Esquemas de distribución de trabajos para un sistema ERP en la nube

110300083 Canul Canul Cindy Guadalupe
110300083@ucaribe.edu.mx
110300097 Kumul Uc Cristian Ariel
110300097@ucaribe.edu.mx
110300079 Peraza Feliciano Jonathan
110300079@ucaribe.edu.mx

Asesor: Joel Antonio Trejo Sánchez

Otoño 2015

Ingeniería en Telemática

Universidad del Caribe
CONOCIMIENTO Y CULTURA PARA EL DESARROLLO HUMANO

Departamento de Ciencias Básicas e Ingenierías
IT0427 / Proyecto terminal

CARTA DE ACEPTACIÓN DE PROYECTO TERMINAL

TÍTULO
Esquemas de distribución de trabajo para un sistemas ERP en la nube.

AUTORES
110300083
Cindy Guadalupe Canul Canul
110300083@ucaribe.edu.mx

110300097
Cristian Ariel Kumul Uc
110300097@ucaribe.edu.mx

110300079
Jonathan Peraza Feliciano
110300079@ucaribe.edu.mx

ASESOR
Joel Antonio Trejo Sánchez

RESUMEN
El cómputo en la nube es una tecnología prometedora como plataforma del futuro, permite el acceso a recursos de hardware de manera remota. Dentro del funcionamiento de los servicios es necesario optimizar los recursos en el centro de datos y para esto se utilizan los esquemas de calendarización. El problema incrementa con el número de tareas, pues es un problema NP-difícil, debido a las distintas características de cada host.
En este proyecto se estudiará y se hará una comparación de varios esquemas de calendarización para proponer una mejora en la optimización de recursos de acuerdo a las peticiones de un sistema ERP en la nube.

FIRMAS

AUTORES	ASESOR	TITULAR DE LA ASIGNATURA

Resumen

Abstract

Índice general

Índice de figuras

Índice de cuadros

Introducción

Antecedentes y Planteamiento del problema

El cómputo en la nube es una tecnología que ha abarcado gran parte de los negocios para dar soporte a ellos. Ésta tecnología provee a los negocios una ventaja competitiva en los costos de recursos como se ve en los negocios tradicionales, además de la versatilidad que provee al ajustarse a la necesidad de las empresas (Srinivasan, 2014).

La tecnología en la nube ha desarrollado una infraestructura fuerte después del surgimiento del cómputo distribuido (Chen y Deng, 2009). Para obtener las ventajas de dicha tecnología los usuarios simplemente necesitarán conectarse a internet y de esta manera tendrán el acceso al procesamiento de manera remota (Aranganathan, 2011). Sin embargo, para aprovechar el máximo potencial de éstos recursos, es necesario tener en consideración algunas variables, ya que en un entorno en la nube existe un comportamiento dinámico de los recursos a manera que se les provea a los usuarios el servicio (Shimpy y Sidhu, 2014). Una de las prácticas con mayor importancia en la nube es la calendarización, ya que tiene como objetivo administrar las tareas del centro de datos para optimizar los recursos del mismo. De esta manera la eficiencia de la carga de trabajo en la nube aumenta (Shimpy y Sidhu, 2014). En general, el objetivo de la calendarización en la nube es utilizar los recursos de manera apropiada, mientras la carga de trabajo es distribuida uniformemente para mejorar los tiempos de ejecución (Shimpy y Sidhu, 2014). Debido a la atención que se tiene en la tecnología en la nube, los centros de datos han tomado un papel muy importante para los servicios empresariales (Shimpy y Sidhu, 2014). Un centro de datos está compuesto por miles de servidores virtuales ejecutándose en una instancia de

tiempo alojando muchas tareas, al mismo tiempo el centro de datos recibe miles de peticiones a esas tareas. Es aquí en donde la programación de trabajos tiene un rol demasiado importante para el cómputo en la nube, ya que influye en el rendimiento del mismo (Srinivasan, 2014).

El problema de la calendarización pertenece a los algoritmos NP-Difícil, lo cual tiene un amplio rango de soluciones posibles y se toma mucho más tiempo de encontrar una respuesta optima, ya que no existe un método para resolver estas incógnitas. Sin embargo, es posible estar cerca de la mejor solución contemplando algunos entornos (Shimpy y Sidhu, 2014).

Propuesta de solución y Justificación

Los centros de datos son una parte esencial en la tecnología en la nube, están conformados por varios servidores virtuales que alojan varios trabajos, ejecutándose por estancias de tiempo y a la vez recibiendo peticiones a esos trabajos (Shimpy y Sidhu, 2014). Para que el cómputo en la nube pueda rendir correctamente, se necesita una buena programación de trabajos o calendarización. Este problema tiene un gran número de soluciones posibles, toma mucho tiempo en encontrar un respuesta óptima y por eso está dentro de los algoritmos NP-Difícil. No existe un método para resolver lo anterior, sin embargo es posible obtener una solución cercana comtemplando algunos entornos (Shimpy y Sidhu, 2014). En esta investigación se propone una mejora a un algoritmo de calendarización para el caso de estudio de un sistema ERP en la nube, contemplando sus posibles escenarios y peticiones heterogéneas. De esta manera se podrá contar con un esquema de distribución de trabajos que satisfaga las necesidades de un sistema ERP, mejorando el tiempo de respuesta y aprovechando los recursos. Con beneficios a los proveedores de servicios (SaaS) de sistemas ERP.

En la figura 1 se puede observar la arquitectura del centro de datos que se implementará en la investigación, el cual está compuesto de los siguientes elementos:

- **Servidor administrador:** Es el servidor que tendrá el rol de front end en el centro de datos, teniendo una interacción directa con las peticiones de los usuarios, que son especificados en éste documento como trabajos. El principal

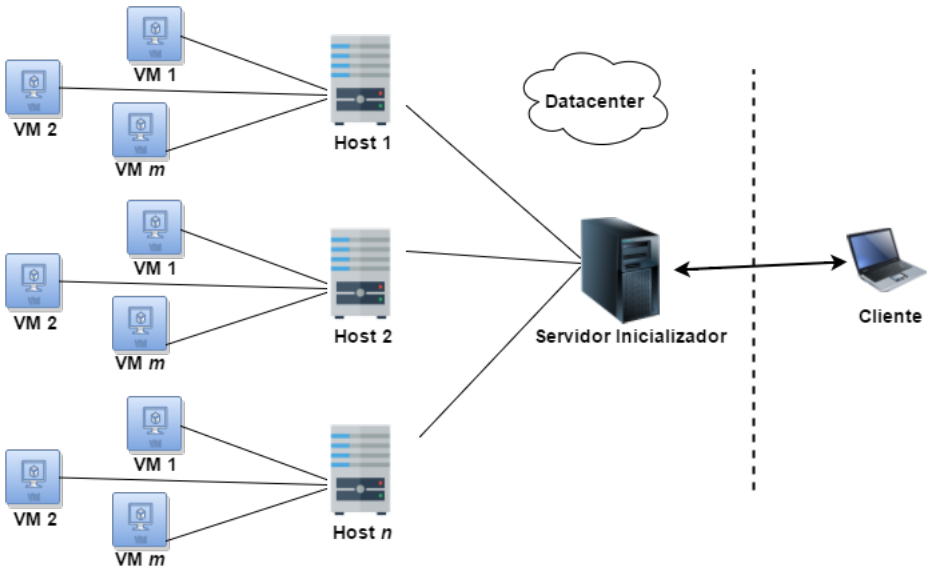


Figura 1: Propuesta de arquitectura en la nube

objetivo de éste elemento es delegar los trabajos a las máquinas virtuales en los distintos hosts del centro de datos.

