tomado los recursos que son parte del proyecto Formiga Cloud [28], que propone la creación de una infraestructura en la nube basada en el modelo IaaS para aprovechar los recursos computacionales disponibles en los laboratorios informáticos de la una universidad. Por ejemplo, la Universidad de Santiago de Compostela (USC) [92] tiene alrededor de 1800 ordenadores personales disponibles en sus diferentes aulas de informática que están inactivos fuera del horario lectivo; este recurso puede ser utilizado conforme a los objetivos del proyecto mencionado. Por lo tanto, Formiga Cloud pretende utilizar estos recursos no sólo para actividades de enseñanza sino también para la informática científica.

Para evaluar el rendimiento de la infraestructura, se ha seleccionado una aplicación de informática científica, específicamente un simulador de dispositivos semiconductores 1D-SIM [27], el cual se ha desarrollado en la Universidad Santiago de Compostela y de la misma manera se ha hecho uso de una referencia sintética como ser la biblioteca numérica de Linpack [13] [19]. Usando estas aplicaciones, se ha evaluado el impacto en el rendimiento de varios elementos clave en la computación en la nube como ser la cantidad de máquinas virtuales por anfitrión, la influencia del *hyperthreading* y la E/S del disco duro. Ambas amplicaciones resuelven sistemas lineales de ecuaciones conformadas por matrices dispersas para 1D-SIM y densas para Linpack.

1D-SIM es un simulador de tipo deriva-difusión (*drift-diffusion*, (D-D)) unidimensional para dispositivos semiconductores. En la figura 4.17 se muestra un diagrama de flujo del proceso de simulación. Las ecuaciones básicas que se deben resolver en el modelo de deriva-difusión son la ecuación de Poisson y las ecuaciones de continuidad de electrón-hueco. [77]. El método de elementos finitos se aplica para discretizar el dispositivo. Para un caso discretizado en N nodos, se debe resolver un conjunto de ecuaciones no lineales acopladas de dimensión 3N. Las incógnitas del problema son el potencial electrostático (ψ) y los potenciales cuasi-Fermi para los electrones (ϕ_n) y los huecos (ϕ_p). El método de Gummel se aplica [75] para desacoplar estas ecuaciones, que luego se linealizan utilizando el método de Newton [7]. El sistema lineal resultante de ecuaciones se resuelve utilizando el método iterativo BiCGSTAB preacondicionado con una factorización LU incompleta, que depende tanto de un cierto relleno como de un umbral numérico. En este caso, se ha utilizado un transistor bipolar de heterounión como prueba de referencia, incluidos los efectos de emisión termoiónica y efecto túnel.

La prueba de referencia Intel Linpack resuelve un sistema denso de ecuaciones lineales, mide la cantidad de tiempo que toma para factorizar y resolver el sistema, luego convierte ese tiempo en una tasa de rendimiento y finalmente, evalúa la precisión de los resultados. Hay que destacar que esta es una de las pruebas que se utiliza para crear la lista TOP500 [53] que da una clasificación de las súpercomputadoras más potentes del mundo.

La máquina utilizada como anfitrión tiene un procesador Intel Core i7-2600@3.40 GHz con cuatro núcleos y 8 GB de RAM. La red de interconexión es una Ethernet GigaBit. Los sistemas operativos del anfitrión y de las máquinas virtuales (MV) son CentOS 64 bit versión 6.1 y Debian 6.0.4 64 bit, respectivamente. Para realizar las pruebas, se desplegaron varias máquinas virtuales en el mismo anfitrión utilizando el hipervisor KVM (qemu-kvm-0.12.1.2). La API de virtualización [45] es libvirtd (versión 0.9.4). Cada una de las máquinas virtuales tiene un núcleo y 1 GB de RAM disponibles. La CPU virtual es QEMU Virtual CPU (versión cpu64-rhel6). El sistema de archivos de red utilizado es NFS v3.

Resultados

Inicialmente, hemos modificado el simulador 1D-SIM para eliminar las operaciones de escritura en el disco duro. En el anfitrión físico con el *hyperthreading* deshabilitado, ejecutamos el código y se necesitaron 64.7 segundos para obtener la solución. En las mismas circunstancias, una máquina virtual implementada con el agente de Apache CloudStack requiere sólo un 2.5 % más de tiempo para alcanzar la misma solución. Para evaluar el impacto de la cantidad de máquinas virtuales implementadas por el anfitrión en el rendimiento del simulador unidimensional, consideramos *n* máquinas virtuales con n = 2, 4, 6, 8. Las primeras MV n-1 ejecutan el simulador una cierta cantidad de veces y la n-ésima ejecuta el código y estima los tiempos de simulación.

La figura 4.18 muestra los tiempos de simulación frente a la cantidad de máquinas virtuales desplegadas por anfitrión cuando no se generan operaciones de E/S en el disco local. La influencia del *hyperthreading* también se muestra en la figura. Hay un aumento en el tiempo de simulación cuando se incrementa el número de MV por anfitrión. Cuando se habilita el *hyperthreading*, este incremento es inferior al 6 %, al ejecutarse cuatro o menos MV y alcanza el 13 % y el 23 %, respectivamente, para 6 y 8 VM. Sin embargo, si el *hyperthreading* está desactivado, la pérdida en la eficiencia es más notable. En este caso, cuando se implementan dos o menos máquinas virtuales, el aumento en el tiempo de simulación es inferior al 5 %, pero alcanza el 10 % cuando se utilizan cuatro máquinas virtuales. Para un mayor número de

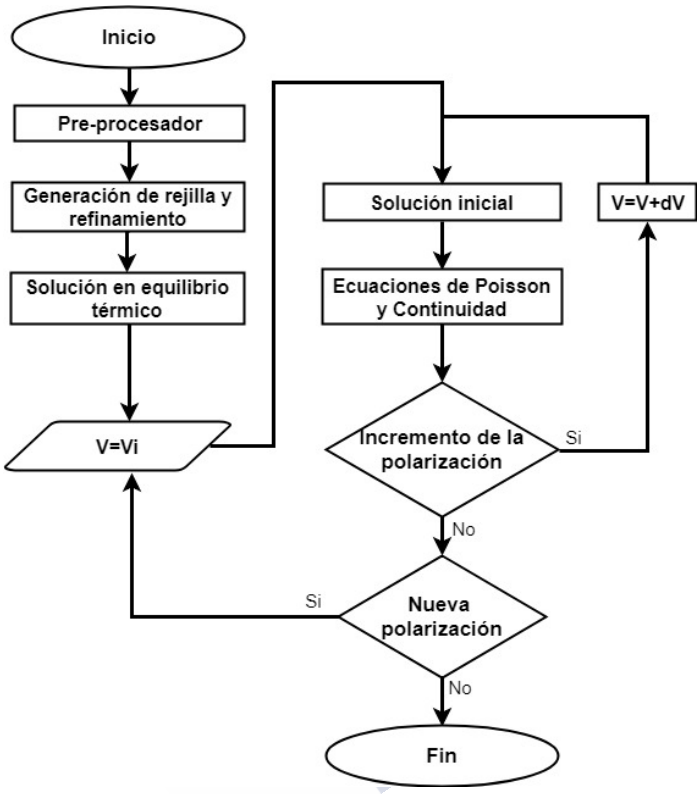

Figura 4.17: Diagrama de flujo de la simulación deriva-difusión (*drift-diffusion*) del dispositivo 1D.

MV por anfitrión, hay un deterioro importante en el tiempo requerido, observando en seis y ocho máquinas virtuales, los tiempos de simulación incrementan entre 65 % y 120 % respectivamente, al compararse con los obtenidos con una sola MV.

Se estudió la influencia de las operaciones de E/S en el rendimiento de la simulación 1D-SIM. Para ello, durante cada iteración del proceso de simulación, los valores de las diferentes variables que se calculan durante la simulación (concentraciones de electrón-hueco y potencial electrostático) se escriben en un archivo para cada nodo de la malla. Esta es una prueba más realista y se realiza de la misma manera que la anterior, pero incluye una MV adicional en otro anfitrión que exporta un sistema de archivos NFS que se comparte entre las otras MV para almacenar los resultados de la simulación. Esta MV adicional tiene 4 núcleos y se implementa en un procesador Intel Core i7-2600 @ 3.4 GHz.

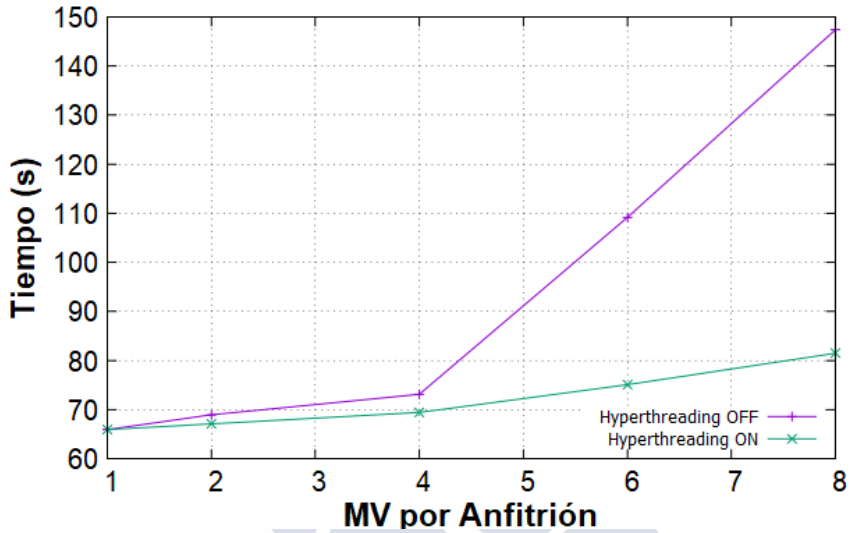

Figura 4.18: Tiempo de simulación deriva-difusión del dispositivo 1D versus el número de máquinas virtuales por anfitrión cuando no se generan operaciones de E/S. Se comparan los resultados con *hyperthreading* activado o desactivado.

La figura 4.19 muestra, para este caso que los tiempos de simulación frente al número de máquinas virtuales implementadas por anfitrión. El impacto del *hyperthreading* también se muestra en la figura. Cuando se incrementa el número de procesos que usan el sistema de almacenamiento NFS, se generan más interrupciones en la aplicación debido a la cantidad de procesos que compiten por obtener acceso al sistema de archivos. Por lo tanto, hay un aumento considerable en el tiempo de simulación cuando se incrementa el número de máquinas virtuales. Cuando el *hyperthreading* está activado, disminuye considerablemente el tiempo requerido para la simulación, siendo más evidente cuando se implementan dos o más máquinas virtuales. Cuando el *hyperthreading* está deshabilitado y se usan 8 VM en el mismo anfitrión, observamos el mayor aumento en el tiempo de simulación de todas las pruebas, duplicando el tiempo en comparación con el valor obtenido cuando se usa el soporte *hyperthreading* en un caso similar con el mismo número de MV.