- **Host:** es el recurso de hardware que se tiene en el centro de datos. Una de las características de éste elemento es que es finito.

- **Máquinas Virtuales:** son instancias dentro de los host, tienen como objetivo resolver los trabajos que se les sea asignado.

Objetivos

Objetivo general

Determinar y proponer un nuevo esquema que pueda resolver de manera eficiente el problema de la calendarización de un centro de datos y satisfacer las necesidades de un sistema ERP en la nube.

Objetivos específicos

- Definir una arquitectura para un centro de datos en la nube con las necesidades de un sistema ERP.
- Realizar la simulación en base a la arquitectura del centro de datos.
- Realizar e implementar diferentes esquemas de distribución de trabajos, rescatando sus características principales.
- Reunir las características principales y proponer el nuevo esquema de distribución.
- Aplicar el nuevo esquema de distribución a las necesidades de un ERP.
- Comparar los resultados obtenidos del nuevo esquema propuesto y de los anteriores para determinar si hubo una mejora.

Hipótesis

La modificación de un esquema de distribución mejorará el rendimiento de la calendarización para un sistema ERP en la nube.

Para realizar la verificación de validez de la hipótesis se usarán:

- **Pruebas de hardware y software:**
  - **Prueba de desempeño:** Validar el tiempo de respuesta para las tareas bajo condiciones normales y máximas.
  - **Prueba de carga:** Verificar el tiempo de respuesta del sistema, bajo diferentes condiciones de carga. Se mide el tiempo de respuesta y otros requisitos sensibles al tiempo.
- **Estadísticas:** en base a los resultados que se obtendrán en las pruebas de desempeño y de carga, se va realizar comparaciones y posteriormente visualizarlas por medio de gráficas y tablas.

Capítulo 1

Marco contextual

Introducción

En este capítulo se describen algunos marcos conceptuales necesarios para fundamentar y analizar los requisitos de la investigación, las secciones son las siguientes: Situación actual, Estado del Arte, Marco Teórico y el Marco Metodológico. En la primera sección se muestra cómo se encuentra la investigación en la actualidad, cómo se encuentra nacional e internacionalmente y cuál es el punto de partida del proyecto; la siguiente sección se presenta la información sobre los diferentes esquemas de distribución que serán implementados durante la elaboracón del proyecto. El Marco teórico nos describe los conceptos necesarios para fundamentar y comprender mejor el proyecto y su propósito, finalmente en la última sección se describe los pasos que se van a seguir para realizar y alcanzar los objetivos del proyecto.

1.1. Situación Actual

Actualmente, hay un mayor número de organizaciones que utilizan el cómputo en la nube, ya que trae beneficios en cuanto a gastos y recursos. Ya no hay necesidad de invertir tanto en la administración de TI (Information Technology) y existe una mejor capacidad de almacenamiento para los recursos que el ámbito empresarial necesita. El cómputo en la nube utiliza los servicios SaaS (Software as a Service) y HaaS (Hardware as a Service) para hacer más eficiente y flexible el uso de la infraestructura de la nube (Mariscal y Gil-Garcia, 2013).

México utiliza, en cuanto a la infraestructura, los servicios ofrecidos por Amazon, Google y Triada (TELMEX). Estos proporcionan almacenaje de información virtual, plataformas de hospedaje de aplicaciones y servicios en línea (Mariscal y Gil-Garcia, 2013).

Al proveer el servicio se tiene dos escenarios diferentes: cuando hay una mayor demanda de recursos y cuando no la hay, eso va dependiendo de las necesidades de las empresas. Para que se tenga una mejor eficacia durante los servicios, se tiene que mejorar el uso de los recursos en el centro de datos. Dentro del centro de datos hay muchos servidores virtuales que están recibiendo trabajos, mientras la nube las mantiene en los lotes de solicitud de trabajos. Es aquí, donde resaltamos la importancia de la distribución de trabajos dentro del centro de datos (Shimpy y Sidhu, 2014).

En este momento, la distribución de trabajos es un problema NP- difícil, pues que no se ha encontrado una solución óptima. Sin embargo, existen diferentes propuestas para encontrar una mejor solución, utilizando diferentes esquemas de distribución o dicho de otra forma algoritmos (Shimpy y Sidhu, 2014).

Existen una gran variedad de algoritmos utilizados en el problema de distribución de trabajos. Los algoritmos más utilizados en las investigaciones son: FCFS, Round Robin, Min-Min, Max-Min y Metaheurística. Cabe mencionar que en las investigaciones solo hacen una selección de una serie de algoritmos para que, por medio de pruebas, determinen quién es el mejor de ellos.

1.2. Estado del Arte

Los siguientes algoritmos de calendarización actualmente están prevaleciendo en la nube:

- **Resource-Aware-Scheduling Algorithm (RASA):** Parsa, Entezari-Maleki (2009) proponen el algoritmo RASA, el cual utiliza las ventajas de dos algoritmos tradicionales (Max-min y Min-min) y cubre sus desventajas. Aunque el tiempo limite, la tasa de llegada, costo de ejecucion y costo de comunicación no están considerados (Parsa y Entezari-Maleki, 2009).

- **RSDC (Reliable Scheduling Distributed In Cloud Computing):** Delevar, Javanmard, Shabestari y Talebi (2012) proponen un algoritmo confiable en un entorno en la nube, en este algoritmo los trabajos importantes son divididos en sub trabajos, de tal manera que se puedan balancear las peticiones (Delavar et al., 2012).

- **An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment:** Dakshayini, Gurupased (2011), proponen un nuevo algoritmo de calendarización que se basa en la prioridad y un esquema de control de admisión. En este algoritmo, la prioridad se asigna a cada proceso admitido en la cola (Dakshayini y Guruprasad, 2011) .

- **Pre-emptable Shortest Job Next Scheduling algorithm (PSJN) :** Este algoritmo se propone en una nube privada. Utiliza la técnica de suscripción preferente del algoritmo de Round Robin junto con el siguiente proceso más corto (PSN). Brinda beneficios de costos y mejora tiempo de respuesta y tiempo de ejecución (nis, 2015).