Hemos realizado un estudio similar utilizando el *benchmark* Intel Linpack. En esta prueba, se utilizó un tamaño de problema de 5000, con valores de alineación de 4 Kbytes. Linpack proporciona el rendimiento promedio en GFlops por todas las ejecuciones Linpack (cinco en nuestro caso) para una sola prueba. Estos resultados se muestran en la figura 4.20, donde se

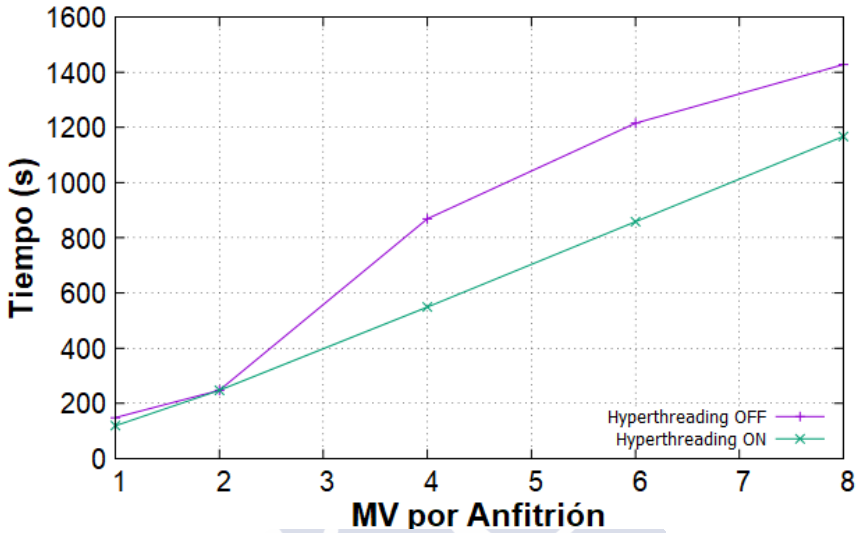

Figura 4.19: Tiempo de simulación de *drift-diffusion* del dispositivo 1D versus el número de máquinas virtuales por anfitrión cuando existen operaciones de E/S. Se comparan los resultados con *hyperthreading* activado o desactivado.

puede observar las operaciones de coma flotante frente al número de MV por anfitrión. Con el fin de hacer una comparación adecuada entre los resultados obtenidos al usar Linpack y el simulador 1D-SIM, en la figura 4.21 se muestra el tiempo de ejecución frente al número de máquinas virtuales para la prueba Linpack. En estas figuras, también se muestra el impacto del *hyperthreading*. Cuando sólo se implementa una MV por anfitrión, Linpack requiere alrededor de 43 segundos para resolver la prueba y entrega un valor de 11.1 GFlops. Cuando el *hyperthreading* está desactivado, los tiempos de simulación siguen exactamente la misma tendencia observada en la figura 4.18. Cuando se compara con los resultados obtenidos con una MV, hay un aumento mínimo en el tiempo de simulación para 2 MV, este valor se eleva a un 9 % para 4 MV. Para más de 4 MV por anfitrión, notamos la misma degradación de las operaciones de coma flotante como lo mostró el simulador 1D-SIM, observando que en 6 y 8 MV los tiempos de simulación se incrementan en alrededor de un 63 % y 124 %, respectivamente al ser comparados con los obtenidos con una MV. Por lo tanto, aunque la disminución en los resultados de Linpack es inferior a 1 GFlop cuando se implementan 4 o menos MV, hay una caída entre 4 GFlops al utilizar 6 MV, y una disminución de 6 GFlops cuando se utilizan 8 MV. Cuando se habilita el *hyperthreading*, los resultados de la simulación de hasta

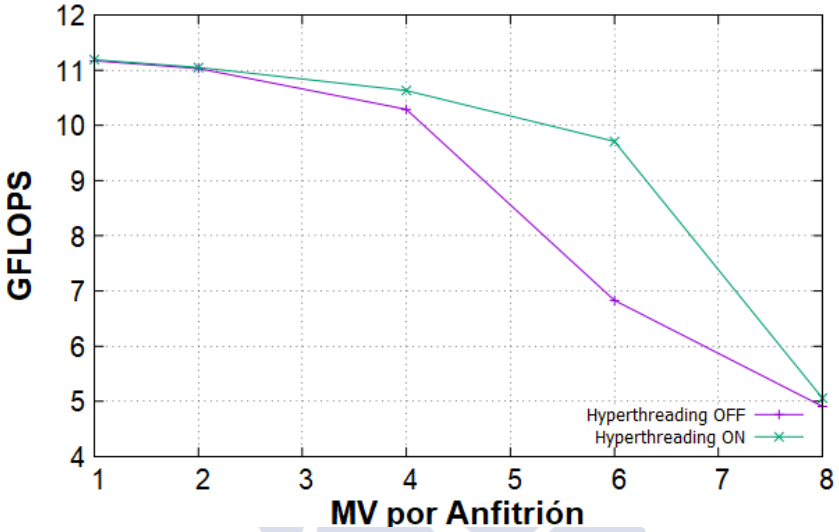

Figura 4.20: Operaciones de coma flotante por segundo versus la cantidad de máquinas virtuales por anfitrión cuando se utiliza Linpack. La influencia del *hyperthreading* se ha incluido en la gráfica.

6 máquinas virtuales también siguen el mismo comportamiento que en la figura 4.18, mostrando porcentajes similares del aumento en el tiempo de simulación cuando se incrementa el número de máquinas virtuales. Esta penalización en el tiempo de simulación es inferior al 5 % cuando se utilizan 4 o menos máquinas virtuales, y alcanza hasta el 13 % para 6 máquinas virtuales. Para todas estas configuraciones, la disminución en el rendimiento es menor a 1.5 GFlops. Sin embargo, cuando se implementan 8 MV en un anfitrión, la eficiencia disminuye muy rápidamente, lo que lleva a un aumento del 112 % en el tiempo de simulación y a una disminución de 6.1 GFlops en las operaciones de coma flotante.

Finalmente, es necesario tener en cuenta que cuando Linpack se ejecuta simultáneamente con procesos que hacen un uso intensivo del disco duro, no se espera ningún efecto importante en el resultado, ya que esta prueba de referencia no realiza operaciones de escritura en el disco duro. Por lo tanto, no presentamos un estudio de la influencia de las operaciones de E/S usando Linpack.

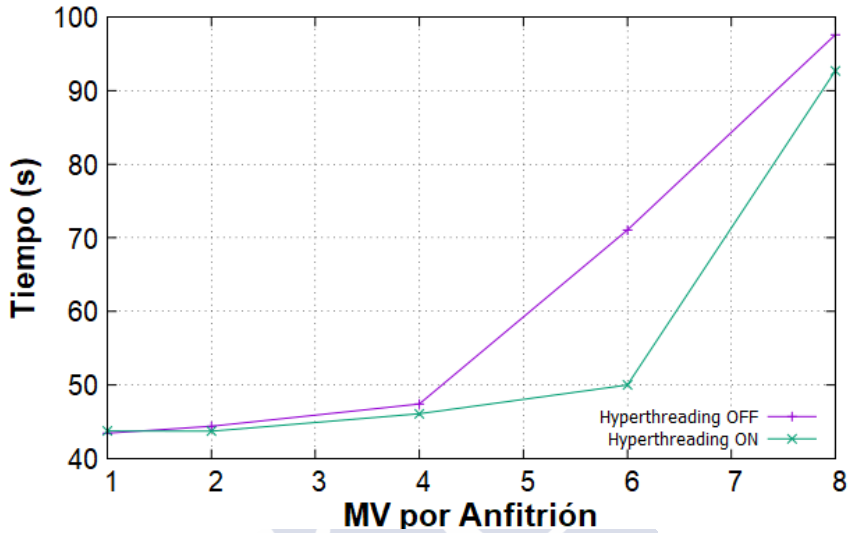

Figura 4.21: Tiempo de simulación versus el número de máquinas virtuales por anfitrión cuando se utiliza Linpack. La influencia del *hyperthreading* se ha incluido en la gráfica.

Conclusiones

Se ha estudiado la influencia del *hyperthreading*, la cantidad de máquinas virtuales empleadas por anfitrión y la E/S en dos pruebas de ejecución, una de ellos es un simulador de dispositivos semiconductores unidimensionales y el otro es el Linpack. Los resultados de ambas pruebas al ser comparadas muestran un comportamiento similar, con una importante degradación en la eficiencia cuando el *hyperthreading* está deshabilitado y con más de 4 máquinas virtuales implementadas en ejecución en un mismo anfitrión. El efecto de las operaciones de E/S es aún más notable en la simulación, con aumentos en el tiempo de simulación de más del 200 % cuando se utilizan 4 o más máquinas virtuales.

Los resultados obtenidos muestran una importante pérdida en el tiempo de ejecución cuando se implementan más de cuatro máquinas virtuales en un anfitrión físico de cuatro núcleos y estando el *hyperthreading* deshabilitado. El efecto de las operaciones de E/S en los resultados es aún más dramático, mostrando que el tiempo de ejecución cuando cuatro o más máquinas virtuales concurrentes en el mismo anfitrión físico puede ser más del doble que cuando se ejecuta una sola MV. De forma que es viable utilizar la nube para completar tareas de simulación numérica teniendo el cuidado de no generar sobre demanda de recursos, ya que el tiempo

requerido aumenta cuando se utilizan en las máquinas virtuales más núcleos físicos que los disponibles en el sistema anfitrión.

4.4.3. Redes de comunicación mejoradas para la enseñanza de MPI en la nube

En este caso de uso se pretende probar una infraestructura en la nube como una herramienta para aprender MPI, se evaluaron dos escenarios diferentes utilizando Apache CloudStack: el *primero*, se ejecutó un clúster virtual como un entorno de ejecución MPI y, el *segundo* un clúster virtual mejorado cuya latencia de comunicación MPI se mejoró.

En los procesos de enseñanza-aprendizaje, las nubes bajo el modelo de infraestructura como servicio (IaaS), podrían ser muy útiles debido a que los usuarios emplean recursos virtualizados. Los hipervisores proporcionan la capa de abstracción y el aislamiento necesario, como se haría en un modelo de caja de arena (*sandbox*). Como resultado, los entornos de aprendizaje virtualizados nos permiten usar la potencia computacional de los nodos informáticos sin la necesidad de cambiar la configuración física del anfitrión, reduciendo el esfuerzo requerido para la administración del sistema y aislando la configuración del anfitrión físico del entorno virtualizado del alumno. Además, permiten a los usuarios instalar diferentes sistemas operativos invitados y probar software que puede coexistir bajo los mismos anfitriones físicos sin comprometer o modificar su configuración [29].

El objetivo principal de este caso de uso es mostrar cómo se puede usar una infraestructura en la nube para enseñar MPI pero siguiendo la teoría del constructivismo que permite a los estudiantes construir su propio conocimiento a través de sus experiencia e interpretaciones personales. El papel del maestro es ayudar a comprender, mejorar la calidad del aprendizaje y fomentar la construcción del conocimiento [40][30].

Los estudiantes pueden implementar una infraestructura segura bajo Apache CloudStack para aprender todo el proceso, incluida la instalación de una MV, la configuración del sistema operativo, la instalación del entorno MPI y las herramientas de desarrollo relacionadas. Bajo esta infraestructura, los estudiantes deben ser capaces de llevar a cabo el análisis de rendimiento de sus aplicaciones, así mismo puedan ellos por si solos probar e implementar diferentes enfoques para lanzar una solución de MPI para un problema computacional dado.

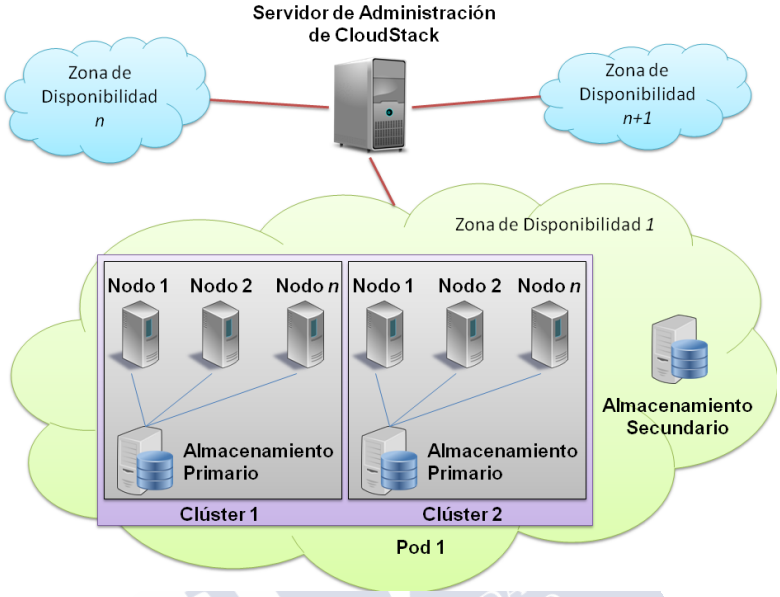

Figura 4.22: Arquitectura de operación de Apache CloudStack.

Infraestructura

La infraestructura en la nube utilizada se basa en Apache CloudStack 2.2.14 empleando hardware básico. Los nodos de computación (CN) cuentan con procesadores Intel Core i5 con 8 GB de RAM, tienen instalado Linux CentOS 6.3 64 bits como sistema operativo y KVM como el hipervisor administrado por apache CloudStack. El servidor NFS, que actúa como el almacenamiento primario de Apache CloudStack, es un ordenador Core 2 Duo 6600 a 2.40 GHz, con 4 GB de RAM, disco duro de 500 GB (7200 RPM SATA) y corriendo el sistema operativo CentOS 6.3 64 bits. La red de interconexión de esta nube es del tipo Ethernet Gigabit con una MTU de 1500 bytes. La operación se puede observar en la figura 4.22

Para fines de enseñanza, Apache CloudStack tiene características muy llamativas como la interfaz web que proporciona una administración completa de la nube. Algo interesante es la capacidad de definir máquinas virtuales altamente disponibles, estas se mantienen operativas por Apache CloudStack sin intervención del usuario o administrador en ningún punto. Otra opción es la capacidad de instalar un sistema operativo por medio de una imagen ISO estándar, la misma puede llevarse a cabo a través de la interfaz web sin la necesidad de uti-

lizar herramientas adicionales. Esta característica permitiría a los profesores o estudiantes poder preparar sus propias MV sin la necesidad de utilizar plantillas de predefinidas. Además, los profesores y los estudiantes pueden crear sus propias plantillas de máquinas virtuales de forma privada y se podrán hacer visibles sólo a los usuarios de una cuenta específica, también pueden hacerse públicas o si se gusta hacerse visibles para todos los usuarios de la nube.

Message Passing Interface (MPI) [60] es un protocolo de comunicaciones independiente del lenguaje donde se implemente, se ha convertido en un estándar de facto para la comunicación entre procesos, usando el modelo de paso de mensajes. Los clústeres de súper ordenadores de memoria distribuida a menudo ofrecen el uso de MPI a sus usuarios, entre otras infraestructuras computacionales.

Con el fin de probar la infraestructura en la nube de Apache CloudStack como una herramienta de enseñanza y aprendizaje para el paradigma de programación MPI, se prepararon e implementaron dos escenarios. El *primero* es un escenario de configuración de prueba básico desplegado como prueba de concepto y el *segundo* constituyó una mejora con respecto al primero, con el propósito de obtener un mejor rendimiento. El propósito de estos escenarios es preparar a los estudiantes para resolver problemas en entornos complejos. La tecnología de la nube ayudará a lograr este objetivo y por medio de la interfaz web de Apache CloudStack, el docente puede ayudar a los alumnos con facilidad y revisar su progreso, centrando la atención en los temas más relevantes. Apache CloudStack proporciona flexibilidad al proceso de enseñanza aprendizaje, incrementando la independencia de tiempo y espacio. Además tiene la ventaja de que los estudiantes pueden realizar sus actividades sin la necesidad de estar presentes en el laboratorio de computación.

- **Escenario básico:** Para probar la infraestructura en la nube con Apache CloudStack como una herramienta útil para el aprendizaje de programación con MPI, fue necesario implementar un clúster virtual, este se puede definir como un clúster compuesto por máquinas virtuales (MV) donde se ejecutan las aplicaciones paralelas. En informática, un clúster es un grupo de computadoras interconectadas que trabajan en conjunto y que se pueden ver como un sistema único. Típicamente, como se muestra en la figura 4.23, dos tipos de componentes de acuerdo a su función son parte de un clúster, uno de ellos es el maestro y el otro son los nodos. El maestro es la computadora donde se conectan los usuarios. Los nodos están pensados como recursos computacionales que se emplearán para ejecutar aplicaciones de usuario. Por lo general, los usuarios no tienen acceso directo a los nodos, por lo que no pueden iniciar sesión en ellos. Los usuarios iniciarán

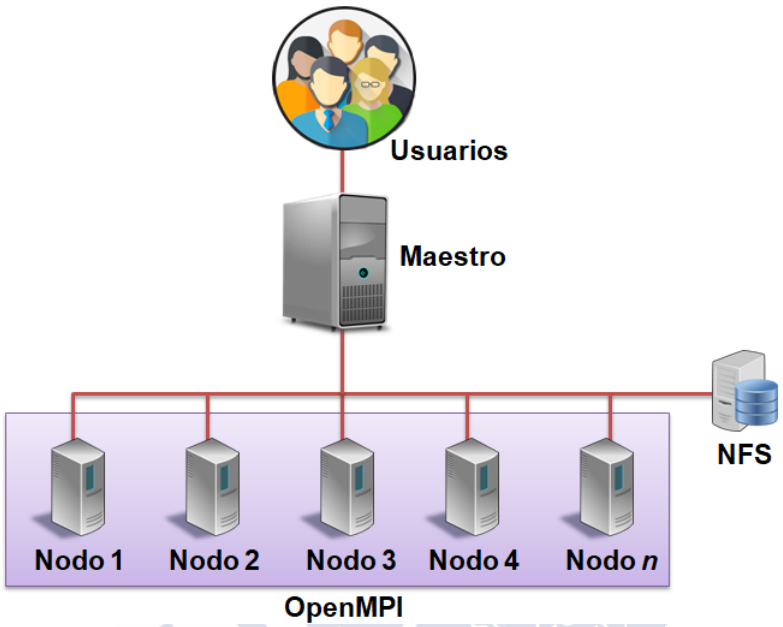

Figura 4.23: Esquema de operación de un clúster para MPI.

las aplicaciones desde el maestro el cual los ejecutará en los nodos. Cada computadora que compone el clúster ejecuta su propia instancia de un sistema operativo, así mismo pueden existir recursos compartidos accesibles como ser servidores de almacenamiento NFS, entre otros.

El clúster virtual que los estudiantes deben implementar en Apache CloudStack está compuesto por una máquina virtual configurada como maestro y dos máquinas virtuales configuradas como nodos. El maestro desplegado es una MV que emplea un disco duro de 10 GB, un núcleo de CPU, 1 GB de memoria RAM y ejecuta como base el sistema operativo CentOS 6.3. También sirve el directorio inicial a los nodos que componen el clúster virtual empleando el sistema de archivos de red (NFS) como un protocolo de sistema de archivos distribuido. Los nodos implementados son máquinas virtuales con un disco duro de 10 GB, cada uno tiene un CPU central con 1 GB de RAM y ejecutan el sistema operativo CentOS 6.3. Los nodos montan los directorios compartidos que provee el maestro. Las máquinas virtuales utilizadas comparten una interconexión de red Ethernet Gigabit.

El clúster virtual utiliza OpenMPI 1.6 [88] como implementación de MPI. OpenMPI es de código abierto y soporta el protocolo MPI-2 [60], el cual fue desarrollado por un consorcio conformado por investigadores, académicos y socios industriales. Las características más notables de MPI-2, son la seguridad, la concurrencia de subprocesos, generación de procesos dinámicos, tolerancia a errores de red y de proceso, heterogeneidad en cuanto a redes, instrumentación en tiempo de ejecución, entre otras características.

- **Escenario mejorado:** En las redes Ethernet que utilizan TCP existe una alta latencia de las comunicaciones MPI, es por ello que el rendimiento obtenido es limitado. Sin embargo, esta latencia se puede reducir utilizando Open-MX [36]. Esta es una implementación de alto rendimiento de la pila de transmisión de mensajes Myrinet Express [14] sobre redes Ethernet genéricas. Tiene las capacidades del *firmware* MX que se ejecuta en Myri-10G NIC como un controlador en el kernel de Linux. Para aplicaciones heredadas, una biblioteca de espacio de usuario expone la interfaz MX a las aplicaciones. Open-MX admite Linux en cualquier arquitectura y funciona al menos en kernels Linux igual o superior a la versión 2.6.15, al igual ocurre con los dispositivos Ethernet que están soportados en él. Para funcionar correctamente todos los equipos que deseen aprovechar esta librería deben encontrarse en el mismo segmento de red e interconectados mediante un conmutador o *switch*. Open-MX es compatible con el tráfico IP y puede coexistir perfectamente en la misma red. Para configurar Open-MX para ser utilizado por OpenMPI, es necesario tener en cuenta que este último debe compilarse e instalarse con la opción de habilitar la compatibilidad con Open-MX. El propósito de este escenario es hacer que los estudiantes presten atención a la importancia del análisis del rendimiento de los ordenadores que ejecuten aplicaciones que requieran alto desempeño. El clúster virtual empleado en este caso tiene la misma configuración que la descrita en el escenario básico, pero el cambio es contar con Open-MX para mantener las comunicaciones MPI, evitando la sobrecarga de TCP para comunicar procesos.

Pruebas de referencia

Para probar los escenarios descritos anteriormente, se ejecutaron tres tipos de aplicaciones, la *primera* es el Intel MPI Benchmarks [37], el *segundo* es el ejemplo HEAT MPI [10] y la *tercera* es la aplicación el GADGET2 [83].

La *primera* aplicación para prueba es el Intel MPI Benchmarks 3.2.3 (IMB) que proporciona un conjunto conciso de kernels de referencia MPI elementales, tiene varios parámetros como ser la longitud de los mensajes o la selección de comunicadores para ejecutar un punto de referencia específico. IMB también proporciona una configuración estándar, pero el usuario puede configurar a medida. Si se utiliza el modo estándar, todos los parámetros mencionados anteriormente son fijos y no deben modificarse. El modo seleccionado para probar la infraestructura virtual es el estándar.

La versión utilizada de IMB, contiene diferentes clases de pruebas de referencia como ser la transferencia simple o *single transfer*, la transferencia paralela o *parallel transfer* y la transferencia colectiva o *collective transfer*. La simple son dos pruebas, una de ellas la *PingPong* y la otra es *PingPing*. La paralela se denominan *Exchange* y *Sendrecv*. La pruebas colectivas son *Bcast, Allgather, Allgatherv, Alltoall, Alltoallv, Reduce, Reduce_scatter, Allreduce* y la *Barrier*.

Las pruebas de transferencia simple, se describen a continuación:

- ***PingPong*:** Se usa para medir el inicio y la producción de un sólo mensaje enviado entre dos procesos.

- ***PingPing*:** Mide el inicio y el rendimiento de los mensajes individuales con la diferencia de que los mensajes que se aproximan obstruyen a los mensajes enviados.

Las pruebas de transferencia paralela, se describen a continuación:

- ***MPI_Exchange*:** Es un patrón de comunicación que suele ocurrir en los algoritmos de división de cuadrículas (intercambios de límites). El grupo de procesos es similar a una cadena periódica y cada proceso intercambia datos con el vecino izquierdo y derecho de la cadena.

- ***MPI_Sendrecv*:** Cada proceso se envía al proceso de la derecha y se recibe desde su vecino izquierdo en una especie de cadena. El intercambio es un patrón de comunicación que se usa a menudo en los algoritmos de división de cuadrículas en el que el grupo de procesos se ve como una cadena periódica y cada proceso intercambia datos con los vecinos izquierdo y derecho de la cadena.