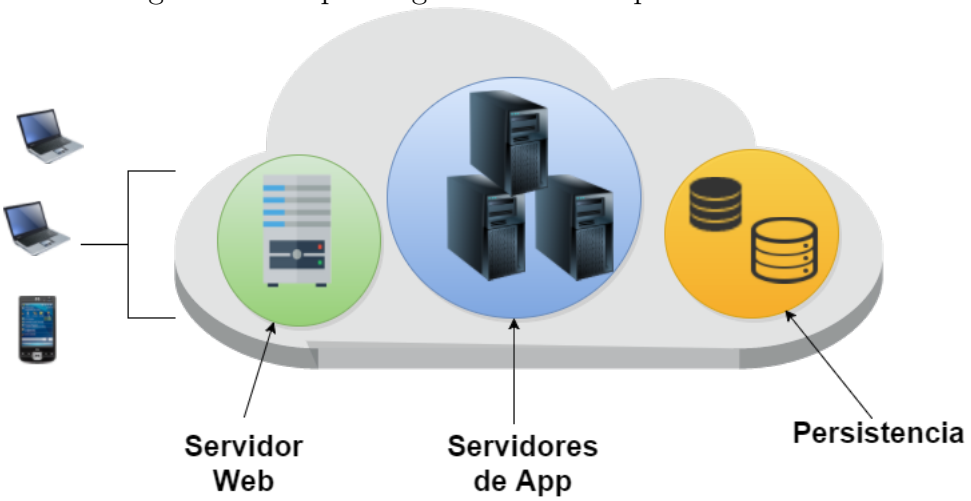
- **User-priority Guided Min-min scheduling algorithm:** Se realiza una mejora para el algoritmo de balanceo de cargas, a través del algoritmo Min-min para la calendarización de trabajos con el fin de minimizar el tiempo de terminación del ultimo trabajo (makespan) y maximizar la utilización de los recursos (Chen et al., 2013).

1.3. Marco teórico

El cómputo en la nube es una tecnología emergente, la cual está compuesta por un grupo de recursos heterogéneos que proveen servicios a través de internet (Agarwal et al., 2014). Esta tecnología permite a los consumidores y negocios utilizar aplicaciones sin necesidad de instalación y con acceso a sus archivos personales en cualquier computadora con acceso a internet (Ahmed et al., 2012). El cómputo en la nube se puede clasificar de dos maneras:

- Por la ubicación:
  - **Nube Pública:** La infraestructura de cómputo se puede compartir entre cualquier organización (Ahmed et al., 2012).
  - **Nube Privada:** La infraestructura de cómputo es dedicada a una organización en particular y no se comparte con otras organizaciones (Ahmed et al., 2012).
  - **Nube Híbrida:** Las organizaciones pueden albergar sus aplicaciones críticas en nubes privadas y las aplicaciones con menos problemas de seguridad las puede albergar en nubes públicas (Ahmed et al., 2012).
- Por el tipo de servicios ofrecidos:
  - **Infrastructure as a Service (IaaS):** En éste nivel, la infraestructura se ofrece como servicio hacia los solicitantes en forma de Máquinas Virtuales (VM) (Agarwal et al., 2014).
  - **Platform as a Service (PaaS):** Es una plataforma de desarrollo de aplicaciones que se provee como servicio hacia los desarrolladores para crear aplicaciones basadas en web (Agarwal et al., 2014).
  - **Software as a Service (SaaS):** En éste nivel el proveedor en la nube provee las aplicaciones de software (Agarwal et al., 2014).

Figura 1.1: Esquema general del Cómputo en la Nube



Centro de Datos

Un centro de datos (datacenter) es un espacio dedicado donde las organizaciones almacenan y operan la mayoría de su información y la infraestructura de las tecnologías de comunicaciones que respaldan sus negocios (EPI, 2015).

Esquemas de Distribución de Trabajos

La distribución de trabajo (calendarización) es uno de los puntos principales además de los más retadores problemas en el cómputo en la nube (Li et al., 2014). Esta consiste en un conjunto de políticas para controlar el orden en el cual los procesos serán ejecutados por el sistema (Agarwal et al., 2014).

Existen varios tipos de algoritmos de calendarización, su principal ventaja es obtener un alto rendimiento al cumplir con las trabajos. Los principales ejemplos de algoritmos de calendarización son (Salot, 2013):

- **FCFS (First Come First Serve) Algorithm:** significa que el trabajo que llegue primero será el primero en ser ejecutado.
- **Round Robin Algorithm:** consiste en agregar un tiempo de ejecución para cada trabajo, si dicho tiempo se agota y el trabajo actual no ha finalizado, pasa a un estado de espera y continúa con el siguiente trabajo, al finalizar el

último trabajo regresa a revisar si existen trabajos en espera para proceder a ejecutarlas.

- **Min-Min Algorithm:** se va ejecutando una por una iniciando por los trabajos de menor tamaño.
- **Max-Min Algorithm:** se va ejecutando una por una iniciando por los trabajos de mayor tamaño.
- **Most fit task scheduling algorithm:** este algoritmo busca los elementos más aptos en la cola de trabajos para ejecutarlo primero. Tiene un alto rádio de error.
- **Priority scheduling algorithm:** la idea básica es que a cada proceso se le asigna una prioridad, y dependiendo a las prioridades de los demás procesos es cómo se ejecutarán cada uno de ellos.

Balanceo de Cargas

Dentro de nuestro entorno en la nube cada host tiene recursos de hardware finitos y son susceptibles a fallos. Para mitigar contra las fallas y el exhaustivo uso de los recursos, los host son agrupados en clusters, los cuales son esencialmente un agrupamiento de recursos compartidos. El administrador (manager) es capaz de asegurarse que ninguno de los host dentro del cluster es responsable de todas las máquinas virtuales dentro de ese cluster (Dover, 2013).

Un entorno de virtualización responde a los cambios en la demanda para los recursos en cada host haciendo uso del balanceo de cargas, calendarización de trabajos y migración.

Una política de balanceo de cargas es configurada para un cluster, que a su vez contiene multiples hosts, cada uno de ellos con distintos recursos de hardware y disponibilidad de memoria (Dover, 2013). Existen tres políticas para el balanceo de cargas:

- Sin Balanceo.
- Distribución Uniforme.

- Ahorro de Energía.

Migración

La migración se utiliza para hacer cumplir las políticas dentro del balanceo de cargas. La migración de una máquina virtual toma lugar de acuerdo a las políticas de balanceo de cargas para un cluster y la demanda actual sobre los hosts dentro de dicho cluster. La migración de igual manera puede ser configurada para ocurrir automáticamente cuando un host es bloqueado o movido a modo de mantenimiento (Dover, 2013).