Las pruebas colectivas y sus funciones, se describen a continuación:

- ***MPI_reduce*:** Su objetivo es disminuir un vector de elementos flotantes de longitud *L* empleando la operación *MPI_SUM*.

- ***Reduce_scatter*:** Decrementa un vector de elementos con longitud flotantes *L* que emplea la operación *MPI_SUM*. En la etapa de dispersión, los elementos *L* se dividen lo más uniformemente posible.

- ***MPI_Allreduce*:** Disminuye un vector de elementos con longitud flotantes *L* empleando la operación *MPI_SUM*.

- ***MPI_Allgather*:** Es una función en la que cada proceso envía *r* bytes y recibe una cantidad de bytes que es igual a *r* multiplicada por la cantidad de procesos.

- ***MPI_Allgatherv*:** Busca mostrar si MPI produce sobrecarga al crear una situación más compleja en comparación con *MPI_Allgather*.

- ***MPI_Alltoall*:** Hace que cada proceso ingrese un número de bytes igual a *r* multiplicado por el número de procesos (*r*, en cada uno de ellos) y recibe un número de bytes igual a *r* multiplicado por el número de procesos (*r* de cada proceso).

- ***MPI_Alltoallv*:** Esta función incrementa aún más la carga de bytes que con la función *MPI_Alltoall*.

- ***MPI_Bcast*:** Se busca que el proceso raíz transmita *r* bytes a todos. En este punto de referencia, el proceso raíz de la operación se cambia cíclicamente.

La *segunda* aplicación para prueba de referencia del clúster virtual se ha hecho empleando HEAT MPI de John Burkardt [10], que es una implementación en lenguaje C de la ecuación de calor dependiente de tiempo 1D que emplea una forma de descomposición de dominio.

La *tercera* aplicación para prueba se ha hecho empleando el software GADGET2 [83]. Este programa es un código de libre disposición para simulaciones cosmológicas de N-cuerpos/SPH en computadoras paralelas con memoria distribuida. Utiliza un modelo de comunicación explícito implementado con la interfaz de comunicación MPI estandarizada. GADGET2 calcula las fuerzas gravitatorias con un algoritmo de árbol jerárquico y representa los fluidos por medio de la hidrodinámica de partículas suavizadas (SPH). GADGET2 puede usarse para estudios de sistemas aislados, o en simulaciones que incluyen la expansión cosmológica del espacio, con o sin condiciones de contorno periódicas en ambos casos. En este tipo de

simulaciones, GADGET2 sigue la evolución de un sistema de N-cuerpos sin colisiones autogravitantes y permite incluir opcionalmente dinámicas de gas.

Resultados

Se inicia mostrando los resultados obtenidos para el protocolo TCP y Open-MX medidos con el Intel MPI Benchmarks. Los resultados de latencia obtenidos para los puntos de referencia de transferencia simple *PingPong* y *PingPing* se pueden observar en la figura 4.24. Nótese que en todas las figuras presentadas, el eje X está en escala logarítmica. Para el punto de referencia de la prueba *PingPong*, la latencia se reduce en alrededor de un 30 % cuando se utiliza Open-MX para la comunicación en comparación con TCP, como se muestra en las marcas cuadradas de la figura. El índice de referencia *PingPing* también reduce su latencia, incluso en una cantidad mayor (alrededor de un 36 %) que *PingPong*, cuando se utiliza Open-MX.

Los resultados obtenidos tanto para *Exchange* como para los puntos de referencia de transferencia paralelos de *Sendrecv* se muestran en la 4.25. Para la prueba de rendimiento de *Exchange*, la latencia se reduce en aproximadamente un 45 % cuando se utiliza Open-MX en comparación con TCP, como se muestra en las marcas cuadradas de la figura. Respecto a *Sendrecv* también reduce su latencia en un 35 % cuando Open-MX se utiliza para las comunicaciones.

Los colectivos *Allgather* y *Allgatherv* se muestran en la figura 4.26. Para el punto de referencia *Allgather*, la latencia se reduce alrededor del 33 % cuando se usa Open-MX en comparación con TCP, como se muestra en las marcas cuadradas de la figura. El índice de referencia *Allgatherv* también obtiene su latencia reducida en torno al 31 %.

En las pruebas del conjunto *Alltoall* y *Alltoallv* se indican en la figura 4.27. Para *Alltoall*, la latencia se redujo en alrededor del 35 % cuando se utiliza Open-MX en comparación con TCP, como se muestra en las marcas cuadradas de la figura. El índice de referencia de *Alltoallv* también reduce su latencia en aproximadamente un 35 % de forma similar a la descrita anteriormente.

Los resultados obtenidos para los puntos de referencia colectivos *Reduce* y *Reduce_scatter* se representan en la figura 4.28. Para la referencia de *Reduce*, la latencia disminuye alrededor del 35 % como se muestra en las marcas cuadradas de la figura. Para *Reduce_scatter* también fue menor, bajando su latencia en un 32 %, en ambos casos cuando Open-MX se utiliza para comunicaciones en sustitución de TCP.

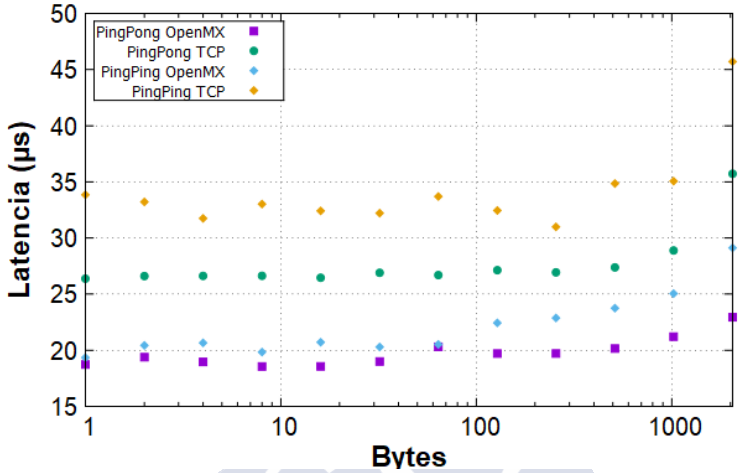

Figura 4.24: Latencia frente a bytes en la prueba de transferencia simple de red (PinPong) y (PingPing).

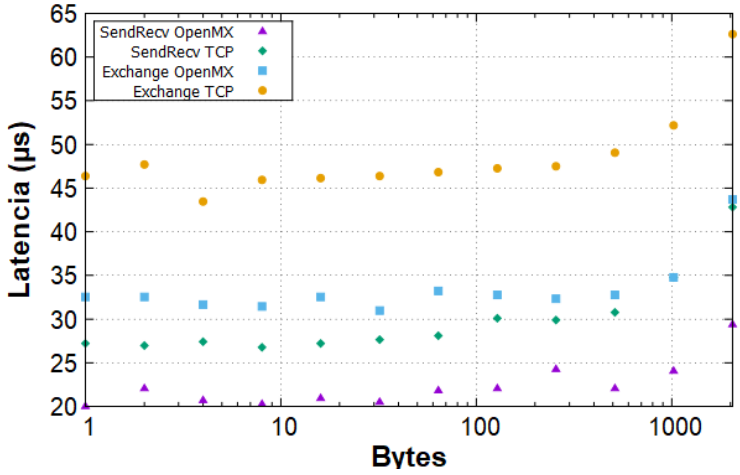

Figura 4.25: Latencia frente a bytes en la prueba de transferencia paralela de red (Exchange) y (Sendrecv).

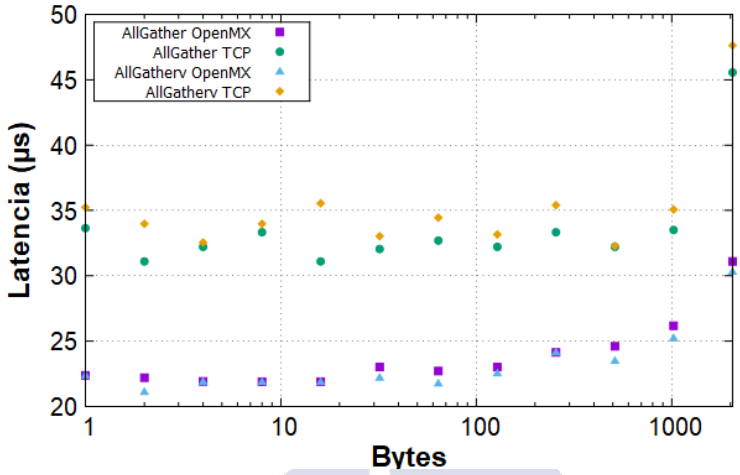

Figura 4.26: Latencia frente a bytes en la prueba detransferencia colectiva (Allgather) y (Allgatherv).

Para las pruebas del ejemplo HEAT MPI se mide el tiempo computacional transcurrido. Se usó el protocolo TCP para comunicar procesos MPI, empleando dos nodos virtuales desplegados en Apache CloudStack. El tiempo transcurrido fue de 22.708 milisegundos. Cuando se usa Open-MX, el tiempo transcurrido obtenido fue de 15.878 milisegundos (un 30 % mejor que TCP).

Finalmente, de forma contraria a lo obtenido en las pruebas previas, GADGET2 no obtiene un mejoría de operación cuando se utiliza Open-MX para comunicar procesos MPI. Esto muestra que eficiencia obtenidas dependen en gran medida del problema.

Conclusiones

Los procesos de enseñanza-aprendizaje están siendo apoyados por el desarrollo de nuevas tecnologías. En el pasado reciente, recursos como el correo electrónico, el chat, la audioconferencia, la videoconferencia y la conferencia vía web, se incorporaron como nuevas herramientas en el proceso lectivo. Actualmente, se está dando un paso más con el desarrollo y la popularización de las tecnologías en la nube, ya que están despertando un gran interés en los entornos educativos. Actualmente hay un desarrollo activo de plataformas en la nube con el lanzamiento de varias soluciones de código abierto para construir nubes privadas, públicas e

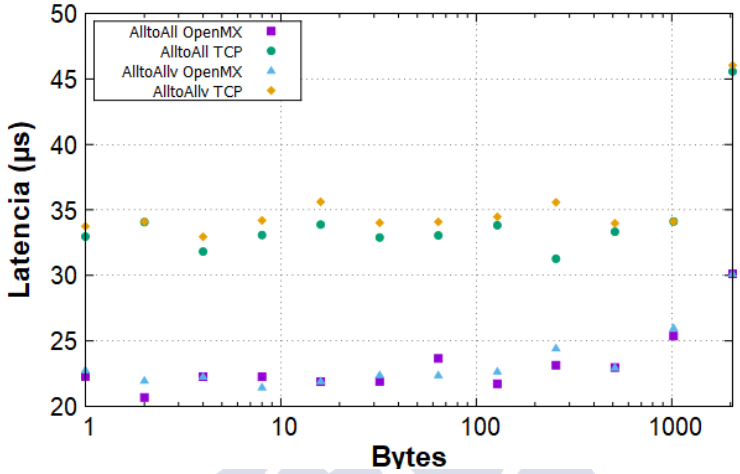

Figura 4.27: Latencia frente a bytes en la prueba de transferencia colectiva (AlltoAll) y (AlltoAllv).

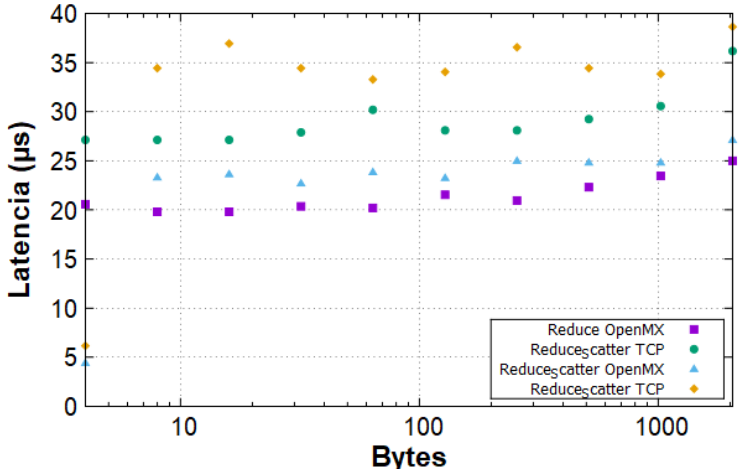

Figura 4.28: Latencia frente a bytes en la prueba de transferencia colectiva (Reduce) y (Reduce_Scatter).

híbridas, como ser los gestores OpenNebula, Eucalyptus, OpenStack y Apache CloudStack. Cada uno de ellos tiene características únicas.

En el modelo de servicio en la nube más básico como ser la infraestructura como servicio (IaaS), se puede proporcionar recursos computacionales como máquinas virtuales. En informática, este modelo ofrece a docentes y estudiantes la posibilidad de gestionar infraestructuras virtuales en las que se pueden realizar prácticas de administración de sistemas y lenguajes de programación sin comprometer la configuración de los nodos físicos.

Como hemos visto y entre otras ventajas, la enseñanza de MPI en la nube siguiendo la teoría del constructivismo se puede realizar con el propósito de preparar a los estudiantes para la resolución de problemas en entornos complejos. Por lo tanto, se implementaron dos escenarios diferentes ambos usando equipo e interconexión a red con *hardware* básico, bajo Apache CloudStack usando KVM como hipervisor. El *primero* constituye un clúster virtual para ejecutar aplicaciones MPI y el *segundo* es un clúster virtual mejorado que utiliza Open-MX para obtener una menor latencia de las comunicaciones de MPI para que los estudiantes tomen conciencia de la importancia de realizar el análisis de rendimiento de la computadora. Ambas infraestructuras MPI virtuales se probaron empleando los puntos de referencia Intel MPI, el ejemplo HEAT MPI y el software GADGET2.

Los resultados obtenidos muestran que la ejecución de aplicaciones MPI en la nube es adecuada y la latencia de las comunicaciones MPI se redujo en un 30 % con Open-MX en comparación con TCP. El tiempo transcurrido obtenido para el ejemplo de HEAT MPI también es alrededor de un 30 % mejor que usando TCP. Sin embargo, dependiendo de la implementación de las aplicaciones que utilizan MPI, hay casos en los que no se observan mejoras en la latencia, como sucedió con GADGET2. Estos casos de prueba demuestran que se pueden desarrollar experimentos en la nube por parte de profesores o estudiantes (estos actuando hasta como administración de sistemas), con diversidad de lenguajes de programación y desarrollando prácticas para la medición del rendimiento.

Conclusiones

En este documento se ha evaluado si una infraestructura virtualizada y en la nube puede ser de utilidad para reemplazar soluciones dedicadas de procesamiento, asi mismo se pretendió determinar que si esta a su vez puede emplearse en necesidades de cálculo intensivo que actualmente no pueden ser atendidas por falta de capacidad computacional. Los resultados de los diferentes casos de uso han sido favorables, haciendo viable emplear la nube y la virtualización para estas y otras necesidades.

La utilidad de la tecnología de virtualización parte fundamental del paradigma de la computación en la nube, ofrece al campo de la astronomía y astrofísica; específicamente en las aplicaciones de CASA y GADGET2 una opción para el procesamiento de datos observacionales y para el modelamiento teórico de la ocurrencia y evolución de fenómenos en cuerpos celestes. Los tiempos obtenidos son similares a los que se reportan en un ordenador físico y la diferencia existente no es significativa, ya que en la mejor de las condiciones de ejecución convierten en una alternativa de solución a los entornos de procesamiento virtualizados sobre infraestructuras flexibles frente a los ordenadores físicos dedicados.

En los casos de uso relacionados con multimedia, la nube es una opción, sobre todo para las actividades de renderización y transcodaje. La ventaja de utilizar la capacidad elástica y bajo demanda del paradigma permite realizar actividades que antes causaban atraso a los profesionales de esta área, portándolas a la nube ya que ahora pueden realizarse de forma desatendida. Al compararse las capacidades de un anfitrión físico con los virtuales se aprecia que en las mejores condiciones de ejecución la diferencia entre ambas infraestructuras no es significativa. El mayor valor que aporta virtualización es la de reutilizar los recursos computacionales físicos y la posibilidad de liberar estaciones de trabajo con mayor demanda, delegando carga de trabajo a una solución virtual.

La infraestructura virtualizada también es de utilidad si se enfoca a un entorno de forma-

ción de profesionales que trabajen con multimedios, permitiendo que ellos completen proyectos de renderización y edición sacando provecho de una solución virtualizada. Pudiendo de esta forma realizar múltiples tareas en menos tiempo y formando capacidades a través de la experimentación; de nuevo un entorno que reutilice equipos físicos facilita el acceso a multiples usuarios bajo demanda. Los resultados de las pruebas realizadas demuestran que la eficiencia de este tipo de entornos es similar a las alternativas físicas, volviendolas una opción en la optimización de recursos.

En procesos de transcodaje, la nube aporta una vez más beneficios, ya que los tiempos para completar estas tareas son similares al compararse con ordenadores físicos. En todo lo relacionado a la virtualización hay que tener el cuidado que el número de maquinas virtuales no supere las capacidades físicas en cuanto a número de núcleos del equipo anfitrión. De igual manera el *hyperthreading* aporta beneficios en la sobredemanda de las aplicaciones. Un aspecto a considerar es el almacenamiento, ya que puede resultar en un punto que afecte la eficiencia de las operaciones de multimedios, ya que son altamente demandantes en el acceso a disco debido a que los formatos son pesados, especialmente cuando se utilizan resoluciones de alta definición y formatos de vídeo sin comprimir.

Las capacidades de la nube para aplicaciones médicas fueron ventajosas, en el caso de uso descrito en la que se realizaron pruebas con Freesurfer, que es una aplicación que modela y mide diferentes elementos fisiológicos del cerebro humano enfocándose en el grosor de la corteza cerebral. Aquí se demostró que las infrastructuras virtualizadas son una opción, sobre todo porque la evaluación de las diferentes ejecuciones resultaron adecuadas al compararlas con un anfitrión físico, sobre todo si se logra identificar la configuración idónea de los nodos computacionales mediante pruebas de referencia. En conclusión, Freesurfer al procesar imágenes biomédicas provenientes de resonancia magnética para realizar la reconstrucción funcional y estructural del cerebro, puede ser portada con buenos resultados a la nube; aportando una alternativa de análisis a los investigadores médicos.

En lo relacionado a la reducción de texto no estructurado usando un gestor relacional de base de datos, la nube puede ser utilizada aportando sobre todo la capacidad de realizar múltiples operaciones en diferentes instancias de forma concurrente. Un elemento que puede deteriorar los buenos resultados obtenidos es el sistema de almacenamiento donde se ejecuten las tareas, ya que este tipo de aplicaciones son demandantes en acceso al sistema de archivos.

También se evaluó la simulación numérica con resultados alentadores. En este sentido, la simulación de nanodispositivos demostró tener tiempos de ejecución similares a la de un

ordenador físico. En este caso de uso fue de mucha importacia conocer el punto de equilobrio de los buenos resultados, ya que si se lanzaban un número de máquinas virtuales que sobrepasara la disponibilidad de núcleos físicos disponibles en el equipo anfitrión, ocasionaba un incremento del tiempo requerido de simulación. Esto se debe a la sobre demanda de recursos en el procesador. De la misma manera, esto afecta las operaciones de entrada y salidad del sistema de archivos, ocasionando cuello de botella. Desactivar el soporte del *hyperthreading* no aportó beneficios. De forma que la nube es viable para la simulación numérica, teniendo en consideración que se ve sacrificada por el uso intensivo de recursos.

La nube al centrarse en procesos de enseñanza aprendizaje como ser el caso de MPI permite un beneficio adicional como ser la conformación de una caja de arena para pruebas, situación que no sería posible en un anfitrión físico. En general, la nube es una alternativa viable para ser utilizada en aplicaciones relacionadas que requieran el uso de técnicas de programación concurrente, permitiendo exclusión mutua y sincronización de procesos. MPI es un estándard que permite el desarrollo de aplicaciones que aprovechan multiples procesadores, en el caso de uso expuesto se portaron las librerías de OpenMPI a la nube permite que los estudiantes puedan diseñar, desarrollar y analizar la ejecución de soluciones creadas en esta herramienta.

Las pruebas demostraron que la ejecución de aplicaciones basadas en MPI en la nube es adecuada, aunado con el uso de Open-MX se logró una disminución en la latencia de comunicaciones en un tercio al ser comaparada a que si este soporte no se empleara. Pero, se utilizó en el caso de uso la aplicación GADGET2, que no se vió beneficiada de una comunicación mejorada con Open-MX. De manera que la nube ofrece una alternativa flexible para poder hacer pruebas de referencia y evaluar el comportamiento de aplicaciones desarrolladas en diversidad de lenguajes que requieran concurrencia.

Una infraestructura computacional de alto rendimiento desarrollada con el paradigma de la nube es una opción para las actividades de reducción de datos y cálculo intensivo, sobre todo para las necesidades relacionadas a los casos de uso aqui evaluados. De la misma forma, contar con una metodología para evaluar y comparar las variables de referencia entre sistemas sería el indicador de la viabilidad de la migración de aplicaciones entre entornos de ejecución. En conclusión este tipo de abordaje para las necesidades de procesamiento es viable y útil.

Después de todas las pruebas realizadas, a nivel general al migrar aplicaciones a la nube se debe tener en cuenta los siguientes elementos:

• **Control de la sobredemanda:** La sobresuscripción de recursos es una condición a evitar,

ya que esto vuelve cualquier aplicación protada a la nube menos eficiente, resultando con tiempos de ejecución demasiados largos y poco práctivos.

- **Efecto del hyperthreading:** Es beneficioso siempre y cuando no se supere el número de núcleos físicos, de otra manera se sacrifica la eficiencia de la ejecución.

- **CPU Turbo:** Sólo es beneficioso en los anfitriones físicos, en las instancias en la nube tiene efecto negativo en los tiempos de ejecución.

- **Almacenamiento SSD:** Las aplicaciones que requieren acceso frecuente y amplio al sistema de archivos se debe de considerar utilizar la tecnología de estado sólido de almacenamiento para poder evitar cuellos de botella en la ejecución, sobre todo de las simulaciones.

- **Afinamiento:** Remover aquellas aplicaciones y servicios de los sistemas operativos utilizados evita comportamientos extraños en las ejecuciones, dentro del afinamiento necesario puede considerarse el uso de las soluciones de ahorro de energía, ubicacíón de particiones de almacenamiento, sincronización del reloj del sistema, desactivación de filtros de red, entre otros.

- **CPU Bound:** Al ejecutar este tipo de aplicaciones, se debe tener en cuenta los elementos que pueden interferir la capacidad de procesamiento de los microprocesadores, ya que la más mínima interrupción puede afectar los tiempos para completar tareas.

- **Optimzación:** El uso de controladores y librerias optimizadas para la infraestructura que estemos usando puede agregar mejores condiciones de ejecución que se traduce en ahorro de tiempo.