Sistemas ERP

La sigla ERP, en inglés Enterprise Resource Planning, significa Planificación de los Recursos de la Empresa. Un sistema ERP constituye un marco de trabajo que incluye aplicaciones comerciales, administrativas (finanzas, contabilidad), recursos humanos, planeamiento de manufactura y gestión de proyectos (Saroka, 2002).

1.4. Marco Metodológico

A continuación se propone la siguiente metodología para llevar a cabo el proyecto:

- **Simular:** Se implementará a manera de simulación un centro de datos con un entorno en la nube, las máquinas virtuales y el servidor inicializador que lo conforman.
- **Implementar:** En el centro de datos se desarrollarán los algoritmos de calendarización que se mencionan con antelación.
- **Evaluar:** Se simulará el comportamiento de las peticiones de un sistema contemplando los distintos escenarios del ERP y consumiendo el servicio en la nube (SaaS), para conocer el rendimiento en tiempo de ejecución de los algoritmos.
- **Mejorar:** Se propondrá una mejora a algún algoritmo de acuerdo a las necesidades y comportamiento de un sistema ERP en la nube.

- **Comparar:** Se realizará una comparativa de tiempo de ejecución entre la versión mejorada y la original para el caso de estudio de un sistema ERP.

Capítulo 2

Desarrollo

Introducción

En éste capítulo se muestra la arquitectura de los distintos esquemas de calendarización de trabajos en un entorno en la nube, para la implementación de dichos esquemas hace uso de un framework llamado CloudSim, el cual ser definido de igual manera.

2.1. Aplicación del marco metodológico y de actividades de experimentación

En base de los dos primeros puntos descritos anteriormente en el marco metodológico se realizarón las siguientes activides:

- **Simular:** Se implementará a manera de simulación un centro de datos con un entorno en la nube, las máquinas virtuales y el servidor inicializador que lo conforman.

- **Implementar:** En el centro de datos se desarrollarán los algoritmos de calendarización que se mencionan con antelación.

2.1.1. Simulación del Centro de Datos en la Nube

Cloudsim es un nuevo, generalizado y extensible framework de simulación, que permite el modelaje, simulación y experimentación de infraestructuras emergentes de cómputo en la nube y servicios de aplicación. (Calheiros et al, 2010, p.2).

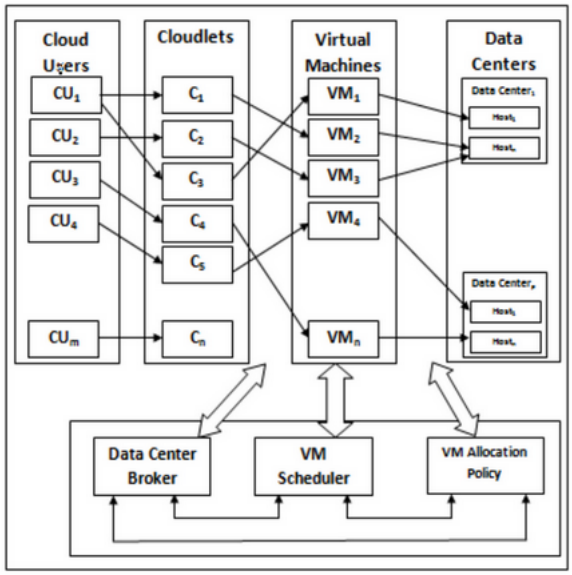
Entre los componentes que proporciona dicho framework se encuentran los siguientes (Calheiros et al, 2010, cap. 4):

- **Cloudlet:** Esta clase modela las aplicaciones de servicio basadas en la nube como pueden ser envío de contenido, redes sociales, y flujo de trabajo empresarial.

- **Datacenter:** Esta clase modela el nucleo de los servicios en un nivel de infraestructura (hardware) que son ofrecidos por Cloud Providers (Amazon, Azure, App Engine). Estos son encapsulados en un conjunto de host que pueden ser homogeneos o heterogeneos con respecto a sus configuraciones de hardware (memoria, núcleos, capacidad, y almacenamiento).

- **DatacenterBroker:** Esta clase modela un broker, el cual es responsable de mediar las negociaciones entre el SaaS y los Cloud providers; y dichas negociaciones son manejadas por los requerimientos QoS.

Figura 2.1: Estilo de Trabajo de Cloudsim, Fuente: Chatterjee et al.



- **Host:** Esta clase modela los recursos fsicos como una computadora o un servidor de almacenamiento.

- **Vm:** Esta clase modela una Máquina Virtual (VM), la cual es adinistrada y hosteada por un componente host en la nube. Cada VM tiene acceso a un componente que almacena las siguientes caractersticas relacionadas a una VM: memoria accesible, procesador, tamaño de almacenamiento.

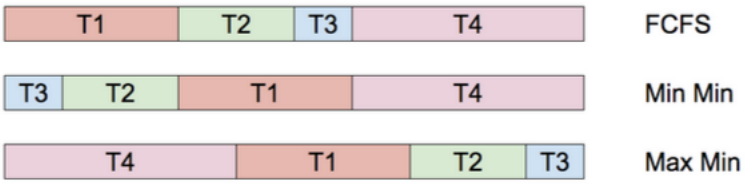
2.1.2. Implementación de los Algoritmos

Existen varios algoritmos para calendarizar los trabajos en el cómputo en la nube. La mayor ventaja de estos algoritmos es obtener el mayor rendimiento. Los principales ejemplos de algoritmos de calendarización son: FCFS, Round Robin, Min-Min, Max-Min y algoritmos de meta heursticas.(Shimpy y Jagandeep, 2014, p. 1).

De los algoritmos mencionados anteriormente en éste trabajo se presentan los siguientes:

- **FCFS:** Ejecuta las tareas en orden de llegada, es decir, el primero en llegar es el primero en ser atendido.