- **Pruebas de rendimiento:** El proceso de calibración y evaluación de las capacidades de las infraestructuras empleadas pueden aportar un parámetro de comparación entre anfitriones físicos y máquinas virtuales, así como los resultados de las aplicaciones que se ejecuten.

Trabajo futuro

En los países en vías de desaroolo, la inversión en investigación e innovación es usualmente un aspecto relegado y no es considerado como una prioridad, este hecho hace que estos

países se vuelvan importadores y consumidores de soluciones tecnológicas. Para mejorar esta situación los estados deben hacer inversiones graduales y constantes para cerrar las brechas tecnológicas resultantes y una solución de capacidades computacionales de alto rendimiento basado en el paradigma de la nube puede brindar recursos útiles para las tareas científicas. En este sentido encontrar un nicho de uso que una los campos académicos y de negocios debería ser un objeto de estudio posterior.

Una línea que se debería seguir en cuanto a capacidades computacionales de alto rendimiento es la incorporación de las mismas en actividades de minería de datos, sistemas de información geográfica, cambio climático y gobierno electrónico; esta última de inmensa ayuda debido a que fortalecería actividades económicas relacionadas a trámites burocráticos. Otro aspecto muy útil a desarrollar es la capacidad de contar con seguridad informática activa empleando análisis de vulnerabilidades, estudios de paquetes y telecomunicaciones, entre otros; sobre todo por el incremento de los ataques cibernéticos.

En los próximos años, la tendencia de transformar la industria a la nube, convirtiéndose esto en una nueva forma de revolución productiva que convertirá la economía a estar basada en el conocimiento, esto es una gran oportunidad ya que estará basada en tecnología elástica monetizada bajo demanda. Esto puede permitir nuevas formas de realizar negocios y facilitando la producción de soluciones de forma local por medio de la transferencia tecnológica. Esto representa opoertunidades económicas de interés para los estados que debe ser explotado mediante la figura de parque tecnológicos, poniendo a disposición capacidad computacional de alto rendimiento para que las empresas y los emprendedores tenga acceso recursos tecnológicos que les aceleren sus actividades.

Una infraestructura flexible de computación de alto rendimiento basada en la nube aportaría beneficios a las necesidades de investigación de la Universidad Nacional Autónoma de Honduras, es por esto que el paso siguiente es la conformación de una solución permamente y accesible para los usuarios con necesidades computacionales elevadas. El trabajo futuro debe ir enfocado a portar y evaluar nuevas aplicaciones, desarrollar una infraestructura accesible a los investigadores. A la vez debe de dar soporte a necesidades de gobierno electrónico y de tecnología de bloques de ser necesario.

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Bibliografía

[1] Agudelo Santos, Carlos Alberto y Isaac Zablah: *Sistematización de Experiencias del Centro de Diagnóstico de Imágenes Biomédicas Investigación y Rehabilitación*. Revista del Departamento de Salud Pública / FCM-UNAH, 2(1), 2017.

[2] Alfred, R.: *March 25, 1954: RCA TVs Get The Color For Money*. Wired Magazine, mar 2008. https://www.wired.com/2008/03/dayintech-0325/.

[3] Amazon: *Amazon PrimeVideo*, 2018. https://www.primevideo.com, visitado el 08ENE2018.

[4] Apple Inc.: *iTunes*, 2018. https://www.apple.com/la/itunes/, visitado el 08ENE2018.

[5] Asenov, A.: *Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 µm MOSFET's: A 3-D Atomistic simulation study*. IEEE Transactions on Electron Devices, 45(12):2505–2513, 1998, ISSN 00189383. http://ieeexplore.ieee.org/document/735728/.

[6] Bakulin, P.I., E.V. Kononovich y V. Moroz: *Curso de Astronomía General*. Informe técnico, Sputnik, 1992, ISBN 8480410035, 9788480410038.

[7] Bank, R. E. y D. J. Rose: *Parameter Selection for Newton-Like Methods Applicable to Nonlinear Partial Differential Equations*. SIAM Journal on Numerical Analysis, 17(6):806–822, dec 1980, ISSN 0036-1429. http://epubs.siam.org/doi/10.1137/0717068.

[8] Benz, Willy: *Applications of Smooth Particle Hydrodynamics (SPH) to astrophysical problems*. Computer Physics Communications, 48(1):97–105, jan 1988,

ISSN 00104655. http://linkinghub.elsevier.com/retrieve/pii/0010465588900276.

[9] Bhardwaj, S., L. Jain y S. Jain: *Cloud computing: A study of infrastructure as a service (IAAS)*. International Journal of engineering and information Technology, 2(1):60–63, 2010.

[10] Burkardt, John: *MPI Examples*, 2011. https://people.sc.fsu.edu/~jburkardt/, visitado el 07FEB2018.

[11] Byrd, G.G. y M.J. Valtonen: *Internal Tail Structure and Orbit Dynamics of the Radio Galaxy 3C129*. En *Bulletin of the American Astronomical Society*, volumen 12, página 503, 1980.

[12] Cellary, W. y S. Strykowski: *e-government based on cloud computing and service-oriented architecture*. En *Proceedings of the 3rd International Conference on Theory and Practice of Electronic Governance - ICEGOV '09*, página 5, New York, New York, USA, 2009. ACM Press, ISBN 9781605586632. http://portal.acm.org/citation.cfm?doid=1693042.1693045.

[13] Corp., INTEL: *Intel Math Kernel Library Benchmarks*, 2017. https://software.intel.com/en-us/articles/intel-mkl-benchmarks-suite, visitado el 21JUN2017.

[14] CSPI: *CSPI Technology Solutions*, 2018. https://www.cspi.com/, visitado el 07FEB2018.

[15] Dale, A., B. Fischl y M. Sereno: *Cortical Surface-Based Analysis I: Segmentation and Surface Reconstruction*. Neuroimage, 9(2):179–194, 1999.

[16] Delgado, J., J.C. Moure, Y. Vives-Gilabert, M. Delfino, A. Espinosa y B. Gómez-Ansón: *Improving the Execution Performance of FreeSurfer*. Neuroinformatics, 12(3):413–421, jul 2014, ISSN 1539-2791. http://link.springer.com/10.1007/s12021-013-9214-1.

[17] Dikaiakos, M.D., D. Katsaros, P. Mehra, G. Pallis y A. Vakali: *Cloud computing: Distributed internet computing for IT and scientific research*. IEEE Internet computing, 13(5), 2009.

[18] Dongarra, Jack J.: *The LINPACK Benchmark: An explanation*. páginas 456–474. 1988. https://dl.acm.org/citation.cfm?id=73977.

[19] Dongarra, J.J., J.R. Bunch, C.B. Moler y G.W. Stewart: *LINPACK Users' Guide*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 1979, ISBN 9780898711721. https://books.google.hn/books?id=AmSm1n3Vw0cC.

[20] Ercan, T.: *Effective use of cloud computing in educational institutions*. Procedia - Social and Behavioral Sciences, 2(2):938–942, 2010, ISSN 18770428. http://linkinghub.elsevier.com/retrieve/pii/S1877042810001709.

[21] Fischl, B.: *FreeSurfer*. NeuroImage, 62(2):774–781, aug 2012, ISSN 10538119. http://linkinghub.elsevier.com/retrieve/pii/S1053811912000389.

[22] Forouzanfar, M.H., L. Alexander, H.R. Anderson, V.F. Bachman, S. Biryukov, M. Brauer y et al. Burnett: *Global, regional, and national comparative risk assessment of 79 behavioural, environmental and occupational, and metabolic risks or clusters of risks in 188 countries, 1990–2013: a systematic analysis for the Global Burden of Disease Study 2013*. The Lancet, 386(10010):2287–2323, dec 2015, ISSN 01406736. http://linkinghub.elsevier.com/retrieve/pii/S0140673615001282.

[23] Forum, MPEG4 Industry: *MPEG-4 – The Media Standard*. Informe técnico, m4if, 2002. http://www.oipf.tv/docs/mpegif/m4-out-20027.pdf.

[24] Foundation, The Linux: *The Xen Project*, 2017. https://xenproject.org/developers/teams/hypervisor.html, visitado el 21JUN2017.

[25] FreeSurfer: *FreeSurfer Software Suite*, 2015. http://freesurfer.net, visitado el 15SEP2015.

[26] FreeSurfer: *Recon-all Step-wise Directives*, 2016. https://surfer.nmr.mgh.harvard.edu/fswiki/recon-all{#}StepwiseDirectives-{%}0A1, visitado el 15SEP2016.

[27] Garcıa-Loureiro, A.J., T.F. Pena, J.M. Lopez-Gonzalez y Ll. Prat: *Parallel implementation of a simulator for heterojunction bipolar transistors*. VIII Symp. on Parallelism, páginas 41–50, 1997.

[28] Gomez-Folgar, F., J. Cacheiro, C. Sánchez-Fernández, A. Garcia-Loureiro y R. Valin: *An e-Science infrastructure for nanoeletronic simulations based on grid and cloud technologies*. En *Proceedings of the 8th Spanish Conference on Electron Devices, CDE'2011*, páginas 1–4. IEEE, feb 2011, ISBN 978-1-4244-7863-7. http://ieeexplore.ieee.org/document/5744153/.

[29] Gomez-Folgar, F., R. Valin, A. Garcia-Loureiro, T. F. Pena y I. Zablah: *Cloud computing for teaching and learning MPI with improved network communications*. En Mikroyannidis, Alexander, Rocael Hernández Rizzardini y Hans Christian Schmitz (editores): *1st International Workshop on Cloud Education Environments (WCLOUD 2012)*, páginas 22–27, Antigua, Guatemala, 2012. CEUR Workshop Proceedings. http://ceur-ws.org/Vol-945/paper5.pdf.

[30] Gomez Folgar, Fernando, Guillermo Indalecio Fernandez, Jose Isaac Zablah Avila, Natalia Seoane Iglesias, Antonio Jesus Garcia Loureiro y Tomas Fernandez Pena: *A study of the influence of VM allocation policies on MPI Bcast and MPI Exchange latency in cloud*. IEEE Latin America Transactions, 15(8):1490–1496, 2017, ISSN 1548-0992. http://ieeexplore.ieee.org/document/7994797/.

[31] Greengard, S.: *Cloud computing and developing nations*. Communications of the ACM, 53(5):18, may 2010, ISSN 00010782. http://portal.acm.org/citation.cfm?doid=1735223.1735232.

[32] Gronenschild, E.H.B.M., P. Habets, H.I.L. Jacobs, R. Mengelers, N. Rozendaal, J. van Os y M. Marcelis: *The Effects of FreeSurfer Version, Workstation Type, and Macintosh Operating System Version on Anatomical Volume and Cortical Thickness Measurements*. PLoS ONE, 7(6):e38234, jun 2012, ISSN 1932-6203. http://dx.plos.org/10.1371/journal.pone.0038234.

[33] Habib, Irfan: *Virtualization with KVM*. Linux J., 2008(166), feb 2008, ISSN 1075-3583. http://dl.acm.org/citation.cfm?id=1344209.1344217.

[34] Helfer, T.T., M.D. Thornley, M.W. Regan, T. Wong, K. Sheth, S.N. Vogel, L. Blitz y D.C.J. Bock: *The BIMA Survey of Nearby Galaxies (BIMA SONG). II. The CO Data*. The Astrophysical Journal Supplement Series, 145(2):259–327, apr 2003, ISSN 0067-0049. http://stacks.iop.org/0067-0049/145/i=2/a=259.

[35] Hoffa, C., G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman y J. Good: *On the Use of Cloud Computing for Scientific Workflows*. En *2008 IEEE Fourth International Conference on eScience*, páginas 640–645. IEEE, dec 2008, ISBN 978-1-4244-3380-3. http://ieeexplore.ieee.org/document/4736878/.

[36] Inria Bordeaux Research Centre: *Myrinet Express over Generic Ethernet Hardware*, 2018. http://open-mx.gforge.inria.fr/, visitado el 07FEB2018.

[37] Intel, M P I: *Benchmarks: Users Guide and Methodology Description*. Intel GmbH, Germany, 452, 2004.

[38] Jacobini, C. y P. Lugli: *The Monte Carlo method for semiconductor device simulation*. Springer Science & Business Media, 2012.

[39] Jaeger, S.: *The common astronomy software application (CASA)*. En *Astronomical Data Analysis Software and Systems XVII*, volumen 394, páginas 623–624, 2008.

[40] Jonassen, David, Mark Davidson, Mauri Collins, John Campbell y Brenda Bannan Haag: *Constructivism and computer mediated communication in distance education*. American Journal of Distance Education, 9(2):7–26, jan 1995, ISSN 0892-3647. http://www.tandfonline.com/doi/abs/10.1080/08923649509526885.

[41] Jones, M.T.: *Virtio: marco de virtualización de entrada/salida para Linux*. Informe técnico, IBM, 2010. https://www.ibm.com/developerworks/ssa/linux/library/l-virtio/.

[42] Klein, A. y J. Tourville: *101 Labeled Brain Images and a Consistent Human Cortical Labeling Protocol*. Frontiers in Neuroscience, 6, 2012, ISSN 1662-4548. http://journal.frontiersin.org/article/10.3389/fnins.2012.00171/abstract.

[43] Lawson, C. L., R. J. Hanson, D. R. Kincaid y F. T. Krogh: *Basic Linear Algebra Subprograms for Fortran Usage*. ACM Transactions on Mathematical Software, 5(3):308–323, sep 1979, ISSN 00983500. http://portal.acm.org/citation.cfm?doid=355841.355847.

[44] Letaifa, A.B., A.l Haji, M. Jebalia y S. Tabbane: *State of the Art and Research Challenges of new services architecture technologies: Virtualization, SOA and Cloud Computing*. International Journal of Grid and Distributed Computing, 3(4):69–88, 2010.

[45] Libvirt Project: *The virtualization API*, 2018. http://libvirt.org/, visitado el 31ENE2018.

[46] Lin, G., D. Fu, J. Zhu y G. Dasmalchi: *Cloud computing: IT as a service*. IT Professional Magazine, 11(2):10, 2009.

[47] Liu, Q., R. Safavi-Naini y N.P. Sheppard: *Digital Rights Management for Content Distribution*. En *Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers 2003 - Volume 21*, ACSW Frontiers '03, páginas 49–58, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc., ISBN 1-920682-00-7. http://dl.acm.org/citation.cfm?id=827987.827994.

[48] Liu, S., W. Cai, S. Liu, F. Zhang, M. Fulham, D. Feng, S. Pujol y R. Kikinis: *Multimodal neuroimaging computing: a review of the applications in neuropsychiatric disorders*. Brain Informatics, 2(3):167–180, sep 2015, ISSN 2198-4018. http://link.springer.com/10.1007/s40708-015-0019-x.

[49] Markowich, Peter A.: *The stationary semiconductor device equations*. Springer Science & Business Media, 2013.

[50] Matroska Organization: *Matroska Media Container*, 2018. https://matroska.org/, visitado el 08ENE2018.

[51] MCBIN: *Testing FreeSurfer*. Informe técnico, Imaging, Martinos Center for Biomedical and Neuroimaging, Laboratory for Computational, 2016. https://surfer.nmr.mgh.harvard.edu/fswiki/TestingFreeSurfer.

[52] Mell, P. M. y T. Grance: *The NIST definition of cloud computing*. Informe técnico, National Institute of Standards and Technology, Gaithersburg, MD, 2011. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

[53] Meuer, H.W., E. Strohmaier, J. Dongarra, H. Simon y M. Meuer: *TOP500 Supercomputer Sites*, 2015. https://www.top500.org/, visitado el 21MAY2017.

[54] Meyer, Kenneth, Glen Hall y Dan Offin: *Introduction to Hamiltonian dynamical systems and the N-body problem*, volumen 90. Springer Science & Business Media, 2008.

[55] MGH/HST Athinoula A. Martinos: *Center for Biomedical Imaging*, 2016. https://www.nmr.mgh.harvard.edu, visitado el 02SEP2016.

[56] Miley, G.K., G.C. Perola, P.C. der Kruit y H. der Laan: *Active galaxies with radio trails in Clusters*. Nature, 237(5353):269–272, 1972.

[57] Mills, D.L.: *Network Time Protocol (Version 3) Specification, Implementation and Analysis*, 1992. https://tools.ietf.org/html/rfc1305, visitado el 28DIC2017.

[58] Mohsen, N.: *Global, regional, and national age–sex specific all-cause and cause-specific mortality for 240 causes of death, 1990–2013: a systematic analysis for the Global Burden of Disease Study 2013*. The Lancet, 385(9963):117–171, jan 2015, ISSN 01406736. http://linkinghub.elsevier.com/retrieve/pii/S0140673614616822.

[59] Motahari-Nezhad, H.R. and Stephenson, B. and Singhal, S.: *Outsourcing business to cloud computing services: Opportunities and challenges*. IEEE Internet Computing, 10(4):1–17, 2009.

[60] MPI Forum: *MPI*, 2018. http://mpi-forum.org/docs/, visitado el 06FEB2018.

[61] Murray, C.J.L., R.M. Barber, K.J. Foreman, A.A. Ozgoren, F. Abd-Allah, S.F. Abera, V. Aboyans, J.P. Abraham, I. Abubakar, L.J. Abu-Raddad, N.M. Abu-Rmeileh y et al. Achoki: *Global, regional, and national disability-adjusted life years (DALYs) for 306 diseases and injuries and healthy life expectancy (HALE) for 188 countries,*

1990–2013: quantifying the epidemiological transition. The Lancet, 386(10009):2145–2191, nov 2015, ISSN 01406736. http://linkinghub.elsevier.com/retrieve/pii/S014067361561340X.

[62] NetFlix Inc.: *NetFlix*, 2018. https://www.netflix.com/hn/, visitado el 08ENE2018.

[63] NINDS: *Neurological diagnostic tests and procedures*, 2016. https://www.ninds.nih.gov/Disorders/Patient-Caregiver-Education/Fact-Sheets/Neurological-Diagnostic-Tests-and-Procedures-Fact.

[64] NIST: *NIST Big Data Interoperability Framework: Volume 1, Definitions*. Informe técnico, National Institute of Standards and Technology, Gaithersburg, MD, oct 2015. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-1.pdf.

[65] Norcott, W.D.: *IOZone Filesystem Benchmark*, 2017. http://www.iozone.org, visitado el 22JUN2017.

[66] OpenNebula Project: *OpenNebula*, 2017. https://opennebula.org/, visitado el 20SEP2017.

[67] Oracle Corp.: *MySQL 5.6 Reference Manual*, 2015. https://dev.mysql.com/doc/refman/5.6/en/innodb-fulltext-index.html, visitado el 30NOV2017.

[68] Oracle Corp.: *MySQL*, 2017. https://www.mysql.com/, visitado el 28DIC2017.

[69] Ott, J. y J. Kern: *CASA Synthesis & Single Dish Reduction Reference Manual & Cookbook*. CASA Supplement Series, 2017. https://casa.nrao.edu/casa_cookbook.pdf.

[70] Peng, Junjie, Xuejun Zhang, Zhou Lei, Bofeng Zhang, Wu Zhang y Qing Li: *Comparison of Several Cloud Computing Platforms*. En *2009 Second International Symposium on Information Science and Engineering*, páginas 23–27. IEEE, dec 2009, ISBN 978-1-4244-6325-1. http://ieeexplore.ieee.org/document/5447227/.

[71] Philpott, G. y A. Kattukaran: *Evolution of TV: How TV's Migration to the Cloud Might Upend TV As We Know It*. Informe técnico, Google, 2015. https://www.thinkwithgoogle.com/marketing-resources/evolution-of-tv-migration-to-the-cloud/.

[72] Roth, G.D. (editor): *Handbook of Practical Astronomy*. Springer Berlin Heidelberg, Berlin, H., 2009, ISBN 978-3-540-76377-2. http://www.springerlink.com/index/10.1007/978-3-540-76379-6.

[73] Roy, Gareth, Andrew R. Brown, Fikru Adamu-Lema, Scott Roy y Asen Asenov: *Simulation Study of Individual and Combined Sources of Intrinsic Parameter Fluctuations in Conventional Nano-MOSFETs*. IEEE Transactions on Electron Devices, 53(12):3063–3070, dec 2006, ISSN 0018-9383. http://ieeexplore.ieee.org/document/4016359/.

[74] Saad, Yousef, Gen Ching Lo y Sergey Kuznetsov: *PSPARSLIB users manual: A portable library of parallel sparse iterative solvers*. Univ. of Minnesota, 1997.

[75] Scharfetter, D.L. y H.K. Gummel: *Large-signal analysis of a silicon Read diode oscillator*. IEEE Transactions on Electron Devices, 16(1):64–77, jan 1969, ISSN 0018-9383. http://ieeexplore.ieee.org/document/1475609/.

[76] SDRT/UNAH: *Utv*, 2017. https://utv.unah.edu.hn/, visitado el 11NOV2017.

[77] Selberherr, S.: *Analysis and Simulation of Semiconductor Devices*. Springer Vienna, 2012, ISBN 9783709187524. https://books.google.hn/books?id=E7roCAAAQBAJ.

[78] Selberherr, Siegfried: *Analysis and Simulation of Semiconductor Devices*. Springer Vienna, Vienna, 1984, ISBN 978-3-7091-8754-8. http://link.springer.com/10.1007/978-3-7091-8752-4.

[79] Sempolinski, Peter y Douglas Thain: *A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus*. En *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, páginas 417–426. IEEE, nov 2010, ISBN 978-1-4244-9405-7. http://ieeexplore.ieee.org/document/5708480/.

[80] Seoane, N., A.J. García-Loureiro, K. Kalna y A. Asenov: *Impact of intrinsic parameter fluctuations on the performance of HEMTs studied with a 3D parallel drift-diffusion simulator*. Solid-State Electronics, 51(3):481–488, mar 2007, ISSN 00381101. http://linkinghub.elsevier.com/retrieve/pii/S003811010700069X.

[81] Seoane, N., R. Valin, A. Garcia-Loureiro, T.F. Pena y I. Zablah: *Performance of numerical simulations on the cloud*. En *Actas de las XXIII Jornadas de Paralelismo (JP2012)*, páginas 395–399, Elche, Alicante, 2012. Universidad de Alicante.

[82] Seoane, Natalia, A. J. García-Loureiro, K. Kalna y A. Asenov: *Random dopant related variability in the 30 nm gate length In0.75Ga0.25As implant free MOSFET*. Journal of Computational Electronics, 7(3):159–163, sep 2008, ISSN 1569-8025. http://link.springer.com/10.1007/s10825-008-0233-3.

[83] Springel, V.: *GADGET2*, 2016. http://wwwmpa.mpa-garching.mpg.de/gadget/, visitado el 20SEP2017.

[84] The Apache Software Foundation: *Apache CloudStack*, 2017. http://cloudstack.apache.org/, visitado el 20SEP2017.

[85] The CentOS Project: *CentOS*, 2017. https://www.centos.org/, visitado el 28DIC2017.

[86] The HandBrake Team: *Handbrake: The open source video transcoder*, 2017. https://handbrake.fr/, visitado el 20OCT2017.

[87] The KVM Project: *KVM*, 2017. https://www.linux-kvm.org/page/Main_Page, visitado el 21JUN2017.