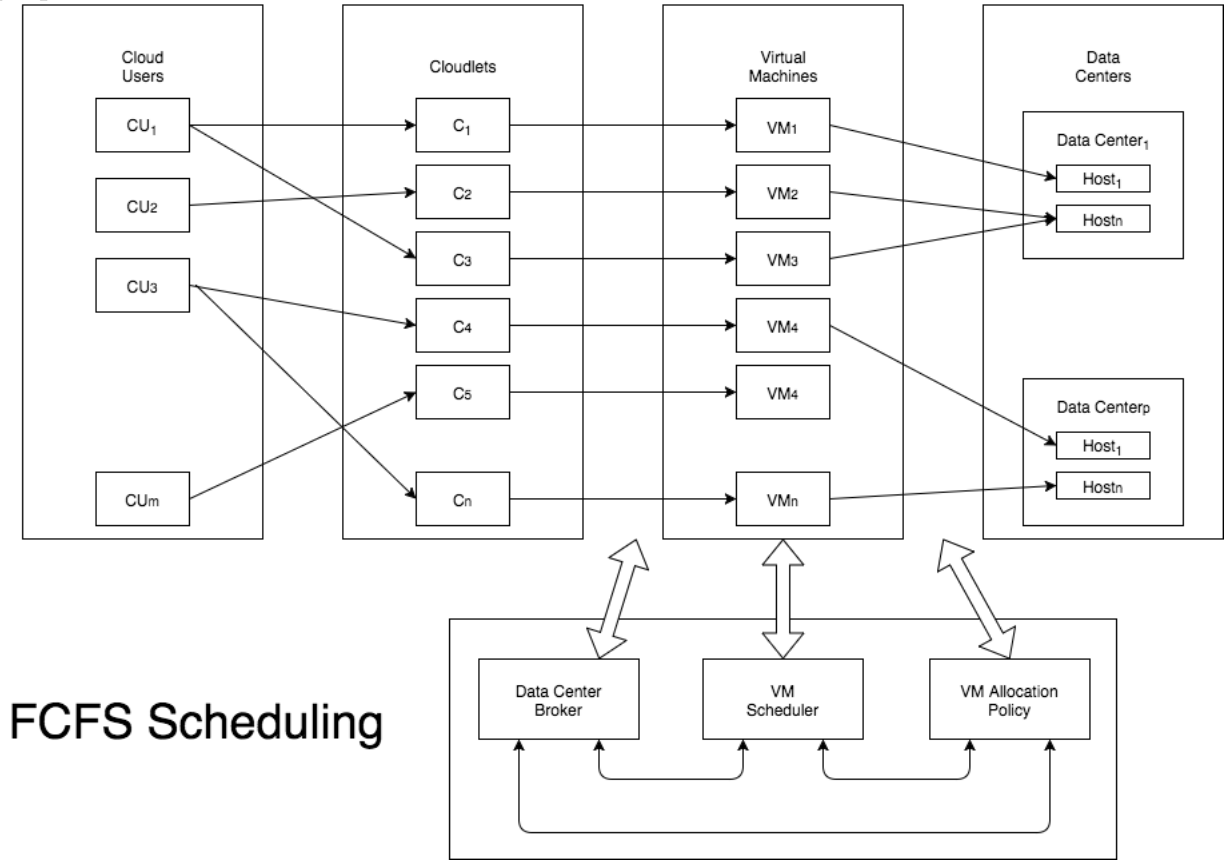
Figura 2.2: Esquema de trabajo de algoritmos de calendarizacin Fuente: Elaboracin propia.



- **Min-Min:** Selecciona las tareas más pequeas para ser ejecutadas primero.
- **Max-Min:** Selecciona las tareas más grandes para ser ejecutadas primero.

En un entorno de trabajo normal, la forma descrita anteriormente la podemos ver de la siguiente manera:

Figura 2.3: Arquitectura FCFS para un entorno en la nube. Fuente: Elaboracin propia.



Sin embargo, en un entorno en la nube al ser múltiples máquinas virtuales alojadas en distintos hosts que a su vez pueden formar parte de uno o ms datacenters dicho esquema tiene que ser adaptado para poder adoptar un estilo de trabajo similar al proporcionado por nuestro framework, por lo que los resultados quedan de la distinta manera:

Como podemos apreciar en el diagrama, podemos tener m usuarios ejecutando n tareas, sin embargo la asignación de máquinas virtuales va dependiendo del orden de llegada de dichas tareas, y quien se encarga de repartir las tareas es el datacenterBroker.

De manera similar tenemos los siguientes algoritmos min-min y max-min:

Figura 2.4: Arquitectura de Max-Min para un entorno en la nube. Fuente: Elaboracin propia.

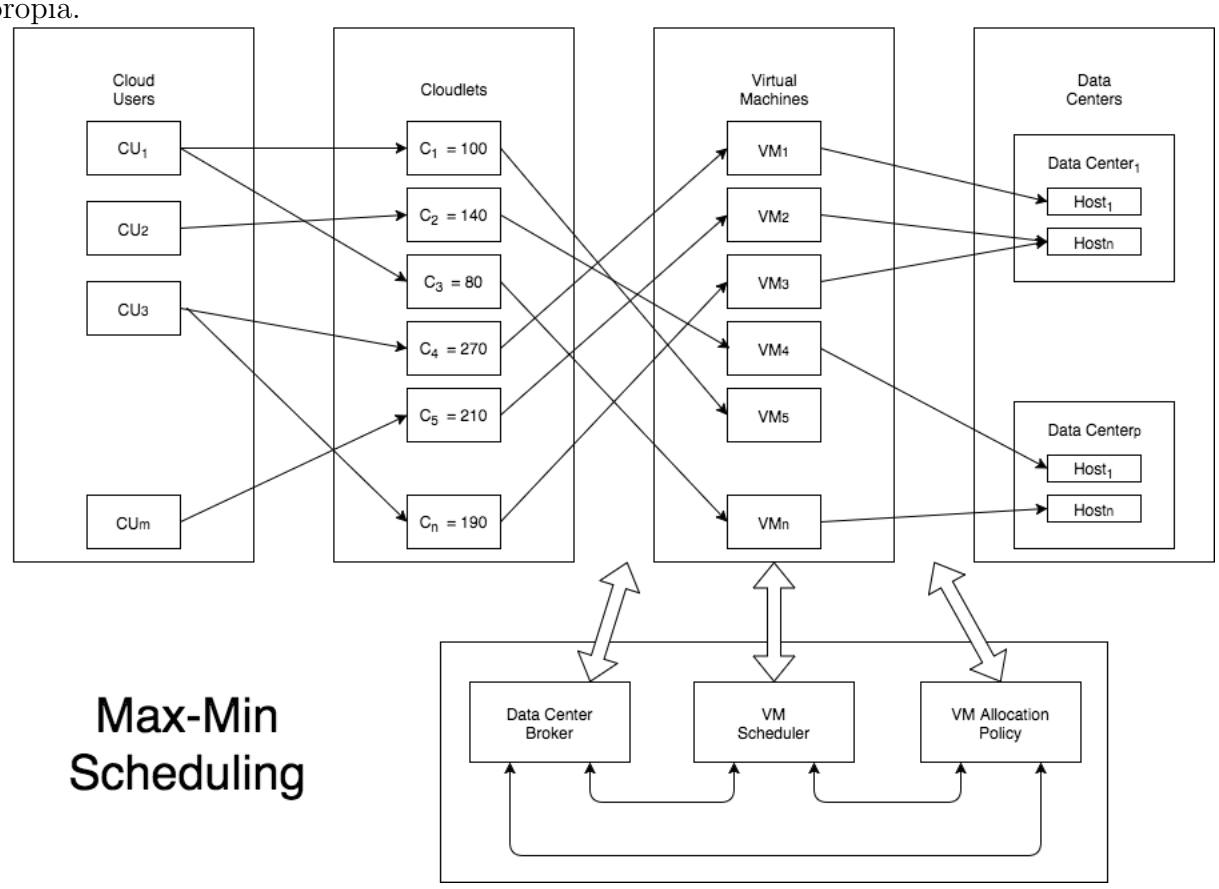
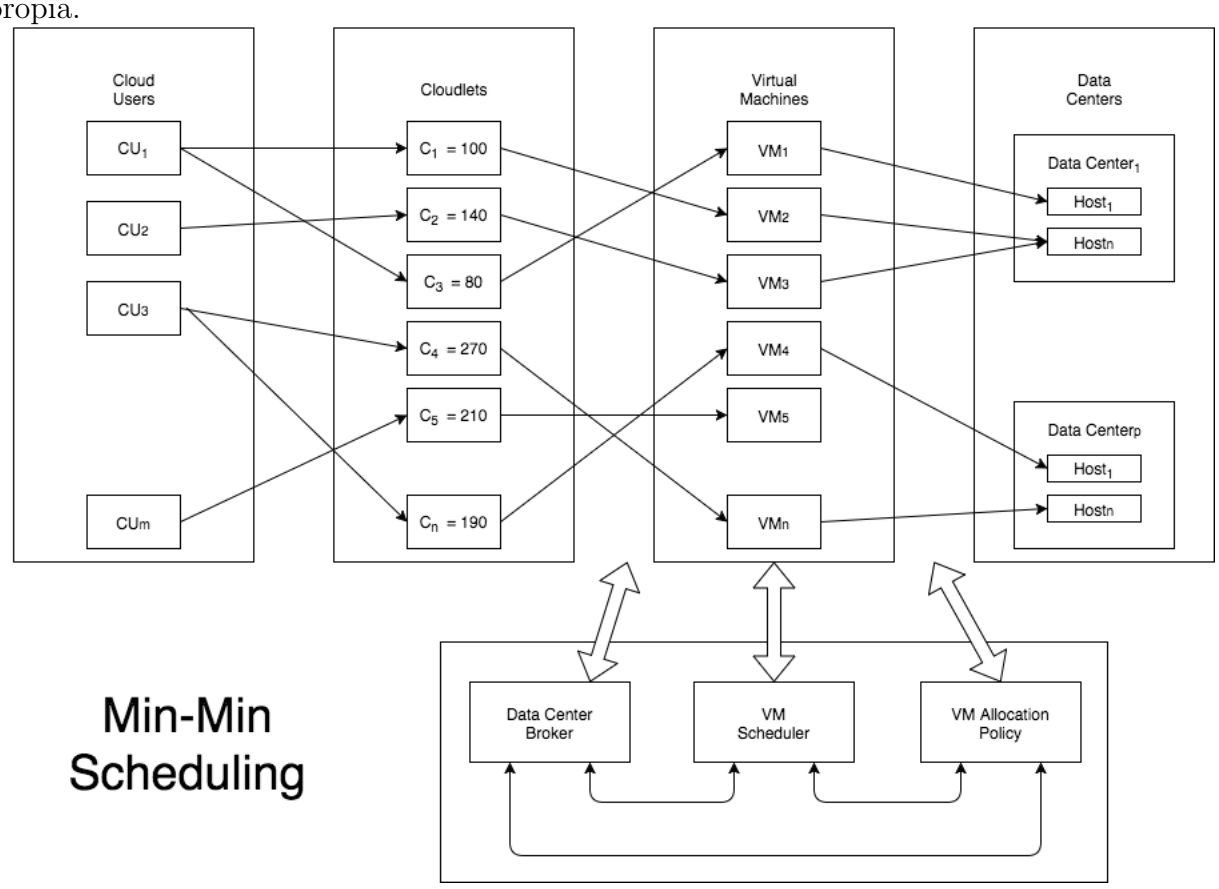


Figura 2.5: Arquitectura de Min-Min para un entorno en la nube. Fuente: Elaboracin propia.



Capítulo 3

Resultados

Introducción

En éste capítulo se describirán los resultados obtenidos en modelado y simulación del centro de datos en CloudSim. La experimentación fue desarrollada en una computadora con un procesador Intel Core i5 con la siguiente configuración: 2.5 GHz, 3 MB de caché, 4 GB de RAM , Linux Ubuntu 14.04 lts 64 bits como sistema operativo y JDK 8.6. Para evaluar los algoritmos de calendarización seleccionados se ha

implementado un entorno de cómputo en la nube que consiste en un datacenter,un broker, máquinas virtuales y host. Con el objetivo de evaluar el tiempo de ejecución y el costo de procesamiento en el datacenter tras ejecutar ciertos número de tareas (cloudlets).

Para la configuración del centro de datos en CloudSim se utilizaron diez host, cinco máquinas virtuales por host y las tareas fueron establecidas en un intervalo de 100 500 (tabla 3.1). Como se puede apreciar en la tabla 3.2, cada host tuvo 2048 Mb

Cuadro 3.1: Configuración Datacenter

Datacenter	
Host	10
VM	5
Cloudlet	100-500

Cuadro 3.2: Configuración de Host

Host	
RAM	2048 MB
CPU	2
Storage	800GB-1TB
BW	1 GB/s

Cuadro 3.3: My caption

VirtualMachine	
RAM	512 MB— 1GB
MIPS	250 — 500
Storage	10 GB
BW	1 GB/s

de memoria RAM, Dos núcleos de procesamiento, de almacenamiento de 800 GB o 1TB elegidas de manera aleatoria, el ancho de banda fue de 1GB/s.

En la tabla 3.3 se muestran los parámetros que se consideró para las máquinas virtuales, donde cada VM tendrá 10 GB de almacenamiento, 512 MB o 1 GB seleccionado de manera aleatoria, además para la característica del procesador se tiene la propiedad MIPS (million instructions per second) con 250 o 500 y un ancho de banda de 1000kb/s.

Para la configuración de las tareas, se tomó el tamaño con un intervalo de 1kb a 10kb, el parámetro fileSize que representa el tamaño del archivo de entrada va de 300b a 1.8kb así como el archivo de salida, como parámetro final se tiene los MIPS que irán de 1 a 3.

En la figura 3.1, se puede observar el promedio del tiempo de ejecución en ms para diferentes cantidades de tareas (de 100 a 500). A primera vista con el algoritmo FCFS y Min-Min se mantiene un tiempo de ejecución sin muchos cambios a pesar del aumento en la carga de tareas, a diferencia del algoritmo Max-Min que aumentó el tiempo de ejecución a medida que se incrementó el número de tareas.

El costo de procesamiento de acuerdo a cada algoritmo se puede observar en la figura 3.2, en donde el algoritmo FCFS tiene un incremento drástico al realizar la

Cuadro 3.4: Configuración Cloudlet

Cloudlet	
length	1kb-10kb
MIPSlength	300b-1.8kb
length	300b-1.8kb
length	1000 kb/s

Figura 3.1: Promedio tiempo de ejecución con tareas 100-500

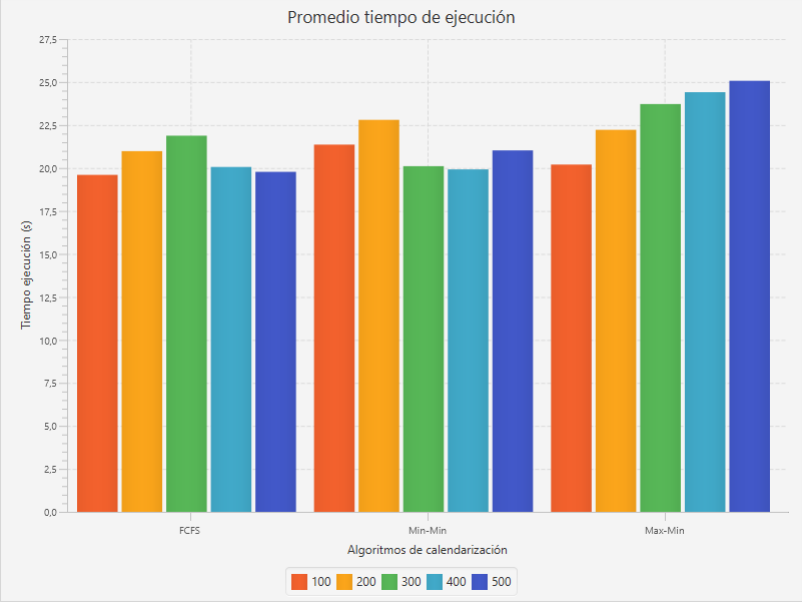
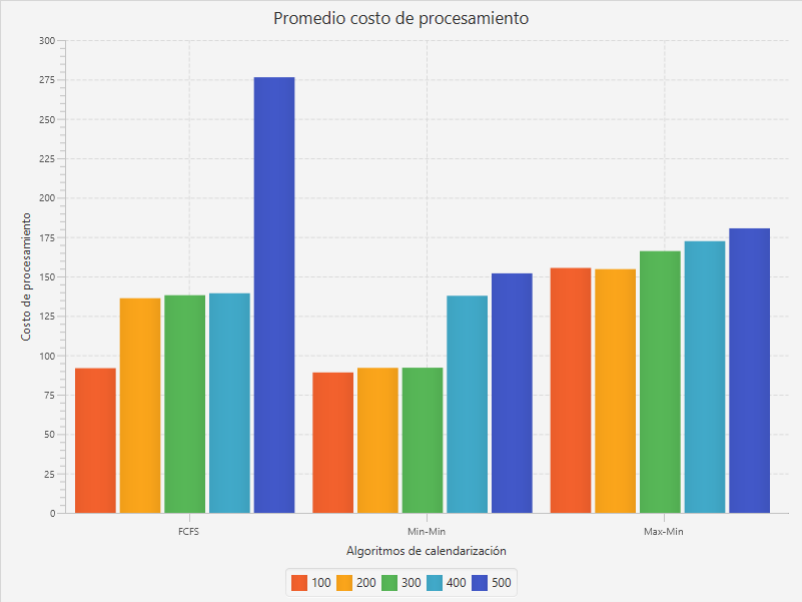


Figura 3.2: Promedio costo de procesamiento con tareas 100-500



prueba con 500 tareas. El algoritmo Max-Min se conservó sin muchos cambios a pasar de los cambios en la cantidad de tareas, mientras que Min-Min tiene un menor costo de procesamiento cuando las tareas son inferiores a 300.

Para mostrar el comportamiento del tiempo de ejecución por cada algoritmo, se tomaron 50 muestras de una simulación de 500 tareas. En la figura 3.3, se puede apreciar que el algoritmo FCFS tiene un comportamiento inestable ya que algunas tareas pueden tener menor complejidad o tamaño, lo que implica una respuesta rápida.

En la figura 3.4 se tiene la misma simulación pero con el algoritmo Max-Min, de acuerdo a las características de este calendarizador, en las primeras tareas se toma un mayor tiempo en responder y va disminuyendo de manera gradual, sin embargo aún es inestable en las últimas muestras ya que no se contempla el grado de complejidad es decir el parámetro MIPS de los cloudlets.

Por último tenemos el algoritmo Min-Min en el que el tiempo de ejecución fue aumentando conforme se resolvían las tareas (figura 3.6).

Figura 3.3: Tiempo ejecucin 50 muestras FCFS

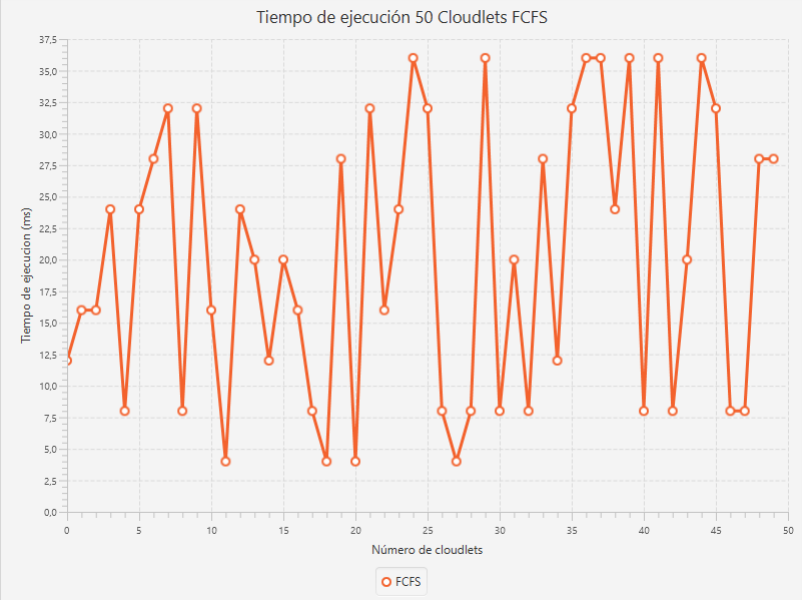


Figura 3.4: Tiempo ejecucin 50 muestras Max-Min

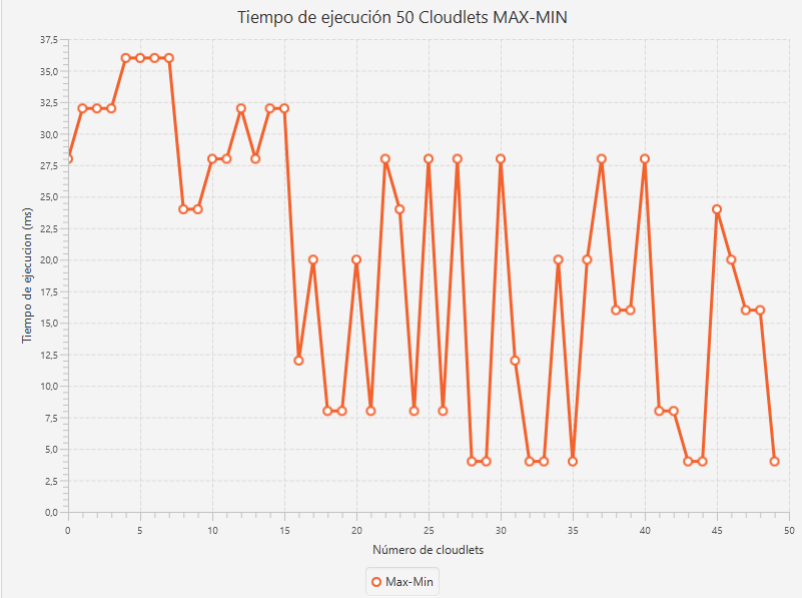
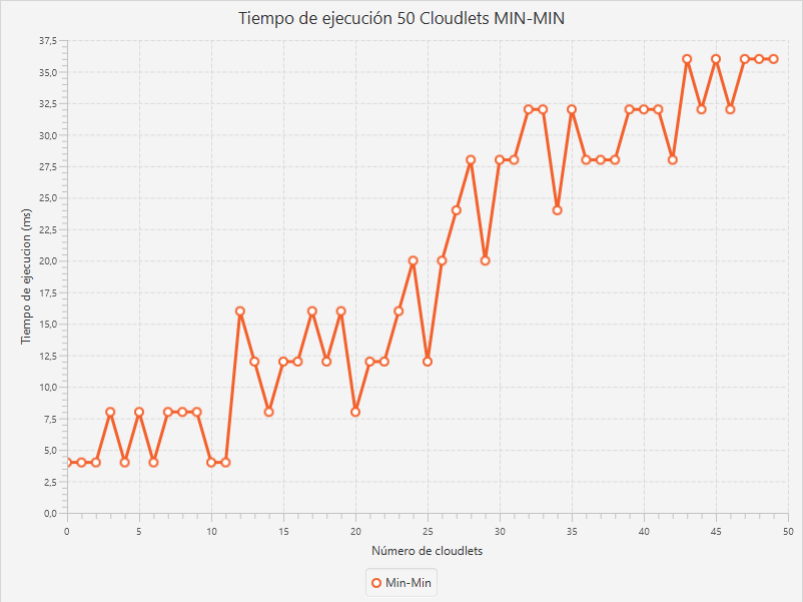


Figura 3.5: Tiempo ejecucin 50 muestras Min-Min



Cuadro 3.5: Desviación estándar del tiempo de ejecución

Algoritmo	**Desviación estándar**
FCFS	11.00619
MAX-MIN	8.91444
MIN-MIN	11.25613

Observando la desviación estándar de éstas muestras anteriores, el algoritmo Min-Min y FCFS tuvieron más variaciones en las muestras con respecto a la media, mientras que el Max-Min tuvo las variaciones por debajo de las dos anteriores(Cuadro 3.5).

Conclusiones

Recomendaciones

Bibliografía

[1] Pre-emptable shortest job next scheduling in private cloud computing in journal of information, knowledge and research computer engineering. En *Knowledge and research computer engineering*. 2015.

[2] Dr Agarwal, Saloni Jain, et al. Efficient optimal algorithm of task scheduling in cloud computing environment. *arXiv preprint arXiv:1404.2076*, 2014.

[3] Mohiuddin Ahmed, ASMR Chowdhury, Mustaq Ahmed, y Md Mahmudul Hasan Rafee. An advanced survey on cloud computing and state-of-the-art research issues. *IJCSI International Journal of Computer Science Issues*, 9(1):1694–0814, 2012.

[4] KM Aranganathan, S y Mehata. An aco algorithm for scheduling data intensive application with various qos requirements. *Int. J. Comput. Appl*, 27(10):10–5120, 2011.

[5] Huankai Chen, Frank Wang, Na Helian, y Gbola Akanmu. User-priority guided min-min scheduling algorithm for load balancing in cloud computing. En *Parallel Computing Technologies (PARCOMPTECH), 2013 National Conference on*, págs. 1–8. IEEE, 2013.

[6] Quan Chen y Qianni Deng. Cloud computing and its key techniques. *Journal of Computer Applications*, 29(9):2565, 2009.

[7] Dr M Dakshayini y Dr HS Guruprasad. An optimal model for priority based service scheduling policy for cloud computing environment. *International Journal of Computer Applications*, 32(9):23–29, 2011.

[8] Arash Ghorbannia Delavar, Mahdi Javanmard, Mehrdad Barzegar Shabestari, y Marjan Khosravi Talebi. Rsdc (reliable scheduling distributed in cloud computing). *International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol*, 2, 2012.

[9] Gordon S. Hildred T Dover, Z. The technical architecture of red hat enterprise virtualization environments. 2013.

[10] EPI.

[11] Ji Li, Longhua Feng, y Shenglong Fang. An greedy-based job scheduling algorithm in cloud computing. *Journal of Software*, 9(4):921–925, 2014.

[12] J Mariscal y JR Gil-Garcia. El cómputo en la nube en méxico: Alcances y desafíos para los sectores público y privado. cide. 2013.

[13] Saeed Parsa y Reza Entezari-Maleki. Rasa-a new grid task scheduling algorithm. *JDCTA*, 3(4):91–99, 2009.

[14] Pinal Salot. A survey of various scheduling algorithm in cloud computing environment. *IJRET: International Journal of Research in Engineering and Technology, ISSN*, págs. 2319–1163, 2013.

[15] Raúl Horacio Saroka. *Sistemas de información en la era digital*. Fundación OSDE, 2002.

[16] Er Shimpy y Mr Jagandeep Sidhu. Different scheduling algorithms in different cloud environment. *algorithms*, 3(9), 2014.

[17] S Srinivasan. Cloud computing evolution. En *Cloud Computing Basics*, págs. 1–16. Springer, 2014.

Glosario

A

- **Amazon:** Compañía de comercio electrónico y servicios de computación en la nube.

C

- **Cloud Computing:** cómputo en la nube, tiene como objetivo ofrecer servicios a través de Internet.
- **CloudSim:** Framework para el modelado y simulación de la infraestructura y los servicios de Cloud Computing.

E

- **ERP:** Planificador de recursos empresariales, son sistemas informáticos destinados a la administración de recursos de una organización.

P

- **Pruebas de software/hardware:** técnicas cuyo objetivo es evaluar la calidad del software/hardware.

Anexos