[88] The Open MPI Project: *Open Source High Performance Computing*, 2018. https://www.open-mpi.org, visitado el 07FEB2018.

[89] The OpenStack project: *OpenStack*, 2017. https://www.openstack.org/, visitado el 20SEP2017.

[90] Thilker, D.A., R. Braun y R.M. Walterbos: *Expanding HI shells in NGC 2403. First results from an automated object recognition package*. Astronomy and Astrophysics, 332:429–448, 1998.

[91] Urquía-Osorio, Hebel., Isaac Zablah y Iscia Lopes-Cendes: *Medicina de precisión: ¿un nuevo paradigma en salud?* Revista Hispanoamericana de Ciencias de la Salud, 2(3):199–202, 2016. http://www.uhsalud.com/index.php/revhispano/article/download/168/107.

[92] USC: *Universidad Santiago de Compostela*, 2018. http://www.usc.es/, visitado el 31ENE2018.

[93] Vecchiola, C., S. Pandey y R. Buyya: *High-Performance Cloud Computing: A View of Scientific Applications*. En *2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, páginas 4–16. IEEE, 2009, ISBN 978-1-4244-5403-7. http://ieeexplore.ieee.org/document/5381983/.

[94] VirtualBox: *VirtualBox*, 2017. https://www.virtualbox.org/, visitado el O1ENE2012.

[95] Wang, L., J. Tao, M. Kunze, D. Rattu y A.C. Castellanos: *The cumulus project: Build a scientific cloud for a data center*. En *First workshop on Cloud Computing and its Applications (CCA), Computation Institute and the National Center for Data Mining at UIC, Chicago, USA, URL http://www. cca08. org*, 2008.

[96] Ward, J.S. y A. Barker: *Undefined By Data: A Survey of Big Data Definitions*. arXiv, sep 2013. http://arxiv.org/abs/1309.5821.

[97] Wu, D., M.J. Greer, D.W. Rosen y D. Schaefer: *Cloud Manufacturing: Drivers, Current Status, and Future Trends*. En *Volume 2: Systems; Micro and Nano Technologies; Sustainable Manufacturing*, página V002T02A003. ASME, jun 2013, ISBN 978-0-7918-5546-1. http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/MSEC2013-1106.

[98] Wu, D., M.J. Greer, D.W. Rosen y D. Schaefer: *Cloud manufacturing: Strategic vision and state-of-the-art*. Journal of Manufacturing Systems, 32(4):564–579, oct 2013, ISSN 02786125. http://linkinghub.elsevier.com/retrieve/pii/S0278612513000411.

[99] Zablah, I., A. Garcia-Loureiro, F. Gomez-Folgar y T. F. Pena: *Render on the cloud: Using cinelerra on virtualized infrastructures*. 1st International Workshop on Cloud

Education Environments (WCLOUD 2012), páginas 28–32, 2012, ISSN 16130073. http://ceur-ws.org/Vol-945/paper6.pdf.

[100] Zablah, Isaac y Hebel. Urquía Osorio: *Las Ciencias de la Salud y la revolución del Big-Data*. Revista del Cuerpo Medico HNAAA, 9(2):88–89, 2016. http://cmhnaaa.org.pe/ojs/index.php/RCMHNAAA/article/view/327/283.

[101] Zablah, Isaac, Raúl Valin, Antonio García-Loureiro, Javier López Cacheiro y Fernando Gomez-Folgar: *Comparación del rendimiento entre los hipervisores XEN y KVM usando virtualización por hardware*. En Almeida, Francisco, Vicente Blanco, Coromoto León, Casiano Rodríguez y Francisco De Sande (editores): *Actas de las XXII Jornadas de Paralelismo (JP2011)*, páginas 543–548, La Laguna, Tenerife, 2016. Servicio de Publicaciones, Universidad de la Laguna, ISBN 978-84-694-1791-1.

[102] Zablah, J.I., F. Gomez-Folgar y A.J. Garcia-Loureiro: *Fostering data reduction and analysis in radio astronomy with CASA through the use of Cloud and virtualization technologies*. 2nd International Conference on Advanced in Computer, Eletrical and Electronics Engineering (ICACEEE 2014) Co-jointed with 2nd International Conference on Computing, E-Learning and Emerging Technologies (ICCEET 2014), 16(1), 2014. http://ijitcs.com/volume16{_}No{_}1/Zablah.pdf.

[103] Zablah, Jose Isaac, Fernando Gomez Folgar, Antonio Garcia Loureiro, Jorge Alberto Almendares y Marco Tulio Medina: *Performance evaluation of FreeSurfer medical application in cloud computing*. IEEE Latin America Transactions, 15(11):2232–2237, nov 2017, ISSN 1548-0992. http://ieeexplore.ieee.org/document/8070431/.

[104] Zablah, José Isaac y Antonio García Loureiro: *Análisis del rendimiento de una aplicación cosmológica sobre máquinas virtuales*. Actas de las XXIII Jornadas de Paralelismo (JP2012), páginas 407–411, 2012.

[105] Zablah, José Isaac, Antonio García Loureiro y Fernando Gómez Folgar: *Performance Evaluation of hypervisors used in cloud infrastructures that leverage hardware virtualization*. Portal de la Ciencia, 6(2):107–120, feb 2015, ISSN 2223-3059. https://dicyp.unah.edu.hn/investigacion-cientifica/publicacion-comunicacion-y-difusion/revistas/

portal-de-la-ciencia/http://www.lamjol.info/index.php/PC/article/view/1846.

[106] Zhang, L.J., J. Zhang, J. Fiaidhi y J.M. Chang: *Hot Topics in Cloud Computing*. IT Professional, 12(5):17–19, sep 2010, ISSN 1520-9202. http://ieeexplore.ieee.org/document/5593035/.

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Índice de figuras

2.1. Mapa ilustrativo de la distribución geográfica de los centros de súper computación según la lista TOP500 de noviembre del 2018. 13
2.2. Niveles de servicio en la nube. 15
2.3. Diagrama de Venn que muestra la convergencia de los diferentes tipos de nubes. 16
2.4. Diagrama que muestra los componentes de OpenNebula [66]. 19
2.5. Diagrama que muestran los componentes de OpenStack [89]. 21
2.6. Diagrama de operación del gestor Apache CloudStack. 23

4.1. Comparación del tiempo requerido (en segundos) para completar las simulación en las diferentes infraestructuras cuando estas no poseen el soporte del *hyperthreading*. 45
4.2. Comparación del tiempo requerido (en segundos) para completar las simulación en las diferentes infraestructuras cuando estas poseen el soporte del *hyperthreading* activado. 47
4.3. Tiempo requerido por las máquina virtual (MV) para completar la simulación de la evolución de NGC4826, se muestra el efecto del soporte del *hyperthreading*. 51

4.4. Tiempo requerido por las máquinas virtuales (MV) para completar la simulación de la evolución de 3C129, se muestra el efecto del soporte del *hyperthreading*. 52
4.5. Tiempo requerido por las máquinas virtuales (MV) para completar la simulación de la evolución de NGC2403, se muestra el efecto del soporte del *hyperthreading*. 53
4.6. La gráfica muestra la capacidad de procesamiento en GFlops obtenido por las máquinas virtuales (MV) y el efecto del soporte del *hyperthreading* en esas mediciones. 54
4.7. Flujo de trabajo del transcodaje. Handbrake se encuentra virtualizado y su uso es transparente al usuario. 58
4.8. Flujo de trabajo con las diferentes etapas involucradas en la operación televisiva. 64
4.9. Flujo de trabajo en un centro de capacitación, nótese que los estudiantes no saben donde se encuentran realmente los nodos físicos, pueden los mismos estar distribuidos en múltiples sitios siempre que estén conectados a través de una red. 66
4.10. Desempeño con 40 segundos de vídeo a una resolución de 1080/60p. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse. 67
4.11. Desempeño con 30 minutos de vídeo a una resolución de 720p. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse 68
4.12. Desempeño con 30 minutos de vídeo a una resolución de 1080/60i. Barras más cortas indican mejores resultados, ya que necesitan menos tiempo para completarse 69
4.13. Tiempo de ejecución de *autorecon 1* para el sujeto *Bert* para cada una de las series de medidas obtenidas sobre la plataforma física (anfitrión), sobre la máquina virtual KVM y sobre la máquina virtual de VirtualBox. 75
4.14. Perfil de utilización de la CPU durante la ejecución de *autorecon 1* para el individuo *Bert* frente al tiempo. 75
4.15. Tiempo (en segundos) utilizado por las infraestructuras para completar el script muestreo, en una serie de diez pruebas consecutivas de ejecución. . . . 82

4.16. Utilización de memoria RAM en las diferentes ejecuciones en cada infraestructura. 85
4.17. Diagrama de flujo de la simulación deriva-difusión (*drift-diffusion*) del dispositivo 1D. 90
4.18. Tiempo de simulación deriva-difusión del dispositivo 1D versus el número de máquinas virtuales por anfitrión cuando no se generan operaciones de E/S. Se comparan los resultados con *hyperthreading* activado o desactivado. . . . 91
4.19. Tiempo de simulación de *drift-diffusion* del dispositivo 1D versus el número de máquinas virtuales por anfitrión cuando existen operaciones de E/S. Se comparan los resultados con *hyperthreading* activado o desactivado. 92
4.20. Operaciones de coma flotante por segundo versus la cantidad de máquinas virtuales por anfitrión cuando se utiliza Linpack. La influencia del *hyperthreading* se ha incluido en la gráfica. 93
4.21. Tiempo de simulación versus el número de máquinas virtuales por anfitrión cuando se utiliza Linpack. La influencia del *hyperthreading* se ha incluido en la gráfica. 94
4.22. Arquitectura de operación de Apache CloudStack. 96
4.23. Esquema de operación de un clúster para MPI. 98
4.24. Latencia frente a bytes en la prueba de transferencia simple de red (PinPong) y (PingPing). 103
4.25. Latencia frente a bytes en la prueba de transferencia paralela de red (Exchange) y (Sendrecv). 103
4.26. Latencia frente a bytes en la prueba detransferencia colectiva (Allgather) y (Allgatherv). 104
4.27. Latencia frente a bytes en la prueba de transferencia colectiva (AlltoAll) y (AlltoAllv). 105
4.28. Latencia frente a bytes en la prueba de transferencia colectiva (Reduce) y (Reduce_Scatter). 105

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

Índice de tablas

2.1. Distribución de los centros de supercomputación según la lista TOP500 de noviembre del 2018 13

3.1. Resultados con Linpack, detalla capacidad de cálculo y tiempo requerido de ejecución. 36

3.2. Tasas de transferencia en Kbytes/s de escritura y re-escritura en IOZone. . . 38

3.3. Resultados de la simulación de nanodispositivos, valores en segundos. . . . 39

4.1. Transcodaje del archivo fuente sobre anfitrión físico. Medidas en segundos (s), menor tiempo indica mejores resultados. De la primera columna de izquierda a derecha, corresponden a la funcionalidad de *Turbo* (T) y la segunda el número de núcleos empleados (N), la última es la media aritmética de las medidas (Prom.) 60

4.2. Transcodaje del archivo fuente sobre máquinas virtuales. Medidas en segundos (s), menor tiempo indica resultados más eficientes. La primera columna de izquierda a derecha, corresponde a la funcionalidad de *Turbo* (T) y la segunda el número de núcleos empleados (N), la última es la media aritmética de las medidas (Prom.) 60

4.3. Tiempo de ejecución por patrón en la máquina física. A menor tiempo mayor eficiencia. 83
4.4. Tiempo de ejecución por patrón de la instancia cloud. A menor tiempo mayor eficiencia. 84

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA