Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California

Doctorado en Ciencias en Ciencias de la Computación

Optimización de costos y calidad de servicio en sistemas de voz sobre IP basados en nubes computacionales

Tesis
para cubrir parcialmente los requisitos necesarios para obtener el grado de Doctor en Ciencias

Presenta:

Jorge Mario Cortés Mendoza

Ensenada, Baja California, México
2018

Tesis defendida por

Jorge Mario Cortés Mendoza

y aprobada por el siguiente Comité

Dr. Andrey Chernykh
Director de tesis

Miembros del comité

Dr. Carlos Alberto Brizuela Rodríguez

Dr. Jesús Favela Vara

Dr. Raúl Rivera Rodríguez

Dr. Ramin Yahyapour

Dr. Jesús Favela Vara
Coordinador del Posgrado en Ciencias de la Computación

Dra. Rufina Hernández Martínez
Directora de Estudios de Posgrado

Resumen de la tesis que presenta **Jorge Mario Cortés Mendoza** como requisito parcial para la obtención del grado de Doctor en Ciencias en Ciencias de la Computación.

Optimización de costos y calidad de servicio en sistemas de voz sobre IP basados en nubes computacionales

Resumen aprobado por:

_____

Dr. Andrey Chernykh
Director de tesis

La tecnología de voz sobre IP (VoIP) permite la comunicación de voz y/o datos a través de Internet de forma confiable y menos costosa que los sistemas tradicionales de telefonía. Esta solución posibilita una interconexión flexible entre organizaciones y empresas. Las soluciones de VoIP en la nube permiten a los proveedores ofrecer un servicio más económico debido a la infraestructura escalable de telefonía virtualizada. Los algoritmos de asignación y balanceo de carga son fundamentales en VoIP para el uso eficiente de la infraestructura. Sin embargo, las técnicas de asignación de llamadas tradicionales, basadas en el número de llamadas, no funcionan adecuadamente en infraestructuras dinámicas. Las características inherentes a los entornos nube limitan el funcionamiento de la mayoría de estas técnicas. En esta tesis, se propone un nuevo modelo de VoIP para ambientes nube y se aborda el problema de optimización bi-objetivo en la calendarización de estos servicios. La formulación del problema se basa en el caso especial de empaquetamiento dinámico en línea no clarividente y las soluciones consideran la optimización de la calidad del servicio y el costo de renta de la infraestructura virtual empleada. A diferencia de la formulación estándar del problema, los contenedores siempre están abiertos, incluso si están completamente empaquetados. Los elementos desaparecen de los contenedores después de la terminación de la llamada y la utilización puede cambiar en cualquier momento. Se evalúa el desempeño de veinte estrategias de asignación y tres modelos de predicción de llamadas mediante simulación con una carga de trabajo real de seis meses de la compañía MIXvoip. Los resultados muestran que el modelo propuesto es razonable y representativo de instalaciones y aplicaciones reales. Las estrategias de asignación pueden reducir el costo, incrementar la calidad de servicio o una combinación de ambos.

Palabras clave: balanceo de carga, asignación de llamadas, cómputo en la nube, voz sobre IP (VoIP), estimación de carga.

Abstract of the thesis presented by **Jorge Mario Cortés Mendoza** as a partial requirement to obtain the Doctor of Science degree in Computer Science.

Cost-Quality Optimization of Cloud-based Voice over IP

Abstract approved by:

_____

Dr. Andrey Chernykh
Thesis Director

Voice over Internet Protocol (VoIP) allows communication of voice and/or data over the Internet to be reliable and less expensive than traditional ISDN systems. This solution typically enables flexible interconnection between organizations and companies. Cloud VoIP solutions allow VoIP providers to offer a cheaper service due to the scalable virtualized telephone infrastructure. Scheduling and load balancing algorithms are fundamental to achieve an efficient use of the infrastructure. Due to the fact that virtual machine parameters are changing over time, traditional scheduling techniques based on number of calls do not adapt well to this dynamism. They do not take into account uncertainty in dynamic and unpredictable cloud environments. In this thesis, we address the problem of scheduling VoIP services in distributed cloud environments and propose a new model for bi-objective optimization. We consider the special case of the on-line nonclairvoyant dynamic bin-packing problem and discuss solutions for reducing provider cost and increasing the quality of service. Unlike the standard formulation of the problem, our bins are always open, even if they are completely packed. Items in bins can disappear after call termination, and utilization can be changed at any moment. We evaluate the performance of twenty call allocation strategies and three calls prediction models by comprehensive simulation analysis on real workload considering six months of the MIXvoip company service. The model is reasonable and representative for real installations and applications. The experimental results indicate that the proposed algorithms can be efficiently used in a VoIP cloud environment.

Keywords: load balancing, calls allocation, cloud computing, Voice over IP (VoIP), load estimation.

Dedicatoria

Esta tesis está dedicada a mis familiares y amigos.

Agradecimientos

Me gustaría expresar mi sincera gratitud al profesor Andrey Chernykh, mi director de tesis, por su constante orientación y apoyo. Sus consejos me han ayudado en todos los aspectos de mi vida.

A los miembros del comité de tesis, Dr. Carlos Alberto Brizuela Rodríguez, Dr. Jesús Favela Vara, Dr. Raúl Rivera Rodríguez y Dr. Ramin Yahyapour, por su generosa orientación, su colaboración en la investigación, sus comentarios constructivos, correcciones de tesis y retroalimentación sobre el trabajo de investigación.

A todos los estudiantes y personal de CICESE, en especial a los miembros del departamento de ciencias de la computación por su amistad y apoyo.

Al Centro de Investigación Científica y de Educación Superior de Ensenada (CICESE), la institución anfitriona de mis estudios, por proporcionar las instalaciones para realizar la investigación.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por brindar el apoyo económico para realizar mis estudios de doctorado.

Tabla de contenido

Lista de figuras

Lista de tablas

Capítulo 1. Introducción

En el modelo de cómputo en la nube, los recursos están disponibles para los usuarios dependiendo de sus necesidades. Como parte de los servicios flexibles, la nube proporciona una manera fácil de acceder a datos y archivos, especialmente para hacer que los datos estén disponibles para los usuarios en todo el mundo. El usuario solo necesita una computadora y acceso a Internet para poder usar la infraestructura nube. Este modelo ha sido ampliamente implementado por compañías como HP, IBM, Microsoft, Sun Microsystem, Google, Amazon, Oracle, entre otros proveedores prominentes.

En los últimos años, el cómputo en la nube ha crecido como un modelo de negocio factible. Diferentes empresas adoptaron este modelo como una solución rentable e innovadora (Amazon Web Services, 2017; Google Cloud Services, 2017). Las empresas están trasladando su infraestructura a la nube debido al ahorro en costo de infraestructura y la escalabilidad, ambos factores permiten a las compañías aumentar sus ganancias y brindar una calidad de servicio adecuada a los usuarios. Algunos ejemplos de migración de negocios son: Netflix en TV (Amazon Web Services, 2017), Spotify en radio (Google Cloud Services, 2017), Skype en VoIP (telefonía)[1], etc.

VoIP es un grupo de tecnologías que permite establecer llamadas a través de Internet mediante la transmisión de datos de voz por medio de una red IP. La tecnología de VoIP requiere menos equipo de hardware (reducción en los equipos dedicados de telefonía) y permite una conectividad más directa (menos puntos físicos de transferencia). Además, VoIP es una estrategia de servicio a largo plazo con mayor flexibilidad que la infraestructura telefónica tradicional. Se estima que para el 2020 la telefonía de VoIP generará ingresos de más de 80 mil millones de dólares y tendrá más de 200 millones de consumidores corporativos (Future Market Insights, 2017).

Bajo costo y altos niveles de calidad de servicio motivaron la adopción y el rápido crecimiento de la tecnología VoIP en la nube, donde los proveedores de VoIP encontraron un modelo de servicio fácil de implementar. Otros beneficios del modelo de VoIP basado en la nube (CVoIP) sobre los modelos de telefonía convencional y VoIP tradicional son la flexibilidad, escalabilidad, variedad de servicios extendidos, etc. El modelo de CVoIP adapta los recursos del sistema dinámicamente en respuesta a la demanda.

[1] https://support.skype.com/en/faq/FA12381/what-does-it-mean-that-skype-is-moving-from-peer-to-peer-to-the-cloud

Asterisk (Meggelen, 2013) es la columna vertebral en la mayoría de soluciones VoIP. Este software alimenta los sistemas IP de la central privada automática[2] (PBX). Las máquinas virtuales (VMs) ejecutan instancias de Asterisk y facilitan las llamadas, correos de voz, conferencias de audio/video, menús interactivos de telefonía, distribución de llamadas, etc. Además, los usuarios pueden transferir imágenes, textos y agregar nuevas funcionalidades, abriendo una experiencia completamente nueva en la comunicación telefónica convencional.

El éxito del modelo de CVoIP depende significativamente de la calidad y el costo del servicio. La calidad de servicio (QoS) es el primer aspecto a considerar porque VoIP tiene factores sensibles e impone restricciones importantes. De acuerdo con la Unión Internacional de Telecomunicaciones (UIT), la QoS se define como: "La totalidad de las características de un servicio de telecomunicaciones que determinan su capacidad para satisfacer las necesidades explícitas e implícitas del usuario del servicio"(UIT-T, 2008). La QoS en VoIP está determinada por dos factores: el procesamiento de llamadas y la transmisión de llamadas.

El aspecto más importante en la transmisión de llamadas es el envío adecuado de voz a través de la red. El tiempo de tránsito de los paquetes, el retardo en la cola de los enrutadores, el tiempo de envío desde el origen al destino, la variación de retardo en los paquetes (jitter) y la pérdida de paquetes sirven para evaluar la calidad en la transmisión de llamadas. Una calidad de voz adecuada es el aspecto más importante en el procesamiento de la llamada, la evaluación de la calidad incluye: el tiempo para inicializar y finalizar la llamada, y el tiempo en convertir la porción de voz de las llamadas en paquetes transportados a través de la red.

El puntaje de opinión medio (MOS) es una medida usada para evaluar la calidad del habla proporcionada por un códec. El códec realiza la conversión de la voz analógica entrante en un flujo digital y convierte ese nuevo flujo digital a un patrón de voz analógica en el destino final. Cada códec proporciona una adecuada calidad de voz si y solo si la utilización del procesador es lo suficientemente baja. Teóricamente, la utilización del 100% de procesador proporciona el mejor rendimiento, sin embargo, cuando la utilización del CPU es mayor al 85% entonces aparece el fenómeno de jitter y audio cortado (Eleftheriou, 2010).

Elementos inherentes a los entornos nube afectan la QoS de las llamadas, por ejemplo: el tiempo para desplegar la infraestructura virtual del sistema CVoIP, el cifrado del tráfico, la distribución geográfica

[2] Una central privada automática es cualquier central telefónica conectada directamente a la red pública de telefonía por medio de líneas troncales para gestionar llamadas internas, entrantes y salientes con autonomía sobre cualquier otra central telefónica.

de los recursos, etc. Estos factores deben ser estudiados para minimizar sus efectos en la calidad de voz y garantizar el correcto procesamiento de las llamadas.

El costo del servicio es el segundo aspecto a considerar para el éxito del modelo CVoIP. El gasto operacional de una empresa es un elemento importante para determinar el costo de un servicio. Los proveedores de CVoIP deben reducir la renta de la infraestructura y maximizar el uso de los recursos para ofrecer precios competitivos a los clientes potenciales. El manejo ineficiente de los recursos tiene un efecto negativo directo sobre el rendimiento y el costo. El alquiler de recursos en la nube es un parámetro usado para estimar el precio del servicio, este permite medir el costo del sistema en términos de demanda de recursos y el tiempo de uso.

Los proveedores de CVoIP pueden ofrecer un servicio económico y adecuado al usuario cuando:

1. La renta de recursos en la nube para proporcionar el servicio de VoIP es lo más baja posible.
2. La utilización de los recursos es suficientemente baja para garantizar un adecuado procesamiento de las llamadas.

Ambos objetivos están en conflicto, el proveedor debe maximizar la utilización de los recursos para reducir el costo de la infraestructura, pero esta situación degrada la QoS. Por otro lado, la reducción de la utilización en los recursos puede garantizar la QoS pero aumenta el costo de renta en infraestructura.

Los proveedores de VoIP usan técnicas de balanceo de carga para alcanzar una alta calidad de servicio. Desafortunadamente, las técnicas actuales de balanceo de carga no son eficientes en sistemas a gran escala como los ambientes de cómputo en la nube (Alakeel, 2010; Tchernykh et al., 2016). En los últimos años, diferentes algoritmos se han introducido para superar los desafíos de distribución de carga en CVoIP (Cheng et al., 2015; Montazerolghaem et al., 2016). Sin embargo, el problema de balanceo de llamadas en CVoIP con calidad de servicio y optimización del costo del proveedor no ha sido estudiado lo suficiente.

Esta tesis se enfoca en el desarrollo de un modelo CVoIP y estrategias de asignación de llamadas. El objetivo principal es asignar llamadas de manera eficiente en el sistema de CVoIP para alcanzar una alta calidad de servicio y reducir la renta de recursos.

La siguiente sección presenta diferentes aspectos de entornos VoIP y destaca las oportunidades y desafíos en el campo de CVoIP.

1.1 Telefonía por Internet

La telefonía por Internet (VoIP) proporciona servicios de comunicación a través del Internet público, en lugar de una red telefónica tradicional. El Instituto nacional de estándares y tecnología de Estados Unidos (NIST) define Voz sobre IP como "VoIP - la transmisión de voz en paquetes conmutados sobre redes IP - es una de las tendencias emergentes más importantes en telecomunicaciones". El documento describe las ventajas de esta tecnología "esta forma de transmisión es conceptualmente superior a la comunicación por conmutación de circuitos convencional en muchos sentidos", y las expectativas de las compañías para VoIP "menor costo y mayor flexibilidad se encuentran entre las promesas de VoIP para la empresa" (Walsh y Kuhn, 2005). Finalmente, en el documento se puntualizan varios componentes de redes VoIP con impacto directo en la calidad de voz (administradores de llamadas, puertas de enlace, enrutadores, cortafuegos y protocolos).

VoIP es el último paso evolutivo en la transmisión de voz a larga distancia. Básicamente, VoIP consiste en la codificación de voz, en forma digital, para ser transmitida a través del Internet, el extremo receptor decodifica la señal para que el destinatario pueda escuchar la voz del emisor. Otros elementos necesarios para establecer la conexión de VoIP incluyen: protocolo de inicio de sesión (SIP), protocolos de voz y video, y servicios actualmente en uso en el Internet. Los servicios de VoIP reducen significativamente las tarifas de llamadas comparados con los servicios telefónicos convencionales. Los principales proveedores de VoIP continúan una feroz competencia por ofrecer servicios adicionales utilizando la tecnología nube. Las ventajas del servicio de VoIP sobre la red telefónica tradicional son:

Costo. El servicio telefónico de VoIP es más económico que un servicio de telefonía convencional. Por ejemplo, llamadas de larga distancia y la conexión con un teléfono fijo o celular es más económica, llamadas con otros teléfonos VoIP son gratuitas, los usuarios pueden tener más de un número de teléfono en diferentes códigos de área, etc.

Portabilidad. Los usuarios reciben y realizan llamadas telefónicas en cualquier lugar donde haya una conexión de banda ancha disponible, simplemente inician sesión en su cuenta de VoIP. El sistema telefónico de VoIP está disponible en todo el mundo, similar a los sistemas de correo electrónico.

Flexibilidad. Un convertidor de VoIP o un adaptador de teléfono VoIP permiten al usuario usar teléfonos convencionales en sistemas de VoIP. Los usuarios pueden adquirir números VoIP para recibir llamadas telefónicas al mismo número en todo el mundo.

Características adicionales. Los sistemas de VoIP agregan funciones avanzadas de comunicación. Algunos servicios adicionales en telefonía VoIP sin costo comprenden: desvío de llamadas, llamadas en espera, correo de voz, identificador de llamadas y llamadas en conferencia.

Las empresas de VoIP necesitan implementar la infraestructura VoIP en diferentes ubicaciones para ofrecer el servicio. En los últimos años, cuantiosos proveedores de VoIP están migrando su infraestructura de VoIP al modelo nube debido a diversos beneficios. La siguiente sección detalla el modelo tradicional de VoIP y la adopción de la infraestructura nube en el modelo de VoIP.

1.1.1 Voz sobre el protocolo de Internet (VoIP)

La arquitectura general del modelo de VoIP contiene elementos de infraestructura en comunicación que conectan teléfonos de forma remota a través de Internet. Los servidores emplean Asterisk para emular una central telefónica y proporcionan la infraestructura necesaria (puertas de enlace, conmutadores de interconexión, controladores de sesión, cortafuegos, etc.).

Asterisk es el marco de trabajo gratuito y de código abierto más popular para el desarrollo de aplicaciones de comunicación. Asterisk ofrece todas las funciones de una PBX: permite llamadas entre teléfonos registrados y la conexión con otros servicios telefónicos, como la red telefónica pública conmutada (PSTN) y otros servicios de VoIP. Este software de PBX soporta varios protocolos standard de VoIP y funciona con casi todos los equipos de telefonía que utilizan hardware de bajo costo. Asterisk permite construir sistemas telefónicos o migrar sistemas existentes a nuevas tecnologías.

La Figura 1 muestra la arquitectura general del modelo VoIP. Los elementos de la infraestructura de comunicación están conectados a Internet y enlazan teléfonos de forma remota a través de Internet.

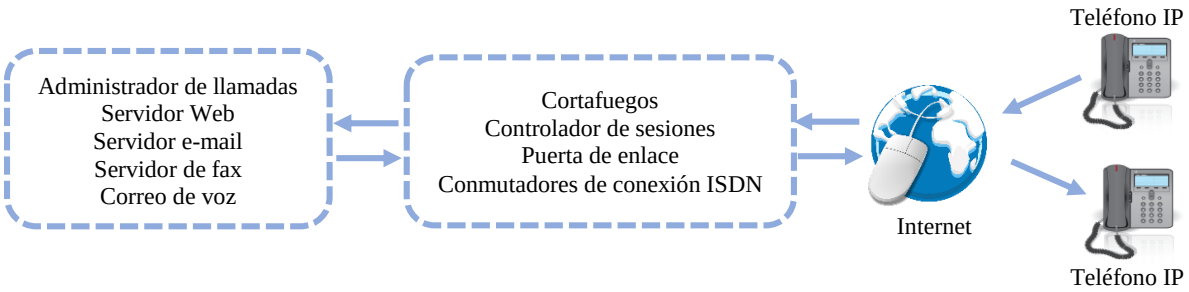

Figura 1. Arquitectura de VoIP.

Una desventaja de esta arquitectura surge cuando la infraestructura alcanza su máxima capacidad. Las soluciones tradicionales de VoIP son difíciles de escalar, por lo que, incrementar la infraestructura

implica frecuentemente reemplazar todo el hardware existente. El aprovisionamiento excesivo y, por lo tanto, el incremento en costos no es una solución eficiente, incluso con el creciente número de clientes y la seguridad de poder prestar servicio durante horas pico o en comportamientos anormales del sistema. La disponibilidad del servicio de VoIP es fundamental en todo momento y para cualquier número de usuarios.

Los proveedores invierten en infraestructuras grandes para lidiar con un número creciente de usuarios y evitar la pérdida de llamadas (y por lo tanto, de clientes). En este caso, la infraestructura generalmente está subutilizada. De forma similar, la degradación de los recursos (obsolescencia tecnológica) provoca un reemplazo periódico de los servidores.

1.1.2 Modelo híbrido de voz sobre IP (HVoIP)

La compañía MiXvoip (MIXvoip, 2017), un proveedor telefónico de voz IP, desarrolló los conceptos de súper nodo de voz (SN) y agrupamiento de súper nodos (SNC) para enriquecer las características de las centrales telefónicas. Los SNs agrupan nodos de voz que proporcionan la funcionalidad del sistema telefónico (correo de voz, transferencia de llamadas, música en espera, función de conferencia, etc.).

La Figura 2 ilustra los elementos constitutivos (capas) de los SNs. Las principales funciones de las capas son: *Sistema Operativo* (SO) ejecuta el software de VoIP (Asterisk) y funciona principalmente en distribuciones GNU/Linux pero otros SO pueden ser usados. La capa de seguridad incluye el acceso al servidor por medio de sesiones SSH, FTP y otros; así como el acceso de los usuarios al teléfono a través de su nombre de usuario y contraseña. La capa de monitoreo verifica el funcionamiento de los recursos e informa el estado de los nodos de voz. La arquitectura distribuida de los SNs reparte el procesamiento entre varias entidades, reduce el riesgo de falla y limita la adquisición de equipos.

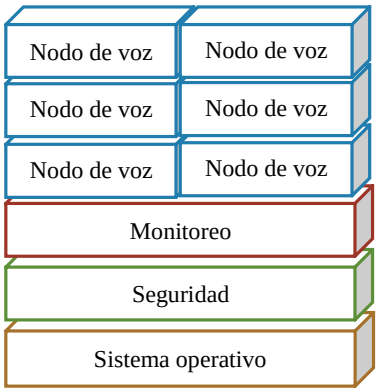

Figura 2. Arquitectura del súper nodo de voz.

Un SNC es un conjunto de SNs desplegados en la nube e interconectados lógicamente a nivel local. Los SNCs minimizan el camino entre dos usuarios locales, por lo tanto, aumentan la calidad de voz. Esta implementación brinda redundancia en un área geográfica determinada y garantiza una alta calidad de voz entre los SNCs a través del Internet público. Los SNCs se despliegan en ubicaciones geográficas ampliamente distribuidas.

La Figura 3 muestra un ejemplo de conectividad entre dos usuarios en el sistema de VoIP basado en SNCs. Cuando un usuario en el área 1 desea establecer una llamada, el SN más cercano al área 1 atiende la petición. Si los clientes son pequeñas empresas, las llamadas son realizadas a nivel local o en países vecinos. El despliegue de esta arquitectura brinda calidad servicio cercana a la proporcionada por la red digital de servicios integrados (ISDN).

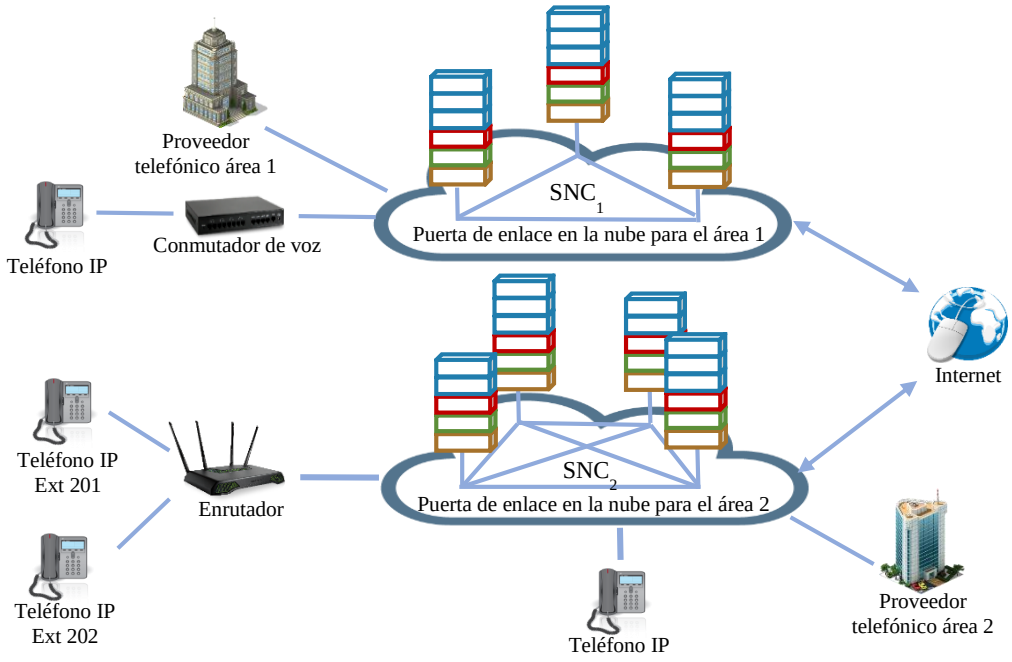

Figure 3. Despliegue de agrupamiento de súper nodos.

Internet proporciona la interconexión del sistema con otros operadores o una conexión física (por cable) conecta dos dispositivos en un centro de datos, donde el operador y la infraestructura se encuentran a corta distancia.

La Figura 4 muestra el modelo tradicional e híbrido de VoIP. En la solución tradicional de VoIP, el proveedor de servicio es propietario de la infraestructura. Desafortunadamente, este modelo tiene varios inconvenientes: requiere implementación de servidores físicos en el área, conflictos de hardware entre la infraestructura central y la estación de trabajo del usuario, y es un modelo difícil de escalar.

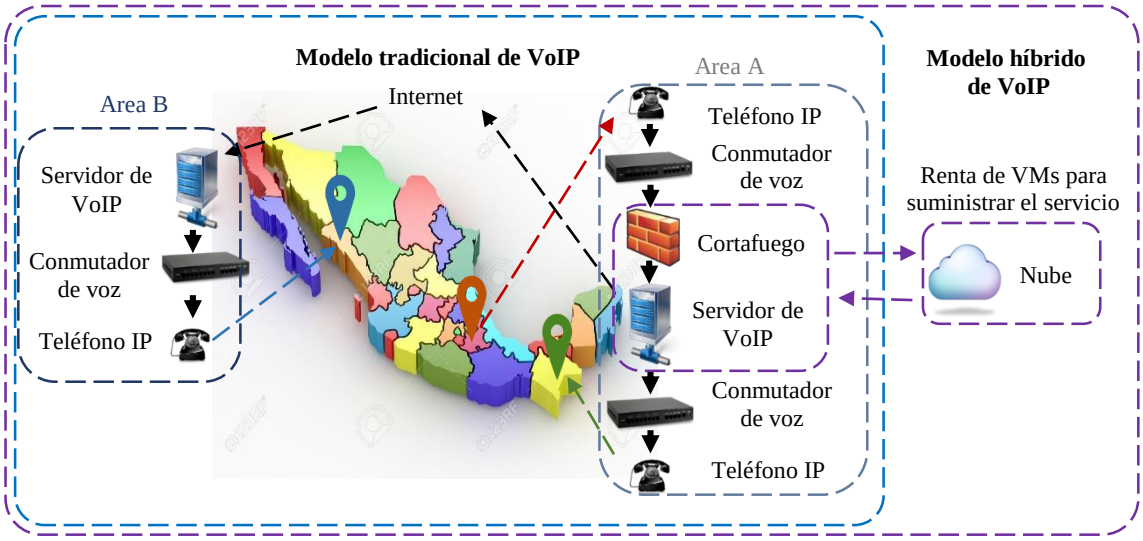

Figura 4. Modelo tradicional e híbrido de VoIP.

El modelo híbrido de VoIP (HVoIP) soluciona varios problemas del modelo tradicional. HVoIP utiliza VMs para proporcionar el servicio VoIP cuando la infraestructura del proveedor alcanza su máxima capacidad. El sistema ejecuta instancias de SNs en las VMs desplegadas en varios proveedores de nube mediante el modelo IaaS.

Las ventajas del modelo HVoIP son: mejora la escalabilidad del sistema de VoIP, reduce el aprovisionamiento excesivo de recursos, incrementa la granularidad y minimiza el costo en infraestructura. Desafortunadamente, HVoIP es un modelo costoso porque los proveedores aún necesitan poseer infraestructura propia para proporcionar el servicio y alquilar oficinas y espacio. La cantidad de recursos ociosos pueden aumentar, el proveedor debe maximizar el uso de los recursos propios y reducir la cantidad de recursos alquilados. Un modelo basado completamente en la nube puede resolver los inconvenientes del modelo HVoIP.

1.1.3 Modelo de voz sobre IP basado en la nube (CVoIP)

En el modelo de VoIP basado en la nube (CVoIP), toda la infraestructura se renta en nubes locales, lo cual permite a los proveedores reducir costos. CVoIP incrementa la redundancia en diferentes áreas geográficas y garantiza una alta calidad de voz entre los SNs. Las VMs desplegadas en diversos proveedores de nube ejecutan instancias de SNs, de forma similar a HVoIP. La Figura 5 muestra el modelo CVoIP, este modelo soluciona la mayoría de los inconvenientes del modelo tradicional e híbrido de VoIP.

Adicionalmente, CVoIP agrega nuevas funciones y capacidades, proporciona una implementación más sencilla e integra servicios que son dinámicamente escalables. Otros beneficios incluyen: la disponibilidad de transferencia de datos, la integridad y la seguridad.

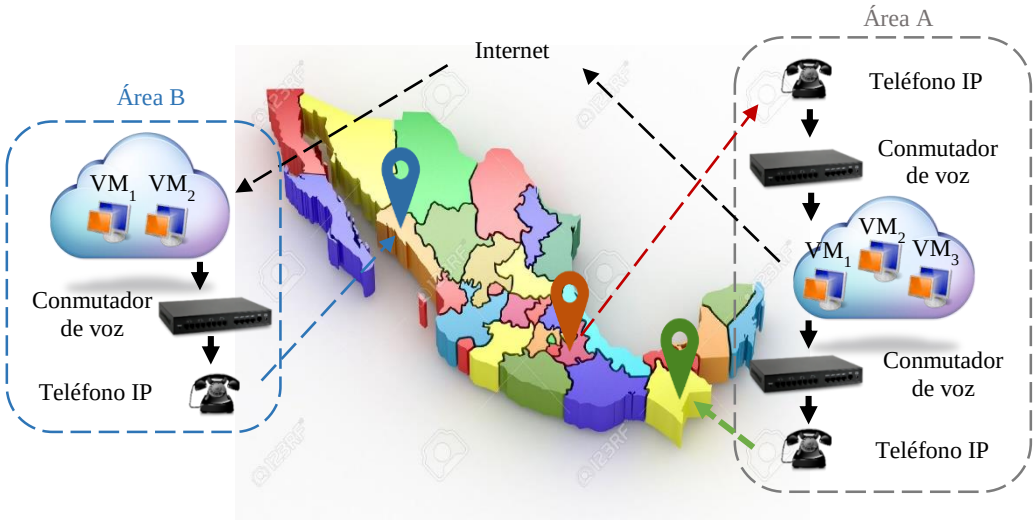

Figura 5. Modelo de VoIP basado en la nube.

La Figura 6 muestra una solución de VoIP basada en la nube. Los nodos de voz ofrecen una variedad de servicios, por ejemplo: transferencia de llamadas, correo de voz, función de conferencia, música en espera, etc. Las ventajas de esta arquitectura son: el incremento de la escalabilidad y la distribución de VMs en varios procesadores.

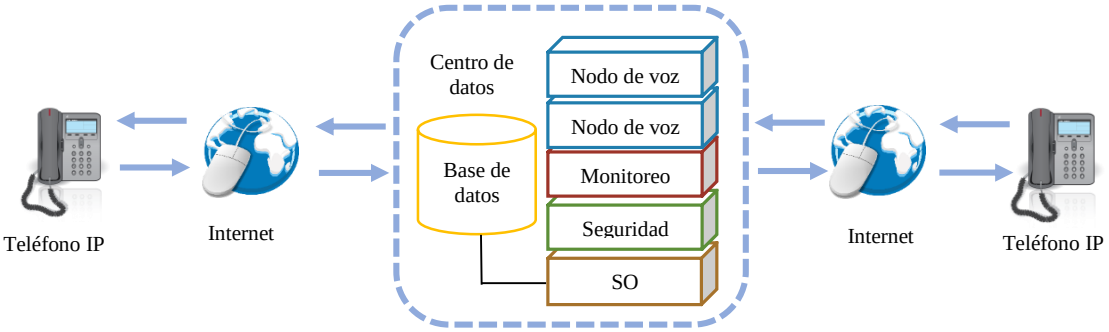

Figura 6. Arquitectura de VoIP basada en la nube.

Desde la perspectiva del cliente, una conexión a Internet y un teléfono IP o un teléfono basado en software (softphone) son suficientes para usar los servicios de VoIP en la nube. En el modelo de CVoIP, los centros de datos en la nube hospedan las VMs que alojan los servidores de voz. Los nodos de voz establecen la interconexión entre los usuarios y mantienen comunicación constante con la base de datos del sistema, donde todos los usuarios están registrados con nombre y contraseña. El sistema almacena detalles sobre cada llamada realizada como: origen, destino, duración, etc.

La Figura 7 presenta el siguiente paso en la arquitectura distribuida de CVoIP, donde los centros de datos geográficamente distribuidos agrupan SNs, esta infraestructura es altamente escalable. Las ventajas de los SNs en CVoIP son:

Granularidad. Los SNs reducen los conflictos entre las estaciones de trabajo de los usuarios y la infraestructura central, por ejemplo, cuando se realiza una actualización en la central, y tienen una compatibilidad total con los fabricantes de teléfonos y las centrales telefónicas (administran múltiples versiones de teléfonos por software).

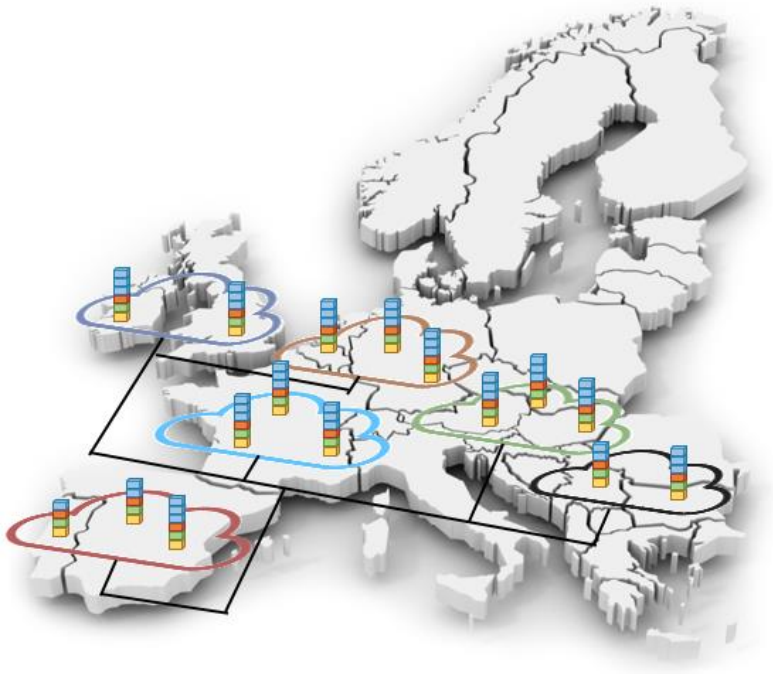

Figura 7. Agrupamiento de SNs en centros de datos.

Escalabilidad. En caso de saturación, la infraestructura aumenta el número de recursos (SNs) sin alterar la arquitectura existente o afectar el servicio proporcionado al cliente. De forma similar, los recursos disminuyen cuando están ociosos.

Distribución. Los proveedores no instalan servidores o equipos físicos en otras áreas geográficas para desplegar los SNs, solo se necesita alquilar infraestructura en nubes locales.

Tolerancia a fallos. La distribución de los SNs incrementa la tolerancia a fallos, el modelo de CVoIP detecta SNs con problemas y redirecciona automáticamente el tráfico a instancias sanas hasta que los SNs alcancen un estado saludable. El proveedor de CVoIP puede rentar infraestructura en diferentes nubes locales (dentro de una misma zona) para reducir las fallas con algún proveedor de nube.

Reducción de costos. El uso de sistemas CVoIP reduce el costo en infraestructura, los proveedores pagan solo por los recursos que utilizan. La renta de infraestructura elimina los gastos de inversión en equipos y espacio físico (oficinas, centros de datos, almacenes, etc.).

La arquitectura de CVoIP supone una distribución geográfica de los SNs, por lo tanto, los recursos están agrupados en centros de datos con diferentes ubicaciones. Los proveedores deben mejorar diferentes características del sistema para implementar y administrar eficientemente los teléfonos a través de la nube, donde lo más importante es la utilización de la infraestructura y la QoS.

Para mejorar el rendimiento general del sistema CVoIP, los nodos de voz deben balancear el procesamiento de las llamadas (señal de voz) para evitar la sobrecarga de los procesadores y, por lo tanto, la reducción de la calidad de voz. Un problema similar ocurre con la capacidad de la red. El sistema debe evitar el tiempo ocioso de los procesadores porque aumenta los gastos en renta de la infraestructura.

1.2 Justificación

Una asignación adecuada de llamadas maximiza el rendimiento de los sistemas de CVoIP al reducir el número de recursos activos y mantener el tiempo ocioso de los recursos lo más bajo posible. Para minimizar la reducción de calidad, los recursos deben mantener la cantidad de trabajo computacional por debajo del umbral que garantiza la QoS. Solo el uso de técnicas eficientes de balanceo de carga permitirá explotar al máximo las ventajas de la tecnología CVoIP.

En esta investigación, se propone un modelo de VoIP para el cómputo en la nube y se diseñan estrategias de asignación de llamadas para mejorar el funcionamiento del sistema. Las estrategias de asignación consideran varios parámetros para adaptarse a la incertidumbre en la llegada de los trabajos y las afectaciones.

En la actualidad, el modelo de CVoIP requiere nuevas técnicas para escalar en la Federación de nubes donde los centros de datos cuentan con miles de procesadores (Figura 8). Estas técnicas deben ser conscientes de los recursos debido al uso creciente de arquitecturas heterogéneas de cómputo en los centros de datos. Los algoritmos de balanceo existentes deben ser mejorados para funcionar adecuadamente en arquitecturas de CVoIP más complejas.

La causa más importante de desequilibrio de carga en VoIP es la naturaleza dinámica del problema a lo largo del tiempo. Otros factores a tener en cuenta son: el tiempo de llegada de las llamadas, la variabilidad de la utilización de los procesadores, los retardos de inicio en los recursos, el códec de la llamada, etc.

Figura 8. Federación de nubes.

La mayoría de los algoritmos de balanceo consideran entornos deterministas suponiendo el conocimiento de los trabajos del usuario y los parámetros del sistema. El entorno dinámico de la nube añade una considerable incertidumbre en el proceso de balanceo. El tiempo de ejecución de las tareas, la cantidad de trabajo, la cantidad de datos a gestionar, entre otros, son factores desconocidos.

En general, la técnica de virtualización dificulta obtener un conocimiento exacto del sistema. Los parámetros como la velocidad del procesador, la cantidad de VMs disponibles, los retardos de inicio y el ancho de banda cambian constantemente en el tiempo. Sin embargo, los algoritmos de balanceo deberían buscar la manera de mejorar el uso de los recursos y garantizar la QoS.

1.3 Objetivos de la investigación

1.3.1 Objetivo general

Diseñar e implementar un modelo de voz sobre IP para ambientes de cómputo en la nube. El modelo incluye el desarrollo de estrategias dinámicas de asignación de llamadas para la optimización multi-objetivo del sistema.

1.3.2 Objetivos específicos

1. Analizar las características de diferentes entornos de VoIP y nube.

2. Definir un modelo matemático para representar un sistema de voz sobre IP para ambientes de cómputo en la nube.

3. Diseñar estrategias de asignación de recursos multi-objetivo para entornos voz IP en la nube.

4. Definir estrategias de asignación de recursos para reducir el costo en renta de infraestructura en sistemas de VoIP basados en la nube.

5. Establecer un umbral de utilización para garantizar la calidad de servicio en el procesamiento de llamadas.

6. Analizar el desempeño de las estrategias propuestas con diferentes configuraciones y cargas de trabajo.

7. Comparar el rendimiento de las estrategias propuestas con estrategias empleadas en sistemas VoIP.

8. Investigar los efectos de la escalabilidad en las estrategias de asignación de llamadas.

9. Estudiar la degradación de la calidad del servicio debido a los retardos de inicio en las máquinas virtuales.

10. Proponer modelos de estimación de llamadas para apoyar el proceso de asignación de recursos.

1.4 Propuesta de solución

La metodología empleada para abordar el problema de diseño de un modelo de voz sobre IP basado en la nube y las estrategias de asignación se describe en la Figura 9.

Inicialmente, se analizaron diferentes modelos de VoIP y ambientes nube para incorporar las características más representativas en el sistema de VoIP basado en la nube. Nuestra propuesta de CVoIP modela el sistema a partir del problema de empaquetamiento dinámico de contenedores donde las máquinas virtuales representan los contenedores y las tareas caracterizan los elementos a empaquetar. El procesamiento de llamadas define la calidad de servicio, donde la saturación y la falta de recursos degradan dicha calidad.

Posteriormente, se estudiaron estrategias de asignación y balanceo de carga en diferentes ambientes de cómputo con el objetivo de proponer estrategias para el problema de asignación de llamadas en CVoIP. Las estrategias de asignación consideran varios parámetros para distribuir las llamadas entre los recursos, por ejemplo: número de llamadas, utilización de la VMs y tiempo de renta de las VMs.

Además de las estrategias de asignación, se desarrollaron tres modelos de predicción de llamadas para reducir los efectos del retardo en el aprovisionamiento de los recursos en la nube.

Finalmente, se evaluó el desempeño de las estrategias de asignación de llamadas bajo dos ambientes. En el primer ambiente, limitar la utilización de las VMs garantiza la máxima calidad de servicio en el sistema, por lo tanto, la renta de infraestructura determina la eficiencia de las estrategias de asignación de llamadas. En el segundo ambiente, no se puede garantizar la cálida de servicio debido a los retardos en el tiempo de aprovisionamiento de los recursos. La renta en infraestructura y la degradación de la calidad de servicio se emplean para evaluar el desempeño de las estrategias de asignación.

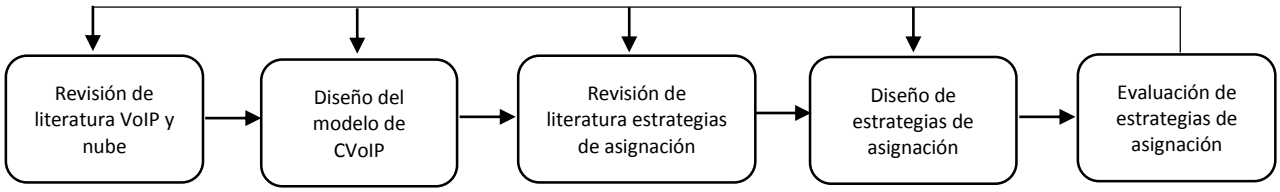

Figura 9. Metodología general propuesta.

1.5 Estructura de la tesis

El presente trabajo está organizado de la siguiente manera:

En el Capítulo II se presentan los fundamentos de esta investigación, conceptos importantes como balanceo de carga, empaquetamiento, software comercial de VoIP para balanceo de carga, estimación de carga y algoritmos de balanceo de carga en diferentes ambientes distribuidos.

En el Capítulo III se describe el modelo formal del ambiente utilizado en esta investigación y las estrategias de asignación; la información incluye: notaciones, métricas, criterios de evaluación, estrategias de asignación de llamadas, modelos de predicción, análisis de la carga de trabajo y el marco de simulación.

En el Capítulo IV se detalla la configuración experimental para validar el desempeño de los algoritmos propuestos, los temas abordados contemplan: configuración de la simulación, regla de predicción del algoritmo tasa de cambio y diferentes versiones de la red neuronal para la predicción de llamadas.

En el Capítulo V se muestra el análisis experimental y la discusión sobre los resultados. El análisis compara el desempeño de los algoritmos propuestos con diferentes valores de simulación en dos escenarios.

Finalmente, en el Capítulo VI se presentan las conclusiones del trabajo de investigación, aportaciones al conocimiento, limitaciones y posibles líneas de investigación motivadas por las conclusiones obtenidas.

En el Anexo A se proporciona información importante relacionada al cómputo en la nube, uno de los pilares de esta investigación.

En el Anexo B se presentan las tablas con información completa sobre el análisis de degradación promedio y rango.

En el Anexo C se proporciona un resumen extenso en inglés del trabajo de tesis realizado.

Capítulo 2. Trabajo relacionado

En este capítulo se ofrece una visión general de los últimos avances en balanceo de carga, empaquetamiento, voz sobre el protocolo de Internet (VoIP), modelos de estimación de carga y balanceo de carga en entornos distribuidos.

Los temas siguientes examinan las principales características de los algoritmos de balanceo de carga, resultados teóricos del problema de empaquetamiento y su relevancia en la asignación de recursos, productos de software comerciales y los esfuerzos en VoIP para mejorar la utilización de los recursos sin afectar la calidad de servicio (QoS), la importancia de la estimación de la carga en el cómputo en la nube y la investigación en algoritmos de balanceo de carga en sistemas distribuidos.

2.1 Balanceo de carga

El balanceo de carga es un proceso de toma de decisiones sobre la distribución del trabajo. Sistemas de producción, servicios industriales, cómputo distribuido y paralelo, cómputo en la nube y otros sistemas utilizan este proceso mediante una implementación con soporte de hardware, software o una combinación de ambos. El balanceo de carga es una técnica ampliamente utilizada para el manejo eficiente los recursos.

Existen dos enfoques generales para el balanceo de carga en sistemas de cómputo: los recursos colaboran para adaptarse a la llegada aleatoria del trabajo y los brokers adecúan el número de trabajos en las colas de trabajo de los recursos. La tasa de llegada de trabajos, los retardos de comunicación, la variabilidad de los parámetros del trabajo y otros factores afectan considerablemente el rendimiento de los sistemas. El diseño de algoritmos escalables y eficientes de balanceo de carga es esencial para tratar con factores complejos como la incertidumbre.

Un conjunto virtualizado de recursos de cómputo, almacenamiento dinámicamente escalable, softwares y servicios de cómputo agregan una nueva dimensión al problema de balanceo de carga. A diferencia de los recursos dedicados administrados por un sistema de colas, la asignación y reasignación del trabajo puede realizarse no solo en función de los recursos y las propiedades del trabajo, sino también de los usuarios que comparten recursos al mismo tiempo.

Brindar una adecuada QoS a los usuarios finales es uno de los mayores desafíos en el cómputo en la nube. Las técnicas clásicas de balanceo de carga deben considerar parámetros adicionales cuando se usan en este tipo de entornos. La nube proporciona recursos de una manera dinámica y escalable. Algunas

características únicas que diferencian estas tecnologías de otras son: (1) recursos bajo demanda, (2) servicios flexibles y (3) recursos administrados por el proveedor de la nube. El cómputo en la nube incorpora características de sus predecesores, sistemas cluster y grid, y agrega sus peculiaridades: rendimiento escalable, capacidad de almacenamiento, elasticidad y funcionalidad extendida.

La gestión de la infraestructura en la nube es una tarea desafiante. La confiabilidad, la seguridad, la calidad del servicio y la relación costo-eficiencia son cuestiones importantes y requieren optimización de recursos en múltiples capas de la infraestructura y aplicaciones. Por ejemplo, un balanceo de carga adecuado hace más competitivo al proveedor del servicio y logra una mayor satisfacción del usuario. El balanceador ayuda a implementar tolerancia a fallos, escalabilidad, evita cuellos de botella, sobre aprovisionamiento, reduce el tiempo de respuesta, minimiza el consumo de energía, etc. (Nidhi et al., 2012). Adicionalmente, la nube suministra servicios en base a tres modelos: IaaS (VMs, red y almacenamiento), SaaS (aplicaciones) y PaaS (herramientas y bibliotecas).

El balanceo de trabajos computacionales, VMs, almacenamiento virtual, solicitudes de bases de datos y tráfico de VoIP se identifican como una gran preocupación para el uso eficiente del cómputo en la nube. Por lo tanto, los algoritmos de balanceo de carga son parte fundamental de la investigación en sistemas de cómputo en la nube.

2.1.1 Clasificación de los algoritmos de balanceo de carga

La clasificación más aceptada de los algoritmos de balanceo de carga toma en cuenta la forma de asignar y reasignar la carga de trabajo en los recursos.

Enfoque centralizado. Un solo nodo es responsable de distribuir la carga en todo el sistema. El nodo central recopila información sobre todos los recursos y toma decisiones basadas en esta información. Este enfoque puede causar cuellos de botella en algunas situaciones, por ejemplo, durante la pérdida de comunicación entre los recursos y el nodo central.

Enfoque distribuido. Los nodos construyen su propia cola de trabajos de forma autónoma y recopilan información de otros nodos. Los nodos toman decisiones de forma local e independiente, por ejemplo, un nodo define el momento de enviar trabajo a otros nodos para aliviar su carga. Este enfoque es más adecuado para sistemas distribuidos.

Enfoque híbrido. Un solo nodo es responsable de distribuir la carga de trabajo entre otros nodos de nivel local, este enfoque combina los dos enfoques anteriores con el objetivo de aprovechar las ventajas

de cada uno. El controlador administra los nodos subyacentes de nivel local y cada nodo de nivel local controla un subconjunto de recursos (nodos, almacenamiento, conmutador, etc.). Un nodo de nivel local conoce la información de sus recursos y comparte la carga de trabajo con otros nodos localmente.

Una clasificación alternativa de los algoritmos de balanceo considera el momento de la toma de decisiones sobre la distribución de la carga, estática y dinámica, y la forma de recibir los trabajos en el sistema, cola central y cola local.

Los *algoritmos estáticos* de balanceo toman decisiones en tiempo de compilación, donde se produce una estimación de los requisitos de los recursos. El balanceador no supervisa los nodos en tiempo de ejecución y distribuye el trabajo entre los nodos (servidores, sitios y VMs) sin tomar en cuenta el estado actual de cada uno. Los algoritmos estáticos reducen el sobrecosto (overhead), los retardos de comunicación y el tiempo de ejecución de los trabajos en entornos estáticos. Varios problemas están asociados a este tipo de algoritmos, por ejemplo, no son adecuados cuando la carga de trabajo es pesada y varía en el tiempo.

Los *algoritmos dinámicos* de balanceo de carga toman decisiones en tiempo de ejecución. El balanceador supervisa la distribución del trabajo entre los nodos en base a la información actual de cada nodo. El costo adicional de recolectar y mantener la información de los nodos no debe ser menospreciado. Los algoritmos dinámicos de balanceo de carga definen cuatro políticas para distribuir la carga del sistema: la *política de transferencia* determina los trabajos en el nodo local para posteriormente transferirlos a un nodo remoto, la *política de selección* define los nodos involucrados en el intercambio de carga, la *política de ubicación* selecciona un nodo destino para enviar trabajos y la *política de información* recopila información sobre los nodos en el sistema.

Los *algoritmos de cola central* reciben los trabajos mediante una cola global, esta situación permite que los trabajos estén listos para una distribución dinámica. El balanceador envía los trabajos a los nodos que solicitan carga.

Los *algoritmos de cola local* reciben los trabajos en colas independientes, cada nodo posee su propia cola de trabajo. Inicialmente, la carga es asignada localmente y posteriormente un proceso de migración redistribuye la carga a otros nodos. El proceso de migración es fundamental en estos algoritmos donde los nodos colaboran en la distribución de la carga local. Existen dos enfoques de migración, un nodo sobrecargado ofrece trabajo a otros nodos en el sistema o un nodo solicita trabajo cuando tiene poca carga.

En general, comparar diferentes enfoques es complicado, sin embargo, los algoritmos dinámicos tienen algunas ventajas sobre los algoritmos estáticos en escenarios reales. Desafortunadamente, la implementación y el uso de algoritmos dinámicos incrementa el número de parámetros a tener en cuenta, por ejemplo: determinar los nodos involucrados en la migración, el número de trabajos a migrar, umbrales de sobrecarga y poca carga, el grado de comunicación, el tamaño del mensaje para intercambiar información, etc.

2.1.2 Asignación de recursos

La asignación de recursos es el primer paso en el proceso de distribución de llamadas. Los algoritmos de asignación deciden el destino de las llamadas cuando llegan al sistema. Las restricciones de tiempo y calidad de servicio no permiten el uso de técnicas de migración en vivo (live-migration) para reasignar llamadas en el modelo de CVoIP, por lo tanto el proceso de asignación es la fase importante que permite mejorar el desempeño del sistema. La asignación de recursos es fundamental en nuestra investigación.

Las técnicas tradicionales de asignación de recursos no son adecuadas para modelos nube porque estos entornos están basados en la tecnología de virtualización con naturaleza distribuida. Además, el cómputo en la nube presenta nuevos desafíos debido a la manejabilidad y flexibilidad en la asignación de recursos. Las técnicas de asignación de recursos en nubes deben considerar: la heterogeneidad de hardware, la estimación de la carga de trabajo y las características de las aplicaciones para cumplir los objetivos de nivel de servicio de los consumidores (Parikh, 2013).

El diseño de estrategias depende significativamente de las características del ambiente. La siguiente sección reseña aspectos importantes del cómputo en la nube con impacto en el proceso de asignación.

2.1.3 Desafíos del balanceo de carga en el cómputo en la nube

El desarrollo de un algoritmo dinámico y eficiente de balanceo de carga implica diversos problemas importantes: estimación de la carga, comparación de niveles de carga, índices de rendimiento, estabilidad del sistema, cantidad de información intercambiada entre nodos, estimación de recursos solicitados por los trabajos, selección de trabajos a transferir, selección de nodos remotos, etc. (Alakeel, 2010). Algunos aspectos a considerar durante el diseño de un algoritmo de balanceo de carga incluyen: la distribución de los nodos en la nube, las réplicas de almacenamiento, migración de VMs y retardos de inicio de las VMs.

Distribución de los nodos. Algunos algoritmos de balanceo funcionan adecuadamente solo si la distribución de los nodos es cercana y los retardos de comunicación son insignificantes. Sin embargo, la distribución geográfica de la infraestructura nube impone restricciones importantes al considerar retardos de comunicación.

Réplicas de almacenamiento. La replicación completa de datos no es una técnica eficiente porque aumenta la sobrecarga de almacenamiento y comunicación en el sistema. La replicación parcial considera las capacidades de cada nodo para almacenar porciones de información en diferentes nodos (con traslape). Algunos beneficios de la replicación parcial son el incremento de la utilización, tolerancia a fallos y la disponibilidad de datos. Los algoritmos de balanceo de carga deben considerar la distribución de la información para evitar cuellos de botella en solicitudes de información.

Migración de VMs. Un nodo sobrecargado puede aliviar su carga mediante la migración de VMs. Un conjunto de archivos conforma una VM, por lo tanto, la migración de VMs consiste en mover archivos de un nodo a otro. Algunas preguntas difíciles de responder en migración de VMs son: seleccionar una VM para migrar, determinar el nodo destino y estimar el beneficio de la migración. Adicionalmente, la migración de VMs incrementa el uso de la red y reduce el desempeño de las VMs durante el proceso de migración.

Retardos en el inicio de la VM (StUp). El tiempo para desplegar VMs en la infraestructura nube afecta considerablemente el rendimiento de sistemas suaves de tiempo real. Por ejemplo, el sistema de CVoIP debe asignar las llamadas tan pronto como sea posible para no degradar la calidad del servicio. El StUp depende de varios factores pero los más importantes son el SO y el software que se ejecuta en la VM (Mao y Humphrey, 2012; Hoffman, 2015).

Algunos de los factores mencionados anteriormente pueden afectar el funcionamiento de los sistemas CVoIP. Las siguientes secciones presentan los últimos avances en diversos temas relacionados con técnicas de empaquetamiento para asignación de recursos, software comercial de VoIP, balanceo y predicción de la carga para sistemas de voz IP y estrategias de asignación de recursos para diferentes ambientes.

2.2 Empaquetamiento

El estudio del problema clásico de empaquetamiento en una dimensión se remonta a la década de 1970 con los trabajos de Ullman y Johnson (Ullman, 1971; Johnson, 1973). El problema de

empaquetamiento es uno de los problemas más estudiados en computación. Una secuencia de elementos, cada uno con un tamaño en el rango (0, 1), debe ser empaquetada en un número mínimo de contenedores.

El problema de empaquetamiento dinámico de contenedores (DBP) se introdujo en 1983 (Coffman et al., 1983) como una generalización del problema clásico de empaquetamiento. En el problema DBP, los elementos llegan y abandonan el sistema en momentos arbitrarios, el objetivo principal es minimizar el número total de contenedores usados para empaquetar elementos de fracciones unitarias (es decir, elementos con tamaños $1/w$ para un entero $w \geq 1$) en contenedores de tamaño unitario. La formulación del problema de asignación de llamadas en CVoIP es semejante al problema DBP, las llamadas (elementos) llegan y abandonan (finalizan su ejecución) el sistema de VoIP en momentos arbitrarios.

Coffman et al. (1983) definen una relación competitiva entre 2.75 y 2.897 para el algoritmo First Fit en línea y demuestran que ningún algoritmo en línea puede lograr una relación competitiva mejor a 2.5. Los resultados consideran un algoritmo fuera de línea donde los elementos pueden ser reempaquetados sin penalización ni costo. También, proporcionan un límite inferior de 2.388 cuando el algoritmo fuera de línea no permite reempaquetar los elementos.

Chan et al. (2008) analizan el rendimiento de la familia de algoritmos Any Fit (Best Fit, First Fit y Worst Fit) y proporcionan límites de rendimientos estrechos y casi ajustados. Los autores muestran que First Fit tiene un mejor rendimiento y su relación competitiva se encuentra entre 2.45 y 2.4942. Finalmente, prueban que ningún algoritmo en línea puede ser mejor que 2.428-competitivo para elementos de fracciones de unidades.

Chan et al. (2009) mejoran el factor de competitividad a 2.5 para el algoritmo fuera de línea. Wong et al. (2012) establecen una cota inferior de 8/3 ≈ 2.666 para el algoritmo fuera de línea, esta cota reduce la brecha entre la cota inferior y superior, que actualmente se encuentra en 2.788.

Las técnicas de asignación y el balanceo de carga en la nube utilizan ampliamente técnicas de empaquetamiento, el principal objetivo es asignar aplicaciones a las VMs y/o las VMs en los recursos. Algunos trabajos de investigación relevantes en esta área se describen a continuación.

Wolke et al. (2015) analizan una amplia variedad de controladores para la asignación y reasignación de VMs considerando la disponibilidad de recursos (CPU, memoria, ancho de banda, etc.). Las estrategias de empaquetamiento asignan las VMs entrantes en los servidores, siempre que la capacidad residual lo permita. El controlador de reasignación activa la migración de VMs para mejorar la asignación cuando los servidores están vacíos o sobrecargados. Los autores muestran que la combinación

de asignación y la reasignaciones periódica logran la mayor eficiencia energética sujeta a niveles de servicio predefinidos.

Li et al. (2015) consideran una variante del problema DBP para procesar solicitudes de juego en ambientes nube. En un sistema de juegos en la nube, cada instancia de juego demanda cierta cantidad de CPU y GPU al servidor donde se ejecuta. Las solicitudes deben atenderse con suficientes recursos para proporcionar una buena experiencia al usuario. El objetivo principal es minimizar el número de servidores (contenedores) necesarios para procesar las solicitudes de juego (elementos) debido a que cada solicitud agrega un costo proporcional a la duración del uso de los recursos. Los autores analizan la relación competitiva para las versiones originales y modificadas de los algoritmos Best Fit y First Fit.

Las estrategias de empaquetamiento son usadas ampliamente en el cómputo en la nube porque tiene un alto desempeño y son fáciles de analizar e implementar. Los algoritmos de empaquetamiento han sido ampliamente estudiados y toman decisiones en base al estado actual del sistema, estas características son requeridas en estrategias de balanceo de carga en CVoIP.

2.3 Productos comerciales de VoIP

Esta sección describe productos de software comercial para balanceo de VoIP. El desempeño de estrategias de balanceo comerciales define un marco de referencia para comparar las estrategias propuestas en este trabajo de investigación.

Los productos de software comerciales para la distribución de la carga implementan algoritmos de balanceo de carga que se ejecutan en los servidores de balanceo. Los productos de hardware y software (cajas negras) están empacados en sistemas operativos y softwares especiales, y los conmutadores comerciales incrementan su funcionalidad al agregar extensiones a la tradicional capa 2/3 del modelo OSI.

Los proveedores de VoIP categorizan el tipo de tráfico que fluye a través de la red al aplicar el balanceo de la carga. Varios métodos de distribución de carga pueden ser aplicados según los objetivos definidos por el proveedor. El balanceo de carga se puede realizar en dos modos: stateful y stateless. En el modo stateful, el protocolo está diseñado para reconocer el inicio y la terminación de la sesión (por ejemplo, el tráfico SIP). En el modo stateless, el protocolo considera que las sesiones individuales son independientes al tráfico entrante y la distribución de la carga (por ejemplo, protocolo en tiempo real).

Los métodos de asignación más comunes son: round robin, mínimas conexiones, distribución ponderada, tiempo de respuesta y umbrales de carga del servidor (Citrix 2017). Algunos algoritmos son

más adecuados para: gestionar el tráfico en sitios web, administrar el tráfico en servidores de sistema de nombres de dominio (DNS) y manejar aplicaciones web complejas utilizadas en el comercio electrónico.

Round Robin (RR) es un método de distribución de carga donde los servidores esperan su turno para atender solicitudes. Las ventajas del algoritmo RR comprenden: consume poca cantidad de recursos y es eficiente para sistemas que manejan un gran número de solicitudes simultáneas, aproximadamente equivalentes en términos de recursos de procesamiento y duración. Este algoritmo no toma en cuenta el número de conexiones simultáneas activas en el sistema.

Mínimas conexiones (LC) realiza un seguimiento del número de conexiones simultáneas activas en cada nodo, el balanceador utiliza esta información para reenviar solicitudes al nodo menos cargado. Distribución ponderada (WD) define una ponderación relativa para los servidores y emplea algún método adicional para balancear la carga.

Tiempo de respuesta (RT) se usa en situaciones donde el rendimiento de las aplicaciones es importante. RT funciona como un método independiente o define valores de umbral para otros métodos de distribución. Los umbrales de carga del servidor (SLT) permiten definir configuraciones para varios indicadores (carga del CPU, disco duro, memoria, ancho de banda) y utiliza algún método de balanceo para distribuir la carga entre los servidores.

Mobicents (Mobicents, 2017) es un proxy basado en el protocolo SIP para infraestructura VoIP con múltiples servidores proxy de entrada. El objetivo principal es evitar la congestión, la mala utilización de los recursos y la sobrecarga. Mobicents usa un algoritmo RR para distribuir el tráfico entre los servidores, y contempla las solicitudes y los parámetros del sistema, como el CPU.

Brocade (Brocade, 2017) es una versión avanzada del balanceador de carga para servidores SIP. El comportamiento del algoritmo es similar a un balanceador de carga para proxy o registro de tráfico, en lugar de servidor proxy SIP. La aplicación proporciona modo stateful y persistencia para el tráfico SIP del protocolo de datagramas de usuario (UDP). Brocade crea sesiones para que las transacciones posteriores sean enviadas al mismo servidor (parámetro de persistencia), esto permite el balanceo de carga de los servidores SIP del Agente de usuario back-to-back (B2BUA).

Radware (Radware, 2017) ofrece soluciones de TI como protección contra denegación de servicio, entrega de aplicaciones y soluciones de balanceo de carga. El balanceo de carga de VoIP garantiza disponibilidad de red, seguridad, escalabilidad y administración del tráfico VoIP. Radware mejora las operaciones comerciales, minimiza la degradación de la entrega del servicio y previene el tiempo ocioso

de los recursos. Esta solución consiste una agrupación local y global de recursos junto con controles de estado en dispositivos SIP.

NetScaler ADC (Citrix, 2017) permite balancear la carga de los mensajes SIP sobre UDP o el protocolo de control de transmisión (TCP). El algoritmo balancea las solicitudes SIP en un grupo de servidores proxy. El usuario crea servidores virtuales de balanceo de carga con métodos de distribución de carga y tipos de persistencia. NetScaler mejora la seguridad y las experiencias administrativas de Skype empresarial en un entorno virtual.

El algoritmo RR es ampliamente utilizado en sistemas de VoIP por lo que resulta indispensable evaluar su comportamiento en el modelo de CVoIP. El diseño de estrategias puede considerar información sobre el número de conexiones, umbrales de carga, monitoreo de recursos (CPU, memoria, ancho de banda, etc.), entre otros. Las siguientes secciones describen los últimos avances en diferentes áreas de VoIP en la nube.

2.4 Calendarización en VoIP

La migración del servicio de VoIP a las nubes detonó la investigación en asignación de llamadas, calidad del servicio, predicción de la carga, etc. El objetivo principal es reducir el costo de infraestructura y garantizar la entrega del servicio de la mejor manera posible. Varios algoritmos han sido propuestos para mejorar el rendimiento de los sistemas de VoIP.

Lee et al. (2006) analizan tres algoritmos de calendarización en sistemas inalámbricos de banda ancha móvil para el estándar IEEE 802.16e. Los autores proponen los algoritmos: rendimiento del servicio de subvención no solicitado (UGS), servicio de sondeo en tiempo real (rtPS) y servicio de sondeo en tiempo real extendido (ertPS). El análisis experimental basado en simulación muestra que el algoritmo ertPS, en conjunto con el códec EVRC y supresión de silencio, admite una mayor cantidad de llamadas en comparación con los algoritmos UGS y rtPS.

Folke et al. (2007) analizan algoritmos de calendarización para una mezcla de tráfico web y VoIP. Las estrategias Proporcional Fair (PF), tasa máxima (MR), MR con mínima tasa de bits (MRmin) y MR con demora estricta ($MRdelay$) comparan su desempeño con variaciones de carga y retardos en la calendarización. Los autores concluyen que un calendarizador funciona bien cuando aumenta gradualmente la prioridad de VoIP y considera la tasa actual del usuario. Sin embargo, la prioridad de VoIP necesita un aumento drástico cuando los retardos son cortos. El sistema se vuelve sensible a situaciones de sobrecarga al intentar mantener la calidad de ambos tipos de tráfico.

Bayer et al. (2010) analizan la calendarización bajo demanda para el control de acceso basado en redes de malla TDMA y el estándar IEEE 802.16. Los autores proponen tres esquemas de acceso: esquema coordinado de recursos descoordinados, esquema coordinado de recursos coordinados y esquema coordinado de recursos consciente de VoIP. Los resultados de la investigación muestran que las redes en malla pueden admitir tráfico de VoIP con buena calidad cuando existe una calendarización persistente. En comparación con otros esquemas de coordinación de recursos, el esquema de acceso consciente de VoIP aumenta significativamente el número de llamadas admitidas.

So (2011) analiza la calendarización dinámica y persistente para servicios de VoIP en sistemas inalámbrico de frecuencia ortogonal dividida con múltiples accesos. El autor desarrolla modelos analíticos y de simulación para evaluar el rendimiento de los servicios de VoIP en términos del rendimiento promedio y la sobrecarga de señalización según los esquemas de calendarización.

Sánchez (2013) analiza el problema de teleconferencia segura bajo demanda en la infraestructura de nubes públicas. El autor propone la arquitectura red privada virtual de teleconferencia segura en la nube (TSCV) y la arquitectura de servidores de teleconferencia ocultos en la nube (HTSC) para abordar posibles violaciones de seguridad y ataques en las comunicaciones de VoIP. TSCV impide el registro de usuarios no autorizados y ataques de denegación de servicio y HTSC evita ataques de denegación de servicio y ataques al servidor Asterisk, el componente principal de VoIP. El estudio evalúa sistemas existentes de servidores de telefonía comercial y de código abierto con respecto a la disponibilidad y la escalabilidad. El trabajo presenta un marco abstracto de elementos para comparar diferentes arquitecturas de VoIP.

Wu et al. (2014) presentan un marco de calendarización en tiempo real en entornos virtualizados consciente de las limitaciones de las aplicaciones en tiempo real. El mecanismo de partición dinámica multinúcleo considera los parámetros de calendarización de las VMs para dividir los CPU físicos en dos grupos. Los autores utilizan la estrategia primero la llamada con plazo global de terminación más corto para calendarizar las llamadas en las VMs. El objetivo del sistema basado en instancias de Asterisk es mejorar el uso del CPU y garantizar la calidad de las llamadas.

Mazalek et al. (2015) estudian el impacto del cifrado IPSec en la utilización del CPU, el ancho de banda y la calidad de voz. Los autores muestran un efecto significativo del período de carga útil de voz en la utilización del CPU y el ancho de banda. Las estrategias ahorran hasta 60% del ancho de banda cuando el período de carga se elige correctamente. La calidad de la llamada, expresada mediante la escala MOS, permanece casi constante hasta el momento que la utilización del CPU es cercana al 80%.

Costa et al. (2015) muestran como un servidor Asterisk puede proporcionar capacidades de comunicación VoIP con calidad MOS aceptable. Los autores usan la métrica de probabilidad de bloqueo para medir la capacidad del servidor VoIP y evalúan la calidad de las llamadas de voz mediante MOS. Los resultados experimentales muestran un manejo eficiente de Asterisk con el protocolo SIP, donde más de 160 llamadas de voz atendidas concurrentemente tienen una probabilidad de bloqueo inferior al 5%.

Cheng et al. (2015) presentan y comparan el calendarizador SRT-Xen con cuatro estrategias de calendarización: Credit, Credit2, rtglobal y rtpartition. Las estrategias utilizan una calendarización en tiempo real amigable para mejorar la administración de las colas de los CPUs virtuales. El sistema emplea Asterisk para evaluar el rendimiento de las estrategias y la calidad de voz. SRT-Xen admite por los menos un 13.61% más de sesiones con evaluación perceptual de la calidad del habla mayor a 4.

Montazerolghaem et al. (2016) presentan y analizan el controlador de balanceo de carga virtual para la admisión de llamadas (VLBCAC). Esta estrategia proporcionar control de admisión para servidores SIP y balanceo de llamadas entrantes. VLBCAC incorpora mecanismos para predecir el número de llamadas, estimar los recursos requeridos y seleccionar instancias adecuadas de VM (considerando CPU, memoria y ancho de banda). Los autores proponen un modelo para maximizar el uso de los recursos y el rendimiento del sistema.

Kim (2016) investiga la escalabilidad de los servidores SIP para un sistema de telecomunicaciones basado en la nube. El sistema puede crecer y reducir su tamaño a pedido, responder a variaciones de latencia en las VMs y tolerar fallas. El administrador de carga escalable (LSM) incorpora diferentes técnicas en un solo sistema coherente. Las funciones principales del LSM son: aumentar o disminuir los recursos según la carga, verificar el estado de todos los nodos, supervisar la cola de llamadas en espera e incorporar la carga al sistema.

La investigación en algoritmos de distribución de recursos para ambientes nube considera diferentes elementos, sin embargo, la mayoría de estas propuestas confirman el uso de la medida MOS para evaluar la calidad de las llamadas e instancias de Asterisk como software estándar para proporcionar el servicio de VoIP. Los trabajos más recientes abordan cuestiones relacionadas con las características de la infraestructura nube (escalabilidad, utilización del CPU, VMs, tipos de instancias, etc.). Sin embargo, ningún trabajo afronta la degradación de la calidad de voz debido al retardo en el despliegue del sistema de VoIP en la infraestructura nube.

2.5 Estimación de la carga

El objetivo de la predicción de carga en el cómputo en la nube es minimizar los costos de infraestructura y mejorar la QoS para el usuario final. Las técnicas de predicción favorecen el proceso de distribución de la carga, la idea primordial es anticipar la cantidad de tráfico entrante, es decir, las llamadas en el caso de VoIP.

Campos y Scherson (1999) proponen un algoritmo de balanceo de carga llamado tasa de cambio (RoC) que incorpora una regla para la estimación de carga en el sistema. El algoritmo descentralizado minimiza los tiempos ociosos del procesador sin incurrir en una sobrecarga excesiva durante el proceso de balanceo. Los elementos de procesamiento (PE) interactúan para balancear la carga actual mediante la migración de trabajos. La estimación de la carga depende de la diferencia de carga (tasa de cambio) en el PE entre dos intervalos de tiempo. La carga actual, la tasa de cambio y la configuración del balanceador determinan el momento de balancear la carga. El proceso de migración considera cuatro fases asíncronas para distribuir la carga de trabajo.

Simionovici et al. (2013) estudian y comparan dos modelos de predicción de carga para entornos de VoIP reales. Los algoritmos sistema de partículas interactivas (IPS) y redes neuronales (NN) estiman el tráfico de voz entrante durante ventanas de tiempo. La investigación busca construir un modelo capaz de predecir tendencias relacionadas con: perfiles de usuario y patrones de horas pico o llamadas. Simionovici et al. (2015) amplían su investigación donde analizan IPS, mezcla de modelos Gaussianos (GMM) y procesos Gaussianos (GP) para estimar el tráfico de voz entrante. El modelo proporciona enfoques de modelado flexible, conformación del tráfico determinada por los clientes y soluciones escalables con buena precisión de predicción. Todos los algoritmos fueron entrenados y probados bajo diferentes escenarios.

Calheiros et al. (2015) proponen un módulo de predicción de carga en la nube para proveedores de SaaS basado en el modelo auto regresivo integrado de promedio móvil (ARIMA). Los autores evalúan la precisión de la predicción en cargas reales de solicitudes a servidores web. La investigación analiza el impacto en términos de eficiencia en la utilización de los recursos y la QoS.

Lu et al. (2016) analiza algoritmos de predicción de carga para mejorar la eficiencia energética en sistemas nube. Los autores evalúan el rendimiento de los predictores: red neuronal con propagación hacia atrás con razón de aprendizaje aleatorio variable (RVLBPNN), modelo oculto de márkov (HMM) y clasificador bayesiano ingenuo (NBC) para estimar cargas de trabajo en centros de datos. Todos los modelos evalúan su eficiencia en la predicción de cargas de trabajo intensivas en memoria y CPU. El

objetivo principal es mantener un alto nivel de calidad de servicio (QoS) y reducir el uso de los recursos en el centro de datos.

La estimación de la carga apoya el proceso de distribución del trabajo. El uso del modelo CVoIP agrupó la investigación en estimación de carga en nubes y VoIP. Varios de los enfoques previamente descritos pueden mejorar la eficiencia de las estrategias de balanceo en CVoIP y reducir los problemas inherentes a los ambientes nube como el retardo de inicio en la VMs. La implementación y evaluación de tres estimadores de carga es objeto de estudio en esta tesis.

En la siguiente sección se reseñan algoritmos de distribución del trabajo para ambiente de cómputo distribuido, la idea primordial es identificar estrategias fuera del domino de VoIP y analizar su posible incursión en el problema de balanceo en CVoIP.

2.6 Algoritmos de distribución de recursos en ambientes de cómputo distribuido

El problema de distribución de los recursos ha sido ampliamente estudiado en diferentes ambientes de cómputo. El diseño de un algoritmo depende significativamente de las características del entorno. El estudio de los últimos avances favorece el proceso de diseño de estas estrategias, la siguiente revisión permite identificar características de ambientes y diferentes enfoques.

En general, agrupar los algoritmos de balanceo de carga es complicado, sin embargo para simplificar el estudio se proponen tres subcategorías: técnicas estáticas, adaptación de meta heurísticas y técnicas con diferentes enfoques.

2.6.1 Técnicas estáticas

Min-Min (Kokilavani y Amalarethinam, 2011) es un algoritmo de balanceo de carga para cómputo en grid. Un conjunto de trabajos no asignados espera en una cola para ser distribuidos, el planificador identifica el trabajo con menor tiempo de ejecución de todos los trabajos y luego asigna este trabajo al recurso que produce el menor tiempo de finalización. El proceso de asignación se repite mientras haya trabajos en la cola de espera. El objetivo del algoritmo es minimizar el tiempo total de ejecución de los trabajos (makespan).

Max-Min (Kokilavani y Amalarethinam, 2011) es un algoritmo de balanceo de carga para cómputo en grid. Un conjunto de trabajos no asignados espera en una cola para ser distribuidos, el planificador identifica el trabajo con menor tiempo de ejecución de todos los trabajos y luego asigna el trabajo al recurso que produce el máximo tiempo de finalización. El proceso de asignación se repite hasta que todos

los trabajos sean asignados. El objetivo del algoritmo es reducir el tiempo de espera de los trabajos más grandes.

Balanceo de carga Min-Min: LBMM (Kokilavani y Amalarethinam, 2011) es un algoritmo de balanceo de carga de dos fases para cómputo en grid. LBMM utiliza las ventajas de Min-Min para reducir el tiempo de finalización del calendario y aumentar la utilización de los recursos. En la primera fase, la estrategia Min-Min asigna las tareas en los recursos. En la segunda fase, el balanceador selecciona un recurso con carga pesada (recurso con alto makespan) y reasigna sus tareas a otros recursos con carga ligera.

Round robin difuso - FBRR (Sethi et al., 2012) es un algoritmo de balanceo de carga basado en lógica difusa y round robin para centro de datos en la nube. El sistema mantiene la información de las VMs y los trabajos actualmente asignados. En la fase de asignación, el balanceador distribuye los trabajos a las VMs con menor carga. En la fase de reasignación, un difusificador considera la velocidad del procesador y la carga de las VMs para balancear el sistema. El sistema de inferencia difusa (FIS) simula la toma de decisiones humana basada en reglas de control difusas y parámetros lingüísticos de entrada.

Porción de tiempo priorizada basada en round robin - TSPBRR (Samal y Mishra, 2013) es un algoritmo de balanceo de carga basado en round robin para cómputo en la nube. TSPBRR soluciona la distribución de carga en varios nodos de un sistema distribuido. El algoritmo mejora la utilización y el tiempo de respuesta de los recursos en comparación con otra versión de RR.

El enfoque de balanceo estático tiene ventajas sobre otros: fácil implementación, mínimo intercambio de información, no requiere monitoreo de los recursos, etc. Desafortunadamente, estos algoritmos necesitan información no disponible en ambientes nube: tiempo de ejecución de los trabajos y la utilización de los recursos (puede cambiar en cualquier momento). El proceso de migración es ampliamente utilizado por estrategias estáticas para incrementar el desempeño, sin embargo el modelo de CVoIP no permite este proceso porque degrada la calidad del servicio. Los retardos de inicio en las VMs y cambio dinámicos de infraestructura afectan el desempeño de estos algoritmos. La mayoría de las investigaciones utilizan abstracciones básicas del sistema nube que no representan ambientes reales.

2.6.2 Técnicas meta heurísticas

Colonia de hormigas y teoría compleja de red - ACCLB (Zhang y Zhang, 2010) es un mecanismo de balanceo de carga para federación de nubes públicas. Una hormiga visita los nodos de procesamiento periódicamente para balancear la carga de trabajo. Durante el viaje, la hormiga recuerda los nodos con

máxima (Nmax) y mínima (Nmin) carga. El proceso de búsqueda termina cuando los nodos Nmax y Nmin exceden un umbral de diferencia de carga o los nodos visitados sobrepasan el número de movimientos permitidos. Finalmente, la hormiga informa a Nmax y Nmin que deben balancear la carga de trabajo entre ellos. El objetivo principal es hacer frente al balanceo dinámico con características de red compleja.

Estrategia de calendarización para balanceo de recursos en VMs - SVB (Hu et al., 2010) es un algoritmo de balanceo de carga para la migración de VM en entornos nube. El algoritmo basado en algoritmos genéticos (GA) considera la variación del sistema, información histórica y el costo de migración. El principal objetivo del algoritmo es reducir la migración de VMs. El cromosoma está representado por una estructura de árbol donde el nodo raíz es el calendario, los nodos intermedios son los recursos y los nodos hoja son las VMs. El trabajo define las estrategias correspondientes de selección, hibridación y variación del GA.

Comportamiento de alimentación de las abejas - HF (Randles et al., 2010) es un algoritmo de balanceo de carga descentralizado para sistemas nube a gran escala. El algoritmo asigna dinámicamente los servicios web en los servidores para regular el sistema frente a la demanda. Los servidores locales colaboran para alcanzar un comportamiento auto-organizado, definido por el equilibrio de la carga global. Un servidor físico agrupa servidores virtuales, cada uno con su propia cola de trabajos. La noción de "anuncios" se utiliza para comunicar el beneficio global de la colonia. Los servidores ociosos leen los anuncios, eligen un anuncio y atienden la solicitud o aleatoriamente atienden una solicitud de la cola de solicitudes.

Una nueva versión de la heurística HF (LD y Krishna, 2013) considera la calendarización de tareas en el cómputo en la nube. El algoritmo usa las prioridades de las tareas durante el balanceo con el objetivo de minimizar el tiempo de espera de las tareas en la cola de la VM. Las abejas (tareas) migran de VMs sobrecargadas, actualizan la carga de las VMs y la prioridad de otras tareas. Las tareas eligen una VM considerando la carga y las prioridades de otras tareas. Tareas con alta prioridad, enviadas a otras VMs, deben tomar en cuenta el número de tareas de alta prioridad en la VM, esta situación garantiza la ejecución de la tarea lo antes posible.

Sistema de agente auto-organizado para el balanceo de carga dinámico - SOAS (Laredo et al., 2012) es un algoritmo descentralizado para balanceo de carga en sistemas distribuidos. El algoritmo emula el comportamiento de una pila de arena donde las avalanchas reconfiguran el estado del sistema. El comportamiento de avalancha modela adecuadamente la calendarización no clarividente, donde las tareas representan granos de arena y el proceso de balanceo de carga simboliza una avalancha. Las tareas

llegan al sistema en forma de bolsas de tareas y el proceso de auto-organización genera una nueva asignación de tareas. Un agente distribuido adquiere propiedades como la auto-organización de tareas en los recursos.

Optimización por cúmulo de partículas - TBSLB-PSO (Ramezani et al., 2014) es un mecanismo de balanceo de carga para cómputo en la nube. El calendarizador central de tareas (CTS) consulta la información de una pizarra para transferir tareas desde una VM sobrecargada. La pizarra contiene toda la información de los calendarizadores: características de las VMs, tareas en ejecución y calidad del servicio (QoS). El proceso de migración considera la cantidad de datos, CPU, memoria, ancho de banda de las VMs. Los recursos físicos inactivos están limitados para recibir tareas, esta situación permite disminuir el consumo de energía.

Balance de carga basado en luciérnagas - LBS-BF (Florence et al., 2014) es un mecanismo de balanceo de carga para el cómputo en la nube. El balanceador distribuye la carga dinámicamente en base a índices de carga de los recursos compartidos. La atracción de las luciérnagas está ligada a la función objetivo y al deterioro del atractivo. El algoritmo genera índices de calendarización y encuentra nodos estrechamente asociados dentro de la red. LBS-BF considera tres parámetros: atracción entre nodos y solicitudes, índice de calendarización y distancia entre nodos. Estos parámetros toman en cuenta la velocidad del CPU, la memoria, el tiempo de procesamiento y la carga de los nodos.

Las ventajas del enfoque basado en técnicas meta heurísticas incluyen: fácil adaptación al problema, mínima información sobre el sistema, colaboran con otras estrategias, etc. Las estrategias pueden adaptarse fácilmente a ambientes dinámicos. Desafortunadamente, estos algoritmos introducen importantes retardos de tiempo en el proceso de balanceo. Similar a las estrategias estáticas, el proceso de migración es ampliamente utilizado. Las investigaciones consideran sistemas estáticos que no corresponden a ambientes nube reales. Este tipo de técnicas no pueden aplicarse en ambientes de CVoIP debido a las restricciones de tiempo.

2.6.3 Técnicas con diferentes enfoques

VectorDot - VD (Singh et al., 2008) es un algoritmo de balanceo de carga para la administración de recursos en sistemas jerárquicos de cómputo distribuido. El algoritmo está inspirado en el método Toyoda para el problema de la mochila multidimensional. VD permite mover VMs, tráfico y almacenamiento virtual entre nodos sin tiempo de inactividad de la aplicación en ejecución. La función Producto Punto (PP) define servidores sobrecargados debido al uso excesivo de CPU, memoria y red o disco de E/S. El planificador usa PP para determinar la migración de VMs o tráfico en servidores o conmutadores (nodos).

Carton - LB + DRL (Stanojevic y Shorten, 2009) es un algoritmo de balanceo de carga para unificar el control de la nube. El sistema utiliza un balanceador de carga (LB) para minimizar los costos asociados y un limitador proporcional distribuido (DRL) para garantizar una asignación de recursos justa. El balanceador de carga distribuye el trabajo en base a un algoritmo de subgradiente. DRL asigna las capacidades del servidor asegurando el mismo nivel de rendimiento en todos los servidores.

Comparar y balancear - CB (Zhao y Huang, 2009) es un algoritmo de balanceo de carga distribuido para sistemas nube. El algoritmo reduce el tiempo de migración y cumple con la reubicación sin interrupciones de las VMs. El sistema alcanza el balanceo de carga mediante un muestreo, CB calcula el costo de los nodos en el sistema y compara el costo del nodo actual con otro elegido al azar. Si la diferencia de costo entre ambos nodos excede cierto umbral entonces el sistema determina la migración de una VM con cierta probabilidad. El algoritmo solo considera la distribución inicial de las VMs en los nodos físicos.

Política de balanceo de carga central para VMs - CLBVM (Bhadani y Chaudhary, 2010) es un algoritmo de balanceo de carga para la distribución de VMs en entornos nube. El algoritmo toma decisiones basadas en información global y define reglas centralizadas de información y ubicación. La regla de transferencia está parcialmente distribuida y parcialmente centralizada. Las VMs pueden migrar a un servidor ligeramente cargado cuando el rendimiento su servidor es afectado por otra VM.

Estrategia de balanceo de carga para almacenamiento virtual - LBVS (Liu et al., 2010) es un algoritmo de balanceo de carga para arquitecturas de almacenamiento a gran escala en la nube. El algoritmo proporciona almacenamiento de datos con tres etapas: una para virtualización de almacenamiento y dos módulos de balanceo de carga. El espacio global, el espacio lógico y los recursos integran el modelo de virtualización de almacenamiento. El módulo de balanceo de réplicas (RBM) comparte la carga de acceso en los servidores réplica y el módulo de balanceo de escritura (WBM) usa ponderación para elegir y enrutar datos / archivos al mejor nodo lógico.

Calendarización de tareas basada en balanceo de carga - TSLB (Fang et al., 2010) es un algoritmo de balanceo de carga de dos niveles para grid y cómputo en la nube. La calendarización de primer nivel asigna las aplicaciones de los usuarios a las VMs y la calendarización de segundo nivel asigna las VMs en los recursos. TSLB considera los requisitos dinámicos de los usuarios, el algoritmo conoce a priori la demanda de recursos de los trabajos pero estos pueden cambiar durante la ejecución. Por esta razón, el calendarizador reasigna recursos de dos maneras: envía VMs a nodos con suficientes recursos libres o recibe VMs de otros nodos.

Muestreo aleatorio con sesgo - BRS (Randles et al., 2010) es un algoritmo de balanceo de carga descentralizado para sistemas nube a gran escala. El sistema alcanza auto-organización en base al muestreo aleatorio del sistema, lo que mantiene la carga del nodo cerca de una media global. La carga de un servidor está representada por aristas, este enfoque permite crear una red en el sistema para proporcionar una medida de disponibilidad del estado inicial. Los nodos aceptan trabajos en base a un umbral, si el nodo sobrepasa el umbral entonces este ejecuta el trabajo. En caso contrario, el nodo selecciona al azar un nodo contiguo para enviar el trabajo.

Agrupamiento activo - AC (Randles et al., 2010) es un algoritmo descentralizado de balanceo de carga de auto-agregación para sistemas nube a gran escala. Este algoritmo conecta servicios similares, agrupa instancias similares, mediante la reconexión local de la red. AC construye iterativamente la red de conexiones para cada nodo. Este enfoque funciona adecuadamente cuando cada nodo conoce a sus similares y puede contactar con ellos para delegar tareas.

Calendarización de dos fases - OLB + LBMM (Wang et al., 2010) es un algoritmo de balanceo de carga para redes de tres niveles en el cómputo en la nube. El algoritmo combina el balanceo de carga oportunista (OLB) y el balanceo de carga Min-Min (LBMM). OLB mantiene los nodos ocupados y LBMM encuentra el nodo con el mínimo tiempo de ejecución para un trabajo en el entorno. Una cola de trabajos almacena tareas que el administrador debe llevar a cabo. En la primera fase, el algoritmo de calendarización OLB asigna tareas mediante el servicio de administración. En la segunda fase, el algoritmo de calendarización LBMM elige el nodo de servicio adecuado para ejecutar subtareas.

Guiado por eventos - ED (Nae et al., 2010) es un algoritmo de balanceo de carga para juegos de tiempo real en línea (MMOG) para ambientes nube. El algoritmo muestra un alojamiento eficiente en costo de sesiones y aprovisionamiento bajo demanda de la cantidad correcta de jugadores. ED utiliza una solución guiada por eventos; el analizador de eventos recibe una solicitud, analiza el contexto y toma decisiones para asignar servicios. Los eventos consisten en sesiones, recursos, estados, monitoreo y predicción.

Azure MapReduce - AMR (Gunarathne et al., 2010) es un algoritmo descentralizado de balanceo de carga para cómputo en la nube de aplicaciones científicas en Microsoft Azure. En tiempo de ejecución, AMR crea servicios de infraestructura en Microsoft Azure. Inicialmente, una cola de mensajes almacena las tareas. Posteriormente, una función de mapeo obtiene todos los mensajes de la cola, el mensaje de la tarea contiene metadatos necesarios para asignar la ejecución de tareas y generar una tabla de tareas reducida. Finalmente, una función de reducción de tareas obtiene datos intermedios basados en la tabla

de tareas reducidas. Los usuarios tienen la capacidad de configurar el mapeo y reducir los trabajadores por instancia de Azure.

Política consciente de la carga de trabajo y del cliente - WCAP (Mehta et al., 2011) es un algoritmo de balanceo de carga para calendarizar solicitudes en servidores y bases de datos en entornos distribuidos. El planificador define el nodo más adecuado para procesar las solicitudes. La división de los recursos en cuatro secciones da como resultado una mejor localización. La propiedad de USP especifica el proveedor de servicio y las solicitudes de los consumidores; la información de contenido reduce la búsqueda. El rendimiento de esta estrategia se compara con la política consciente del cliente (CAP) y la política de distribución de solicitudes consciente de la carga de trabajo (WARD).

Recurso web virtuales - VWR (Nakai et al., 2011) es un algoritmo de balanceo de carga para distribuir solicitudes de clientes entre servidores replicados. El protocolo para limitar la tasa de re direccionamiento evita la sobrecarga de los servidores remotos. VWR permite a los servidores sobrecargados reducir su carga de trabajo y redirigir la carga excedente al servidor web más liviano. Los servidores pueden aceptar o rechazar solicitudes, cuando una solicitud es rechazada entonces otro servidor seleccionado aleatoriamente procesa la solicitud.

Cola de trabajos ociosos - JIQ (Lu et al., 2011) es un algoritmo de balanceo de carga dinámicamente escalable con despachadores distribuidos en centros de datos. La idea central es desacoplar el descubrimiento de servidores ligeramente cargados de la asignación de trabajo. El algoritmo consiste en dos procesos de balanceo de carga para asignar trabajos en servidores inactivos. En el proceso de asignación de trabajo, las estructuras I-cola mantienen la información de los procesadores ociosos; cuando un trabajo llega al sistema, el despachador asigna el trabajo al primer elemento de la I-cola. Si la I-cola está vacía, el despachador envía el trabajo a un procesador elegido al azar. Este algoritmo reduce la duración promedio de los trabajos en las colas de espera sin incurrir en sobrecostos de comunicación.

Multiprocesamiento sin bloqueo - LFM (Liu et al., 2011) es una solución de balanceo de carga para entornos con múltiples núcleos. El núcleo de Linux modificado garantiza la sesión de los usuarios con el mismo servidor de fondo sin bloquear las sesiones o usar memoria compartida. Una función hash, basada en la matriz Toeplitz, reenvía las solicitudes a cada proceso. El objetivo es mejorar el rendimiento general del sistema.

Modelo de decisión de balanceo de carga centralizado - CLBDM (Radojević y Žagar, 2011) es un módulo de balanceo de carga para entornos hospedados en la nube. El algoritmo utiliza la información de los entornos virtualizados y la experiencia del usuario para influir en la decisión del balanceo de carga. Un

módulo central supervisa todo el sistema y trabaja con sensores para medir el tiempo necesario para completar cada acción. CLBDM redistribuye dinámicamente los recursos en servidores virtuales y detecta problemas.

Asignación de VMs a recursos físicos - VMMP (Ni et al., 2011) es una política de balanceo de carga para VM basada en múltiples recursos para entornos de nube privada. El algoritmo calcula el peso de cada nodo donde un valor más alto representa más recursos disponibles. Luego, VMMP usa un enfoque de probabilidad para seleccionar un nodo; esta probabilidad es proporcional al peso de cada nodo. El ambiente considera recursos bidimensionales (CPU y memoria) para evaluar el balanceo de carga.

Balanceo de carga consciente de la potencia - PALB (Galloway et al., 2011) es un enfoque de balanceo de carga para IaaS consciente de la potencia. El balanceador mantiene información sobre el estado de todos los nodos y decide la cantidad de nodos de cómputo que deben estar operando según el porcentaje de utilización. El algoritmo está compuesto por tres secciones: la primera sección asigna las VMs a los nodos, la segunda sección se utiliza para encender nodos adicionales, y la última sección es responsable de apagar nodos ociosos. Las organizaciones pueden ahorrar entre el 70% y el 97% de la energía consumida según la distribución de las tareas y el tamaño de las VMs.

Ejecución actual igualmente distribuida - ESCE (Kaur, 2012) es una estrategia para mejorar la calendarización de trabajos y la asignación de recursos en la nube. ESCE reduce el tiempo de respuesta y el tiempo de procesamiento, por lo tanto, disminuye el costo operativo. El calendarizador estima el tamaño del trabajo y verifica la capacidad y disponibilidad de las VMs. Una vez que el tamaño del trabajo y el recurso disponible coinciden, el planificador de tareas asigna el trabajo en la cola del recurso.

El enfoque de balanceo basado en diferentes técnicas tiene algunas ventajas sobre otros enfoques, sin embargo, varias de estas técnicas no pueden ser aplicadas en CVoIP porque usan técnicas de redistribución de la carga, requieren información no disponible en la nube, implementan esquemas basados en colas de trabajo, utilizan abstracciones básicas de ambientes nube, etc.

El análisis de los trabajos de investigación aporta ideas sobre estas técnicas a tener en cuenta en el diseño de nuevos algoritmos. Apagar o migrar VMs no es completamente adecuado en CVoIP porque ambas implican la degradación de la calidad y dependen significativamente de la utilización y el tiempo de renta de las VMs. Sin embargo, ambas técnicas pueden ser aplicadas teniendo en cuenta las restricciones necesarias. Varias técnicas establecen umbrales para aceptar trabajos y realizan estimaciones de la carga para garantizar la calidad de servicio. Las características descritas previamente son consideradas para el diseño de algoritmos de balanceo de carga en CVoIP.

La Tabla 1 muestra el resumen de las áreas de aplicación de los algoritmos. La Tabla 2 presenta las principales características de los algoritmos de balanceo de carga, y la Tabla 3 resume las métricas utilizadas para estudiar la calidad de los algoritmos.

Tabla 1. Ambientes de los algoritmos de balanceo de carga.

	Algoritmo	Centros de datos	Multi nucleos	Grid	Cómputo distribuido	Cómputo en la nube	Federación de nubes	Servicios web	Almacenamiento de red
VD	VectorDot	•							
WCAP	Decentralized Workload and Client Aware Policy				•				
VWR	Virtual Web Resource							•	
JIQ	Job-Idle-Queue	•							
LFM	Lock-Free Multiprocessing		•						
SVB	Scheduling Strategy on LB of VM Resources					•			
CLBVM	Central LB policy for VMs							•	
LBVS	Load Balance Strategy for Virtual Storage								•
TSLB	Task Scheduling Based on LB					•			
HF-1	Honeybee Foraging Behavior							•	
HF-2	Honey bee Foraging Behavior					•			
BRS	Biased Random Sampling							•	
AC	Active Clustering							•	
TBSLB-PSO	Task-Based System Load Balancing Using Particle Swarm Optimization					•			
LBS-BF	Load Balance Scheduling Based on firefly					•			
ACCLB	Ant Colony and Complex Network Theory						•		
OLB +LBMM	Two-phase scheduling					•			
ED	Event-Driven					•			
Carton	(LB + DRL)					•			
CB	Compare and Balance					•			
TSPBRR	Time Slice Priority Based Round Robin.					•			
ESCE	Equally Spread Current Execution					•			
CLBDM	Central Load Balancing Decision Model.					•			
AMR	Azure MapReduce				•				
VMMP	Virtual Machine to Physical Machine Mapping					•			
FBRR	Fuzzy Based Round Robin	•							
PALB	Power Aware Load Balancing					•			
LBMM	Load Balancing Min Min			•					
Min-Min	Min-Min			•					
Max-Min	Max-Min			•					
SOAS	Self-organized agent system for dynamical				•				
RoC-LB	Rate of change load balancing algorithm				•				

Tabla 2. Principales características de los algoritmos de balanceo de carga.

	Centralizado	Descentralizado	Jerárquico	En línea	Fuera de línea	No clarividente	QoS	Costo de migración	Retardo de red	Información histórica	Estáticos	Dinámicos	Replicación	Migración
VD			•		•	•						•		VM, VS, y tráfico
WCAP			•	•		•						•	Servidor	
VWR	•			•		•			•			•	Servidor	solicitudes
JIQ		•		•		•						•		
LFM	•			•		•						•		
SVB	•			•		•		•		•		•		VM
CLBVM	•			•		•						•		VM
LBVS	•									•		•	Datos/Archivos	
TSLB	•			•		•				•		•		VM
HF-1		•		•		•						•		Trabajos
HF-2		•			•							•		Tareas
BRS		•		•		•						•		Trabajos
AC		•		•		•						•		Trabajos
TBSLB-PSO	•				•			•				•		Tareas
LBS-BF														Tareas
ACCLB		•		•								•		Trabajos
OLB+LBMM	•					•				•	•			
ED	•			•		•	•					•	Servidor	Sesiones
Carton		•			•	•						•		
CB		•			•	•		•				•		VM
TSPBRR	•				•						•			
ESCE	•			•		•				•		•		
CLBDM	•			•		•						•		Sesiones y VM
AMR		•												
VMMP	•			•		•						•		
FBRR	•			•		•						•		
PALB	•			•		•						•		
LBMM	•				•	•				•		•		Tareas
Min-Min	•				•	•				•	•			
Max-Min	•				•	•				•	•			
SOAS		•		•		•						•		Tareas
RoC-LB		•		•	•							•		Trabajos

Tabla 3. Métricas utilizadas en los algoritmos de balanceo de carga.

	Utilización	Desempeño	Sobrecosto	Tiempo de respuesta	Escalabilidad	Tolerancia a fallos	Rendimiento	Consumo de energía
VD	•							
WCAP	•	•			•	•		
VWR	•	•						
JIQ		•	•	•				
LFM		•					•	
SVB	•		•					
CLBVM	•	•		•			•	
LBVS		•		•	•	•		
TSLB	•	•		•				
HF-1		•			•		•	
HF-2				•		•		
BRS		•			•		•	
AC		•			•		•	
TBSLB-PSO								
LBS-BF								
ACCLB	•	•			•	•		
OLB +LBMM	•	•						
ED	•				•			
Carton	•	•	•					
CB	•		•					
TSPBRR	•			•				
ESCE	•			•				
CLBDM	•			•				
AMR		•			•			
VMMP	•							
FBRR	•			•				
PALB	•							•
LBMM	•		•	•			•	
Min-Min	•		•	•			•	
Max-Min	•		•	•			•	
SOAS	•						•	
RoC-LB		•						

Capítulo 3. Definición del problema

En este capítulo se describe el planteamiento del problema y los elementos más representativos que afectan la asignación de llamadas en el modelo de CVoIP. Se establece la definición formal del problema con notaciones, métricas y criterios de evaluación. Posteriormente, se detallan las estrategias de asignación de llamadas y los modelos de predicción. Al final del capítulo se presenta el análisis de la carga de trabajo y el marco de simulación.

3.1 Planteamiento del problema

El objetivo fundamental de esta tesis es proponer un modelo de CVoIP y diseñar y analizar algoritmos de asignación de llamadas en ambientes nube con incertidumbre. El sistema ejecuta los trabajos de forma compartida y la información sobre la llegada de los trabajos y el tiempo de procesamiento son desconocidos; el tiempo de procesamiento de un trabajo es conocido solo después de su finalización. Los parámetros y la utilización de los recursos cambian constantemente a lo largo del tiempo.

Un centro de datos es una infraestructura que agrupa computadoras, comunicaciones y sistemas de almacenamiento para el procesamiento de información, estos son la infraestructura central para la computación en la nube. En los centros de datos tradicionales, las aplicaciones y los datos están vinculados a servidores específicos y subsistemas de almacenamiento. En el cómputo en la nube, la infraestructura subyacente ejecuta las aplicaciones en VMs de manera flexible (leve acoplamiento) y permite compartir recursos fácilmente.

La virtualización de recursos es parte fundamental del cómputo en la nube porque posibilita la ejecución de varias VMs en un único servidor de procesamiento. La virtualización reduce los ciclos ociosos de CPU, maximiza la eficiencia del servidor y minimiza el desperdicio de recursos. Las ventajas del paradigma de VMs son: inician, finalizan y migran fácilmente entre los recursos y ejecutan una gran variedad de trabajos. La virtualización crea sistemas nube elásticos, escalables y ecológicos.

El cómputo en la nube difiere de los entornos informáticos previamente estudiados en la forma de introducir incertidumbre en el proceso de cómputo. El papel de la incertidumbre en el balanceo de carga no se ha examinado adecuadamente en la literatura. Esta investigación estudia el papel de la incertidumbre en el balanceo de carga en la nube computacional. Existen varias fuentes principales de

incertidumbre: elasticidad dinámica, cambio de rendimiento, virtualización de la máquina, aplicaciones flexibles con leve acoplamiento a la infraestructura, retardo en el inicio de los recursos virtuales, etc.

La elasticidad evita la práctica derrochadora de aprovisionamiento excesivo y permite manejar la carga de trabajo impredecible. Los algoritmos elásticos de balanceo de carga distribuyen el tráfico entrante para lograr una mayor QoS, detectan recursos sobrecargados y redireccionan el tráfico a recursos con poca carga. Si todos los nodos de la nube están sobrecargados, el balanceador amplia la capacidad del sistema para manejar solicitudes en respuesta al tráfico entrante. Del mismo modo, la nube reduce el número de nodos cuando estos no tienen carga (nodos ociosos). La capacidad de recursos informáticos aumenta o disminuye en tiempo real, de acuerdo con el uso de CPU, memoria, ancho de banda, etc.

El balanceador equilibra la carga de trabajo en la nube para evitar la sobrecarga de los recursos, sin embargo, la carga puede cambiar drásticamente, lo cual es difícil de predecir. Los modelos de predicción estiman el tiempo de llegada y el tiempo de ejecución de los trabajos enviados; los datos históricos pueden mejorar las estimaciones debido a que los usuarios tienden a ejecutar los mismos programas repetidamente. Adicionalmente, técnicas como la corrección de predicción y propagación mejoran la precisión de la estimación.

Los recursos compartidos entre las VMs y los trabajos de los usuarios generan cambios dinámicos del rendimiento en tiempo de ejecución. La mayoría de los algoritmos de balanceo de carga suponen que el aprovisionamiento de VMs es predecible y estable en su rendimiento: las VMs están configuradas con una cantidad fija de MIPS, memoria y ancho de banda. Las suposiciones anteriores no son válidas en la infraestructura de la nube real, la mayoría de los proveedores garantizan una cierta velocidad de procesador, capacidad de memoria y almacenamiento local para cada VM.

El rendimiento real de una VM está sujeto al hardware físico subyacente, así como al uso de recursos compartidos por otras VMs asignadas al mismo recurso físico. Desafortunadamente, los simuladores de nube disponible no capturan adecuadamente la falta de homogeneidad y los cambios de rendimiento dinámico inherentes a las infraestructuras compartidas no uniformes, como las nubes computacionales (Bux y Leser, 2013).

3.2 Calidad de servicio en CVoIP

Los servicios de VoIP imponen restricciones estrictas y tienen factores más sensibles. El procesamiento y la entrega de las llamadas determinan la QoS. El procesamiento de llamadas incluye factores como el tiempo para inicializar y finalizar la llamada, y el tiempo para convertir la parte de voz de

las llamadas en paquetes transportados a través de la red. La calidad de voz es el aspecto más importante del procesamiento de llamadas y muchos factores en el cómputo en la nube pueden afectarlo.

3.2.1 Utilización del CPU

La calidad de la voz es percibida subjetivamente por el oyente. Un punto de referencia común utilizado para determinar la calidad de la voz es MOS. Esta medida evalúa la calidad del habla proporcionado por un códec. Cada códec proporciona cierta calidad de voz solo si la utilización del procesador es lo suficientemente baja. Teóricamente, la utilización del 100% del procesador proporciona el mejor rendimiento esperado. Sin embargo, Eleftheriou (2010) mostró que el CPU no puede manejar el estrés cuando la utilización supera el 85%, a partir de este nivel de utilización comienzan a aparecer jitter y audio cortado. Adicionalmente, el trabajo no analiza la influencia de la memoria en la reducción de la calidad de la voz.

Durante el procesamiento de una llamada, el códec incrementa el ancho de banda utilizado pero la utilización que aporta el mismo códec al CPU es más significativa. Montazerolghaem et al. (2016) indican que el ancho de banda consumido por 6,500 llamadas no supera los 100 Mbps, y 10,000 llamadas no alcanzan los 400 Mbps.

La Tabla 4 muestra el ancho de banda utilizado por diferentes códecs. Las llamadas de VoIP usan dos flujos de audio, uno por cada punto final, por lo tanto, una llamada emplea el doble de ancho de banda. El número de llamadas admitidas en una conexión de 100 Mbps está entre 6,000 y 25,000, dependiendo del códec usado por las llamadas.

Tabla 4. Ancho de banda de diferentes códecs.

Códec	Bit Rate (kbps)	MOS	Ancho de banda (kbps)
G.711	64	4.1	87.2
G.729	8	3.92	31.2
G.723.1	6.3	3.9	21.9
G.723.1	5.3	3.8	20.8
G.726	32	3.85	55.2
G.726	24	-	47.2
G.728	16	3.61	31.5
G722_64k	64	4.13	87.2
ilbc_mode_20	15.2	NA	38.4
ilbc_mode_30	13.33	NA	28.8

Sin embargo, la cantidad de llamadas manejadas por el CPU es menor a la soportada por una conexión de 100 Mbps. Por lo tanto, el procesamiento de llamadas es una característica clave para garantizar la QoS. En este trabajo, se propone limitar la utilización del procesador para garantizar la QoS.

Las llamadas tienen diferente impacto en la utilización del procesador en función de las operaciones realizadas por Asterisk. La utilización del procesador es más baja cuando no se realizan operaciones de transcodificación, para este caso, Asterisk solo se encarga de enrutar la llamada. Sin embargo, el enrutamiento influye en la carga del procesador independientemente del tipo de códec. La Tabla 5 muestra la utilización del procesador para la llamada sin transcodificación presentada por Montoro y Casilari (2009).

Tabla 5. Utilización de llamadas sin transcodificación.

Protocolo	Códec	10 llamadas (%)	1 llamada (%)
SIP/RTP	G.711	2.36	0.236
SIP/RTP	G.726	2.13	0.213
SIP/RTP	GSM	2.58	0.258
SIP/RTP	LPC10	1.92	0.192

El rendimiento de los procesadores ATOM se analiza para VoIP (Eleftheriou, 2010) teniendo en cuenta las llamadas, la utilización, el consumo de energía, los mensajes de la base de datos, el registro y el rendimiento de las llamadas. El autor concluye que el CPU puede procesar de 70 a 500 llamadas con un 100% de utilización.

Además de la utilización del CPU, los proveedores deben considerar otros elementos inherentes a la infraestructura nube, uno de los más importantes es el despliegue de la infraestructura. La elasticidad del sistema CVoIP es un factor vital para garantizar la QoS y depende del tiempo despliegue de las VMs.

3.2.2 Retardo de inicio

La elasticidad es la capacidad de adquirir o liberar recursos de cómputo bajo demanda de forma dinámica. Algunas ventajas de la elasticidad en la nube abarcan: evitar la práctica de aprovisionamiento excesivo, adaptar la capacidad del sistema a las cargas de trabajo y disminuir el consumo de energía. Sin embargo, si las VMs tardan mucho tiempo en desplegarse pueden generar el problema de bajo aprovisionamiento que afecta el rendimiento de las aplicaciones.

El tiempo de retardos de inicio (StUp) de una VM es el tiempo requerido para inicializar la VM. Un StUp inesperado provoca reducción de la QoS y aprovisionamiento escaso de recursos. Para reducir los factores que afectan la ejecución de aplicaciones de tiempo crítico, es necesario estudiar los efectos de los retrasos de inicio de las VMs en la administración de sistemas CVoIP.

Mao y Humphrey (2012) estudian los StUPs de VMs en nubes y analizan la relación entre StUp y diferentes factores: tamaño de la imagen del SO, tipo de instancia, ubicación del centro de datos y cantidad

de instancias adquiridas al mismo tiempo. El StUp es el período de tiempo que los proveedores de la nube necesitan para encontrar un lugar en los centros de datos para aprovisionar VMs, asignar recursos (por ejemplo, direcciones IP) a las VMs y copiar / iniciar / configurar la imagen del SO.

Razavi et al. (2013) analizan la dependencia entre el tiempo de inicio de la VM, el tamaño del disco y el contenido. Los autores desarrollan y evalúan un enfoque para consolidar instancias de VMs y reducir el tamaño y contenido del disco. El objetivo es seleccionar un conjunto deseado de servicios para crear imágenes de VMs con tamaño mínimo requerido. Los autores aplican este enfoque a las imágenes de las VMs donde el usuario define paquetes de software necesarios para obtener la funcionalidad deseada.

Hoffman (2015) mide la velocidad de creación de la VM y el tiempo de acceso mediante SSH entre proveedores de nube ubicados en diferentes regiones del mundo. El análisis considera información recopilada durante un año y cubre diferentes regiones del mundo (Europa, Asia y EUA). Los resultados muestran que el mejor proveedor es aproximadamente cuatro veces más rápido que el peor, al considerar ambos procesos.

La Tabla 6 resume el StUp para varios SO y proveedores. La Tabla 7 presenta los tiempos promedio de creación de la VM y el tiempo de acceso al servicio SSH entre proveedores nube.

Tabla 6. Tiempo promedio de retardo de inicio para VMs.

Nube	SO	StUp (seg.)
EC2	Linux	96.9
	Windows	810.2
Azure	WebRole	374.8
	WorkedRole	406.2
	VMRole	356.6
Rackspace	Linux	44.2
	Windows	429.2

Tabla 7. Tiempo promedio de retardo de inicio para VMs con acceso SSH.

Nube	StUp (seg.)
Google Cloud Platform	31
Amazon Web Services	47
Vexxhost	47
Linode	57
DigitalOcean	89
Rackspace	128
Azure	138

En esta investigación se estudian los efectos de los retardos en el despliegue de la infraestructura (StUp) para el procesamiento de llamadas en sistema VoIP basado en la nube computacional.

3.3 Definición formal

En esta tesis se aborda el problema de distribución de llamadas de VoIP en un entorno nube. El modelo supone heterogeneidad de los recursos con diferente cantidad de servidores, memoria, ancho de banda, retardo de inicio, etc.

3.3.1 Modelo de la infraestructura

La infraestructura de VoIP en la nube consta de m agrupamientos de súper nodos heterogéneos $SNC_1, SNC_2, \dots, SNC_m$ con velocidades relativas $s_1, s_2, \dots, s_m$. Cada SNC_i, para todo $i = 1, \dots, m$, cuenta con m_i SNs. Cada SN_k^i, para todo $k = 1, \dots, m_i$, ejecuta $k_i(t)$ VMs en el tiempo t. Las VMs de los SNs son idénticas y tienen la misma capacidad de procesamiento. La máquina virtual VM_j se describe mediante una tupla $(st_j, size_j, StUp_j)$ que consiste en el tiempo de llegada $st_j \geq 0$, la capacidad de procesamiento $size_j$ en MIPS y el retardo de inicio $StUp_j$.

El SNC contiene un conjunto de enrutadores y conmutadores que transportan el tráfico entre los SNs y el mundo exterior. Un conmutador conecta un punto de redistribución o nodos de cómputo. Las conexiones de los procesadores son estáticas pero su utilización varia. La red de interconexión del SNC es local y la interconexión entre los SNCs se proporciona a través de Internet público.

3.3.2 Modelo de los trabajos

n llamadas independientes $J_1, J_2, \dots, J_n$ deben ser calendarizadas en un conjunto de SNCs. La llamada J_j está descrita por la tupla (r_j, p_j, u_j) que consiste en el tiempo de llegada $r_j \geq 0$, duración p_j (tiempo de vida) y contribución a la utilización del procesador u_j debido al códec empleado. El tiempo de llegada de una llamada es conocido solo hasta que la llamada entra al sistema y la duración es desconocida hasta que la llamada finaliza su ejecución. La utilización es constante para una llamada y está determinada por el códec y la capacidad de procesamiento de la VM.

Un trabajo se puede asignar a una sola VM, no se considera la duplicación de trabajos. Los trabajos asignados a una VM no pueden migrar a otra.

3.3.3 Criterios de optimización

Las características de los ambientes nube vuelven irrelevantes las métricas tradicionales para evaluar el costo del sistema. Para este tipo de sistemas, la métrica debe permitir al proveedor medir el costo del sistema en términos de la cantidad de VMs. El modelo de costos del proveedor considera una

función que depende del número de VMs alquiladas y su tiempo de alquiler (*rp*). La ecuación (1) denota el número de horas de facturación en el SNC_i.

$$\bar{b}_i = \sum_{r=0}^{c_{max}/rp} k_i(r * rp) \tag{1}$$

El total de horas de facturación en el SNC está definido por la ecuación (2).

$$\bar{b} = \sum_{i=1}^{m} \bar{b}_i \tag{2}$$

La reducción de calidad de servicio depende significativamente de la utilización de los recursos, en específico del CPU. Similar a la métrica de costo, la métrica de QoS debe permitir al proveedor medir la calidad de servicio en base a la utilización del CPU.

La ecuación (3) formula la degradación de la calidad de servicio en función de la utilización del CPU de la VM, donde st_j y ft_j definen el tiempo de renta de la VM_j, la función $vmu_j(t)$ proporciona información sobre la utilización (carga) de la VM_j en el tiempo t y la función de penalización $\gamma(\alpha) = ((\alpha - 0.7) * \overline{3.33})^2$ para $\alpha > 0.7$ y 0 en otro caso. La Figura 10 muestra el comportamiento de la reducción de la calidad de servicio en función de la carga del CPU (utilización) en la VM.

$$\bar{q}_i = \sum_{j=1}^{\bar{b}_i} \sum_{t=st_j}^{ft_j} \gamma(vmu_j(t)) \tag{3}$$

La reducción de calidad del servicio en el SNC está definida por la ecuación (4).

$$\bar{q} = \sum_{i=1}^{m} \bar{q}_i \tag{4}$$

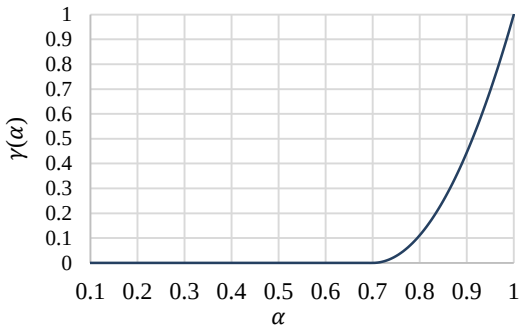

Figura 10. Reducción de la calidad de voz frente a la carga del procesador (utilización).

La cantidad de llamadas en espera es otro factor empleado por el proveedor de CVoIP para determinar la QoS. Esta métrica evalúa el número de llamadas que no son atendidas inmediatamente por falta de recursos. Si las VMs no pueden procesar las llamadas que llegan al sistema entonces las llamadas son enviadas a una cola, en espera por VMs disponibles. La ecuación (5) define la cantidad de llamadas en espera donde s_j precisa el tiempo de inicio de J_j y la función $\delta(\beta)$ es 1 si $\beta > 0$ y 0 en otro caso, determina el total de llamadas enviadas a la cola de espera.

$$\bar{c} = \sum_{j=1}^{n} \delta(s_j - r_j) \tag{5}$$

El problema de asignación de llamadas en sistemas CVoIP se puede formular como un caso especial de empaquetamiento dinámico de contenedores. Los contenedores representan las VMs y la altura de los elementos define la contribución de la llamada a la utilización de la VM. Este enfoque permite mejor adaptación a los factores de incertidumbres en la nube, como la elasticidad dinámica, el cambio de rendimiento, la virtualización, el acoplamiento flexible de la aplicación a la infraestructura, parámetros como la velocidad del procesador, el número de VMs y los retardos de inicio, entre otros.

3.4 Métodos de evaluación

El problema multi-objetivo de asignación de llamadas en CVoIP considera tres criterios para evaluar el rendimiento de los algoritmos: horas de facturación ($\bar{b}$), degradación de la calidad ($\bar{q}$) y llamadas en espera ($\bar{c}$). Un buen algoritmo de balanceo de carga logra un alto rendimiento de los recursos, mientras minimiza la degradación de la calidad y las llamadas en espera. Elegir un método para analizar problemas multi-objetivo no es trivial. Afortunadamente, existen varios enfoques para estudiar este tipo de problemas.

3.4.1 Degradación del desempeño

La metodología de degradación del desempeño fue propuesta por Tsafrir et al. (2007) y ha sido aplicada en diferentes estudios sobre calendarización para determinar la estrategia con mejor desempeño (Tchernykh et al., 2016). El análisis considera la misma importancia para cada métrica y muestra cómo el valor de la métrica generada por los algoritmos se acerca a la mejor solución encontrada. El análisis se lleva a cabo de la siguiente manera; inicialmente, se calcula la degradación del rendimiento de cada estrategia con respecto a la estrategia con mejor rendimiento (error relativo), la ecuación (6) modela el cálculo del error relativo.

$$\left(\frac{valor\ de\ la\ métrica}{mejor\ valor\ obtenido\ de\ la\ métrica} - 1\right) \cdot 100 \quad (6)$$

Posteriormente, los valores para todos los escenarios son promediados y las estrategias son clasificadas, la estrategia con la degradación del desempeño promedio más baja obtiene el rango 1. El análisis identifica las estrategias con mejor funcionamiento en diferentes escenarios, es decir, trata de encontrar un compromiso para todos los casos de prueba. Por ejemplo, el rango de una estrategia no podría ser el mismo para cualquiera de los escenarios individuales.

3.4.2 Frente de Pareto y cubrimiento de conjuntos

La optimización multi-objetivo no se limita a encontrar una única solución para un problema dado, como el enfoque de degradación media, sino un conjunto de soluciones conocido como conjunto óptimo de Pareto. Una solución puede representar la mejor solución con respecto a las horas de facturación, mientras que otra podría ser la mejor con respecto a la QoS. El objetivo es elegir una solución adecuada y obtener un conjunto de soluciones compromiso que represente una buena aproximación al frente de Pareto.

Dos características importantes de una buena técnica de solución son la convergencia al frente de Pareto y la diversidad para muestrear el frente. Las distribuciones de las aproximaciones se pueden comparar con el frente de Pareto para encontrar la mejor estrategia para un problema dado. A menudo, el resultado de los problemas multi-objetivo se compara a través de la observación visual del espacio de la solución.

Un enfoque más formal y estadístico utiliza la métrica de cubrimiento de conjuntos (Zitzler, 1999). La ecuación (7) define la métrica de cubrimiento de conjuntos, donde $SC(A, B)$ calcula la proporción de soluciones en B que son dominadas por soluciones de A.

$$SC(A, B) = \frac{|\{b \in B; | \exists a \in A\ ;\ a \leq b\}}{|B|} \quad (7)$$

Un valor de la métrica $SC(A, B) = 1$ indica que todas las soluciones de B están dominadas por A, mientras que $SC(A, B) = 0$ denota que ningún miembro de B está dominado por A. De esta manera, el frente no dominado A es mejor con respecto a B en proporción al valor de $SC(A, B)$. El operador de dominancia no es simétrico, por lo que $SC(A, B)$ no es necesariamente igual a $1 - SC(A, B)$. $SC(A, B)$ y $SC(B, A)$ deben ser calculados para entender cuántas soluciones de A están cubiertas por B y viceversa.

3.4.3 RMSD y SMAPE

La precisión de los modelos de predicción se evalúa con: la raíz de la desviación cuadrática media (RMSD) y el error simétrico porcentual absoluto medio (SMAPE). Estas métricas son usadas en estadística para medir la precisión de un método para construir valores ajustados de series de tiempo (Calheiros et al., 2015; Simionovici et al., 2013; Simionovici et al., 2015). Las ecuaciones (8) y (9) definen RMSD y SMAPE, respectivamente.

$$\text{RMSD} = \sqrt{\frac{\sum_{i=1}^{n}(F(i) - A(i))^2}{n}} \tag{8}$$

$$SMAPE = \frac{\sum_{i=1}^{n}|F(i) - A(i)|}{\sum_{i=1}^{n}|A(i) + F(i)|} \tag{9}$$

n especifica el tamaño del conjunto, $A(i)$ define el valor real de utilización y $F(i)$ precisa el valor estimado de utilización. Valores más pequeños de RMSD y SMAPE son mejores.

3.5 Asignación de llamadas

El problema de asignación de llamadas es similar al problema de empaquetamiento dinámico de contenedores, una variación del problema clásico de optimización NP difícil con alta relevancia teórica e importancia práctica. El empaquetamiento es un área muy activa de investigación en algoritmos y en la comunidad de investigación de operaciones.

En el modelo de CVoIP, el calendarizador determina si la llamada es atendida por una de las VMs actualmente disponibles o si debe inicializar una nueva instancia de VM. El calendarizador solo conoce la contribución de la llamada u_j a la utilización de la VM y toman decisiones sin conocimiento de la duración de la llamada, la tasa de llegada de llamadas, etc.

La existencia temporal de los elementos es la principal novedad de este problema. La vida útil de las llamadas y la asignación de llamadas determinan el estado de las VMs. A diferencia de la formulación estándar, los contenedores siempre están abiertos y son dinámicos, incluso completamente empaquetados. Los elementos en los contenedores pueden finalizar (terminación de llamadas) y la utilización puede cambiar en cualquier momento, luego las VMs usan el espacio libre para procesar más llamadas.

El modelo de simulación de CVoIP establece tres parámetros configurables: umbral de utilización (UT), umbral de renta (RT) e intervalo de predicción (PI) con el objetivo de estudiar escenarios reales, con

una variedad de entornos heterogéneos. Estos parámetros modelan diferentes preferencias de objetivos, cargas de trabajo y propiedades de la nube.

El umbral de utilización (UT) define un límite para incrementar la cantidad de recursos en el sistema. Si la utilización de todas las VMs en el sistema exceden UT entonces el balanceador inicia más instancias de VMs. UT considera la utilización del CPU para garantizar la calidad del servicio ya que el CPU no puede manejar el estrés cuando la utilización es superior a cierto nivel. La escalabilidad del sistema CVoIP depende completamente de UT.

El umbral de renta (RT) define un intervalo de tiempo previo a la terminación del tiempo de renta de la VM. En el problema de CVoIP, una VM finaliza su ejecución cuando esta ociosa y su tiempo de renta termina. Las estrategias con RT reducen el número de VMs en ejecución al limitar la asignación de llamadas RT minutos antes de finalizar la ejecución las VMs. RT evita el alquiler de la VM por una hora adicional debido a la llegada de llamadas justo antes de finalizar el tiempo de renta de la VM.

El intervalo de predicción (PI) es un intervalo de tiempo para el cual se realiza una predicción. El predictor realiza una estimación de la utilización en cada paso de tiempo (TS). Un TS pequeño proporciona una mejor predicción, pero aumenta la sobrecarga del sistema. En caso contrario, un TS grande reduce la sobrecarga del sistema, pero disminuye la precisión de la predicción.

El modelo de CVoIP considera dos escenarios para analizar el comportamiento de las estrategias de asignación. En ambos escenarios, el planificador no tiene información de la tasa de llegada de llamadas y toma decisiones dependiendo del estado actual del sistema.

En el primer escenario se estudian las estrategias de asignación con diferentes UTs y un StUp igual a cero, el objetivo de esta configuración es tener una referencia de un escenario ideal donde el procesamiento de llamadas es el único factor que afecta la calidad de voz. Este escenario supone que el planificador puede realizar predicciones de carga perfecta, por lo que las nuevas VMs se inicializan a priori y están disponibles cuando se necesitan. Dos casos son analizados bajo estas circunstancias.

En el primer caso, UT es igual al 70% de la utilización de los recursos (UT = 0.7) y se puede garantizar el 100% de la calidad del servicio. Por lo tanto, el problema de asignación multi-objetivo en CVoIP se reduce a un problema mono objetivo. La Figura 11 muestra un ejemplo de asignación de llamadas con garantía de QoS. Los recuadros azules representan las VMs donde la altura define la capacidad máxima de procesamiento y la anchura precisa el tiempo de renta. Las líneas simbolizan la utilización de la VM debido al procesamiento de llamadas y las líneas amarillas definen los UTs. En el ejemplo, VM_2 inicializa

su ejecución cuando la utilización de VM_1 sobrepasa UT (línea roja) y comienza a procesar las llamadas inmediatamente.

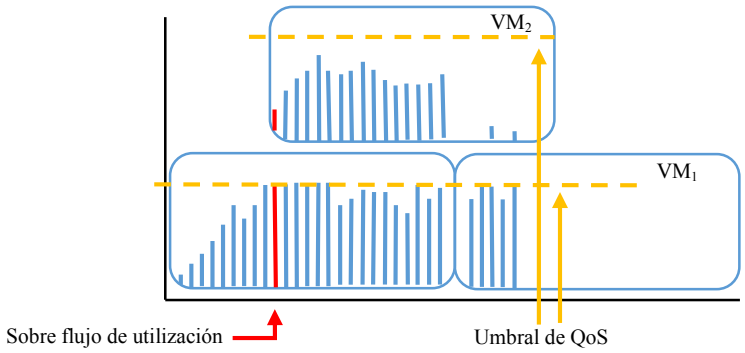

Figura 11. Asignación de llamadas con QoS garantizada.

En el segundo caso, UT es igual al 100% de la utilización de recursos (UT =1.0) y, por lo tanto, el sistema no puede garantizar la calidad de servicio. En este escenario, se estudia el comportamiento de las estrategias con respecto a la utilización de los recursos y la degradación de la QoS. La Figura 12 muestra un ejemplo de asignación de llamadas sin garantía de QoS, la VM_2 inicializa su ejecución cuando la utilización de VM_1 excede la capacidad máxima.

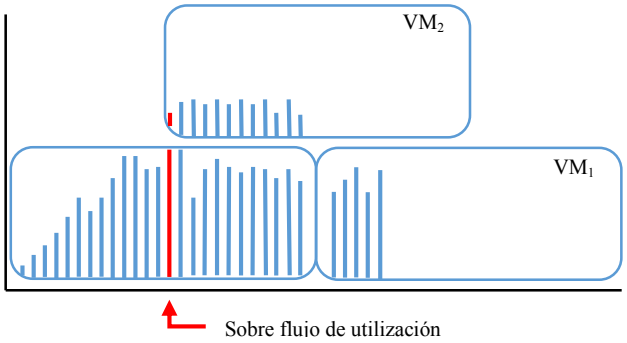

Figura 12. Asignación de llamadas.

En el segundo escenario se estudian las estrategias de asignación con varios StUps. Para este escenario, UT es igual al 70% de la utilización de la VM (UT = 0.7) y se toma en cuenta el StUp de la VM. La Figura 13 muestra un ejemplo de asignación de llamadas con retraso de inicio en la VM. El calendarizador inicializa VM_2 cuando la utilización de VM_1 es superior a UT (línea verde). Durante el StUp de VM_2, VM_1 continúa procesando llamadas con una utilización superior a UT, por lo tanto el sistema reduce la QoS. El peor caso ocurre cuando VM_1 no tiene suficientes recursos para procesar las llamadas entrantes (línea roja). En esta situación, el sistema envía las llamadas entrantes a una cola de espera (llamadas en espera) donde permanecen hasta que el sistema tenga recursos disponibles.

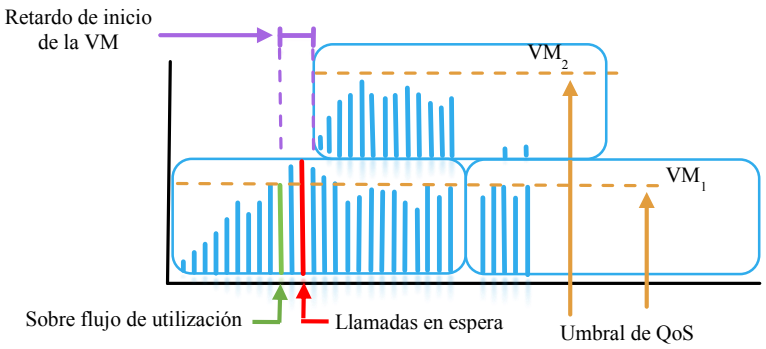

Figura 13. Asignación de llamadas con retardo de inicio.

En el segundo escenario, UT no garantiza la calidad de servicio y las llamadas en espera se convierten en un factor a considerar debido a su impacto en la degradación de la QoS. En este escenario, las estrategias de asignación utilizan varias técnicas de predicción con el objetivo de garantizar la QoS y evitar las llamadas en espera, los modelos de predicción se describen en secciones posteriores.

Las estrategias de asignación de llamadas se agrupan por el tipo y la cantidad de información utilizada para la asignación: (1) libres de conocimiento (KF), sin información sobre aplicaciones y recursos, (2) conscientes de la utilización (UA), con información de utilización del CPU, (3) conscientes del alquiler (RA), con información sobre el tiempo de alquiler de la VM (inicio y finalización) y (4) conscientes de la carga (LA), con información de la carga sobre la VM en cada TS. Además, las estrategias incorporan un mecanismo para reducir la cantidad de VMs activas. El umbral de renta (RT) limita la asignación de llamadas RT minutos antes de que la VM finalice su tiempo de renta.

La Tabla 8 resume las estrategias de asignación que solicitan nuevas VMs cuando la asignación de las llamadas entrantes excede UT en todas las VMs en el sistema. La Tabla 9 muestra las estrategias de asignación de llamadas con estimación de carga. Las estrategias realizan una estimación de la utilización cada TS para el siguiente PI, si la estimación supera UT en todas las VMs entonces el balanceador inicializa una nueva VM.

El Algoritmo 1 describe una de las estrategias propuestas en este trabajo de investigación basada en Best Fit con umbral de renta. BFit_RT genera dos listas: la lista admisible de nodos de voz (AVNL) y no AVNL (nAVNL). La lista AVNL contiene los nodos de voz (VNs) que finalizan su tiempo de renta después de RT minutos. Ambas listas son ordenadas por el nivel de utilización de forma no decreciente. El Algoritmo 1 emplea el término VN en lugar de VM para tener coherencia con la terminología de asignación de llamadas.

Tabla 8. Estrategias de asignación de llamadas.

tipo	nombre	Descripción
KF	Rand	Asigna el trabajo *j* a la VM aleatoriamente usando una distribución uniforme
	RR	Asigna el trabajo *j* a la VM usando el algoritmo Round Robin
UA	FFit	Asigna el trabajo *j* a la primer VM capaz de ejecutarlo
	BFit	Asigna el trabajo *j* a la VM con mayor utilización
	WFit	Asigna el trabajo *j* a la VM con menor utilización
RA	MaxFTFit	Asigna el trabajo *j* a la VM con el tiempo de finalización más lejano
	MidFTFit	Asigna el trabajo *j* a la VM con el tiempo más cercano a la mitad del tiempo de renta.
	MinFTFit	Asigna el trabajo *j* a la VM con el tiempo de finalización más cercano
KF+RT	Rand_05 Rand_10 Rand_15 RR_05 RR_10 RR_15	Asigna el trabajo *j* a la VM con RT igual a 5, 10 y 15 minutos usando las estrategias Rand y RR
UA + RT	BFit_05 BFit_10 BFit_15 FFit_05 FFit_10 FFit_15 WFit_05 WFit_10 WFit_15	Asigna el trabajo *j* a la VM con RT igual a 5, 10 y 15 minutos usando las estrategias FFit, BFit y WFit.
RA + RT	MaxFTFit_05 MaxFTFit_10 MaxFTFit_15 MidFTFit_05 MidFTFit_10 MidFTFit_15 MinFTFit_05 MinFTFit_10 MinFTFit_15	Asigna el trabajo *j* a la VM con RT igual a 5, 10 y 15 minutos usando las estrategias MaxFTFit, MidFTFit y MinFTFit.

El balanceador busca en la lista nAVNL cuando ningún VN de la lista AVNL puede procesar la llamada entrante debido a que excede UT. Un VN de nAVNL procesa la llamada si no excede UT, posteriormente el VN cambia a la lista de AVNL y programa una hora más de tiempo de alquiler. Si ningún VN acepta la llamada porque degrada su QoS entonces el algoritmo busca el VN donde la reducción de la QoS sea menor. El peor caso sucede cuando ningún VN puede dar servicio a la llamada, el sistema está lleno, entonces la llamada queda en espera por un VN disponible.

Tabla 9. Estrategias de asignación de llamadas con predicción dinámica de la carga.

Tipo	Nombre	Descripción
LA	Rand_stUp Rand_s10 Rand_s20 Rand_s30 RR_stUp RR_s10 RR_s20 RR_s30	Asigna el trabajo *j* a la VM usando las estrategias Rand y RR con PI = 10, 20, 30 and StUp seg.
RA+LA	MaxFTFit_stUp MaxFTFit_s10 MaxFTFit_s20 MaxFTFit_s30 MidFTFit_stUp MidFTFit_s10 MidFTFit_s20 MidFTFit_s30 MinFTFit_stUp MinFTFit_s10 MinFTFit_s20 MinFTFit_s30	Asigna el trabajo *j* a la VM usando las estrategias MaxFTFit, MidFTFit y MinFTFit con PI = 10, 20, 30 and StUp seg.
UA +LA	BFit_stUp BFit_s10 BFit_s20 BFit_s30 FFit_stUp FFit_s10 FFit_s20 FFit_s30 WFit_stUp WFit_s10 WFit_s20 WFit_s30	Asigna el trabajo *j* a la VM usando las estrategias BFit, FFit y WFit con PI = 10, 20, 30 and StUp seg.

Algoritmo 1. Best Fit con umbral de renta (BFit_RT)

Entrada: Lista de nodos de voz (VNlist), UT, RT y llamada
Salida: Asignación de llamada en un nodo de voz

```
1   vnIndex ← -1 y Crea lista AVNL y nAVNL con tiempo RT
2   Ordena AVNL por utilización en orden decreciente
3   Ordena nAVNL por utilización en orden decreciente
4   vnIndex ← Best_Fit(AVNL, UT, llamada)
5     si vnIndex < 0 entonces
6       vnIndex ←Best_Fit(nAVNL, UT, llamada)
7       si vnIndex < 0 entonces
8         vnIndex ←Best_Fit(nAVNL, 1.0, llamada)
9         si vnIndex < 0 entonces
10          vnIndex ←Best_Fit(AVNL, 1.0, llamada)
11  si vnIndex < 0 entonces
12    Envia llamada a la cola de espera e inicia un nuevo nodo
13  sino
14      Envia llamada al nodo de voz con índice vnIndex
15      Si (nodo de voz con índice vnIndex está en nAVNL)
16        Mueve el nodo de voz con índice vnIndex a AVNL
17        Renta un hora más el nodo de voz con índice vnIndex
```

Una predicción adecuada del arribo de llamadas reduce la degradación de la calidad y el número de llamadas en espera porque el sistema tiene recursos suficientes para atender las peticiones. Las estrategias de asignación conscientes de la carga utilizan tres modelos de predicción para estimar la carga de trabajo, la siguiente sección describe dichos modelos de predicción.

3.5.1 Asignación de llamadas con predicción dinámica de la carga

En esta sección se describen los modelos de predicción considerados para estimar la carga de trabajo. Una estimación de carga adecuada incrementa el rendimiento de la asignación de llamadas. Las estrategias de balanceo usan predicción para anticipar la carga del sistema debido al número de llamadas entrantes. La estimación de carga favorece el proceso de distribución de carga eficiente, por lo tanto, minimiza los costos de infraestructura y la degradación de la calidad del servicio.

La Figura 13 muestra un ejemplo del impacto de StUp en el proceso de asignación de llamadas. El objetivo de estimar la carga es reducir el efecto de StUP en el sistema CVoIP. La Figura 14 ejemplifica el proceso de asignación de llamadas con predicción de carga. El balanceador estima la carga para el siguiente PI (líneas punteadas), cuando la predicción de carga supera UT (línea roja punteada) entonces el balanceador inicia una nueva instancia de VM, antes de que las llamadas entrantes degraden la QoS. Todos los modelos de predicción estiman la carga del sistema y solicitan VMs en función de la estimación, lo que permite al balanceador calcular la cantidad de VMs en cada TS.

Esta investigación analiza tres modelos de predicción que utilizan diferentes enfoques para estimar la carga en el sistema.

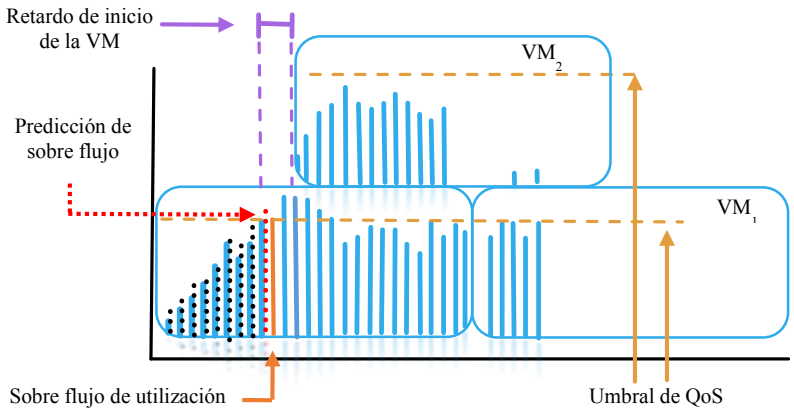

Figura 14. Asignación de llamadas con predicción y retardo de inicio.

3.5.1.1 Asignación de llamadas con predicción basada en tasa de cambio

Tasa de cambio - RoC (Campos y Scherson, 2000) es un algoritmo dinámico y distribuido de balanceo de carga donde el objetivo es minimizar el tiempo ocioso de los procesadores sin incurrir en altos sobrecostos. RoC calcula la diferencia de carga (Δ) entre dos TSs y utiliza Δ como una estimación para el siguiente PI. Δ ayuda a asignar las llamadas de manera eficiente. TS es un parámetro adaptativo y su longitud puede variar, un muestreo más fino de TS mejora el balanceo del sistema, pero aumenta la sobrecarga.

Las estrategias de asignación de llamadas usan el concepto de Δ como un mecanismo para predecir la carga del sistema y por lo tanto las solicitudes de nuevas VMs, las cuales están disponibles después del retardo StUp. El balanceador emplea Δ para estimar el número de VMs cada PI e inicializa VMs antes de que las llamadas entrantes degraden la QoS.

En el modelo CVoIP, $u_i(t)$ define la utilización del SNC_i en el tiempo t, la tasa de cambio de la carga en el tiempo t se define en la ecuación (10).

$$\Delta_i(t) = \big(u_i(t) - u_i(t - TS)\big) \tag{10}$$

La ecuación (11) detalla la estimación de la carga futura del sistema.

$$u_i(t + PI) = u_i(t) + \Delta_i(t) \tag{11}$$

La Figura 15 muestra un escenario de predicción basado en RoC, las líneas continuas representan la utilización real y las líneas punteadas simbolizan la utilización estimada. Si la utilización del sistema es mayor a UT entonces el balanceador solicita una nueva VM. UT es un parámetro ajustable que depende del umbral para garantizar QoS. Por ejemplo, en el momento T_4, el sistema inicia una nueva VM basada en la estimación de utilización para T_5, este es un ejemplo de aprovisionamiento excesivo. En el momento T_5, el sistema estima la utilización menor a UT en T_6 y no solicita una nueva VM. La utilización real es mayor a UT y causa degradación de QoS (ejemplo de aprovisionamiento escaso). Finalmente, en el momento T_9, el sistema estima la utilización mayor a UT en T_{10} y solicita una nueva VM, este es un ejemplo de aprovisionamiento adecuado.

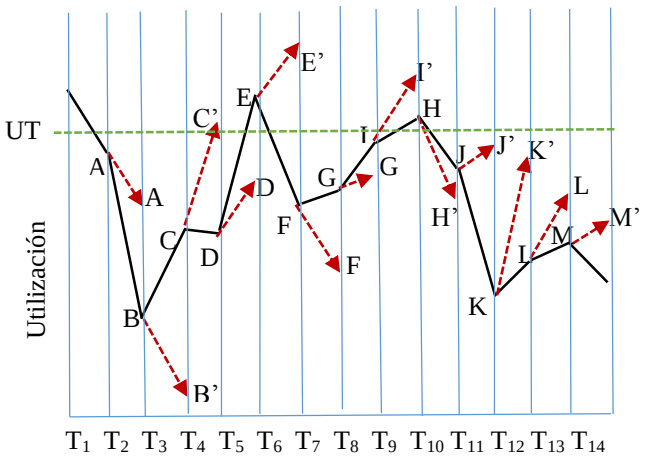

Figura 15. Escenario de predicción basado en tasa de cambio.

3.5.1.2 Asignación de llamadas con predicción basada en redes neuronales

Las redes neuronales artificiales (NNs) son un modelo computacional basado en la estructura, funcionalidad y comportamiento de un cerebro biológico. Las NNs son ampliamente utilizadas en predicción, clasificación o control, reconocimiento de objetos, etc. (Liu et al., 2017). La Figura 16 muestra una arquitectura de NN para predicción de llamadas, los nodos representan neuronas artificiales y las líneas caracterizan las conexiones entre ellas, un grupo de nodos interconectados (la salida de una neurona puede ser la entrada de otro) forma la red.

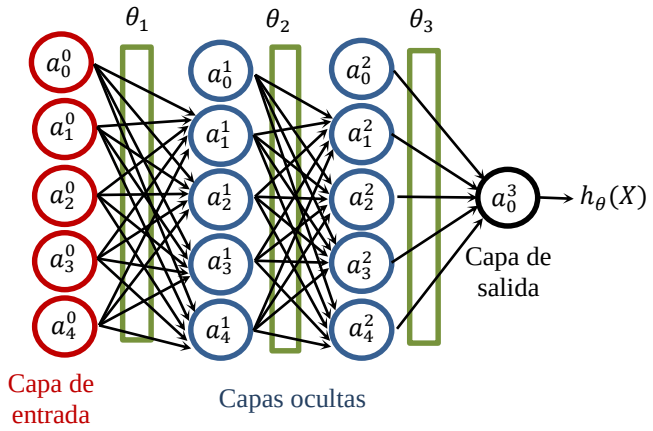

Figura 16. Arquitectura de la red neuronal.

En este modelo, $X = [a_0^0, a_1^0, a_2^0, a_3^0, a_4^0]$ define los datos de entrada a la red neuronal, θ_1, θ_2, y θ_3 representan los factores de peso entre las capas, respectivamente, y $h_\theta(X) = a_0^3$ especifica la salida. Las entradas a_1^0, a_2^0, y a_3^0 constan de valores de utilización dentro de tres TSs previos a la predicción y a_4^0 representa el tiempo actual. En la sección 4.3 se estudian diferentes valores para a_1^0, a_2^0, y a_3^0.

Los elementos a_0^0, a_0^1, y a_0^2 definen el sesgo en cada capa. La red neuronal considera una función de activación sigmoidea (logística) definida por la ecuación (12) y la activación de la unidad i (para $i >0$) en la capa j (para $j >0$) está determinada por la ecuación (13), donde $lon(\theta_i)$ es el número de pesos en θ_i.

$$g(x) = \frac{1}{1 + e^{-z}} \tag{12}$$

$$a_j^i = g\left(\sum_{j=0}^{lon(\theta_i)} a_j^{i-1} * \theta_{i,j}\right) \tag{13}$$

La ecuación (14) define la función de evaluación. El primer término es la suma logarítmica de los errores. Para un conjunto de entrenamiento $T = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$, donde m es el número de elementos en T, $y^{(i)}$ define el valor resultante del i-ésimo elemento y $\mathrm{h}_\theta(x^{(i)})$ precisa el valor estimado de la entrada $x^{(i)}$.

$$J(T) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log\left(\mathrm{h}_\theta(x^{(i)})\right) + \left(1 - y^{(i)}\right) \log\left(1 - \mathrm{h}_\theta(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{L-1} \theta_j^2 \tag{14}$$

El segundo elemento es un término de regularización, donde λ controla la importancia relativa de los dos términos y L es el número de capas en la red neuronal, ver ecuación (15).

$$\theta_i^2 = \sum_{j=1}^{lon(\theta_i)} (\theta_{i,j})^2 \tag{15}$$

Parte fundamental en el diseño de redes neuronales es el proceso de aprendizaje, el objetivo del aprendizaje de la red neuronal es obtener los mejores valores de $\theta_1, \theta_2\ y\ \theta_3$, que minimizan $J(T)$, y utilizar estos valores para predecir la carga del sistema en base a la llegada de las llamadas. El aprendizaje permite ajustar el comportamiento de la salida a partir de la entrada y los valores $\theta_1, \theta_2\ y\ \theta_3$.

El aprendizaje de la red neuronal se realiza mediante un proceso de aprendizaje supervisado donde se requiere un conjunto de entrenamiento compuesto por datos de entrada y los respectivos datos de salida (resultados). Con el fin de obtener la salida deseada, un agente externo modificar los pesos de las conexiones en la red neuronal en función de la salida proporcionada. El objetivo principal es predecir el valor correspondiente a cualquier entrada válida a partir de los ejemplos en el conjunto de entrenamiento.

El Algoritmo 2 muestra los pasos para entrenar la red neuronal donde el aprendizaje por corrección de error ajusta los pesos de las conexiones y el proceso de propagación hacia atrás utiliza el descenso de gradiente (Chauvin y Rumelhart, 1995). El algoritmo ajusta los valores de $\theta_1, \theta_2\ y\ \theta_3$ en función del conjunto de entrenamiento T. El proceso de propagación hacia atrás y la actualización de los pesos $\theta_1, \theta_2\ y\ \theta_3$ se realizan mientras el criterio de paro no se cumpla (líneas 6-8). El criterio de paro consiste en alcanzar un número máximo de iteraciones sin mejora en la evaluación de la función objetivo. Si no hay reducción del error en los pesos, se evalúa la eficiencia de la red neuronal con un conjunto de pruebas R (líneas 9-11). Al finalizar, el algoritmo regresa los valores de $\theta_1, \theta_2\ y\ \theta_3$ que minimicen la función de evaluación para el conjunto R.

En la Sección 4.3 se describe el proceso de aprendizaje y la conformación de los conjuntos de entrenamiento y pruebas para el problema de estimación de la carga en sistemas CVoIP.

	Algoritmo 2. Entrenamiento de la red neuronal
Entrada: Conjunto de entrenamiento *T*, conjunto de pruebas R, criterio de paro.	
Salida: Valores de θ_1, θ_2 y θ_3 que minimicen *J(R)*	
1	**Normalizar** los conjutnos *T* y *R*
2	**Inicializar** δ_1,δ_2, δ_3 y δ
3	**Para** $j \leftarrow 1$ **hasta** 30
4	**Inicializar aleatoriamente** θ_1, θ_2 y θ_3
5	**Mientras** criterio de paro **no se cumpla**
6	**Calcular** $J(T)$
7	**Aplicar** propagación hacia atras
8	**Actualizar** θ_1, θ_2 y θ_3
9	**Si** $\delta > J(R)$
10	$\delta \leftarrow J(R)$
11	$\delta_1 \leftarrow \theta_1, \delta_2 \leftarrow \theta_2, \delta_3 \leftarrow \theta_3$
12	**Regresar** δ_1,δ_2 y δ_3.

3.5.1.3 Asignación de llamadas con predicción basada en información histórica

La predicción basada en información histórica de la carga permite identificar patrones de variación de carga recurrentes (Janssens et al., 2012). En esta investigación se propone un predictor histórico que considera la utilización real y la información histórica sobre la utilización del sistema. El objetivo principal es tener una línea base para comparar la eficiencia de los predictores descritos anteriormente.

En el tiempo t, el predictor histórico estima $u_i\ (t + PI)$ en función de $u_i\ (t)$ y $\widehat{u_i}(t + PI)$, donde $\widehat{u_i}(t)$ define la desviación estándar histórica de la utilización en el tiempo t. La ecuación (16) define la estimación de la carga futura en el sistema.

$$u_i\ (t + PI) = u_i(t) \pm \left(x * \widehat{u_i}(t + PI)\right) \tag{16}$$

Donde x es un número aleatorio con distribución uniforme en el intervalo [0, 1]. Una variable aleatoria con distribución uniforme precisa si la utilización aumenta o disminuye. La Figura 17 muestra un ejemplo de información histórica de la carga con 85 días de utilización y un TS de 315 segundos.

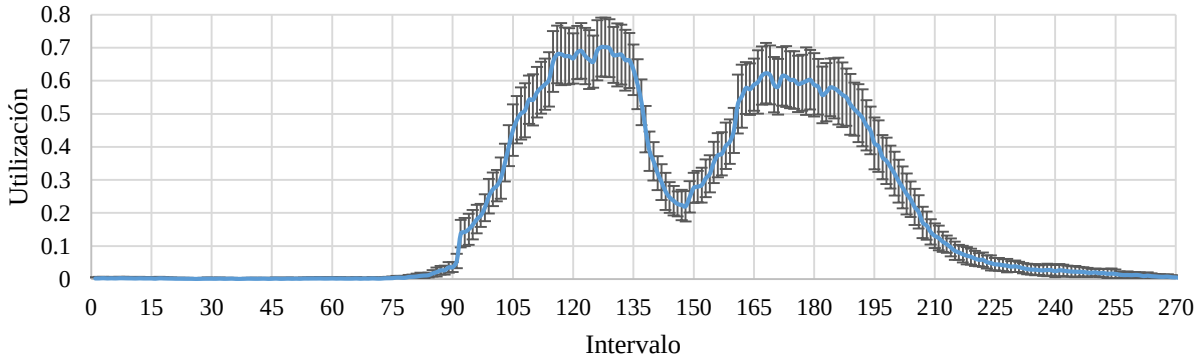

Figura 17. Información histórica de la carga con desviación estándar.

3.6 Carga de trabajo

La carga de trabajo que se utiliza en los experimentos de este trabajo es un conjunto de llamadas telefónicas registradas por el sistema de VoIP de la compañía MIXvoip. Una base de datos registra las llamadas en formato registro detallado de llamadas (CDR) con la siguiente información: índice de la llamada, ID del usuario que realiza la llamada, IP del teléfono desde donde se realiza la llamada, IP del teléfono local, destino de la llamada, código de país de destino, nombre del país de destino, proveedor de servicios de telecomunicaciones; comienzo de la llamada (marca de tiempo); duración de la llamada (en segundos), costo por minuto, etc.

Las estadísticas de las llamadas admitidas incluyen: intentos de llamadas entrantes / salientes (exitosas o no), llamadas rechazadas o fallidas, número de llamadas cuyo tiempo conectado es menor que la duración mínima configurada de la llamada (MCD), número de llamadas que se encuentran con más de la cantidad configurada de latencia, llamadas desconectadas; etc.

La distribución total de llamadas por horas durante seis meses (del 1 de noviembre de 2014 al 17 de abril de 2015) muestra un comportamiento típico de clientes comerciales, con horarios pico: de 8 a 11 AM y de 1 a 5 PM. Durante la semana, el tráfico es alto de lunes a viernes, mientras que los fines de semana disminuye considerablemente.

La Figura 18 muestra la distribución de la duración de las llamadas durante seis meses, que depende significativamente de los clientes (por ejemplo, escuelas, comercios, etc.). La duración de la mayoría de las llamadas es corta, por ejemplo, 1-5 minutos.

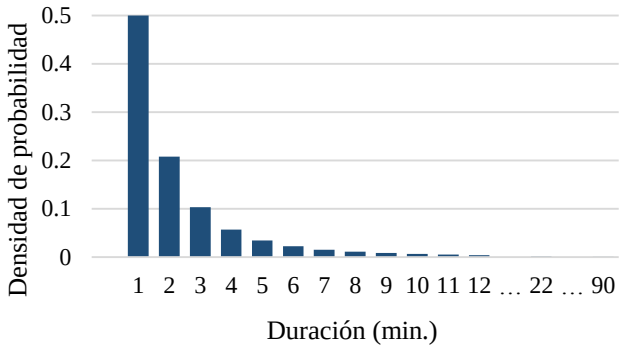

Figura 18. Distribución de la duración de las llamadas.

Dang et al. (2004) demostraron que la llegada de las llamadas concuerda con un proceso de Poisson y la distribución de la duración de la llamada mediante una distribución de Pareto generalizada con valores de parámetros que indican variaciones finitas. Los autores probaron una serie de distribuciones de probabilidad y mostraron que el modelo explica los datos en regiones de alta densidad y también se adapta a regiones de baja densidad, conocidas como colas de distribución.

En el primer escenario, la carga de trabajo comprende 24 semanas con llamadas realizadas de lunes a domingo. La Figura 19 muestra el número promedio de llamadas por horas durante 24 semanas.

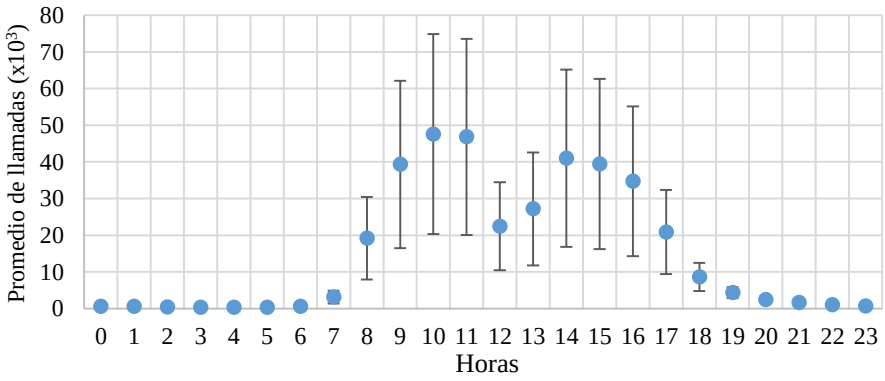

Figura 19. Número promedio de llamadas por horas del día durante 24 semanas con desviación estándar.

Durante los fines de semana, el número de llamadas es bajo, por lo que el sistema de CVoIP solo necesita una VM para atender las llamadas. Por lo tanto, los fines de semana se eliminaron de la carga de trabajo porque pueden ser reemplazadas con 24 horas de facturación por día. Una VM siempre está activa, incluso sin llamadas.

En el segundo escenario, las llamadas telefónicas realizadas durante 30 días conforman la carga de trabajo, el resto de las cargas de trabajo sirve para entrenar la NN y son la base de la predicción basada en información histórica. La Tabla 10 y la Tabla 11 presentan el número de llamadas por día y la duración de las llamadas, la carga de trabajo solo contempla días entre semana.

Tabla 10. Número de llamadas por día.

Día	Total	Promedio
Lunes	131,443	21,906
Martes	129,379	21,563
Miércoles	131,460	21,910
Jueves	130,439	21,739
Viernes	120,999	20,166

Tabla 11. Duración de las llamadas.

Tiempo (min.)	Número de llamadas
0 - 1	310,602
1 - 2	136,211
2 - 3	68,988
3 - 4	39,392
4 - 5	23,397
5 - 6	15,075
6 - 7	10,009
7 - 8	7,256
8 - 9	5,536
…	…
19 - 20	721

3.7 Marco de simulación

La industria y la academia popularizaron el enfoque de simulación para evaluar sistemas de computación en la nube, el comportamiento de las aplicaciones y la seguridad. Desarrollar nuevas propuestas dirigidas a diferentes temas relacionados con las nubes (por ejemplo, administración de centros de datos, aprovisionamiento de recursos, consumo de energía, etc.) requiere mucho trabajo y dinero. Incluso con herramientas automatizadas para hacer este trabajo, sería difícil evaluar el desempeño de una manera repetible y controlada, debido a la variabilidad inherente de la nube. Por lo tanto, la simulación facilita el estudio de estos escenarios complejos.

Las técnicas de simulación se utilizan ampliamente en diferentes campos de la informática porque representan condiciones reales de diferentes escenarios. Diversos estudios usan simulación para investigar el comportamiento de los sistemas distribuidos a gran escala. Algunos de estos simuladores son GridSim (Murshed et al., 2001), MicroGrid (Song et al., 2000), GangSim (Dumitrescu y Foster 2005), OptorSim (Bell et al., 2002), SimGrid (Casanova, 2001), CloudSim (Calheiros et al., 2011), MDCSim (Lim et al., 2009), GreenCloud (Kliazovich et al., 2010), CDOSim (Fittkau et al., 2012), TeachCloud (Jararweh et al., 2013) e iCanCloud (Núñez et al., 2012). Los primeros cinco ambientes de simulación se enfocan en sistemas grid y el resto de ellos son marcos de simulación para estudiar sistemas de cómputo en la nube.

Los simuladores grid evalúan el costo de ejecución de aplicaciones distribuidas para diferentes tipos de recursos. Desafortunadamente, la mayoría de ellos carecen de detalles para simular entornos

nube. Los simuladores nube proporcionan un marco de simulación completo para infraestructuras nube (flexibilidad, escalabilidad, etc.) y su rendimiento ha sido probado mediante validación. Existen varios análisis y comparaciones de simuladores nube para determinar el más sofisticado (Zhao et al., 2012). CloudSim, GreenCloud e iCanCloud tienen mayor funcionalidad que otros simuladores, además son de código abierto y tienen soporte para el modelado de comunicación y consumo de energía. GreenCloud está escrito en C++ y OTcl, esta es una desventaja de esta herramienta porque se deben aprender ambos lenguajes para implementar experimentos. El simulador más sofisticado para el dominio del problema es CloudSim. Varios artículos de investigación con resultados obtenidos mediante simulación usan CloudSim (Armenta et. al, 2017; Rodriguez y Buyya, 2018).

CloudSim es un kit de herramientas extensible para simulación de entornos y sistemas de cómputo en la nube. El marco de simulación, software básico y paquetes, permite realizar estudios científicos de una manera controlable (Calheiros et al., 2011). CloudSim soporta la creación de VMs y trabajos en servidores simulados en centros de datos o en un entorno de computación en la nube. La simulación permite configurar federación de nubes con interfaces personalizadas para implementar políticas y técnicas de aprovisionamiento para la asignación de trabajos y VMs.

CloudSim ofrece un modelado de nube donde la infraestructura básica de hardware son los centros de datos. Un centro de datos es un conjunto de recursos físicos que representan los nodos de cómputo y su función principal es gestionar las solicitudes de servicio. Cada recurso físico define: una configuración de capacidades (procesamiento, memoria, almacenamiento), política de asignación para VMs, modelo de soporte, etc. Los recursos físicos del centro de datos proporcionan servicio a las VMs y establecen las operaciones relacionadas con el ciclo de vida de las VMs: aprovisionamiento, creación, destrucción y migración. El aprovisionamiento de VM crea instancias de VMs y define recursos físicos según las características críticas, la configuración y los requisitos de la solicitud. El aprovisionamiento de aplicaciones asigna los servicios (aplicaciones, trabajos) en las VMs y determina la forma de ejecución de dichos servicios.

Los simuladores nube tienen una capa de virtualización que actúa como un entorno de ejecución y hospedaje para servicios de aplicaciones. CloudSim implementa políticas de asignación de recursos de tiempo compartido y de espacio compartido en ambos niveles de aprovisionamiento de VM. A nivel de recurso, la política de asignación define la parte de procesamiento de cada núcleo en el recurso asignada a cada VM. En el nivel de VM, la VM asigna una cantidad específica de potencia de procesador a las aplicaciones individuales (tareas o trabajos) alojadas dentro de su motor de ejecución.

Capítulo 4. Configuración experimental

En este capítulo se describe la configuración experimental usada para validar el rendimiento de las estrategias de asignación. Posteriormente, se define una mejora a la regla de predicción para el algoritmo de tasa de cambio. Finalmente, se evalúan diferentes valores de configuración de la red neuronal para la predicción de llamadas en CVoIP.

El análisis experimental utiliza CloudSim (Calheiros et al., 2011), un marco de simulación para el modelado de infraestructuras y servicios de cómputo en la nube. CloudSim es un simulador estándar empleado para estudiar la gestión de recursos en la nube. Las funcionalidades añadidas al simulador incluyen: algoritmos de asignación de llamadas, llegada dinámica de llamadas, actualización del sistema antes de distribuir la carga, asignación de recursos, creación dinámica de VMs, pre procesamiento de la carga de trabajo, análisis estadístico y modelos de predicción.

MIXvoip (MIXvoip, 2017), un proveedor de VoIP, proporciona cargas reales de trabajo para evaluar las estrategias de asignación en CVoIP. Las llamadas están definidas en formato estándar de carga de trabajo (SWF) con cuatro campos adicionales para procesar las llamadas en el sistema de CVoIP. En ambos escenarios, cada VM ejecuta una instancia de Asterisk (nodo de voz). Las VMs están desplegadas en varios SNs y todas pertenecen a un SNC. Los proveedores de VoIP alquilan las VMs por hora, una VM finaliza su ejecución solo si no proporciona servicio a ninguna llamada al momento de completar su tiempo de renta, en caso contrario, la VM continúa atendiendo llamadas por una hora más.

4.1 Configuración de la simulación

En el primer escenario, el ambiente no considera retraso de tiempo para desplegar la infraestructura (VMs). Este es un ambiente donde el procesamiento de llamadas es la única causa de degradación de calidad. El comportamiento de las estrategias de asignación se estudia bajo dos UTs, con 70% y 100% de utilización. La Tabla 12 presenta los valores de simulación para este escenario.

Tabla 12. Configuración de la simulación para el primer escenario.

Parámetro	Valores
Agrupamiento de súper nodos	1
Estrategias de asignación de llamadas:	20
Número de cargas de trabajo	24
Longitud de la carga de trabajo	168 horas
Número de códecs	4
Número de llamadas	2,526,595

En el segundo escenario, el ambiente incorpora ocho retardos StUps: 0, 45, 90, 135, 180, 225, 270 y 315 segundos como casos de prueba (consulte las Tablas 7 y 8) y diez PS: 10, 20, 30, StUp (PS = StUp). El objetivo principal es analizar la degradación de la calidad de servicio inherente al StUp y la reducción de la degradación de la calidad debido a la predicción de la carga. La Tabla 13 presenta los valores de simulación para este escenario, la carga de trabajo contempla 30 días, de 24 horas cada uno, sin fines de semana ni vacaciones.

Tabla 13. Configuración de la simulación para el segundo escenario.

Parámetro	Valores
Agrupamiento de súper nodos	1
Total de estrategias de asignación de llamadas	128
Asignación sin predicción	32
Asignación con predicción	96
Número de cargas de trabajo	30
Longitud de la carga de trabajo	24 horas
Número de códecs	4
Número de llamadas	643,718
VM StUp	7

Una de las principales contribuciones de esta investigación es el desarrollo de un marco de simulación para sistemas de VoIP en la nube. La implementación de CVoIP en CloudSim incorpora varios parámetros: umbral de utilización, retraso de inicio de las VMs, tiempo de alquiler de las VMs, umbral de renta, intervalo de predicción, entre otros. La Tabla 14 muestra todos los parámetros configurables y los valores de configuración para ambos escenarios.

Tabla 14. Parámetros configurables.

Parámetro		Valores
UT	Umbral de utilización	0.7 y 1.0
RT	Umbral de renta	5, 10 y 15 min.
PI	Intervalo de predicción	10, 20, 30 y StUp seg.
TS	Paso de tiempo	10, 20, 30 y StUp seg.
StUp	Retardo de inicio	45, 90, 135, 180, 225, 270 y 315 seg.

Los experimentos se realizaron utilizando Java 6.1.7601 en una computadora Dell Express x3650 M4 con dos procesadores Xeon IvyBridge E5-2650v2 de 2.6 GHz con sistema operativo CentOS 7. Cada núcleo tiene dos hilos (16 en Hyper Threading), 32 Kb de caché nivel 1, 256 Kb de caché nivel 2 y 20 Mb de caché nivel 3 con 64 GB de memoria RAM y 2 TB de disco duro.

La red neuronal se implementó y entrenó utilizando Octave 4.0.2 en una computadora Dell Precision T3610 con Windows 7 professional, service pack 1, con sistema operativo de 64 bits. El hardware corresponde a un procesador Intel Xeon E5-1603 de 2,8 GHz, 16 GB de memoria RAM y 500 GB de disco duro.

4.2 Configuración de la regla de predicción del algoritmo tasa de cambio

La regla de predicción en RoC es fundamental para estimar la cantidad de VMs necesarias para proporcionar el servicio de CVoIP. RoC usa una regla de estimación lineal que solo considera la diferencia de carga entre dos TSs, vea ecuación (9). Una adecuada elección de $\Delta_i(t)$ aumenta la exactitud de la estimación, por lo que el objetivo es proponer una $\Delta_i(t)$ para mejorar la precisión de RoC.

La vulnerabilidad del sistema CVoIP aumenta cuando el número de VMs es pequeño, por lo tanto $\Delta_i(t)$ debe considerar la cantidad de VMs en el sistema. La ecuación (15) define una nueva regla de estimación de carga para RoC, donde $k_i(t)$ describe el número de VMs en ejecución. Cuando $k_i(t)$ es alto, el sistema responde mejor a cambios abruptos de carga, ya que cuenta con más recursos para admitir grandes volúmenes de llamadas.

$$\Delta_i(t) = (u_i(t) - u_i(t - \mathrm{TS}))/k_i(t) \tag{15}$$

El desempeño de ambas $\Delta_i(t)$ se evalúa por medio de simulación. La configuración del ambiente de simulación define siete StUps: 45, 90, 135, 180, 225, 270 y 315 (segundos), y diez TSs: 10, 20, 30, StUp (segundos) como casos de prueba.

La Tabla 15 y la Tabla 16 muestran los valores de RMSD para ambas reglas de predicción, nRoC supera a RoC en todos los casos de prueba. El peor valor para ambas reglas aparece en la predicción con PS y StUp iguales a 315. Similar a los valores de RMSD, nRoC domina a RoC en todos los casos de prueba para los valores de SMAPE, vea Tabla 17 y la Tabla 18.

La Figura 20 muestra un ejemplo de valores de carga reales y estimados durante 9 horas con un TS de 315 segundos. De manera similar, la Figura 21 presenta los mismos valores durante 40 minutos con un TS de 10 segundos. En ambos ejemplos, la predicción de nRoC ajusta mejor a los valores reales de carga.

Tabla 15. Valores de RMSD para nRoC.

StUp \ PS	10	20	30	StUp
45	14.067	19.278	22.892	26.637
90	**13.991**	19.084	22.907	33.773
135	14.047	19.180	22.883	38.417
180	14.006	19.155	22.754	41.358
225	14.145	19.227	22.869	44.672
270	14.405	19.462	23.010	46.337
315	14.552	19.553	22.849	**49.615**

Tabla 16. Valores de RMSD para RoC.

StUp \ PS	10	20	30	StUp
45	41.028	56.491	66.746	78.607
90	**41.007**	56.469	66.746	98.481
135	41.015	56.416	66.759	111.995
180	41.022	56.460	66.759	118.451
225	41.139	56.423	66.764	127.266
270	41.051	56.431	66.764	131.595
315	41.216	56.555	66.799	**138.387**

Tabla 17. Valores de SMAPE para nRoC ($x10^{-2}$).

StUp \ PS	10	20	30	StUp
45	0.693	0.997	1.195	1.421
90	**0.690**	0.988	1.195	1.817
135	0.692	0.994	1.193	2.060
180	0.691	0.992	1.187	2.202
225	0.698	0.997	1.192	2.369
270	0.694	0.999	1.192	2.451
315	0.693	1.000	1.180	**2.602**

Tabla 18. Valores de SMAPE para RoC ($x10^{-2}$).

StUp \ PS	10	20	30	StUp
45	1.949	2.732	3.243	3.843
90	**1.948**	2.732	3.244	4.844
135	1.949	2.731	3.244	5.484
180	1.949	2.732	3.244	5.821
225	1.950	2.732	3.244	6.255
270	1.950	2.733	3.244	6.467
315	1.952	2.734	3.245	**6.810**

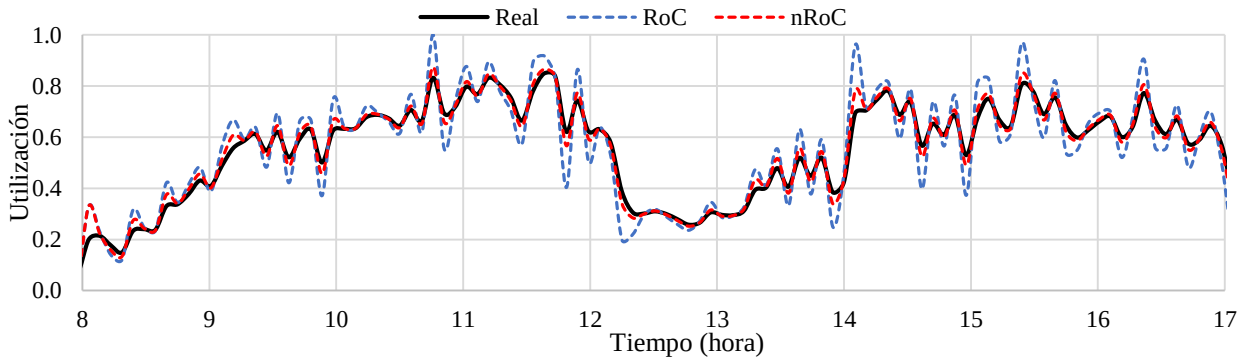

Figura 20. Estimación de carga con TS de 315 segundos.

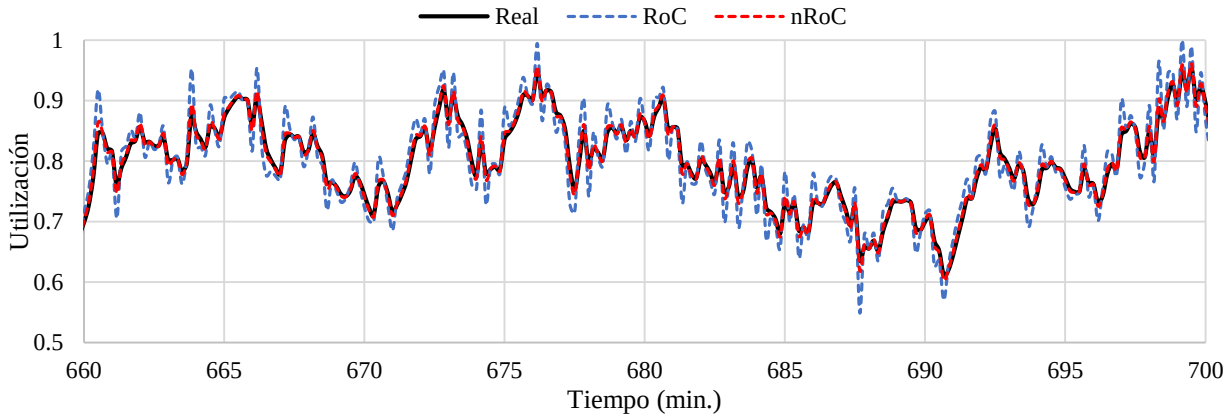

Figura 21. Estimación de carga con TS de 10 segundos.

Además, el desempeño de las reglas de estimación se compara mediante el análisis de cubrimiento de conjuntos. La Tabla 19 presenta los valores $SC(nRoC, RoC)$ para el dominio de la estimación nRoC sobre RoC con ocho estrategias de asignación y siete TSs. nRoC domina las soluciones de RoC en un intervalo de 67.5 - 99.2%, con 84.3% en promedio. La Tabla 20 muestra los valores $SC(RoC, nRoC)$ para el dominio de RoC sobre nRoC. RoC domina las soluciones de nRoC en un intervalo de 8.3 - 37.5%, con 22.2% en promedio. El cubrimiento de conjuntos confirma que la estimación nRoC puede generar soluciones en 84.3%, en promedio, con al menos la misma calidad de RoC, para diferentes StUps y TSs.

Tabla 19. Valores de SC(nRoC, RoC).

Strategy \ StUp	45	90	135	180	225	270	315
BFit	0.908	0.958	0.933	0.933	0.933	0.883	0.858
FFit	0.967	0.983	**0.992**	0.950	0.917	0.900	0.908
MaxFTFit	0.958	0.875	0.942	0.917	0.883	0.833	0.858
MidFTFit	0.867	0.817	0.900	0.792	0.850	0.825	0.842
MinFTFit	0.792	0.858	0.775	0.733	**0.675**	0.700	0.725
Rand	0.775	0.858	0.800	0.833	0.775	0.742	0.742
RR	0.792	0.783	0.842	0.792	0.833	0.800	0.700
WFit	0.775	0.833	0.883	0.833	0.825	0.808	0.750

Tabla 20. Valores de SC(RoC, nRoC).

Strategy \ StUp	45	90	135	180	225	270	315
BFit	0.225	0.142	0.117	0.125	0.125	0.192	0.167
FFit	0.133	**0.083**	0.117	0.117	0.158	0.175	0.117
MaxFTFit	0.142	0.217	0.192	0.108	0.158	0.183	0.100
MidFTFit	0.192	0.242	0.158	0.258	0.217	0.175	0.167
MinFTFit	0.325	0.225	0.308	0.350	**0.375**	0.333	0.167
Rand	0.333	0.208	0.292	0.258	0.300	0.300	0.300
RR	0.250	0.283	0.233	0.333	0.225	0.283	0.358
WFit	0.308	0.267	0.208	0.283	0.250	0.233	0.317

4.3 Configuración de la red neuronal

El diseño de una red neuronal (NN) implica la configuración de varios parámetros. En esta sección se proponen y analizan diferentes valores para λ, valores de utilización de la entrada (a_1^0, a_2^0, y a_3^0) y el tamaño del conjunto de entrenamiento. El objetivo es encontrar los valores que maximicen el desempeño de la NN para el problema de predicción de llamadas.

El ambiente de simulación para evaluar las diferentes versiones de NNs define $PI = TS = stUp = 315$ segundos. El proceso de entrenamiento de la NN considera tres conjuntos: aprendizaje, validación y pruebas. La NN modifica los pesos en la red (θ_1, θ_2, y θ_3) en función del conjunto de aprendizaje, en otras palabras, la NN asimila a partir de este conjunto. El conjunto de validación permite medir la diferencia entre la salida de la NN y los valores deseados, esta diferencia se usa para validar si el aprendizaje de la NN puede finalizar. El conjunto de pruebas sirve para determinar la eficiencia de la NN con otros datos.

Se analizan diferentes valores para λ. El conjunto de valores propuestos abarca un amplio rango de cobertura, $\lambda = \{10.0, 1.0, 0.5, 0.1, 0.01, 0.001, 0.0001, 0.00001\}$. Los tres conjuntos para definir el valor de λ constan de 15 cargas de trabajo cada uno, los conjuntos de aprendizaje y validación cuentan con 5 cargas de trabajo en común y los conjuntos de aprendizaje y pruebas comparten 5 cargas. Una carga de trabajo consiste de llamadas realizadas durante 24 horas. El entrenamiento para cada valor de λ se realiza 30 veces y el análisis compara las NNs con mejor desempeño, con los mejores valores θ_1, θ_2, y θ_3 entrados para cada λ.

La Tabla 21 muestra los valores de RMSD y SMAPE para diferentes valores de λ, valores más pequeños de λ mejoran el desempeño de la NN.

Tabla 21. Valores de RMSD and SMAPE para diferentes configuraciones de λ.

λ	RMSD	SMAPE
0.00001	**106.739660**	**0.055020**
0.0001	107.239957	0.055468
0.001	108.844112	0.056072
0.01	118.981582	0.060927
0.1	120.277697	0.062772
0.5	130.424748	0.072682
1.0	138.546018	0.079169
10.0	167.208843	0.10274

La NN emplea tres valores de utilización, en diferentes TSs, para estimar la carga en el próximo PI. Los valores de utilización pertenecen a los intervalos $(t - TS, t]$, $(t - 2 * TS, t - TS]$ y $(t - 3 * TS, t - 2 * TS]$ y pueden estar definidos por: $u_i(t)$ la utilización en el tiempo t, $u_i(t) = \sum_{k=t-TS}^{t} \frac{u_i(k)}{TS}$ la utilización

promedio en el intervalo $(t - TS, t]$ y $u_i(t) = \max(u_i(t - TS), u_i(t)]$ la utilización máxima en el intervalo $(t - TS, t]$.

Las versiones de NN con diferentes valores de utilización son evaluadas con la configuración previamente descrita y λ = 0.00001, el valor con mejor desempeño para λ. La Tabla 22 muestra los valores de RMSD y SMAPE para diferentes valores de utilización en la predicción: actual, promedio y máximo. La mejor versión de NN considera la utilización en el tiempo t del intervalo $(t - TS, t]$.

Tabla 22. Valores de RMSD y SMAPE para diferentes valores de utilización.

Valor de utilización	RMSD	SMAPE
Actual	**106.739660**	**0.055020**
Promedio	111.681136	0.056846
Máximo	110.478730	0.057947

Finalmente, diferentes tamaños de conjuntos de aprendizaje se analizan para evitar sobreajuste, el número de cargas de trabajo en el conjunto varía. Los diferentes conjuntos de aprendizaje se conforman de 30, 45, 60 y 75 cargas de trabajo.

El tamaño de todos los conjuntos cambia en esta evaluación. El conjunto de aprendizaje comparte 15 cargas de trabajo con el conjunto de pruebas y 5 cargas de trabajo con el conjunto de validación, el resto son cargas de trabajo nuevas. El conjunto de validación contiene 15 cargas de trabajo de las cuales comparte 5 cargas de trabajo con el conjunto de entrenamiento y 10 son cargas de trabajo nuevas. El conjunto de pruebas comprende 30 cargas de trabajo de las cuales comparte 15 cargas de trabajo con el conjunto de aprendizaje y 15 son cargas de trabajo nuevas.

La Tabla 23 muestra los valores de RMSD y SMAPE para los cuatro conjuntos de entrenamiento con diferentes tamaños de carga. La mejor versión de la NN entrenó con 75 días de carga de trabajo y λ = 0.00001. Estos valores de configuración se emplean en la predicción de carga de trabajo para CVoIP.

Tabla 23. Valores de RMSD and SMAPE para diferentes tamaños del conjuntos de aprendizaje.

Conjunto de entrenamiento	RMSD	SMAPE
30	115.679943	0.054781
45	116.057055	0.055013
60	115.563836	0.054721
75	**115.026791**	**0.054598**

Capítulo 5. Análisis experimental

En este capítulo, se analizan los resultados experimentales de ambos escenarios. En el primer escenario, veinte estrategias son analizadas bajo diferentes UTs. Inicialmente, se considera un UT del 70% de utilización para garantizar el 100% de QoS. Posteriormente, el 100% de la capacidad de la VM se emplea para evaluar el comportamiento de las estrategias de asignación y la degradación de la QoS. En el segundo escenario, se estudian los efectos del StUp en el sistema CVoIP. El costo y la degradación de QoS se evalúan para siete StUps con diez TSs. Tres modelos de predicción son incorporados en las estrategias de asignación para mitigar los efectos del StUp.

5.1 Análisis de costo en la asignación de llamadas con QoS garantizada

En el primer escenario, se evalúan veinte estrategias: BFit, FFit, MaxFTFit, MidFTFit, MinFTFit, Rand, RR, WFit y las versiones RT de BFit, FFit, Rand, RR y WFit con 5, 10 y 15 minutos, donde la restricción para garantizar la QoS es UT igual a 0.7. La Tabla 12 muestra la configuración de la simulación para este escenario.

Esta configuración solo evalúa el costo del proveedor generado por el alquiler de la infraestructura. La Figura 22 muestra el número de horas de facturación ($\bar{b}$) durante 24 semanas. La carga de trabajo es baja durante las semanas 8 y 9, de modo que la diferencia de $\bar{b}$ generada por las estrategias es aproximadamente 50. La diferencia es más alta en otras semanas, hasta 160 horas de facturación en la semana 5.

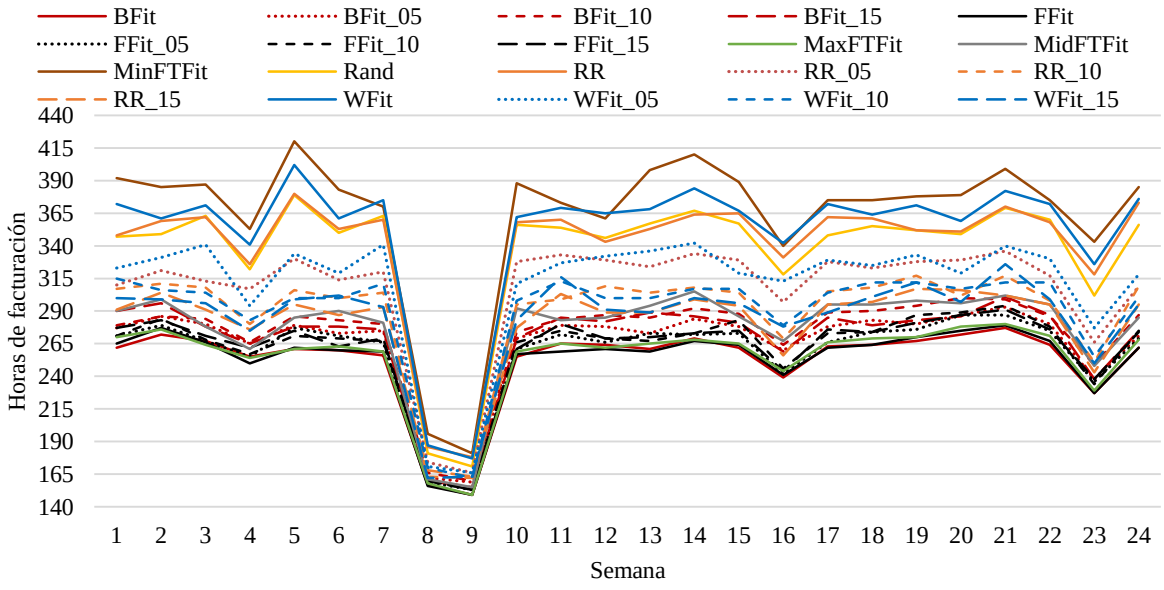

Figura 22. Número de horas de facturación ($\bar{b}$) durante 24 semanas.

La Tabla 24 muestra el promedio de $\bar{b}$ durante 24 semanas para ambos UTs. Para el caso de UT = 0.7, BFit y FFit son las estrategias con mejor desempeño, estas estrategias usan 252.08 y 252.42 horas de facturación, en promedio, por semana para manejar una determinada carga de trabajo. WFit y MinFTFit son las peores estrategias con 351.08 y 363.96 horas de facturación, respectivamente. La reducción de $\bar{b}$ con respecto a Round Robin está entre 25.96% - 1.22% y solo dos estrategias tienen peor rendimiento.

Las estrategias RR_05, RR_10, RR_15 tienen mejor rendimiento que RR. Del mismo modo, WFit_05, WFit_10 y WFit_15 son mejores que WFit. La diferencia entre BFit, la mejor estrategia, y MinFTFit, la peor estrategia, es de aproximadamente 111 horas de facturación, en promedio, por semana.

Para el caso de UT = 1.0, se presenta la clasificación de las estrategias donde 17 estrategias mejoran el rendimiento de Round Robin, la estrategia actualmente utilizada en el servicio de VoIP. Los valores de $\bar{b}$ con UT igual a 1.0 representan el límite inferiores para las estrategias de asignación. Para las versiones en línea de los algoritmos, $\bar{b}$ no puede ser menor porque las VMs usan el 100% de la capacidad. El incremento en $\bar{b}$ para diferentes UTs es de 24.71% a 30.06% aproximadamente.

Tabla 24. Promedio semanal de horas de facturación ($\bar{b}$).

UT / Estrategia	0.7	1.0
BFit	252.08	200.35
FFit	252.42	200.79
MaxFTFit	254.71	204.12
FFit_05	259.46	206.79
FFit_15	261.75	209.28
FFit_10	261.79	208.95
BFit_05	266.13	213.69
BFit_15	269.21	216.22
BFit_10	273.25	217.16
MidFTFit	276.08	221.23
RR_15	279.79	223.97
WFit_15	283.79	227.25
RR_10	289.00	231.59
WFit_10	290.42	232.93
RR_05	306.88	245.84
WFit_05	311.29	248.98
Rand	336.29	263.96
RR	340.46	266.13
WFit	351.08	272.12
MinFTFit	363.96	279.83
Promedio	288.99	229.56

Al comparar las estrategias con respecto a diferentes UTs, las mejores estrategias, BFit y FFit, aumentan $\bar{b}$ en 25.82% y 25.71%, en promedio. Mientras tanto, Round Robin incrementa $\bar{b}$ alrededor del 27.93% y MinFTFit, la peor estrategia, lo hace en 30.06%.

5.2 Análisis bi-objetivo de asignación de llamadas

En las siguientes secciones, se realiza un análisis conjunto de dos métricas de acuerdo con la metodología de degradación del desempeño y se obtiene un conjunto de soluciones compromiso que representan una buena aproximación al frente de Pareto; ambas metodologías se describen en la Sección 3.4.

La configuración es similar al escenario anterior donde el único cambio es la restricción para garantizar la QoS. Esta configuración experimental evalúa el costo del proveedor y la degradación de QoS, con UT igual a 1.0.

5.2.1 Degradación del desempeño

Primero, se presenta un análisis de $\bar{b}$ y $\bar{q}$ por separado. Posteriormente, se encuentra la estrategia que genera el mejor compromiso entre ambas métricas. La Tabla 25 muestra la degradación promedio de $\bar{b}$, la degradación promedio de $\bar{q}$ y el promedio de ambas degradaciones. Las últimas cuatro columnas de la tabla contienen el rango de cada estrategia en relación con el costo del proveedor, la calidad de servicio y los promedios. Rango $\bar{b}$ considera la degradación de $\bar{b}$ para clasificar las estrategias. Rango $\bar{q}$ define la posición de la estrategia en relación con la degradación de $\bar{q}$. Rango promedio establece la posición de la estrategia basada en el promedio de degradaciones de $\bar{b}$ y $\bar{q}$. Rango es la posición relativa con respecto a todas las estrategias, considera el rango de $\bar{b}$ y el rango $\bar{c}$.

BFit es la mejor estrategia para $\bar{b}$, esta estrategia asigna la llamada en base a la estrategia Best Fit, donde la llamada se coloca en la VM más llena, lo que deja el menor espacio de utilización sobrante. Sin embargo, es la peor estrategia para $\bar{q}$. BFit tiende a aumentar la utilización y reducir la calidad.

La mejor estrategia para $\bar{q}$ es WFit, donde la llamada se asigna en la VM que deja la mayor utilización sobrante. WFit tiende a subutilizar las VM por lo que mantiene la calidad de servicio, pero aumenta el número de VMs y el costo de alquiler.

WFit_15 es la estrategia con mejor compromiso entre las dos métricas, esta estrategia asigna la llamada utilizando la estrategia Worst Fit y limita la asignación de llamadas 15 minutos antes de que la VM alcance su tiempo de alquiler. MaxFTFit es la mejor estrategia con posición relativa para todas las estrategias, esta estrategia asigna la llamada a la VM que termina su tiempo de alquiler más lejano, con mayor tiempo de renta restante.

Tabla 25. Degradación y rango.

Estrategia	$\bar{b}$	$\bar{q}$	Promedio	Rango $\bar{b}$	Rango $\bar{q}$	Rango promedio	Rango
BFit	0.263	21.099	10.681	**1**	20	6	4
BFit_05	6.911	17.930	12.420	7	16	12	6
BFit_10	8.405	17.342	12.874	9	13	15	5
BFit_15	8.091	17.240	12.665	8	12	14	3
FFit	0.535	21.031	10.783	**2**	19	7	4
FFit_05	3.483	18.758	11.120	4	18	8	5
FFit_10	4.638	18.069	11.354	5	17	10	5
FFit_15	4.812	17.819	11.315	6	14	9	3
MaxFTFit	2.096	17.835	9.965	**3**	15	4	**1**
MidFTFit	10.536	16.442	13.489	10	11	16	4
MinFTFit	39.147	12.800	25.973	20	10	20	7
Rand	31.263	1.094	16.178	17	4	17	4
RR	32.681	0.732	16.707	**18**	**2**	**18**	**3**
RR_05	22.542	2.144	12.343	15	5	11	**3**
RR_10	15.388	4.666	10.027	13	7	5	**3**
RR_15	11.772	6.980	9.376	11	9	**2**	**3**
WFit	35.435	0.000	17.718	19	**1**	19	**3**
WFit_05	23.904	1.085	12.494	16	**3**	13	**2**
WFit_10	16.606	3.301	9.954	14	6	**3**	**3**
WFit_15	13.292	5.428	9.360	12	8	**1**	**3**

Con respecto a la estrategia actualmente en uso, 17 estrategias mejoran el rendimiento de Round Robin considerando el rango promedio. El desempeño de Round Robin ocupa el tercer puesto del rango, respecto a la posición relativa de todas las estrategias, por lo que al menos 8 estrategias tienen un desempeño igual o mejor que esta estrategia.

5.2.2 Espacio de soluciones y frente de Pareto

Para resolver el problema bi-objetivo general, se requiere obtener un conjunto de soluciones compromiso que representen una buena aproximación al frente de Pareto. Este no es formalmente el frente de Pareto ya que no se lleva a cabo una búsqueda exhaustiva de todas las soluciones posibles, sino que sirve como una aproximación práctica de un frente de Pareto.

La Figura 23 muestra los conjuntos de soluciones para las veinte estrategias obtenidas en base a 109 días de carga de trabajo. Este espacio de solución bidimensional representa un conjunto factible de soluciones que satisfacen las limitaciones del problema, donde se aborda el problema de minimizar $\bar{b}$ y $\bar{q}$.

El espacio de solución cubre un rango de valores en $\bar{b}$ de 0 a 0.65, mientras que los valores de $\bar{q}$ están en el rango de 0 a 0.26. Tres grupos dividen el espacio de solución: el lado inferior derecho, el lado superior izquierdo y en el espacio medio. BFit, FFit y MaxFTFit ocupan la parte inferior derecha y

representan las mejores soluciones en términos de $\bar{b}$. Estas estrategias superan a otras, como RR, que están en uso actual para el servicio de VoIP. WFit reside en el lado superior izquierdo y pertenece a las mejores soluciones en términos de $\bar{q}$. Las tres versiones de WFit con diferentes RT (WFit_05, WFit_10, WFit_15) tienen un buen comportamiento, ocupan el espacio medio.

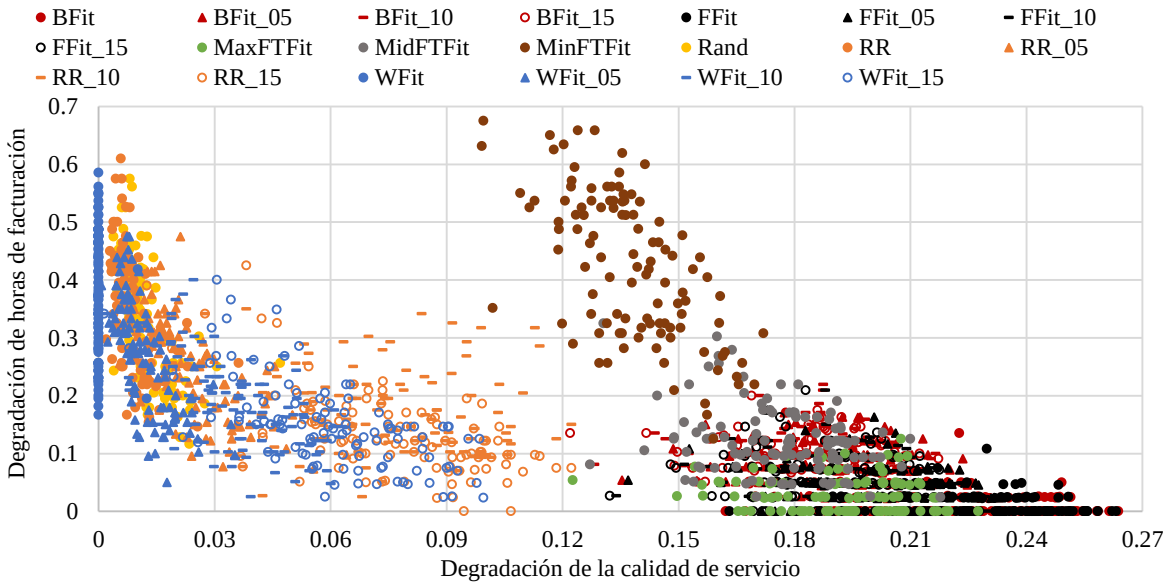

Figura 23. Espacio de soluciones.

WFit es la mejor estrategia con respecto a la degradación de $\bar{q}$. El rango de las degradaciones en $\bar{b}$ es de 0.16 a 0.56. WFit_05 aumenta la degradación de $\bar{q}$ a 0.017, pero reduce $\bar{b}$ hasta 0.05. Para WFit_10, la degradación de $\bar{q}$ aumenta de 0.017 a 0.06, pero reduce $\bar{b}$ a 0.023. Finalmente, WFit_15 tiene una amplia gama de soluciones para la degradación de $\bar{q}$ (de 0.019 a 0.099) pero solo el 20% de sus soluciones supera el 20% de degradación.

Diferentes versiones de WFit cubren varios sectores en el frente de Pareto, y muestran el mejor compromiso entre ambos objetivos para las veinte estrategias. El espacio de la solución MaxFTFit está en el mismo rango de $\bar{b}$ que BFit y FFit pero supera en $\bar{q}$ a ambas estrategias. Las estrategias WFit_05, WFit_10, WFit_15 y MaxFTFit cubren mejor el espacio de solución y el frente de Pareto. Son buenas opciones para los proveedores de VoIP. La Figura 24 muestra las veinte aproximaciones a los frentes de Pareto generados por las estrategias estudiadas.

La métrica de cobertura de conjuntos determina la dominancia entre dos conjuntos de soluciones. Las filas de la tabla muestran los valores $SC(A, B)$ para el dominio de las soluciones de la estrategia A sobre la estrategia B. Las columnas indican los valores $SC\ (B, A)$, es decir, dominio de B sobre A. Las últimas dos columnas muestran el promedio de $SC\ (A, B)$ para la fila A sobre la columna B, y la clasificación basada

en el dominio promedio. De manera similar, las últimas dos filas muestran el dominio promedio de B sobre A, y el rango de la estrategia en cada columna.

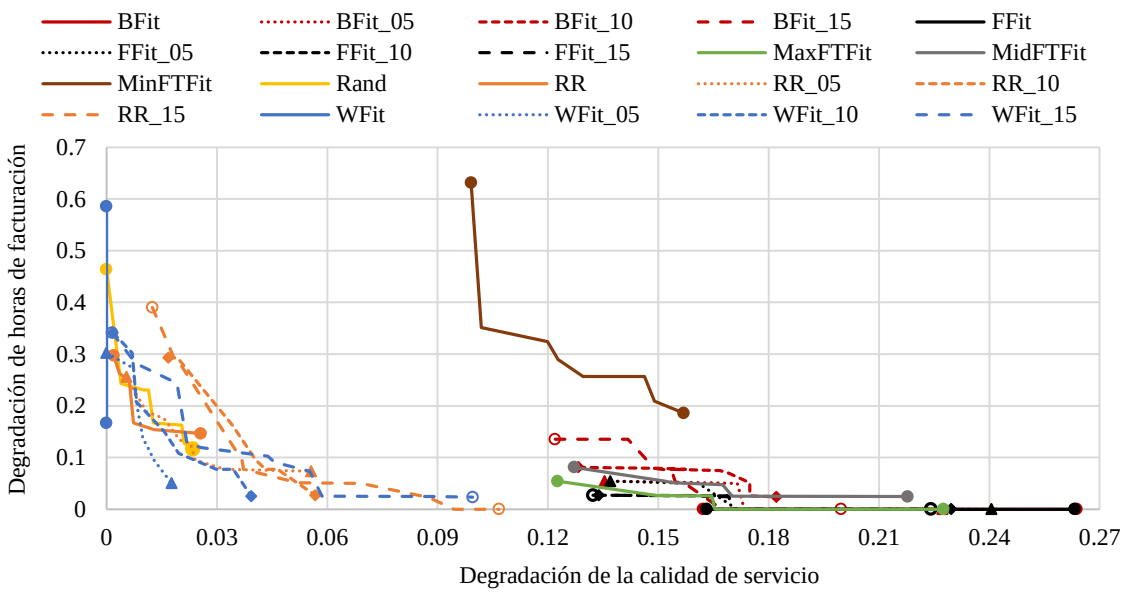

Figura 24. Frentes de Pareto.

La clasificación de las estrategias se basa en el porcentaje de cobertura. Una clasificación más alta implica un mejor frente. La Tabla 26 muestra los resultados para cada uno de los veinte frentes. De acuerdo con la métrica de cobertura de conjuntos, la estrategia con mejor compromiso entre $\bar{q}$ y $\bar{b}$ son WFit_15, seguida de MaxFTFit, RR_15 y WFit_05.

MaxFTFit domina los frentes de otras estrategias en el rango 0-60%, con un promedio de 21.8% ocupando el segundo lugar. $SC(A, MaxFTFit)$ muestra que MaxFTFit está dominado por otros frentes con un promedio de 6.9%. Mientras tanto, WFit_05 y MaxFTFit, con el segundo y tercer rango, están dominados por otras estrategias en 19.5% y 21.9%, en promedio, respectivamente. Ambas estrategias son dominadas por otras estrategias en un rango de 6.2% al 6.9%.

Sin embargo, no se debe considerar solo los frentes de Pareto, cuando muchas soluciones están fuera de las soluciones óptimas de Pareto. Este es el caso de BFit_xx, FFit_xx, RR_xx, aunque los frentes de Pareto son de buena calidad, muchas de las soluciones generadas están bastante lejos de este, y, por lo tanto, una sola ejecución del algoritmo puede producir resultados significativamente peores.

Tabla 26. Cubrimiento de conjuntos y rango (días).

A \ B	BFit	BFit_05	BFit_10	BFit_15	FFit	FFit_05	FFit_10	FFit_15	MaxFTFit	MidFTFit	MinFTFit	Rand	RR	RR_05	RR_10	RR_15	WFit	WFit_05	WFit_10	WFit_15	Promedio	Rango
BFit	1	0	0	0	0.30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.065	18
BFit_05	0.07	1	0.06	0.02	0.08	0.17	0.05	0.01	0.01	0.01	0	0	0	0	0	0	0	0	0	0	0.074	17
BFit_10	0.03	0.17	1	0.07	0.02	0.11	0.10	0.05	0.02	0.02	0	0	0	0	0	0	0	0	0	0	0.079	16
BFit_15	0.03	0.23	0.28	1	0.05	0.10	0.11	0.10	0.03	0.01	0	0	0	0	0	0	0	0	0	0	0.096	14
FFit	0.21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.061	19
FFit_05	0.33	0.06	0.04	0.02	0.30	1	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0.088	15
FFit_10	0.18	0.17	0.09	0.06	0.18	0.38	1	0.06	0.04	0.03	0	0	0	0	0	0	0	0	0	0	0.109	11
FFit_15	0.21	0.29	0.19	0.12	0.21	0.33	0.34	1	0.03	0.06	0	0	0	0	0	0	0	0	0	0	0.139	8
MaxFTFit	0.43	0.42	0.28	0.24	0.46	0.60	0.47	0.38	1	0.09	0	0	0	0	0	0	0	0	0	0	0.218	2
MidFTFit	0.02	0.20	0.31	0.18	0	0.14	0.13	0.08	0.03	1	0	0	0	0	0	0	0	0	0	0	0.105	13
MinFTFit	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0.050	20
Rand	0	0	0	0	0	0	0	0	0	0.01	0.94	1	0.01	0.09	0.03	0	0	0.05	0.01	0.02	0.107	12
RR	0	0	0	0	0	0	0	0.01	0	0	0.90	0.52	1	0.10	0.02	0.03	0	0.07	0.00	0.02	0.133	10
RR_05	0	0.04	0.04	0.04	0	0	0	0.01	0.01	0.08	0.97	0.03	0.02	1	0.21	0.11	0.01	0.01	0.19	0.14	0.145	7
RR_10	0.01	0.13	0.20	0.17	0	0.06	0.06	0.07	0.04	0.28	0.99	0.02	0	0.02	1	0.31	0	0	0.01	0.21	0.178	6
RR_15	0.02	0.32	0.39	0.39	0.02	0.11	0.17	0.16	0.06	0.52	0.98	0	0	0.02	0.01	1	0	0	0	0.02	0.210	3
WFit	0	0	0	0	0	0	0	0	0	0	0.84	0.27	0.41	0.08	0.01	0.02	1	0.09	0.03	0.01	0.138	9
WFit_05	0	0.04	0.02	0.02	0	0.02	0	0.01	0	0.03	0.96	0.42	0.15	0.51	0.20	0.15	0.01	1	0.22	0.15	0.195	4
WFit_10	0	0.09	0.15	0.17	0.01	0.05	0.06	0.06	0.03	0.25	0.97	0.01	0.04	0.06	0.38	0.28	0	0.02	1	0.28	0.194	5
WFit_15	0.02	0.26	0.32	0.32	0.01	0.14	0.16	0.14	0.08	0.45	0.98	0	0.01	0.02	0.06	0.39	0	0	0.02	1	0.219	**1**
Promedio	0.128	0.171	0.169	0.140	0.132	0.160	0.132	0.106	0.069	0.142	0.477	0.113	0.082	0.095	0.096	0.114	0.051	0.062	0.074	0.092		
Rango	12	19	18	15	14	17	13	9	3	16	20	10	5	7	8	11	**1**	2	4	6		

5.3 Análisis de asignación de llamadas con retardo de inicio

En las siguientes secciones, se analiza el comportamiento de las estrategias afectadas por el retraso en el inicio de las VMs. En primer lugar, se estudian los beneficios de las versiones RT, cuando las estrategias toman en cuenta el tiempo de renta de la VMs. En segundo lugar, se analizan las ventajas de las versiones conscientes de la carga con tres modelos de predicción: tasa de cambio, red neuronal y predicción basada en información histórica de la carga. Finalmente, se realiza un estudio comparativo de las mejores estrategias encontradas.

En general, la reducción de la calidad ($\bar{q}$) no se analiza debido a que la diferencia entre la peor estrategia y la calidad del servicio al 100% es de aproximadamente 9.292×10^{-3}. En este caso, las llamadas en espera ($\bar{c}$) representan mejor la reducción de QoS. La Tabla 13 muestra la configuración de la simulación para este escenario.

5.3.1 Estrategias con umbral de renta

Se evalúan ocho estrategias: BFit, FFit, MaxFTFit, MidFTFit, MinFTFit, Rand, RR y WFit, y tres versiones RT para cada estrategia. La Figura 25 muestra la degradación de las horas de facturación ($\bar{b}$) frente a diferentes StUps. Las estrategias con mejor rendimiento son BFit y FFit, y las peores estrategias son MinFTFit y WFit. Las estrategias de asignación tienden a ser robustas con respecto a $\bar{b}$, el StUp no afecta considerablemente $\bar{b}$ para todas las estrategias.

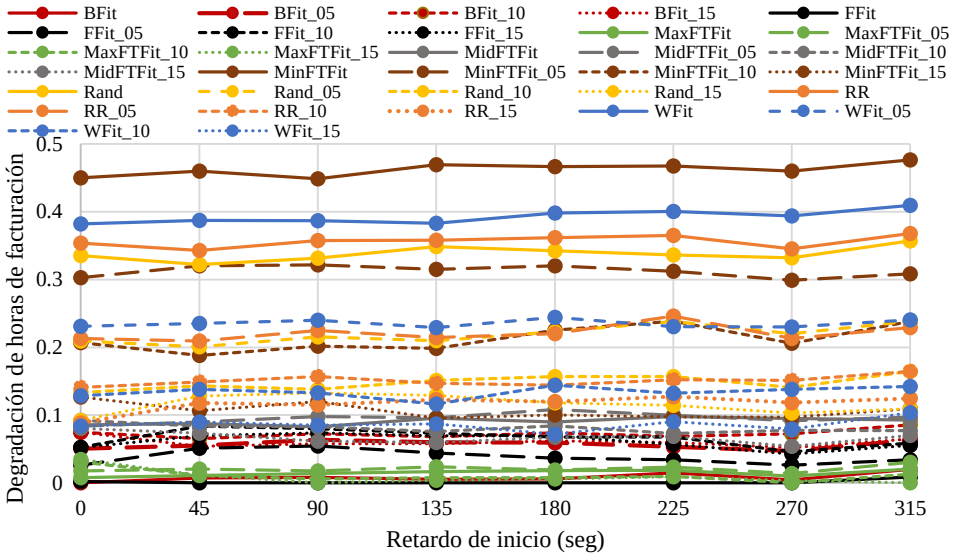

Figura 25. Degradación de horas de facturación para estrategias con umbral de renta.

El promedio de $\bar{c}$ es aproximadamente de 6 llamadas por día para MaxFTFit_15 durante 30 días (la peor estrategia) con StUp igual a 270 segundos (Figura 26). La Tabla 27 muestra la reducción promedio

en $\bar{b}$, la técnica RT beneficia estrategias con alta fluctuación de $\bar{b}$, como MinFTFit, Rand, RR y WFit, y afecta a otras con baja fluctuación de $\bar{b}$ (BFit y FFit). La Tabla 28 presenta el promedio de $\bar{c}$ de las estrategias. BFit_RT y FFit_RT aumentan $\bar{b}$ y $\bar{c}$, mientras que RT reduce $\bar{b}$ pero aumenta $\bar{c}$ en otras estrategias.

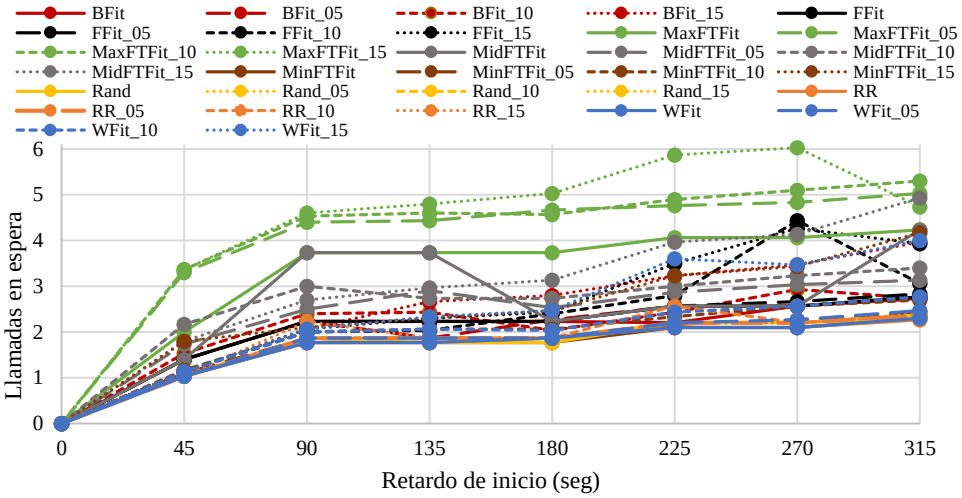

Figura 26. Número promedio de llamadas en espera para estrategias con umbral de renta.

Tabla 27. Reducción promedio de horas de facturación (%).

xx / Strategia	05 min	10 min	15 min.
BFit	- 4.69	- 6.27	- 4.8
FFit	- 4.79	- 6.4	- 5.18
MaxFTFit	- 0.61	0.38	0.59
MidFTFit	- 0.44	0.91	2.19
MinFTFit	10.25	17.07	24.34
Rand	8.88	14.21	16.61
RR	9.97	15.17	17.67
WFit	11.3	18.56	21.98

Tabla 28. Promedio de llamadas en espera por día.

xx / Strategia	No TA	05 min.	10 min.	15 min.
BFit	1.95	1.95	2.06	2.42
FFit	2.00	2.02	2.23	2.45
MaxFTFit	3.20	3.93	4.05	4.30
MidFTFit	2.56	2.34	2.53	2.95
MinFTFit	1.60	1.70	1.86	2.44
Rand	1.64	1.68	1.68	1.68
RR	1.61	1.68	1.72	1.72
WFit	1.62	1.70	1.88	2.38

5.3.2 Estrategias conscientes de la carga con predicción basada en tasa de cambio

Para la predicción con RoC, la ecuación (15) define la regla de predicción para realizar la estimación. La Figura 27 muestra la degradación de $\bar{b}$ frente a diversos StUps. Las estrategias con mejor

rendimiento son BFit_stUp, FFit_stUp y MaxFTFit_stUp, y las peores estrategias son versiones diferentes de MinFTFit.

De forma similar a las estrategias de RT, las estrategias con predicción RoC tienden a ser robustas con respecto a $\bar{b}$. El StUp no afecta considerablemente $\bar{b}$ (en todas las estrategias) y el promedio de $\bar{c}$ es aproximadamente 6 llamadas por día para la peor estrategia, MaxFTFit_stUp con StUp igual a 315 segundos (Figura 28). Las mejores estrategias son MinFTFit_s30, Rand_s30, RR_s30 y WFit_s30, donde $\bar{c}$ varía entre 0.9 y 2.26 llamadas por día.

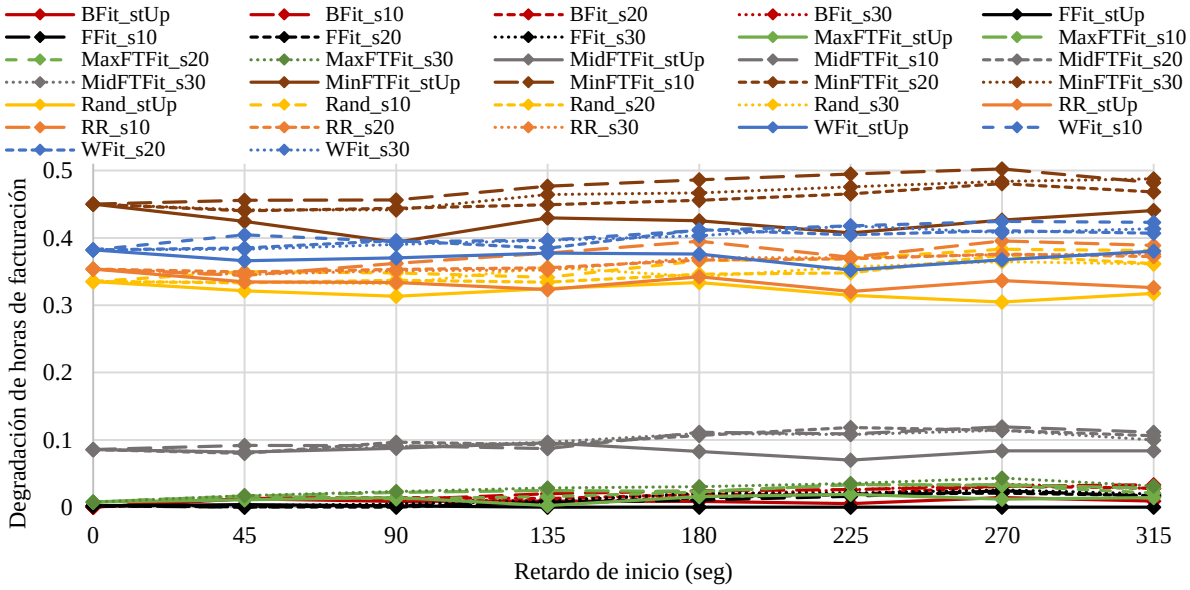

Figura 27. Degradación de horas de facturación para estrategias con predicción basada en tasa de cambio.

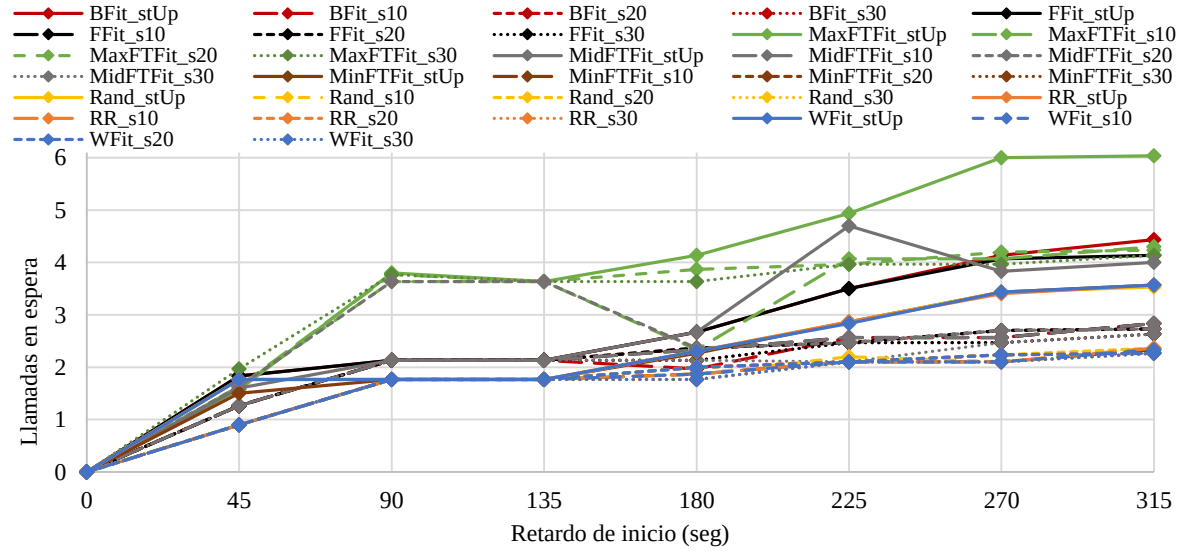

Figure 28. Número promedio de llamadas en espera para estrategias con predicción basada en tasa de cambio.

5.3.3 Estrategias conscientes de la carga con predicción basada en redes neuronales

Para la predicción con NN, las estrategias con el mejor rendimiento en $\bar{b}$ son FFit_stUp, BFit_stUp y MaxFTFit_stUp, todas usan el retardo de tiempo StUp como tiempo de predicción y tienen el peor rendimiento en $\bar{c}$, vea Figura 29 y Figura 30. El promedio de $\bar{c}$ es aproximadamente 16 llamadas por día para la peor estrategia, MaxFTFit_stUp con StUp igual a 270 segundos.

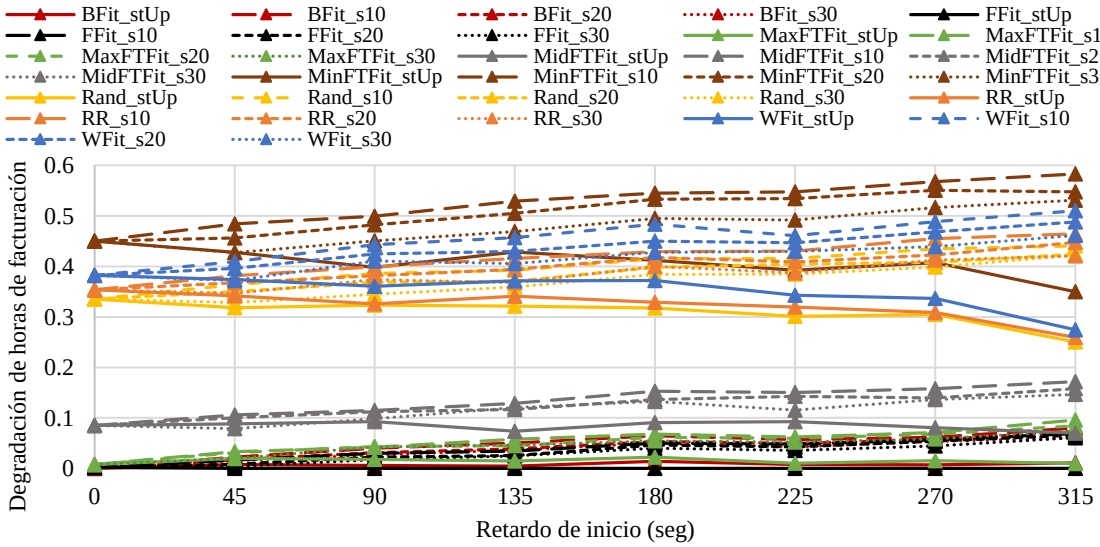

Figura 29. Degradación de horas de facturación para estrategias con predicción basada en redes neuronales.

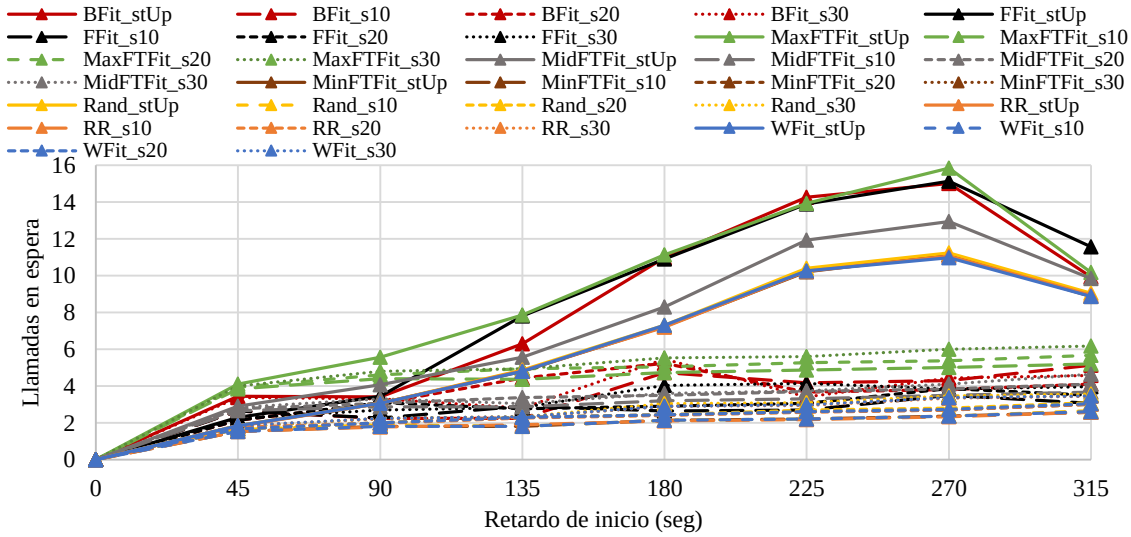

Figura 30. Número promedio de llamadas en espera para estrategias con predicción basada en redes neuronales.

5.3.4 Estrategias conscientes de la carga con predicción basada en información histórica de la carga

De forma similar a RoC y NN, las mejores estrategias en $\bar{b}$ con predicción basada en información histórica de la carga definen StUp igual al tiempo de predicción. Estas estrategias tienen el peor

rendimiento con respecto a $\bar{c}$ (Figura 31 y Figura 32). El promedio de $\bar{c}$ es de aproximadamente 8 llamadas por día para la peor estrategia, MaxFTFit_stUp con StUp igual a 270 segundos.

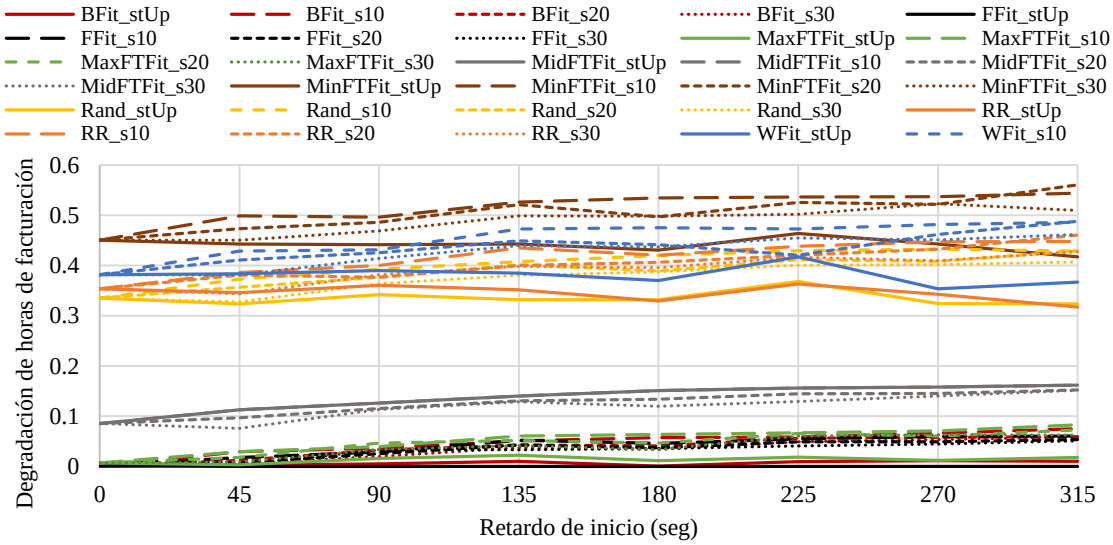

Figura 31. Degradación de horas de facturación para estrategias con predicción basada en información histórica.

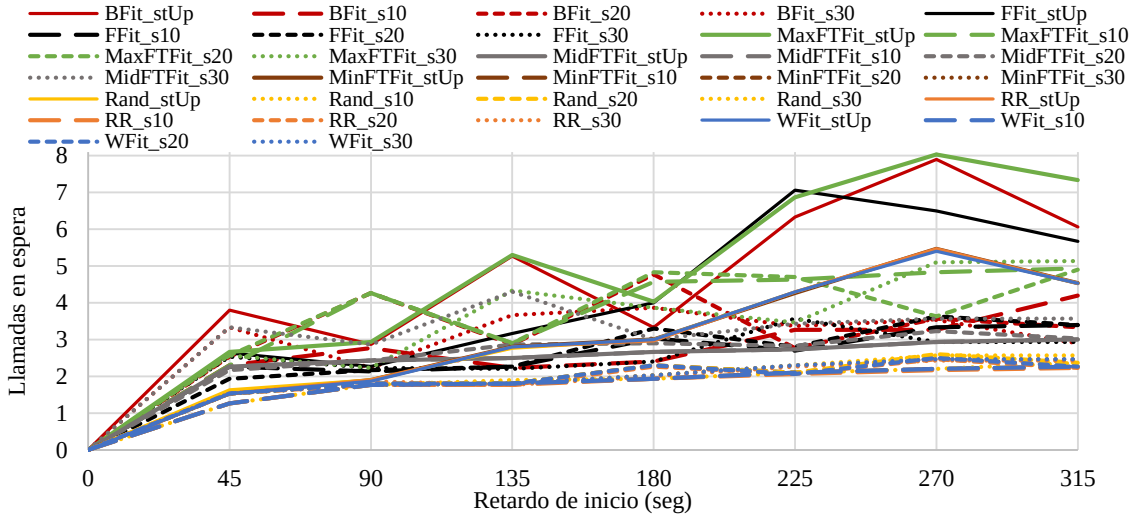

Figura 32. Número promedio de llamadas en espera para estrategias con predicción basada en información histórica.

La Figura 33 muestra un ejemplo de estimación de carga de 12 horas con TS y PI de 315 segundos para los tres modelos de predicción y los valores reales de utilización. La Figura 34 presenta valores similares durante media hora con TS y PI de 10 segundos. En ambas figuras, la predicción de RoC ajusta mejor, más cerca, a los valores reales de carga.

Las tablas 30 a 35 informan los valores de SMAPE y RMSD para los tres modelos de predicción. De acuerdo a las tablas, RoC es el mejor de los tres métodos para construir series de tiempo ajustadas para estimar la carga, incluso con valores altos de StUps y PI.

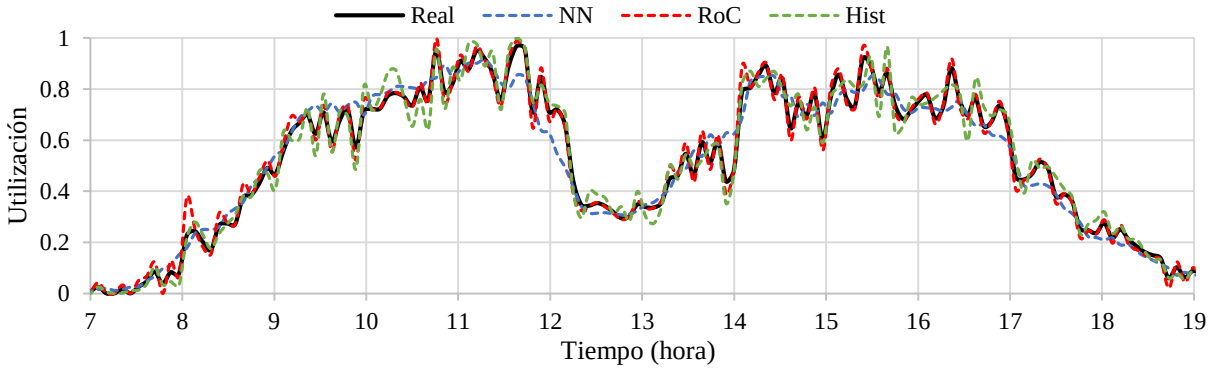

Figura 33. Estimación de la utilización con TS y PI de 315 segundos.

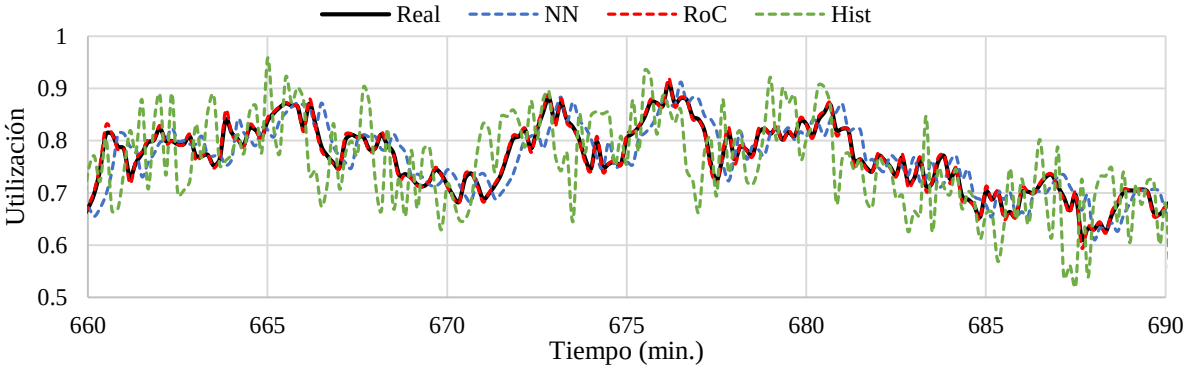

Figura 34. Estimación de la utilización con TS y PI de 10 segundos.

Tabla 29. Valores de SMAPE para la predicción con tasa de cambio.

StUp \ PI	10	20	30	stUp
45	0.006932	0.009974	0.011948	0.014214
90	**0.006895**	0.009883	0.011952	0.018169
135	0.006925	0.009938	0.011934	0.020597
180	0.006908	0.009921	0.011872	0.022022
225	0.006977	0.009967	0.011925	0.023693
270	0.00694	0.009993	0.01192	0.024511
315	0.006931	0.010003	0.011799	**0.02602**

Table 30. Valores de SMAPE para la predicción con redes neuronales.

StUp \ PI	10	20	30	stUp
45	0.019759	0.027758	0.033024	0.039295
90	0.019759	0.027757	0.033035	0.049772
135	0.019759	0.02775	0.033037	0.056384
180	0.**019753**	0.02775	0.033023	0.059849
225	0.019754	0.027757	0.033039	0.064287
270	0.019754	0.027757	0.033037	**0.066532**
315	0.019753	0.027756	0.033037	0.055351

Tabla 31. Valores de SMAPE para la predicción con información histórica de la carga.

StUp \ PI	10	20	30	StUp
45	**0.043347**	0.04312	0.043227	**0.043325**
90	0.043352	0.043123	0.043215	0.042697
135	0.043354	0.043123	0.043218	0.04163
180	0.043353	0.04312	0.043226	0.041102
225	0.04335	0.043115	0.04322	0.040396
270	0.04335	0.043121	0.04322	0.040158
315	0.043353	0.043131	0.043224	0.039967

Tabla 32. Valores de RMSD para la predicción con tasa de cambio.

StUp \ PI	10	20	30	StUp
45	14.0665	19.27764	22.89199	26.63699
90	**13.99144**	19.08354	22.90658	33.77345
135	14.04669	19.17989	22.88319	38.41716
180	14.00646	19.15486	22.75428	41.35787
225	14.1448	19.22651	22.86926	44.67246
270	14.40532	19.46228	23.00976	46.3365
315	14.55159	19.55294	22.84851	**49.61536**

Tabla 33. Valores de RMSD para la predicción con redes neuronales.

StUp \ PI	10	20	30	StUp
45	41.19145	56.64631	67.08032	79.052
90	41.20377	56.64322	67.23338	99.35752
135	41.20377	56.56529	67.24447	112.6882
180	**41.14388**	56.55746	67.06956	118.9171
225	41.15842	56.64213	67.25708	127.746
270	41.15755	56.64164	67.24428	**132.0231**
315	41.14429	56.63986	67.24572	108.897

Tabla 34. Valores de RMSD para la predicción con información histórica de la carga.

StUp \ PI	10	20	30	StUp
45	**81.62947**	80.9945	81.28238	**81.51011**
90	81.63547	80.99913	81.26736	80.95486
135	81.63887	80.99464	81.26956	79.9183
180	81.63899	80.99087	81.28024	77.53849
225	81.63536	80.98568	81.273	77.07457
270	81.6339	80.99453	81.27479	76.93256
315	81.63966	81.00516	81.28059	77.33668

5.4 Análisis bi-objetivo de asignación de llamadas con retardo de inicio

En análisis multi-objetivo, el problema se puede simplificar a un problema de un solo objetivo a través de diferentes métodos de agregación ponderada. Hay varias formas de modelar las preferencias,

por ejemplo, se pueden dar la importancia explícita a cada criterio o una importancia relativa entre los criterios. Esto se puede hacer mediante una definición de ponderaciones de criterios o clasificación de criterios por su importancia.

5.4.1 Degradación del desempeño

En esta sección, la metodología degradación del desempeño se emplea para efectuar un análisis conjunto de dos métricas. Primero, el análisis de la horas de facturación de máquinas virtuales ($\bar{b}$) y el número de llamadas en espera ($\bar{c}$) se realiza por separado. Luego, se encuentra la estrategia que genera el mejor compromiso entre ambas métricas.

La Tabla 36 presenta la degradación promedio de $\bar{b}$, $\bar{c}$ y sus promedios para las mejores estrategias. Las últimas cuatro columnas de la tabla contienen el rango de cada estrategia en relación con el costo del proveedor, la calidad de servicio y los promedios. Rango $\bar{b}$ considera la degradación de $\bar{b}$ para clasificar las estrategias. Rango $\bar{c}$ define la posición de la estrategia en relación con la degradación de $\bar{c}$. Rango promedio establece la posición de la estrategia basada en el promedio de degradación de $\bar{b}$ y $\bar{c}$. Rango es la posición relativa con respecto a todas las estrategias, considera el rango de $\bar{b}$ y el rango $\bar{c}$. Las tablas 36 a 39 contienen la información completa sobre degradación promedio y clasificación para todas las estrategias, ver anexo B.

Tabla 35. Degradación promedio y rango para las mejores estrategias de asignación.

Estrategia	Deg $\bar{b}$	Deg $\bar{c}$	Promedio	Rango $\bar{b}$	Rango $\bar{c}$	Rango promedio	Rango
NN_FFit_stUp	0.0000	4.1342	2.0671	**1**	100	127	32
NN_BFit_stUp	0.0084	3.9974	2.0029	**2**	99	126	32
NN_MaxFTFit_stUp	0.0157	4.4158	2.2158	**3**	101	128	34
RoC_WFit_s30	0.4455	0.0000	0.2227	105	**1**	30	36
RoC_MinFTFit_s30	0.5118	0.0000	0.2559	118	**1**	36	49
RoC_Rand_s30	0.3918	0.0000	0.1959	85	**1**	21	18
RoC_RR_s30	0.4056	0.0000	0.2028	91	**1**	22	23
Roc_MinFTFit_s10	0.5253	0.0132	0.2692	121	**2**	41	53
RoC_WFit_s10	0.4543	0.0132	0.2338	106	**2**	33	38
RoC_RR_s10	0.4192	0.0158	0.2175	95	**3**	27	29
TA_Rand_15	0.1731	0.0579	0.1155	64	11	**1**	12
RoC_FFit_s30	0.0464	0.2026	0.1245	15	28	**2**	**2**
RoC_BFit_s30	0.0517	0.2026	0.1272	20	28	**3**	**3**
RoC_FFit_s10	0.0425	0.2500	0.1462	9	33	9	**1**
RoC_FFit_s20	0.0427	0.2474	0.1451	10	32	8	**1**
RR	0.4222	0.0184	0.2203	97	4	29	32

La mejor estrategia para $\bar{b}$ es NN_FFit_stUp que asigna las llamadas en función de la estrategia First Fit, donde se coloca la llamada en la primera VM. El tiempo de predicción es igual al tiempo de inicio

de VM. Sin embargo, es la segunda peor estrategia para $\bar{c}$. Tiende a aumentar la utilización y reducir la calidad. NN_BFit_stUp y NN_MaxFTFit_stUp son la segunda y tercera mejores estrategias (con respecto a $\bar{b}$), todas las estrategias usan redes neuronales para predecir la llegada de llamadas futuras.

Las mejores estrategias para $\bar{c}$ son RoC_MinFTFit_s30 (donde se ponen las llamadas en la VM donde el tiempo de alquiler es el más cercano a finalizar), RoC_Rand_s30, RoC_RR_s30 y RoC_WFit_s30. Las estrategias usan el algoritmo de tasa de cambio para predecir la carga de trabajo y el tiempo de predicción es de 30 segundos. Estas estrategias tienden a subutilizar las VMs reduciendo $\bar{c}$, pero aumenta $\bar{b}$.

Las estrategias con buen compromiso entre $\bar{b}$ y $\bar{c}$ son: Rand_15, donde se asignan las llamadas a las VMs aleatoriamente. Se trata de reducir $\bar{b}$ mediante limitar la asignación de llamadas 15 minutos antes de que las VMs alcancen su tiempo de renta. Las estrategias RoC_FFit_s30 y RoC_BFit_s30 asignan las llamadas a las VMs utilizando FFit y BFit, y predicen la carga con tasa de cambio cada 30 segundos.

Las estrategias con el mejor rango entre todas son: RoC_FFit_s10, RoC_FFit_s20, RoC_FFit_s30 y RoC_BFit_s30. Todas ellas tienden a estar en la posición más alta con respecto a otras estrategias en ambos criterios.

Con respecto a la estrategia para sistemas VoIP actualmente en uso, 28 estrategias mejoran el rendimiento de Round Robin considerando la degradación promedio de $\bar{b}$ y $\bar{q}$. El desempeño de Round Robin ocupa el puesto 32 del rango, respecto a la posición relativa de todas las estrategias, por lo que al menos 52 estrategias tienen un desempeño igual o mejor que esta estrategia.

Capítulo 6. Conclusiones

En este capítulo se presenta un resumen del trabajo general, las conclusiones finales del estudio, la contribución al conocimiento de esta investigación, las limitaciones del trabajo y algunas ideas para trabajo futuro.

Este trabajo de investigación formula y estudia el problema de asignación de llamadas de VoIP en la nube mediante estrategias conscientes de la carga con varios parámetros. La formulación define las características del ambiente, las características de las llamadas y los criterios de optimización para evaluar el rendimiento de las estrategias.

El modelo de CVoIP considera el costo del proveedor, determinado por las horas de facturación de las máquinas virtuales usadas en la nube, y la calidad del servicio. La calidad del servicio se expresar por el número de llamadas afectadas por la falta de recursos (llamadas en espera) y el deterioro en la calidad de voz debido al procesamiento de las llamadas.

6.1 Conclusiones finales

En este trabajo de investigación se propuso un nuevo modelo de voz sobre IP basado en ambientes nube con optimización bi-objetivo. El modelo considera un caso especial del problema de empaquetamiento en línea dinámico no clarividente y define los elementos más representativos de la arquitectura de VoIP: arquitectura distribuida del sistema, recursos virtuales, software, modelo de costo, modelo de calidad de servicio, asignación de recursos, etc. Se demuestra que el ambiente es razonable y representativo de instalaciones y aplicaciones reales.

Las estrategias de asignación basadas en empaquetamiento multi-objetivo consideran varios parámetros para asignar las llamadas en el sistema de CVoIP. Dependiendo de la configuración de los parámetros, las estrategias pueden reducir el costo de renta en infraestructura, incrementar la calidad de servicio o una combinación de ambas. La evaluación demuestra los beneficios potenciales y estabilidad con respecto al manejo de las tasas de llegada de llamadas y la variación de los retardos de inicio en los recursos virtuales.

El análisis de la evaluación de las estrategias de asignación, mediante simulación con cargas de trabajo reales de la compañía MIXvoip, muestra que los algoritmos propuestos pueden ser usados en entornos reales de VoIP basados en la nube.

Los resultados muestran la robustez de las estrategias frente a variaciones en el tiempo de inicio de las máquinas virtuales, incluso con una gran dispersión del tiempo de inicio. Las estrategias de asignación conscientes de la carga superan a las conocidas porque ofrecen una calidad de servicio adecuada y un menor costo.

Sin embargo, se requiere más estudios para validar el desempeño y eficiencia de las estrategias en un dominio real, esto será tema del trabajo futuro.

6.2 Contribución al conocimiento

La investigación contribuye al problema de asignación de llamadas en voz sobre IP en el campo del cómputo en la nube, las aportaciones de este trabajo de tesis son:

- Definimos un modelo de voz sobre IP basado en la nube.
- Abordamos el problema de asignación de llamadas multi-objetivo en sistemas de CVoIP.
- Proponemos estrategias de asignación de llamadas basadas en empaquetamiento para adaptarse al contexto dinámico de los ambientes CVoIP.
- Analizamos el impacto de los retardos de inicio en los algoritmos de asignación de llamadas en ambientes de voz sobre IP en la nube.
- Evaluamos modelos de predicción de llamadas para reducir la degradación de la calidad de servicio y el número de llamadas afectadas por la falta de recursos.
- Desarrollamos un marco de simulación para sistemas de VoIP en la nube.

Estas aportaciones son la base de las siguientes publicaciones:

6.2.1 Artículos de revista

1. Andrei Tchernykh, ***Jorge M. Cortés-Mendoza***, Alexander Feoktistov, Igor Bychkov, Loic Didelot, Pascal Bouvry, Gleb Radchenko, and Kirill Borodulin. Configurable Cost-Quality Optimization of Cloud-based VoIP. Journal of Parallel and Distributed Computing, Special issue on "Advances in Parallel and Distributed Computing and Optimization", Elsevier, 2017. IF 1.930, Q2 (under review).

2. ***Jorge M. Cortés-Mendoza***, Andrei Tchernykh, Fermin A. Armenta-Cano, Pascal Bouvry, Alexander Yu. Drozdov, and Loic Didelot, 2016. Biobjective VoIP Service Management in Cloud Infrastructure. Scientific Programming, vol. 2016, Article ID 5706790, 14 pages, IF 0.455, Q3.

3. ***Jorge M. Cortés-Mendoza***, Andrei Tchernykh, Ana-Maria Simionovici, Pascal Bouvry, Sergio Nesmachnow, Bernabe Dorronsoro, and Loic Didelot, 2015. VoIP Service Model for Multi-objective Scheduling in Cloud Infrastructure. IJMHeur - International Journal of Metaheuristics. Inderscience. Genève, Switzerland, Vol. 4, No. 2, p. 185-203.

6.2.2 Artículos de conferencia

1. ***Jorge M. Cortés-Mendoza***, Andrei Tchernykh, Gleb Radchenko, and Alexander Yu. Drozdov, 2017. RoC Prediction for Bi-Objective Cost-QoS Optimization of Cloud VoIP Call Allocations. In *Engineering and Telecommunication (EnT), 2017 IVth International Conference on* (pp. 119-123).

2. ***Jorge Mario Cortés-Mendoza***, Andrei Tchernykh, Alexander Feoktistov, Igor Bychkov, and Loic Didelot, 2017. Load-Aware Strategies for Cloud-Based VoIP Optimization with VM Startup Prediction. In *Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017 IEEE International* (pp. 472-481).

3. ***Jorge M. Cortés-Mendoza***, Andrei Tchernykh, Alexander Yu. Drozdov, and Loic Didelot, 2016. Robust cloud VoIP scheduling under VMs startup time delay uncertainty. In *Utility and Cloud Computing (UCC), 2016 IEEE/ACM 9th International Conference on* (pp. 234-239).

4. Ana-Maria Simionovici, Alexandru Adrian Tantar, Pascal Bouvry, Andrei Tchernykh, ***Jorge M. Cortés-Mendoza***, and Loic Didelot, 2015. Voip traffic modelling using gaussian mixture models, gaussian processes and interactive particle algorithms. In *Globecom Workshops (GC Wkshps), 2015 IEEE* (pp. 1-6).

5. ***Jorge M. Cortés-Mendoza*****,** Andrei Tchernykh, Alexander Yu. Drozdov, Pascal Bouvry, Ana-Maria Simionovici, and Arutyun Avetisyan, 2015. Distributed Adaptive VoIP Load Balancing in Hybrid Clouds. NC&SC'2015 - Network Computing and Supercomputing workshop. In conjunction with RuSCDays'15 - The Russian Supercomputing Days, CEUR-WS: Vol-1482 (pp. 676-686).

6. Andrei Tchernykh, ***Jorge M. Cortés-Mendoza***, Johnatan Pecero, Pascal Bouvry, and Dzmitry Kliazovich, 2014. Adaptive energy efficient distributed VoIP load balancing in federated cloud infrastructure. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on* (pp. 27-32).

6.3 Limitaciones de la investigación

La siguiente sección describe las limitaciones de esta investigación, estás se dividen en tres principales aspectos: el ambiente nube de VoIP, la carga de trabajo y los modelos de predicción.

El ambiente de simulación es una abstracción limitada de un ambiente real de nube VoIP donde los aspectos más representativos son modelados. El enfoque contempla elementos relevantes que afectan significativamente el costo y la calidad del servicio, y omite otros menos importantes. El modelo propuesto es representativo de entornos nubes reales y puede ser la base de futuros trabajos donde componentes adicionales pueden ser agregados.

La carga de trabajo es proporcionada por la compañía MIXvoip, un proveedor de VoIP para pequeñas y medianas empresas. Las limitaciones con respecto a la carga de trabajo incluyen:

El número de llamadas restringe los resultados de la investigación a proveedores de VoIP pequeños, similares a MIXvoip, y no permite un estudio del comportamiento complejo en sistemas grandes de CVoIP, como Skype. Adicionalmente, el número de llamadas en la carga de trabajo confina el número de VMs ejecutándose en el sistema al mismo tiempo, esta situación limita el estudio de la interacción entre el balanceador y los recursos para hacer frente a la llegada de llamadas.

El patrón de llegada de las llamadas no permite evaluar adecuadamente los algoritmos de predicción de carga. MIXvoip proporciona servicio a compañías con horario de trabajo similar, la mayoría de estas compañías están localizadas relativamente cerca (en países vecinos con zonas horarias similares). Esta circunstancia facilita estimar el comportamiento de la carga de trabajo(por ejemplo, baja llegada de llamadas los fines de semana y durante vacaciones, incremento de la carga en horas de trabajo y bajos niveles de carga en horario no laboral, etc.).

Similar al punto anterior, la distribución geográfica del sistema de MIXvoip abarca una región específica de Europa. Todo el tráfico que fluye en el SNC es local, ambos puntos reciben servicio dentro del mismo SNC. La interacción de usuarios que pertenecen a diferentes áreas está fuera del alcance de esta investigación.

Los modelos de predicción utilizados en la investigación son sencillos. Modelos de predicción más poderosos y complejos se pueden encontrar en la literatura, estos se pueden adaptar para tratar los factores cambiantes de la nube. Nuestra contribución es definir un entorno VoIP en la nube que pueda incorporar diferentes tipos de predictores: 1) modelo pre entrenado como redes neuronales, 2) modelos con información pre procesada como predicción basada en el historial y 3) modelos con información actual del sistema como tasa de cambio. El sistema de CVoIP desarrollado facilita la implementación de otros modelos de predicción.

6.4 Trabajo futuro

Esta investigación proporciona las bases para futuros estudios en el área de VoIP en la nube. Los estudios futuros deben reducir las limitaciones del modelo actual de CVoIP. La investigación en este campo se puede llevar a cabo en cinco áreas:

1. Implementación de estrategias de asignación en sistemas reales de CVoIP, la implementación de las estrategias en un entorno de nube real puede validar los resultados de esta investigación. Un análisis riguroso de las estrategias en un entorno controlado puede ratificar los beneficios de las estrategias o mostrar su ineficacia en ambientes reales. En ambos casos, el análisis puede revelar factores críticos que limitan el rendimiento de las estrategias propuestas.

2. Uso de nuevos modelos de servicio, el modelo de contenedores como servicio (CaaS) puede simplificar la implementación y administración de los sistemas CVoIP. El modelo actual de CVoIP está basado en VMs para proporcionar el servicio, las VMs incrementan el tiempo de despliegue de la infraestructura, por lo tanto, disminuyen la calidad del servicio. Los contenedores pueden reducir o eliminar el problema del retardo de inicio porque este factor está asociado directamente con la tecnología de VMs. Por lo tanto, una implementación de sistemas CVoIP basado en CaaS involucra mejoras significativas en la calidad del servicio.

3. Mejorar la abstracción del entorno de simulación, varios elementos del entorno nube y ambientes VoIP no se consideran en esta investigación. Limitaciones de tiempo y bajo impacto en la calidad de servicio restringen el alcance de esta investigación. Algunos factores importantes que afectan sistemas de CVoIP reales y deberían estudiarse comprenden: interacción entre varios SNCs, diferentes tamaños de instancias de VMs (por ejemplo, CPU, ancho de banda, memoria, etc.), mayor número de códecs y fallas en la infraestructura.

4. Analizar el comportamiento de las estrategias con cargas de trabajo de sistemas VoIP más grandes, el objetivo de la investigación es entender el funcionamiento del sistema CVoIP con diferentes tamaños de entrada. Lamentablemente, el rendimiento de las estrategias no se puede generalizar, por lo tanto, resulta indispensable mostrar la eficacia de las estrategias con cargas de trabajo de mayor tamaño. El objetivo principal es analizar el comportamiento de las estrategias con variaciones de carga impredecibles.

5. Introducir modelos de predicción más precisos, los resultados de la investigación muestran las mejoras de las estrategias de asignación con modelos básicos de predicción. Una configuración adecuada de los parámetros de los algoritmos de predicción aumenta la precisión de los modelos, por lo tanto, reduce la degradación de la calidad de servicio. El marco de simulación facilita la implementación de otros enfoques de predicción para CVoIP, por ejemplo, procesos Gaussiano, modelo de mezcla Gaussiana y sistema de partículas interactivas demostraron su eficiencia en el dominio de este problema.

Referencias bibliográficas

Alakeel, A.M., 2010. A guide to dynamic load balancing in distributed computer systems. International Journal of Computer Science and Information Security, 10(6), pp.153-160.

Almorsy, M., Grundy, J. and Müller, I., 2016. An analysis of the cloud computing security problem. arXiv preprint arXiv:1609.01107.

Amazon Web Services (AWS), 2018. https://aws.amazon.com/es/solutions/case-studies/. Consultado el 1 de marzo, 2018.

Armenta-Cano, F., Tchernykh, A., Cortés-Mendoza, J. M., Yahyapour, R., Drozdov, A. Y., Bouvry, P., and Nesmachnow, S., 2015. Min_c: heterogeneous concentration policy for power aware scheduling. *Proceedings of the Institute for System Programming of the RAS*, *27*(6), 355-380.

Bhadani, A. and Chaudhary, S., 2010. Performance evaluation of web servers using central load balancing policy over virtual machines on cloud. In Proceedings of the Third Annual ACM Bangalore Conference (p. 16).

Bayer, N., Xu, B., Rakocevic, V. and Habermann, J., 2010. Application-aware scheduling for VoIP in wireless mesh networks. Computer Networks, 54(2), pp.257-277.

Bell, W., Cameron, D., Capozza, L., Millar, A., Stockinger, K. and Zini, F., 2002. Simulation of dynamic grid replication strategies in optorsim. Grid Computing—GRID 2002, pp.46-57.

Beloglazov, A., Buyya, R., Lee, Y.C. and Zomaya, A., 2011. A taxonomy and survey of energy-efficient data centers and cloud computing systems. Advances in computers, 82(2), pp.47-111.

Brocade, 2018. http://www.brocade.com/en.html. Consultado el 1 de marzo, 2018.

Bux, M. and Leser, U., 2015. DynamicCloudsim: Simulating heterogeneity in computational clouds. Future Generation Computer Systems, 46, pp.85-99.

Buyya, R., Calheiros, R.N. and Li, X., 2012. Autonomic cloud computing: Open challenges and architectural elements. In Emerging Applications of Information Technology (EAIT), 2012 Third International Conference on (pp. 3-10).

Calheiros, R.N., Masoumi, E., Ranjan, R. and Buyya, R., 2015. Workload prediction using ARIMA model and its impact on cloud applications' QoS. IEEE Transactions on Cloud Computing, 3(4), pp.449-458.

Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A. and Buyya, R., 2011. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and experience, 41(1), pp.23-50.

Campos, L.M. and Scherson, I.D., 2000. Rate of change load balancing in distributed and parallel systems. Parallel Computing, 26(9), pp.1213-1230.

Casanova, H., 2001. Simgrid: A toolkit for the simulation of application scheduling. In Cluster computing and the grid, proceedings. First ieee/acm international symposium on (pp. 430-437).

Chan, J.W.T., Lam, T.W. and Wong, P.W., 2008. Dynamic bin packing of unit fractions items. Theoretical Computer Science, 409(3), pp.521-529.

Chan, J.W.T., Wong, P.W. and Yung, F.C., 2009. On dynamic bin packing: An improved lower bound and resource augmentation analysis. Algorithmica, 53(2), pp.172-206.

Chauvin, Y., y Rumelhart, D. E., 1995. Backpropagation: theory, architectures, and applications. Psychology Press.

Cheng, K., Bai, Y., Wang, R. and Ma, Y., 2015. Optimizing soft real-time scheduling performance for virtual machines with srt-xen. In Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on (pp. 169-178).

Citrix, 2018. Load Balancing Algorithms. https://docs.citrix.com/en-us/netscaler/10-1/ns-tmg-wrapper-10-con/ns-lb-wrapper-con-10/ns-lb-customizing-lbalgorithms-wrapper-con.html/. Consultado el 1 de marzo, 2018.

Coffman, Jr, E.G., Garey, M.R. and Johnson, D.S., 1983. Dynamic bin packing. SIAM Journal on Computing, 12(2), pp.227-258.

Corrado, E.M. and Moulaison, H.L., 2011. Getting started with cloud computing: A LITA guide. Neal-Schuman Publishers, Inc.

Costa, L.R., Nunes, L.S.N., Bordim, J.L. and Nakano, K., 2015. Asterisk PBX capacity evaluation. In Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International (pp. 519-524).

Dang, T.D., Sonkoly, B. and Molnár, S., 2004. Fractal analysis and modeling of VoIP traffic. In Telecommunications Network Strategy and Planning Symposium. NETWORKS 2004, 11th International (pp. 123-130).

Dumitrescu, C.L. and Foster, I., 2005. GangSim: a simulator for grid scheduling studies. In Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International Symposium on (Vol. 2, pp. 1151-1158).

Eleftheriou, Charalambos. 2010. 3CX Phone System and ATOM N270 Processor Benchmarking. http://www.3cx.com/blog/voip-howto/atom-processor-n270-benchmarking/. Consultado el 1 de marzo, 2018.

Fang, Y., Wang, F. and Ge, J., 2010. A task scheduling algorithm based on load balancing in cloud computing. Web Information Systems and Mining, pp.271-277.

Fittkau, F., Frey, S. and Hasselbring, W., 2012. CDOSim: Simulating cloud deployment options for software migration support. In Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2012 IEEE 6th International Workshop on the (pp. 37-46).

Florence, A.P. and Shanthi, V., 2014. A load balancing model using firefly algorithm in cloud computing. Journal of Computer Science, 10(7), p. 1156.

Folke, M., Landstrom, S., Bodin, U. and Wanstedt, S., 2007. Scheduling support for mixed VoIP and web traffic over HSDPA. In Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th (pp. 814-818).

Future market insights, 2017. VOIP Services Market: Global Industry Analysis and Opportunity Assessment 2015 – 2025, https://www.futuremarketinsights.com/reports/global-voip-services-market. Consultado el 1 de marzo, 2018.

Galloway, J.M., Smith, K.L. and Vrbsky, S.S., 2011. Power aware load balancing for cloud computing. In Proceedings of the World Congress on Engineering and Computer Science (Vol. 1, pp. 19-21).

Goldberg, R.P., 1974. Survey of virtual machine research. Computer, 7(6), pp.34-45.

González-García, J.L., Yahyapour, R. and Tchernykh, A., 2013. Load Balancing for Parallel Computations with the Finite Element Method. Computación y Sistemas, 17(3).

Google Cloud Services, 201. Google Cloud Platform Customer Success. https://cloud.google.com/customers/. Consultado el 1 de marzo, 2018.

Gunarathne, T., Wu, T.L., Qiu, J. and Fox, G., 2010. MapReduce in the Clouds for Science. In Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on (pp. 565-572).

Hoffman, K., 2015. Ready. Steady. Go! The Speed of VM Creation and SSH Access. http://blog.cloud66.com/ready-steady-go-the-speed-of-vm-creation-and-ssh-key-access-on-aws-digitalocean-linode-vexxhost-google-cloud-rackspace-and-microsoft-azure/. Consultado el 1 de marzo, 2018.

Hu, J., Gu, J., Sun, G. and Zhao, T., 2010. A scheduling strategy on load balancing of virtual machine resources in cloud computing environment. In Parallel Architectures, Algorithms and Programming (PAAP), 2010 Third International Symposium on (pp. 89-96).

Janssens, N., An, X., Daenen, K., and Forlivesi, C., 2012. Dynamic scaling of call-stateful SIP services in the cloud. NETWORKING 2012, 175-189.

Jararweh, Y., Alshara, Z., Jarrah, M., Kharbutli, M. and Alsaleh, M.N., 2013. Teachcloud: a cloud computing educational toolkit. International Journal of Cloud Computing 1, 2(2-3), pp.237-257.

Johnson, D.S., 1973. Near-optimal bin packing algorithms (Doctoral dissertation), Massachusetts Institute of Technology.

Kansal, N.J. and Chana, I., 2012. Cloud load balancing techniques: A step towards green computing. IJCSI International Journal of Computer Science Issues, 9(1), pp.238-246.

Kaur, J., 2012. Comparison of load balancing algorithms in a cloud. International Journal of Engineering Research and Applications, 2(3), pp.1169-173.

Kim, J. Y., 2016. On SIP Server Clusters and the Migration to Cloud Computing Platforms. Columbia University.

Kliazovich, D., Bouvry, P. and Khan, S.U., 2012. GreenCloud: a packet-level simulator of energy-aware cloud computing data centers. The Journal of Supercomputing, 62(3), pp.1263-1283.

Kokilavani, T. and Amalarethinam, D.G., 2011. Load balanced min-min algorithm for static meta-task scheduling in grid computing. International Journal of Computer Applications, 20(2), pp.43-49.

Laredo, J.J., Dorronsoro, B., Pecero, J., Bouvry, P., Durillo, J.J. and Fernandes, C., 2012. Designing a self-organized approach for scheduling bag-of-tasks. In P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2012 Seventh International Conference on (pp. 315-320).

LD, D.B. and Krishna, P.V., 2013. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing, 13(5), pp.2292-2303.

Lee, H., Kwon, T., Cho, D.H., Lim, G. and Chang, Y., 2006. Performance analysis of scheduling algorithms for VoIP services in IEEE 802.16 e systems. In Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd (Vol. 3, pp. 1231-1235).

Li, Y., Tang, X. and Cai, W., 2015. Dynamic bin packing for on-demand cloud resource allocation. IEEE Transactions on Parallel and Distributed Systems, 27(1), pp.157-170.

Lim, S.H., Sharma, B., Nam, G., Kim, E.K. and Das, C.R., 2009. MDCSim: A multi-tier data center simulation, platform. In Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on (pp. 1-9).

Liu, H., Liu, S., Meng, X., Yang, C. and Zhang, Y., 2010. LBVS: A load balancing strategy for virtual storage. In Service Sciences (ICSS), 2010 International Conference on (pp. 257-262).

Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y. and Alsaadi, F.E., 2017. A survey of deep neural network architectures and their applications. Neurocomputing, 234, pp.11-26.

Liu, X., Pan, L., Wang, C.J. and Xie, J.Y., 2011. A lock-free solution for load balancing in multi-core environment. In Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on (pp. 1-4).

Lu, Y., Panneerselvam, J., Liu, L. and Wu, Y., 2016. RVLBPNN: A workload forecasting model for smart cloud computing. Scientific Programming.

Lu, Y., Xie, Q., Kliot, G., Geller, A., Larus, J.R. and Greenberg, A., 2011. Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services. Performance Evaluation, 68(11), pp.1056-1071.

Mao, M. and Humphrey, M., 2012. A performance study on the vm startup time in the cloud. In Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on (pp. 423-430).

Mazalek, A., Vranova, Z. and Stankova, E., 2015. Analysis of the impact of IPSec on performance characteristics of VoIP networks and voice quality. In Military Technologies (ICMT), 2015 International Conference on (pp. 1-5).

Meggelen J.V., Madsen L., and Bryant R., 2013. Asterisk: The Definitive Guide. Fourth edition, O'Reilly Media. 846 pp.

Mehta, H., Kanungo, P. and Chandwani, M., 2011. Decentralized content aware load balancing algorithm for distributed computing environments. In Proceedings of the International Conference & Workshop on Emerging Trends in Technology (pp. 370-375).

Mell, P. and Grance, T., 2011. The NIST definition of cloud computing. United States: National Institute of Standards & Technology.

MIXvoip, 2018. https://www.mixvoip.com/. Consultado el 1 de marzo, 2018.

Mobicents, 2018. http://www.mobicents.org/. Consultado el 1 de marzo, 2018.

Montazerolghaem, A., Yaghmaee, M.H., Leon-Garcia, A., Naghibzadeh, M. and Tashtarian, F., 2016. A load-balanced call admission controller for ims cloud computing. IEEE Transactions on Network and Service Management, 13(4), pp.806-822.

Montoro, P. and Casilari, E., 2009. A comparative study of VoIP standards with asterisk. In Digital Telecommunications, 2009. ICDT'09. Fourth International Conference on (pp. 1-6).

Murshed, M., Buyya, R. and Abramson, D., 2001. Gridsim: A toolkit for the modeling and simulation of global grids. Monash University Journal, pp.1-15.

Nae, V., Prodan, R. and Fahringer, T., 2010. Cost-efficient hosting and load balancing of massively multiplayer online games. In Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on (pp. 9-16).

Nakai, A.M., Madeira, E. and Buzato, L.E., 2011. Load balancing for internet distributed services using limited redirection rates. In Dependable Computing (LADC), 2011 5th Latin-American Symposium on (pp. 156-165).

Ni, J., Huang, Y., Luan, Z., Zhang, J. and Qian, D., 2011. Virtual machine mapping policy based on load balancing in private cloud environment. In Cloud and Service Computing (CSC), 2011 International Conference on (pp. 292-295).

Núñez, A., Vázquez Poletti, J.L., Caminero, C., González Castañé, G., Carretero Pérez, J. and Llorente, I.M., 2012. iCanCloud: A flexible and scalable cloud infrastructure simulator. J Grid Computing 10(1), pp. 185–209.

Parikh, S.M., 2013. A survey on cloud computing resource allocation techniques. In Engineering (NUiCONE), 2013 Nirma University International Conference on (pp. 1-5).

Piraghaj, S.F., Dastjerdi, A.V., Calheiros, R.N. and Buyya, R., 2017. ContainerCloudSim: An environment for modeling and simulation of containers in cloud data centers. Software: Practice and Experience, 47(4), pp.505-521.

Radojević, B. and Žagar, M., 2011. Analysis of issues with load balancing algorithms in hosted (cloud) environments. In MIPRO, 2011 Proceedings of the 34th International Convention (pp. 416-420).

Radware, 2018. VoIP Load Balancing: Radware VoIP Traffic Management Solutions. https://www.radware.com/Resources/voip_load_balancing.aspx. Consultado el 1 de marzo, 2018.

Ramezani, F., Lu, J. and Hussain, F.K., 2014. Task-based system load balancing in cloud computing using particle swarm optimization. International journal of parallel programming, 42(5), pp.739-754.

Randles, M., Lamb, D. and Taleb-Bendiab, A., 2010. A comparative study into distributed load balancing algorithms for cloud computing. In Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on (pp. 551-556).

Razavi, K., Razorea, L.M. and Kielmann, T., 2013. Reducing vm startup time and storage costs by vm image content consolidation. In European Conference on Parallel Processing (pp. 75-84). Springer, Berlin, Heidelberg.

Rodriguez, M. A., and Buyya, R., 2018. Scheduling dynamic workloads in multi-tenant scientific workflow as a service platforms. *Future Generation Computer Systems*,*79*, 739-750.

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E., 2002. RFC 3261: SIP: Session Initiation Protocol.

Samal, P. and Mishra, P., 2013. Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing. International Journal of computer science and Information Technologies, 4(3), pp.416-419.

Sánchez, B. P., 2013. On-Demand Secure Teleconferencing on Public Cloud Infrastructures. Universidad Complutense de Madrid, Master Thesys.

Sethi, S., Sahu, A. and Jena, S.K., 2012. Efficient load balancing in cloud computing using fuzzy logic. IOSR Journal of Engineering, 2(7), pp.65-71.

Simionovici, A.M., Tantar, A.A., Bouvry, P., Tchernykh, A., Cortés-Mendoza, J.M. and Didelot, L., 2015. Voip traffic modelling using gaussian mixture models, gaussian processes and interactive particle algorithms. In Globecom Workshops (GC Wkshps), 2015 IEEE (pp. 1-6).

Simionovici, A.M., Tantar, A., Bouvry, P. and Didelot, L., 2013. Predictive modeling in a voip system. Journal of Telecommunications and Information Technology, 4, pp.32-40.

Singh, A., Korupolu, M. and Mohapatra, D., 2008. Server-storage virtualization: integration and load balancing in data centers. In Proceedings of the 2008 ACM/IEEE conference on Supercomputing (p. 53).

So, J., 2011. Scheduling and capacity of VoIP services in wireless OFDMA systems. In VoIP Technologies. InTech.

Song, H.J., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K. and Chien, A., 2000. The microgrid: a scientific tool for modeling computational grids. In Supercomputing, ACM/IEEE 2000 Conference (pp. 53-53).

Stanojevic, R. and Shorten, R., 2009. Load balancing vs. distributed rate limiting: an unifying framework for cloud control. In Communications, 2009. ICC'09. IEEE International Conference on (pp. 1-6).

Tchernykh, A., Pecero, J.E., Barrondo, A. and Schaeffer, E., 2014. Adaptive energy efficient scheduling in Peer-to-Peer desktop grids. Future Generation Computer Systems, 36, pp.209-220.

Tchernykh, A., Schwiegelsohn, U., Talbi, E. G., and Babenko, M., 2016. Towards understanding uncertainty in cloud computing with risks of confidentiality, integrity, and availability. Journal of Computational Science.

Tchernykh, A., Lozano, L., Schwiegelshohn, U., Bouvry, P., Pecero, J.E., Nesmachnow, S. and Drozdov, A.Y., 2016. Online bi-objective scheduling for IaaS clouds ensuring quality of service. Journal of Grid Computing, 14(1), pp.5-22.

Tsafrir, D., Etsion, Y. and Feitelson, D.G., 2007. Backfilling using system-generated predictions rather than user runtime estimates. IEEE Transactions on Parallel and Distributed Systems, 18(6), pp.789-803.

Ullman, J.D., 1971. The performance of a memory allocation algorithm. Department of Electrical Engineering. Computer Science Laboratory: Princeton University.

UIT-T, 2008. Definiciones de términos relativos a la calidad de servicio. https://www.itu.int/rec/T-REC-E.800-200809-I/es. Consultado el 1 de marzo, 2018.

Walsh, T.J. and Kuhn, D.R., 2005. Securing Voice Over IP Networks. IEEE Computer Security and Privacy, 3(3).

Wang, S.C., Yan, K.Q., Liao, W.P. and Wang, S.S., 2010. Towards a load balancing in a three-level cloud computing network. In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on (Vol. 1, pp. 108-113).

Wolke, A., Tsend-Ayush, B., Pfeiffer, C. and Bichler, M., 2015. More than bin packing: Dynamic resource allocation strategies in cloud data centers. Information Systems, 52, pp.83-95.

Wong, P.W., Yung, F.C. and Burcea, M., 2012. An 8/3 Lower Bound for Online Dynamic Bin Packing. In ISAAC (Vol. 7676, pp. 44-53). In Algorithms and Computation, 44–53. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg.

Wu, S., Zhou, L., Fu, D., Jin, H. and Shi, X., 2014. A real-time scheduling framework based on multi-core dynamic partitioning in virtualized environment. In IFIP International Conference on Network and Parallel Computing (pp. 195-207). Springer, Berlin, Heidelberg.

Zhang, Z. and Zhang, X., 2010. A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation. In Industrial Mechatronics and Automation (ICIMA), 2010 2nd International Conference on (Vol. 2, pp. 240-243).

Zhao, W., Peng, Y., Xie, F. and Dai, Z., 2012. Modeling and simulation of cloud computing: A review. In Cloud Computing Congress (APCloudCC), 2012 IEEE Asia Pacific (pp. 20-24).

Zhao, Y. and Huang, W., 2009. Adaptive distributed load balancing algorithm based on live migration of virtual machines in cloud. In INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on (pp. 170-175).

Zitzler, E., 1999. Evolutionary algorithms for multiobjective optimization: Methods and applications. Federal Institute of Technology, Swiss.

Anexo A. Cómputo en la nube

El cómputo en la nube es un modelo de pago por uso que permite alquilar recursos de cómputo bajo demanda. El instituto nacional de estándares y tecnología de Estados Unidos (NIST) define el cómputo en la nube como "un modelo que permite el acceso a la red omnipresente, conveniente y bajo demanda a un grupo compartido de recursos de cómputo configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser rápidamente provisionados y ejecutados con un mínimo esfuerzo de gestión o interacción del proveedor de servicios" (Mell y Grance, 2011). Las principales características del modelo nube se describen a continuación.

Costo. En el cómputo en la nube, los recursos no pertenecen a los usuarios y los usuarios no deben comprar o mantener los recursos. La inversión inicial no es necesaria y los usuarios pagan acorde al tipo de servicio, tiempo de uso y almacenamiento. Los proveedores tienen toda la responsabilidad de la infraestructura. Por otro lado, el cómputo en la nube reduce el costo de mantenimiento de software y hardware porque los proveedores deben mejorar las condiciones de la nube que utilizan los usuarios.

Desempeño. De forma similar a los sistemas distribuidos, la nube computacional incrementa la velocidad de procesamiento y maximiza la capacidad de almacenamiento mediante la consolidación de CPU, memoria, almacenamiento y ancho de banda.

Escalabilidad. El usuario aumenta o disminuye los recursos (CPU, memoria, red, almacenamiento, etc.) en cualquier momento, esta es una ventaja importante para el usuario porque solo paga por lo que realmente usa. El proveedor nube hace uso de los recursos para proporcionar servicio a otros usuarios cuando los recursos están subutilizados. Los recursos no están limitados a una cantidad fija, el usuario agrega o elimina recursos cuando el tamaño del problema cambia.

Movilidad. El usuario accede a los datos en todo momento y desde cualquier lugar, solo necesita un dispositivo con conexión a Internet (computadora portátil, computadora de escritorio, tableta, teléfono inteligente, etc.). Los datos permanecen accesibles sin importar lo que le ocurra al dispositivo.

Virtualización. La tecnología de virtualización permite compartir recursos, un único recurso físico simula varios recursos separados individualmente. La virtualización emplea la capacidad del servidor de forma eficiente al reducir los ciclos de CPU ociosos y minimizar el desperdicio de energía. El sistema ejecuta varias VMs en un recurso físico para proporcionar servicio a varios usuarios. Las ventajas del paradigma de VMs son: pueden iniciar, finalizar y migrar fácilmente entre recursos y ejecutan diversas tareas. En general, la virtualización permite crear nubes elásticas, escalables y ecológicas.

El creciente número de proveedores de nube complica la tarea de seleccionar un servicio apropiado, que satisfaga las necesidades de los usuarios. Los usuarios deben entender las ventajas y limitaciones de los diferentes modelos nubes al elegir un proveedor, esto les permitirá alcanzar sus objetivos más fácilmente. La computación en la nube contempla dos modelos de clasificación: modelo de implementación y modelo de servicio, las siguientes secciones describen ambos modelos.

Modelo de implementación

El cómputo en la nube cuenta con cuatro modelos de implementación basados en el propietario y el acceso. Esta clasificación define el acceso según el propietario del centro de datos en la nube.

Nube pública (acceso público) o nube externa. Varias organizaciones comparten la infraestructura y proporcionan servicio al público en general. Los usuarios obtienen acceso a los recursos mediante suscripción y acceden a los servicios a través un navegador web u otras aplicaciones de software. Las nubes públicas no restringen el número de usuarios, la cantidad de usuarios potenciales no está limitada.

Nube privada (acceso privado) o nube interna. Una compañía es la propietaria de la infraestructura y la única con acceso a los recursos, no proporciona servicio al público general. El proveedor tiene control total sobre las aplicaciones, los recursos y las personas u organizaciones que pueden utilizan la infraestructura. Las nubes privadas ofrecen las mismas características y beneficios que las nubes públicas pero agregan una serie de limitaciones, las restricciones incluyen: control de datos, seguridad y cumplimiento normativo. Los expertos consideran este modelo de nube menos vital y más costosa.

Nube comunitaria (enfoque comunitario). Organizaciones con inquietudes comunes o políticas similares comparten la infraestructura. Los proveedores generan comunidades gestionadas por terceros y alojadas interna o externamente. El costo operativo de estas nubes puede ser mayor que las nubes públicas debido a la pequeña cantidad de usuarios. Sin embargo, este modelo satisface mejor las necesidades de los usuarios porque permite una mayor personalización.

La *nube híbrida (acceso híbrido) o "bursting cloud"* combina nubes privadas, públicas y comunitarias. Una infraestructura privada y/o pública proporciona el servicio a los usuarios. En este modelo, una empresa usa la nube pública para ejecutar aplicaciones pero las aplicaciones con importancia relativa se alojan en nubes privadas. Los servicios de la nube híbrida incrementan la funcionalidad pero agregan complejidad en la distribución de aplicaciones en diferentes entornos. El monitoreo de la infraestructura involucra seguridad y privacidad, por lo tanto, este modelo es menos adecuado para aplicaciones que requieren bases de datos complejas o sincronización (Corrado y Moulaison, 2011).

La Figura 35 presenta los cuatro modelos de implementación. Los modelos de implementación puede ser interno o externo, el modelo interno agrupa recursos bajo políticas de seguridad de organizaciones, pero el modelo externo no.

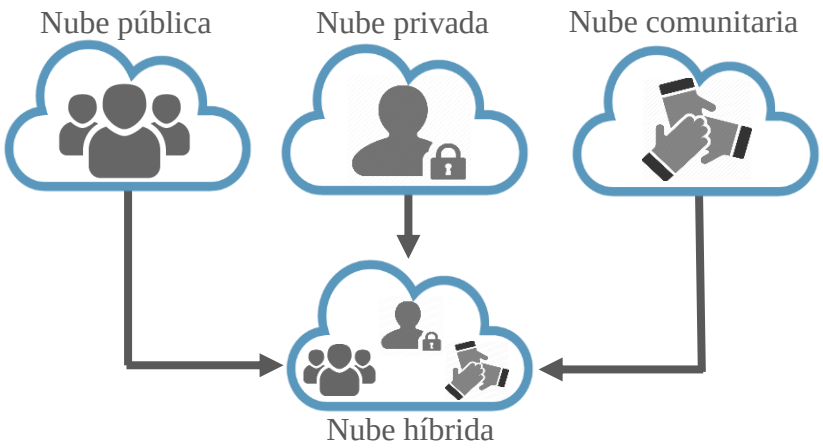

Figura 35. Modelos de implementación en la nube.

Modelo de servicio

El modelo de servicio describe la estructura genérica y los componentes de la nube mediante el concepto de capas, donde cada capa define un servicio. El cómputo en la nube considera tres capas fundamentales para proporcionar servicio: infraestructura como servicio, plataforma como servicio y software como servicio.

Infraestructura como Servicio (IaaS), también conocida como *Hardware como Servicio* (HaaS), facilita el alquiler de recursos de cómputo que pueden incluir procesamiento, almacenamiento, red y otros. Garner[3] define IaaS como "una oferta estandarizada y altamente automatizada, donde los recursos de cómputo, complementados por capacidades de almacenamiento y redes, son propiedad y están alojados por un proveedor de servicios y se ofrecen a los clientes a pedido".

IaaS es la capa funcional inferior y proporciona hardware para implementar y ejecutar software arbitrario. Amazon Web Services (EC2), uno de los principales proveedores, brinda servicio de procesamiento y almacenamiento a la medida. Los usuarios tienen control sobre el software, el almacenamiento y la capacidad de procesamiento, pero los recursos son administrados por el proveedor.

[3] https://www.gartner.com/it-glossary/infrastructure-as-a-service-iaas/

Plataforma como Servicio (PaaS) brinda una plataforma de desarrollo en la nube. El modelo PaaS funciona como una capa de abstracción entre la aplicación de software (SaaS) y la infraestructura (IaaS). Gartner[4] bautizó el año 2015 como "el año de la PaaS" y define el modelo PaaS como "una amplia colección de servicios de infraestructura de aplicaciones (middleware), incluye plataformas de aplicaciones, integración, gestión de procesos comerciales y servicios de bases de datos".

En el modelo PaaS, el usuario adquiere bibliotecas, servicios y herramientas compatibles con el proveedor. Los desarrolladores escriben aplicaciones para una plataforma en particular sin preocuparse por la infraestructura de hardware subyacente. El usuario implementa aplicaciones y posibles configuraciones para las aplicaciones. Por ejemplo, Google App Engine, un servicio de PaaS, permite la ejecución de aplicaciones arbitrarias en la infraestructura de Google.

Software como Servicio (SaaS) ofrece distribución de software por medio de la nube. El usuario renta aplicaciones de software que se ejecutan en una infraestructura bajo demanda. Garner[5] define SaaS como "software propietario entregado y administrado remotamente por uno o más proveedores. El proveedor ofrece software basado en un conjunto de definiciones de datos y códigos comunes que todos los clientes consumen en cualquier momento, en base a un pago por uso o como una suscripción basada en las métricas de uso".

Los clientes acceden al software desde varios dispositivos o a través de diferentes interfaces, como navegadores web o una interfaz de programa. El usuario solo define posibles configuraciones de la aplicación. Aplicaciones de correo electrónico, como Googles Gmail y Microsoft Hotmail, y aplicaciones de ofimática, como Google Docs y Microsoft Office 365, son algunos ejemplos de SaaS.

La Figura 36 muestra los modelos de servicio en nubes privadas, públicas, comunitarias e híbridas. Los recuadros azules delimitan la administración de la infraestructura subyacente por parte del proveedor de la nube en base al modelo de servicio.

El modelo de servicios descrito anteriormente es ampliamente aceptado, sin embargo, existen otras taxonomías con mayor granularidad. Una clasificación alterna define modelos como *Todo como Servicio* (EaaS) o *Cualquier Cosa como Servicio* (XaaS), ejemplos de estos modelos comprenden: comunicaciones, escritorios virtuales, recuperación de desastres, mercadotecnia, salud, etc.

[4] https://www.gartner.com/it-glossary/platform-as-a-service-paas
[5] https://www.gartner.com/it-glossary/software-as-a-service-saas

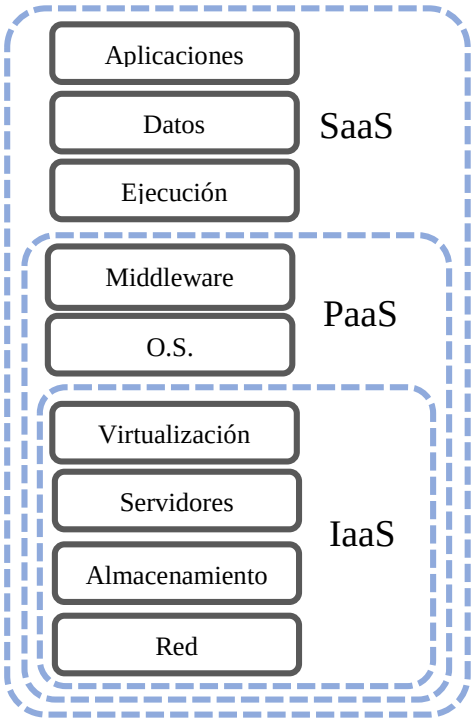

Figura 36. Administración de la infraestructura nube por parte del proveedor.

Contenedores como Servicio (CaaS) es un ejemplo de modelo de servicio recientemente incorporado por los proveedores nube. CaaS mitiga los problemas derivados de la dicotomía entre IaaS y PaaS. Los contenedores, los componentes básicos en CaaS, proporcionan control detallado sobre los recursos compartidos. Entornos virtuales basados en CaaS ofrecen aislamiento sin necesidad de medios de supervisión como los hipervisores (Piraghaj et al., 2017). Google Container Engine (GCE), Amazon EC2 Container Service (ECS) y Azure Container Service (ACS) son ejemplos de CaaS en entornos nube.

Los beneficios del cómputo en la nube están directamente relacionados con la virtualización, esta tecnología crea entornos a partir de recursos de hardware no utilizados. La siguiente sección describe las ventajas de la virtualización en ambientes nube.

Virtualización

La virtualización se introdujo en la década de 1960 (Goldberg, 1974). Esta tecnología es un factor crucial en el cómputo en la nube porque permite crear sistemas elásticos, escalables y ecológicos. Garner[6] define la virtualización como “la abstracción de los recursos informáticos que enmascara la naturaleza física y los límites de esos recursos de los usuarios”. En general, la virtualización crea recursos tecnológicos

[6] https://www.gartner.com/it-glossary/virtualization

virtuales a través de software. Hoy en día, la virtualización existe a diferentes niveles: hardware, VM, almacenamiento, escritorio, red, entre otros.

Las ventajas de la virtualización son: uso eficiente de la capacidad del servidor, reduce ciclos ociosos del CPU y minimiza el desperdicio de energía. La virtualización permite compartir recursos de forma transparente porque el usuario percibe muchos recursos individuales de un único recurso físico (por ejemplo, un servidor). El cómputo en la nube utiliza ampliamente la virtualización a nivel de sistema o VM, los recursos virtualizados incluyen comúnmente una combinación de CPU, memoria, dispositivos de almacenamiento, recursos de red, etc. De acuerdo con Garner[7], "Una VM es una implementación de software de una arquitectura tipo hardware, que ejecuta instrucciones predefinidas de forma similar a una unidad de procesamiento central (CPU) física".

La VM tiene un SO que se ejecuta en la parte superior del hipervisor y proporciona la funcionalidad de una computadora física. Un servidor puede ejecutar varias VMs al mismo tiempo, por ende las VMs dividen el recurso en uno o más entornos de ejecución. El hipervisor gestiona la creación y ejecución de las VMs, libera las VMs de los recursos físicos y asigna dinámicamente los recursos virtuales. Entre los ejemplos de hipervisores[8] se encuentran VMware ESXi, máquina virtual basada en kernel (KVM), Citrix Xen y Microsoft Hyper-V. La Figura 37 muestra la virtualización a nivel de SO donde las VMs son los principales elementos del modelo.

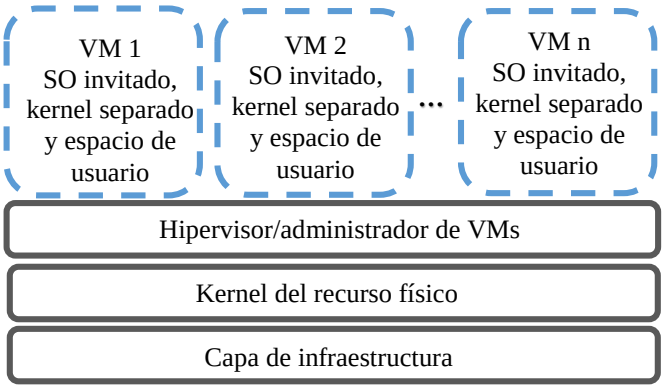

Figura 37. Virtualización a nivel de sistema.

La adopción del cómputo en la nube no está exenta de problemas, las empresas deben considera ciertos desafíos cuando trasladan su infraestructura a la nube. Algunos de los desafíos más importantes se describen a continuación.

[7] https://www.gartner.com/it-glossary/vm-virtual-machine
[8] http://searchservervirtualization.techtarget.com/tip/Top-10-hypervisors-Choosing-the-best-hypervisor-technology

Desafíos del cómputo en la nube

La gestión de recursos en el cómputo en la nube ha estado sujeta a investigación y desarrollo durante muchos años. Actualmente, existen importantes resultados de investigación para diferentes variantes de este problema. Sin embargo, a medida que la nube se ha convertido en un entorno más común se ve una tendencia a problemas de optimización más complejos. Varios aspectos relacionados con los desafíos en sistemas nube comprenden:

La *calidad de servicio* (QoS) es un tema importante y de interés en la nube computacional. Los proveedores deben suministrar suficiente cantidad de recursos y cumplir con los requisitos de QoS como: fechas límite, tiempos de respuesta y restricciones de presupuesto (Buyya et al., 2012). Los acuerdos de nivel de servicio (SLAs) son contratos vinculantes entre un proveedor de servicios y el usuario. Los SLAs contienen la lista de servicios, métricas, responsabilidades del proveedor y mecanismos de auditoría, donde cualquier violación dará lugar a una sanción. Los proveedores deben proporcionar la cantidad adecuada de recursos oportunamente. La infraestructura en la nube a menudo involucra elementos fuera del control del proveedor de servicio, por ejemplo, el Internet. Además, la medición y el control de las garantías de QoS es un desafío. Existen varias formas de proporcionar niveles adecuados de QoS: calendarización, balanceo de carga, asignación de recursos, control de admisión, control de tráfico, suministro dinámico de recursos, etc.

El *consumo de energía* está determinado por la eficiencia del hardware, el sistema administrador de recursos y la ejecución de las aplicaciones. La eficiencia energética tiene impacto sobre el costo del servicio para los usuarios, que generalmente está determinado por el costo total del proveedor (Beloglazov et al., 2011). El objetivo principal es evitar la utilización de más recursos de los requeridos por las aplicaciones. La reducción del consumo de energía se realiza a diferentes niveles: compilador, SO, aplicación y virtualización, que impactan en las optimización de recursos térmicos, de red y de otros sistemas Una solución a este problema es la migración de VMs entre nodos con el objetivo de apagar nodos inactivos, dicha política de desactivación dinámica de componentes apaga partes del sistema de cómputo que no se utilizan (Tchernykh et al., 2014), también conocida como modo reposo, modo encendido/apagado, etc.

La S*eguridad* es la principal preocupación que obstaculiza la adopción del cómputo en la nube en la industria y el uso personal. Las empresas delegan la administración de seguridad a un tercero que aloja sus activos de cómputo (pérdida de control). Adicionalmente, la disponibilidad pública de la infraestructura aumenta las probabilidades de ataques. Los ambientes nube complican la seguridad

porque agrega nuevas dimensiones al problema relacionado con la arquitectura, múltiple arrendamiento, elasticidad y la dependencia de capas. Según Almorsy et al. (2016) "basándonos en la gran cantidad de interesados en la nube, la profunda dependencia y la gran cantidad de controles de seguridad para cumplir con los requisitos de seguridad, la gestión de seguridad en la nube se convierte en un problema de investigación más complicado".

El *balanceo de carga* es un mecanismo empleado para mejorar el rendimiento general del sistema mediante la distribución equitativa de la carga de trabajo entre los nodos. El balanceador mejora la disponibilidad de recursos y lograr una mayor satisfacción del usuario al reducir los tiempos ociosos de los recursos. Un balanceo de carga adecuado usa los recursos disponibles eficientemente, lo que minimiza el consumo de recursos (González-García et al., 2013). El balanceo de carga también incrementa la escalabilidad, evita los cuellos de botella e impide el abastecimiento excesivo, estas ventajas se traducen en una reducción del tiempo de respuesta y del consumo de energía. A diferencia de otros sistemas distribuidos, los principales objetos de balanceo en el cómputo en la nube son las VMs y el almacenamiento virtual.

Anexo B. Degradación promedio y rango

Las Tablas 36 - 39 presentan la degradación promedio de $\bar{b}$, $\bar{c}$ y su promedio. Las últimas cuatro columnas de la tabla contienen el rango de cada estrategia en relación con el costo del proveedor, la calidad de servicio y los promedios. Rango $\bar{b}$ considera la degradación de $\bar{b}$ para clasificar las estrategias. Rango $\bar{c}$ define la posición de la estrategia en relación con la degradación de $\bar{c}$. Rango promedio establece la posición de la estrategia basada en el promedio de la degradación de $\bar{b}$ y $\bar{c}$. Rango es la posición relativa con respecto a todas las estrategias, considera el rango de $\bar{b}$ y el rango $\bar{c}$.

Tabla 36. Degradación promedio y rango para estrategias de asignación con umbral de renta.

Estrategia	Deg $\bar{b}$	Deg $\bar{c}$	Promedio	Rango $\bar{b}$	Rango $\bar{c}$	Rango promedio	Rango
BFit	0.0578	0.2316	0.1447	27	30	7	7
BFit_05	0.1081	0.2342	0.1712	43	31	16	11
BFit_10	0.1238	0.3026	0.2132	50	36	23	18
BFit_15	0.1091	0.5263	0.3177	44	52	59	27
FFit	0.0493	0.2632	0.1562	16	34	13	4
FFit_05	0.0901	0.2763	0.1832	40	35	18	12
FFit_10	0.1195	0.4105	0.2650	49	43	38	23
FFit_15	0.1134	0.5500	0.3317	45	55	64	31
MaxFTFit	0.0643	1.0184	0.5414	30	78	99	38
MaxFTFit_05	0.0702	1.4816	0.7759	31	87	113	48
MaxFTFit_10	0.0562	1.5553	0.8057	26	88	114	44
MaxFTFit_15	0.0530	1.7184	0.8857	21	90	116	41
MidFTFit	0.1440	0.6158	0.3799	59	59	78	48
MidFTFit_05	0.1500	0.4763	0.3132	60	45	57	35
MidFTFit_10	0.1310	0.6000	0.3655	53	57	75	40
MidFTFit_15	0.1161	0.8658	0.4910	46	68	91	44
MinFTFit	0.5346	0.0132	0.2739	122	**2**	42	54
MinFTFit_05	0.3771	0.0711	0.2241	80	14	31	25
MinFTFit_10	0.2720	0.1763	0.2242	70	26	32	27
MinFTFit_15	0.1567	0.5395	0.3481	61	54	69	45
Rand	0.4029	0.0368	0.2199	88	8	28	27
Rand_05	0.2795	0.0605	0.1700	71	12	15	16
Rand_10	0.2054	0.0579	0.1317	68	11	5	14
Rand_15	0.1731	0.0579	0.1155	64	11	**1**	12
RR	0.4222	0.0184	0.2203	97	4	29	32
RR_05	0.2812	0.0579	0.1695	72	11	14	16
RR_10	0.2075	0.0868	0.1472	69	15	10	17
RR_15	0.1747	0.0868	0.1308	65	15	4	15
WFit	0.4611	0.0211	0.2411	108	5	35	43
WFit_05	0.2951	0.0711	0.1831	73	14	17	19
WFit_10	0.1894	0.1842	0.1868	67	27	19	25
WFit_15	0.1390	0.5053	0.3221	57	48	60	35

Tabla 37. Degradación promedio y rango para estrategias con predicción basada en tasa de cambio.

Estrategia	Deg $\bar{b}$	Deg $\bar{c}$	Promedio	Rango $\bar{b}$	Rango $\bar{c}$	Rango promedio	Rango
BFit_s10	0.0550	0.2211	0.1380	24	29	6	6
BFit_s20	0.0513	0.2474	0.1493	19	32	11	5
BFit_s30	0.0517	0.2026	0.1272	20	28	**3**	**3**
BFit_stUp	0.0411	0.6447	0.3429	8	60	67	9
FFit_s10	0.0425	0.2500	0.1462	9	33	9	**1**
FFit_s20	0.0427	0.2474	0.1451	10	32	8	**1**
FFit_s30	0.0464	0.2026	0.1245	15	28	**2**	**2**
FFit_stUp	0.0322	0.6158	0.3240	4	59	62	8
MaxFTFit_s10	0.0578	0.8789	0.4684	27	70	90	28
MaxFTFit_s20	0.0538	0.9868	0.5203	23	75	94	29
MaxFTFit_s30	0.0620	0.9789	0.5205	29	74	95	33
MaxFTFit_stUp	0.0443	1.3763	0.7103	13	85	111	29
MidFTFit_s10	0.1373	0.2500	0.1936	56	33	20	21
MidFTFit_s20	0.1362	0.5132	0.3247	55	51	63	36
MidFTFit_s30	0.1341	0.1737	0.1539	54	25	12	14
MidFTFit_stUp	0.1174	0.6632	0.3903	47	61	80	38
MinFTFit_s10	0.5253	0.0132	0.2692	121	**2**	41	53
MinFTFit_s20	0.5032	0.0289	0.2661	117	7	40	54
MinFTFit_s30	0.5118	0.0000	0.2559	118	**1**	36	49
MinFTFit_stUp	0.4655	0.3553	0.4104	111	41	82	67
Rand_s10	0.4053	0.0237	0.2145	89	6	25	26
Rand_s20	0.3898	0.0368	0.2133	84	8	24	23
Rand_s30	0.3918	0.0000	0.1959	85	**1**	21	18
Rand_stUp	0.3598	0.3763	0.3681	77	42	76	49
RR_s10	0.4192	0.0158	0.2175	95	**3**	27	29
RR_s20	0.4057	0.0289	0.2173	92	7	26	30
RR_s30	0.4056	0.0000	0.2028	91	**1**	22	23
RR_stUp	0.3727	0.3763	0.3745	78	42	77	50
WFit_s10	0.4543	0.0132	0.2338	106	**2**	33	38
WFit_s20	0.4438	0.0289	0.2364	102	7	34	39
WFit_s30	0.4455	0.0000	0.2227	105	**1**	30	36
WFit_stUp	0.4128	0.3763	0.3946	94	42	81	62

Tabla 38. Degradación promedio y rango para estrategias con predicción basada en redes neuronales.

Estrategia	Deg $\bar{b}$	Deg $\bar{c}$	Promedio	Rango $\bar{b}$	Rango $\bar{c}$	Rango promedio	Rango
BFit_s10	0.0534	1.0079	0.5307	22	77	97	30
BFit_s20	0.0455	1.0868	0.5662	14	79	101	24
BFit_s30	0.0407	1.1579	0.5993	7	81	103	20
BFit_stUp	0.0084	3.9974	2.0029	**2**	99	126	32
FFit_s10	0.0434	0.5553	0.2993	12	56	52	9
FFit_s20	0.0377	0.6842	0.3609	6	62	73	9
FFit_s30	0.0322	0.8737	0.4530	4	69	87	10
FFit_stUp	0.0000	4.1342	2.0671	**1**	100	127	32
MaxFTFit_s10	0.0611	1.5579	0.8095	28	89	115	47
MaxFTFit_s20	0.0559	1.7474	0.9016	25	91	117	46
MaxFTFit_s30	0.0494	1.9237	0.9866	17	93	119	40
MaxFTFit_stUp	0.0157	4.4158	2.2158	**3**	101	128	34
MidFTFit_s10	0.1401	0.7421	0.4411	58	63	85	51
MidFTFit_s20	0.1292	0.9211	0.5251	51	73	96	54
MidFTFit_s30	0.1180	0.9921	0.5551	48	76	100	54
MidFTFit_stUp	0.0842	3.3842	1.7342	38	98	125	62
MinFTFit_s10	0.5361	0.1368	0.3365	123	20	65	64
MinFTFit_s20	0.5149	0.3184	0.4167	119	38	83	68
MinFTFit_s30	0.4825	0.5053	0.4939	113	48	93	69
MinFTFit_stUp	0.4027	2.7211	1.5619	87	96	124	73
Rand_s10	0.4066	0.1605	0.2836	93	24	45	47
Rand_s20	0.3888	0.3421	0.3655	83	40	74	53
Rand_s30	0.3740	0.5395	0.4567	79	54	88	61
Rand_stUp	0.3057	2.7684	1.5371	74	97	123	71
RR_s10	0.4247	0.1342	0.2794	98	19	44	47
RR_s20	0.4054	0.3158	0.3606	90	37	72	57
RR_s30	0.3866	0.4974	0.4420	82	47	86	58
RR_stUp	0.3183	2.7211	1.5197	75	96	121	71
WFit_s10	0.4643	0.1395	0.3019	110	21	53	60
WFit_s20	0.4429	0.3237	0.3833	101	39	79	63
WFit_s30	0.4205	0.5079	0.4642	96	49	89	65
WFit_stUp	0.3480	2.7184	1.5332	76	95	122	71

Tabla 39. Degradación promedio y rango para estrategias con predicción basada en información histórica de la carga.

Estrategia	Deg $\bar{b}$	Deg $\bar{c}$	Promedio	Rango $\bar{b}$	Rango $\bar{c}$	Rango promedio	Rango
BFit_s10	0.0905	0.6132	0.3519	41	58	70	30
BFit_s20	0.0800	0.9026	0.4913	35	72	92	37
BFit_s30	0.0737	0.8079	0.4408	33	64	84	28
BFit_stUp	0.0429	1.8079	0.9254	11	92	118	33
FFit_s10	0.0826	0.5105	0.2966	37	50	51	19
FFit_s20	0.0746	0.5395	0.3070	34	54	54	20
FFit_s30	0.0712	0.4842	0.2777	32	46	43	13
FFit_stUp	0.0354	1.4711	0.7532	5	86	112	22
MaxFTFit_s10	0.0965	1.2447	0.6706	42	83	108	55
MaxFTFit_s20	0.0888	1.1947	0.6418	39	82	107	51
MaxFTFit_s30	0.0804	1.1079	0.5942	36	80	102	46
MaxFTFit_stUp	0.0507	1.9342	0.9924	18	94	120	42
MidFTFit_s10	0.1840	0.4632	0.3236	66	44	61	40
MidFTFit_s20	0.1713	0.5368	0.3541	63	53	71	46
MidFTFit_s30	0.1622	0.9000	0.5311	62	71	98	61
MidFTFit_stUp	0.1293	1.2474	0.6883	52	84	110	62
MinFTFit_s10	0.5787	0.0526	0.3156	126	10	58	62
MinFTFit_s20	0.5654	0.1289	0.3472	125	18	68	64
MinFTFit_s30	0.5455	0.1342	0.3398	124	19	66	64
MinFTFit_stUp	0.4914	0.8553	0.6733	115	66	109	72
Rand_s10	0.4618	0.0684	0.2651	109	13	39	52
Rand_s20	0.4454	0.1447	0.2951	104	22	50	56
Rand_s30	0.4299	0.1500	0.2900	99	23	49	52
Rand_stUp	0.3826	0.8605	0.6216	81	67	104	66
RR_s10	0.4753	0.0421	0.2587	112	9	37	51
RR_s20	0.4608	0.1132	0.2870	107	16	47	53
RR_s30	0.4452	0.1289	0.2871	103	18	48	51
RR_stUp	0.3924	0.8553	0.6238	86	66	105	67
WFit_s10	0.5156	0.0526	0.2841	120	10	46	59
WFit_s20	0.4934	0.1263	0.3098	116	17	56	61
WFit_s30	0.4848	0.1342	0.3095	114	19	55	61
WFit_stUp	0.4301	0.8500	0.6401	100	65	106	70

Anexo C. Resumen extenso en inglés

Cost-Quality Optimization of Cloud-based VoIP

1. Introduction

Voice over IP (VoIP) is a fast growing technology in cloud computing considered as a long-term service roadmap, offering more features than traditional telephony (PSTN) infrastructure. The main benefits of this technology, over traditional telephones and VoIP, are cost effectiveness, higher flexibility, scalability, extended service variety, etc. It can cope with different workloads, objective preferences, and cloud properties.

Asterisk (Madsen et al., 2011) is the backbone in Cloud VoIP (CVoIP) solution. This software powers IP Private Branch Exchange (PBX) systems. Virtual Machines (VMs) execute Asterisk instances, and provide calls, voice mails, video/audio conferences, interactive phone menus, call distribution, etc. Additionally, users can transfer images and texts, and they can create new functionalities, opening up a complete new experience in telephonic communication.

The success of the business depends significantly on the price and Quality of Service (QoS) factors. CVoIP providers look to offer an adequate service considering various parameters: the quality of voice, transit time of packets across the Internet, queuing delays at the routers, end-to-end delay, jitter, call set-up and tear-down time, codec compression technique, processing capability, etc. (Singh et al., 2014).

In this research work, we formulated the problem of scheduling of VoIP services in cloud environments, and proposed two models for a mono-objective and a bi-objective optimization problem. We considered the particular case of the dynamic bin packing problem and discussed solutions for provider cost, quality of service, and robustness optimization.

We proposed and evaluated several call allocation strategies that use information about VMs utilization, VM provisioning time, and VM startup time delays. These factors impact on resource under- or over-provisioning, call quality and cost. We analyzed strategies focusing on estimation of the amount of VMs needed to provide VoIP services without quality degradation and on techniques to reduce the number of active VMs.

1.1 Internet telephone

The Internet telephony VoIP refers to making calls according to the IP standard. It achieves significant call rate reduction decreasing the infrastructure and communication costs. VoIP providers have to deal with several problems of traditional VoIP systems. For instance, availability of the service for any number of users is fundamental. With the increasing number of clients, providers need to invest in a large infrastructure to avoid loss of calls (hence, users). This situation leads to two significant problems: over-provisioning and over-running cost. Even with a high number of resources, the system cannot be able to deliver services during peak hours or abnormal system behavior.

A Cloud-based VoIP (CVoIP) reduces the costs and increases availability without fall into over-provisioning. The deployment of the virtual infrastructure is easy to implement, faster to provision, and integrate services that are dynamically scalable. Further, it adds new features and capabilities for users (data transfer availability, integrity, and security).

Voice Nodes (VNs) are the core part of the CVoIP telephony system (Figure 1). They execute specialized software to emulate a telephone exchange, gateways, interconnection switches, session controllers, firewall, etc. VNs verify the credentials of the user and try to reach the other end-device. After the call is finished, call details are stored in the Call-Detail-Record (e.g., client id, destination, duration call).

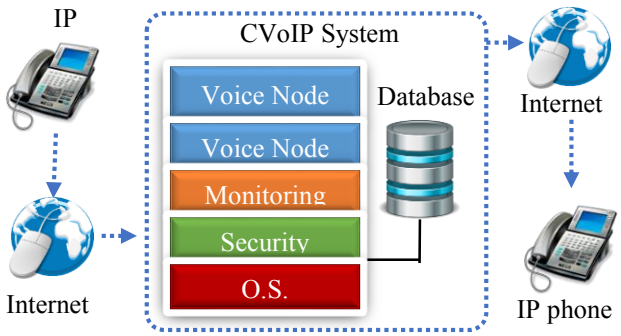

Figure 1. Cloud VoIP architecture.

1.2 Infrastructure

MIXvoip company (MIXvoip, 2018) that hosts and delivers VoIP services developed the concept of Super-Node (SN) and Super Nodes Cluster (SNC) to enrich features for telephone exchanges. SN model combines cloud service with smart business telephony, VoIP, and other telephony services. It brings redundancy in communication in a given geographical area to ensure a high voice quality between SNs through the public Internet and provides short paths between two local users.

The most known telephone software for processing calls and providing a robust control over call activity is Asterisk (Madsen et al., 2011). It is a framework for building multi-protocol, real-time communication solutions providing a powerful control over call activity. It processes calls and connects to other telephone services, such as the Public Switched Telephone Network (PSTN) and VoIP services.

The CVoIP system consists of multiple SNs that run and handle calls. Each SN runs an Asterisk process to connect users inside and outside of the network.

2. Related work

The migration of VoIP businesses to clouds computing triggers intensive research on several fields: call allocation, load balancing, quality of service, load prediction, load estimation, etc. The next section highlights recent works on load balancing, bin-packing, and load estimation related to VoIP management.

2.1 Call load balancing

The main objective of call load balancing is to reduce the infrastructure cost and guarantee that service will be delivered in the best possible way. Several algorithms have been proposed to improve the performance of CVoIP system.

Mazalek et al. (2015) study the impact of the IPSec encryption on the CPU utilization, bandwidth and voice quality. It has a significant effect on the voice payload period on the CPU utilization and bandwidth. Results show that strategies can save up to 40-60 % of bandwidth when the period is chosen properly. The call quality, expressed by MOS scale, remains almost constant up to the moment when the CPU utilization is close to 80%.

Rodrigues et al. (2015) show that Asterisk PBX server is able to provide VoIP communication capabilities with an acceptable MOS quality. The authors use the blocking probability metric to measure the capacity of the VoIP server and MOS to assess the quality of the voice calls. The experimental results show that the Asterisk PBX using SIP effectively handled more than 160 concurrent voice calls with a blocking probability below 5%.

Montazerolghaem et al. (2016) present and analyze the Virtual Load-Balanced Call Admission Controller (VLBCAC) strategy to provide admission control for SIP servers and balance the arriving calls. VLBCAC has mechanisms to predict the call number, required resources, and select the most appropriate VM instances considering CPU, memory, and bandwidth. The proposed model considers maximizing the resource usage and system throughput.

2.2 Bin packing

Bin packing heuristics are used widely in cloud management. Their primary function is to allocate applications into the VMs and/or VMs on the resources.

Song et al., (2014) propose a relaxed on-line bin packing algorithm called Variable Item Size Bin Packing (VIS-BP). It allocates data center resources using live VM migration. The main goal is to restrict the combinations of tasks in a bin to minimize the amount of wasted space. The authors evaluate the effectiveness of the algorithm, and provide a theoretical proof for the number of used servers and VM migrations. The multi-dimensional version considers a mix of CPU and network.

Wolke et al., (2015) analyze a wide variety of controllers for VMs placement and reallocation using the resources availability (CPU, memory, etc.). Incoming VMs are placed on servers as long as their residual capacity allows. Bin packing algorithms are used in this stage. The reallocation controller triggers VM migration, when under loaded servers are emptied and overloaded ones are relieved. The authors found that combinations of placement controllers and periodic reallocations achieve the highest energy efficiency subject to predefined service levels.

Li et al. (2016) consider a variant of dynamic bin packing problem to solve the request dispatching problem arising from cloud gaming. In cloud gaming system, computer games run on cloud servers, where each game instance demands certain amount of resources. In order to provide a good user experiences, requests must be dispatched with enough resources of CPU and GPU. The main objective is to minimize the number of servers (bins) to process the gaming requests (items) due to each resource adds a proportional cost to the duration of its usage. The authors analyze the competition ratio for original and modified versions of Best Fit and First Fit algorithms.

2.3 Load estimation

Several studies are being conducted with the motivation of load prediction for an efficient distribution of the load in the system. The main idea is to anticipate the incoming traffic (calls for VoIP) to minimize the infrastructure costs and improve the QoS to the end user.

Simionovici et al. (2013) study and compare two prediction models for real VoIP environment. Interactive Particle System (IPS), and Neural Network (NN) are used to predict the incoming voice traffic during time frames. In (Simionovici et al., 2015), the authors extend previous work, and use IPS, Gaussian Mixture Model (GMM), and Gaussian Process (GP) to estimate the incoming voice traffic. The authors

provide flexible modeling approaches, traffic shaping determined by clients, and scalable solutions with good prediction precision. All algorithms were trained and tested under different scenarios.

Lu et al. (2016) investigate the dynamic characteristics of cloud workloads, and analyze three algorithms as workload predictors to improve the energy efficient on cloud systems. Authors evaluate the performance of Rand Variable Learning Rate Backpropagation Neural Network (RVLBPNN), Hidden Markov Modelling (HMM), and Naive Bayes Classifier (NBC) for predicting of future workloads in cloud data centers. All models are evaluated for their efficiencies in predicting memory and CPU intensive workloads. The main goal is to maintain high level of QoS and to reduce the usage of the data center resources.

Rate of Change – RoC (Campos and Scherson, 2000) is a strategy for load balancing tasks in distributed system. It allows to trigger a dynamic, distributed, and implicit load balancing mechanism. The balancer (Bal) makes job distribution decisions at run-time, locally and asynchronously. Each Bal considers its own load; migration does not depend on the load of other Bals. The migration decision depends on current load, load changes in the time interval (rate of load change), and current load balancing parameters. Difference in load is used as an estimation of load, its prediction for the next time slot, and detection of the need for load balancing process.

3. Model

Call processing and call delivery are two key issues, which determine the quality of calls. Call processing focuses on the time to set-up and tear-down the call, and on converting the voice portion of the calls into packets transported over the network. Adequate quality of voice is the most important aspect in call processing.

3.1 Quality of service

The quality of voice is subjectively perceived by the listener. A common benchmark used to determine the quality of voice is the Mean Opinion Score (MOS). It evaluates the quality of speech provided by a codec. Each codec provides a certain quality of speech only if processor utilization is low enough. Theoretically, processor utilization of 100% provides the best expected performance. However, Eleftheriou (2017) showed that CPU cannot handle the stress when utilization is up to 85%, then jitters and broken audio symptoms appear. Additionally, the author did not report any influence of memory on the voice quality reduction.

During a call processing, the codec increases the used bandwidth but it is less significant than the same codec adds to CPU utilization. Montazerolghaem et al. (2015) reported that the consumed bandwidth of 6,500 calls per second not exceed 100 Mbps, and 10,000 calls not reach 400 Mbps. Table 1 shows the bandwidth used by different codecs considering that VoIP calls use audio streams for endpoints (a call between two parties will use double of bandwidth). The number of calls supported with 100 Mbps connection is between 6,000 and 25,000 depending on codec of the calls.

Table 1. Codec bandwidth.

Codec	Bit Rate (kbps)	MOS	Bandwidth (kbps)
G.711	64	4.1	87.2
G.729	8	3.92	31.2
G.723.1	6.3	3.9	21.9
G.723.1	5.3	3.8	20.8
G.726	32	3.85	55.2
G.726	24	-	47.2
G.728	16	3.61	31.5
G722_64k	64	4.13	87.2
ilbc_mode_20	15.2	NA	38.4
ilbc_mode_30	13.33	NA	28.8

However, the amount of calls handled by CPU is less than supported by 100 Mbps connection. Hence, the call processing is the key feature to guarantee the QoS. In this work, we proposed to limit processor utilization in order to ensure QoS.

Calls have different impact on the processor utilization depending on the operations performed by Asterisk. If transcoding operations are performed, the utilization is higher than when transcoding is not used. In the latter case, Asterisk is in charge of only routing the call. However, depending on the codec, the processor load is influenced as well. Table 2 shows processor utilization for call without transcoding.

Table 2. Utilization of calls without transcoding (Montoro et al., 2009).

Protocol	Codec	10 Calls (%)	1 Call (%)
SIP/RTP	G.711	2.36	0.236
SIP/RTP	G.726	2.13	0.213
SIP/RTP	GSM	2.58	0.258
SIP/RTP	LPC10	1.92	0.192

The performance of ATOM processors is analyzed for VoIP considering calls amount, utilization, power consumption, database messaging, registration and call performance (Eleftheriou, 2017). The author concludes that CPU can process from 70 to 500 calls with 100% of utilization.

3.2 Startup time delay

VM startup time delay (StUp) is the time period that cloud providers need to find a spot in the data centers to VMs provisioning, allocating resources (e.g., IP addresses) to VMs and copy/boot/configure

the OS image (Mao and Humphreyet, 2012). StUp depends on several factors, but the most importants are the O.S. and software running in the VM (Razavi et al., 2013). Table 3 shows the StUp for several O.S. and providers.

Hoffman (2015) measures the speed of VM provisioning and SSH accessing time among cloud providers located in different regions in the world (Asia, Europe, and USA). Table 4 presents the average times of VM creation and the time of accessing the SSH key among cloud vendors.

Table 3. Average VM startup time.

Cloud	O.S.	StUp (sec.)
EC2	Linux	96.9
	Windows	810.2
Azure	WebRole	374.8
	WorkedRole	406.2
	VMRole	356.6
Rackspace	Linux	44.2
	Windows	429.2

Table 4. Average VM startup time with SSH access.

Cloud	StUp (sec.)
Google Cloud Platform	31
Amazon Web Services	47
Vexxhost	47
Linode	57
DigitalOcean	89
Rackspace	128
Azure	138

3.3 Formal definition

A cloud VoIP infrastructure consists of m heterogeneous Super Node Clusters $SNC_1, SNC_2, \dots, SNC_m$ with relative speeds $s_1, s_2, \dots, s_m$. Each SNC_i, for all $i = 1, \dots, m$, consists of m_i SNs. Each SN_k^i, for all $k = 1, \dots, m_i$, runs $k_i(t)$ VMs at time t (sec.). We assume that VMs of one SN are identical and have the same processing capacity. The virtual machine VM_k is described by a tuple $\{st_k, size_k, stUp_k\}$ that consists of its start of rental period $st_k \geq 0$, startup delay $stUp_k$, the processing capacity $size_k$ in MIPS.

We consider n independent calls $J_1, J_2, \dots, J_n$ that must be scheduled on a set of SNCs. The call J_j is described by a tuple $\{r_j, p_j, u_j\}$ that consists of its release time $r_j \geq 0$, duration p_j (lifespan), and contribution to the processor utilization u_j due to the used codec. The release time of a call is not available before the call is submitted, and its duration is unknown until the call has been completed. The utilization is a constant for a given call that depends on the used codec and VM processing capacity.

We define the provider cost model by considering a function that depends on the number of provisioned VMs and their rental time. We denote the number of billing hours in SNC_i by:

$$\bar{b}_i = \sum_{r=0}^{C_{max}/rp} k_i(r * rp) \quad (1)$$

where rp defines the minimum rental period of a VM, typically $rp = 60$ min. Total billing hours in all $SNCs$ is defined by:

$$\bar{b} = \sum_{i=1}^{m} \bar{b}_i \quad (2)$$

In addition to $st_j, stUp_j, size_j$, the VM is characterized by $vmu_j(t)$ the utilization (load) of the VM_j at time t. We introduce a quality reduction as a function of the VMs utilization by:

$$\bar{q}_i = \sum_{j=1}^{\bar{b}_i} \sum_{t=st_j}^{ft_j} \gamma(vmu_j(t)) \quad (3)$$

where st_j and ft_j define the rental time of the VM_j and the penalization function $\gamma(\alpha) = ((\alpha - 0.7) * \overline{3.33})^2$ when α>0.7 and 0 otherwise. Total Quality Reduction is defined by:

$$\bar{q} = \sum_{i=1}^{m} \bar{q}_i \quad (4)$$

Moreover, we analyze the number of calls on hold. It occurs when the VMs cannot process the arriving calls, so the calls wait in the queue for an available VM. The amount of calls placed on hold is defined by (5), where s_j is the initiation time of J_j, and $\delta(\beta)$ is 1 if $\beta > 0$ and 0 otherwise.

$$\bar{c} = \sum_{j=1}^{n} \delta(s_j - r_j) \quad (5)$$

We consider this problem as a special case of dynamic bin packing (on-line and non-clairvoyant). Bins represent VMs, and the items height define the call contribution to the VM utilization.

To compare strategies, we perform an analysis based on the degradation methodology proposed in (Tsafrir et al., 2007), and applied for scheduling in (Tchernykh et al., 2016). It shows how the metric generated by our algorithms gets closer to the best-found solution as:

$$\left(\frac{strategy\ metric\ value}{best\ found\ metric\ value} - 1\right) \cdot 100 \quad (6)$$

In multi-objective optimization, one solution can represent the best solution concerning provider cost, while another solution could be the best one concerning the QoS. The goal is to choose the most adequate solution and obtain a set of compromise solutions that represents a good approximation to the Pareto front.

A solution is Pareto optimal if no other solution improves it in terms of all objective functions. Any solution not belonging to the front can be considered of inferior quality to those that are included. If one objective is considered more important than the other one, then preference is given to those solutions that are near-optimal in the preferred objective, even if values of the secondary objective are not among

the best obtained. One of formal and statistical approaches uses a set coverage metric *SC(A,B)* that calculates the proportion of solutions in B, which are dominated by solutions in *A*:

$$SC(A,B) = \frac{|\{b \in B; | \exists a \in A\ ;\ a \leq b\}}{|B|} \quad (7)$$

A metric value SC(A,B) = 1 means that all solutions of B are dominated by A, whereas SC(A,B) = 0 means that no member of *B* is dominated by *A*. This way, the larger the value of SC(A,B), the better the Pareto front A with respect to B. Both SC(A,B) and SC(B,A) have to be computed since the dominance operator is not symmetric.

3.4 Call allocation

The call allocation problem is similar to a well-known dynamic bin-packing problem, a variation of the classical NP-hard optimization problem with high theoretical relevance and practical importance. The classic bin packing concerns placing items of arbitrary height into a minimum number of bins with fixed capacity (of one-dimensional space) efficiently. Bin-packing is a very active area of research in the algorithms and operations research communities.

In VoIP, the scheduler decides whether the call is placed into one of the currently available VMs or new VM should be run. The scheduler only knows the contribution of the call to the VM utilization u_j. All decisions have to be made without knowledge of duration of the call, call arrival rate, etc.

Temporal existence of the items is the principal novelty of this problem. Call lifespan and allocations determine the state of the VMs. Unlike the standard formulation, bins are always open and dynamic, even completely packed. Items in bins can be terminated (call termination) and utilization can be changed at any moments, then VMs can use free space to process more calls.

To study realistic scenarios, with gran variety of heterogeneous environments, we define three configurable parameters:

Utilization Threshold (UT) is parameter that is used as a trigger to request a new VM, when current utilization is reached the threshold. It is known that CPU cannot handle the stress when utilization is up to the certain level. Jitters and broken audio symptoms can appear. It depends the CPU characteristics, and allows to avoid voice quality reduction.

Rental Threshold (RT) is a time interval before the VM finishes its renting period. It restricts calls allocation to VMs before rental time is finished. It avoids the case when a call arrives just before finishing of renting time and terminate after renting period causing continue VM renting for one hour more.

Prediction Interval (PI) is time interval for which a prediction is performed. Predictor performs an estimation of utilization each Time Step (TS). A smaller TS provides a better prediction but increase the overhead of the systems. In the opposite way, a larger TS reduces the system overhead but decreases prediction accuracy.

Parameters can be configured and dynamically adapted to cope with different objective preferences, workloads, and cloud properties. This approach allows to adapt to cloud uncertainties such as dynamic elasticity, performance changing, virtualization, parameters such as an effective processor speed, number of running VMs and actual bandwidth, among many others.

In the first scenario, StUp=0. We assume that the scheduler uses a perfect prediction, so new VMs are initialized just in time to start processing the calls immediately. This scenario can be used as a reference of an ideal case, where the processing of the call is the only factor that affects the quality of voice.

If UT = 0.7, the system can guarantee the quality of service. Hence, the problem is reduced to a single objective problem. Figure 2 shows an example of call allocation with QoS guarantee. When VM_1 utilization overpasses UT=0.7 then VM_2 is initialized.

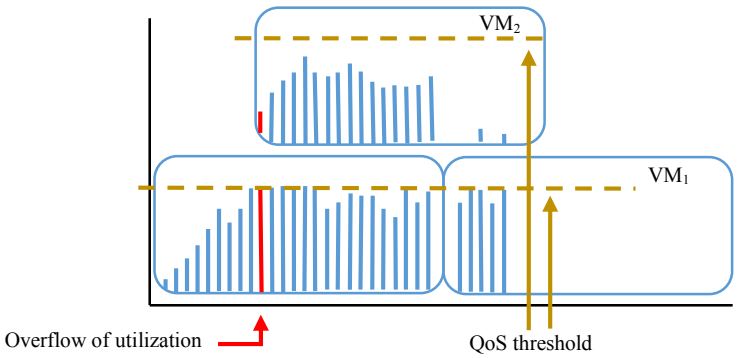

Figure 2. Calls allocation with QoS guarantee.

If UT = 1.0, the system cannot guarantee the QoS. Figure 3 shows an example of call allocation. When VM_1 utilization overpasses the capacity, then VM_2 is initialized and starts to process the calls immediately.

In the second scenario, StUp≠0, and UT= 0.7. Figure 4 shows an example of call allocation with VM startup time delay. VM_2 is requested when the utilization of VM_1 is over UT. During VM_2 StUp, VM_1

continues call processing with utilization more than threshold reducing QoS. The worst case appears when VM_1 does not have enough resources to process arriving calls. In this case, the system puts the calls on hold, waiting for available resources.

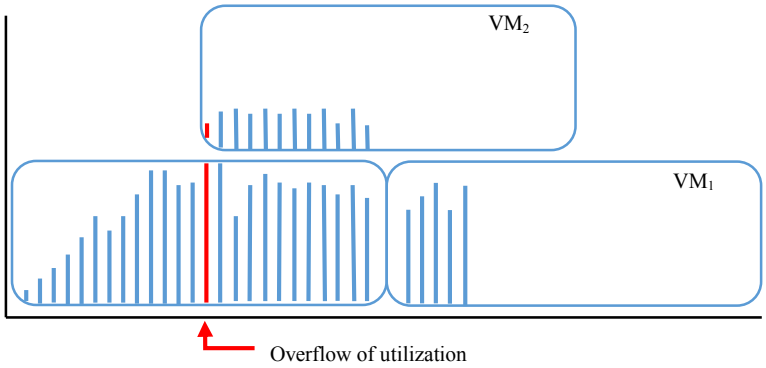

Figure 3. Calls allocation.

We classify call allocation strategies by the type and amount of information used for allocation: (1) Knowledge-Free (KF), with no information about calls and needed resources; (2) Utilization-Aware (UA), with VM utilization information; (3) Rental-Aware (RA) with known VM start time; and (4) Load-Aware (LA) with VM load information.

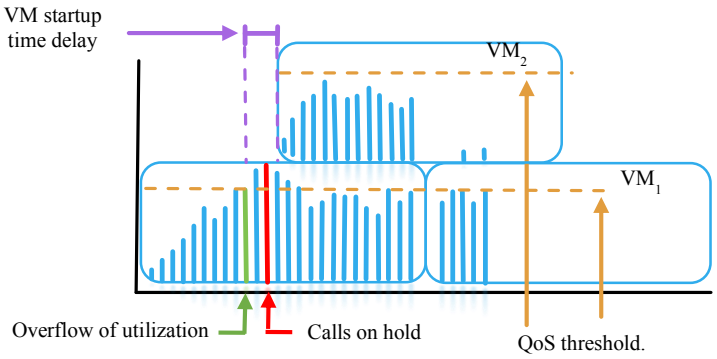

Figure 4. Calls allocation with startup time delay.

Table 5 summarizes the call allocation strategies that request VM, when arriving call exceeds UT. Additionally, we incorporate a mechanism to reduce the number of active VMs. Rental-Threshold strategies limits the allocation of calls RT minutes before the VM finishes its renting.

Table 6 describes the call allocation strategies with load prediction. Each given TS, the strategy estimates the utilization for next PI, and, if it exceeds the UT, it requests an additional VM.

Table 5. Call allocation strategies.

Type	Strategy	Description
KF	Rand	Allocates job *j* to VM randomly using a uniform distribution.
	RR	Allocates job *j* to VM using a Round Robin algorithm.
RA	MaxFTFit	Allocates job *j* to VM with farthest finish time.
	MidFTFit	Allocates job *j* to VM with shortest time to the half of its rental time.
	MinFTFit	Allocates job *j* to VM with closest finish time.
UA	FFit	Allocates job *j* to the first VM capable to execute it.
	BFit	Allocates job *j* to VM with smallest utilization left.
	WFit	Allocates job *j* to VM with largest utilization left.
KF + RT	Rand_05, 10, 15	Allocates job *j* to VM with RT equal to 5, 10 and 15 minutes using the Rand, and RR strategies.
	RR_05, 10, 15	
RA + RT	MaxFTFit_05, 10, 15	Allocates job *j* to VM with RT equal to 5, 10 and 15 minutes using the MaxFTFit, MidFTFit, and MinFTFit strategies.
	MidFTFit_05, 10, 15	
	MinFTFit_05, 10, 15	
UA + RT	BFit_05, 10, 15	Allocates job *j* to VM with RT equal to 5, 10 and 15 minutes using the BFit, FFit, and WFit strategies.
	FFit_05, 10, 15	
	WFit_05, 10, 15	

Table 6. Call allocation strategies with prediction.

Type	Strategy	Description
LA	Rand_StUp	Allocates job *j* to VM using the Rand, and RR strategies. They use PI equal to 10, 20, 30 and StUp seconds to estimate future load.
	Rand_s10, 20, 30	
	RR_StUp	
	RR_s10, 20, 30	
RA + LA	MaxFTFit_StUp	Allocates job *j* to VM using the MaxFTFit, MidFTFit, and MinFTFit strategies. They use PI equal to 10, 20, 30 and StUp seconds to estimate future load.
	MaxFTFit_s10, 20, 30	
	MidFTFit_StUp	
	MidFTFit_s10, 20, 30	
	MinFTFit_StUp	
	MinFTFit_s10, 201, 30	
UA + LA	BFit_StUp	Allocates job *j* to VM using BFit, FFit, and WFit strategies. They use PI equal to 10, 20, 30 and StUp seconds to estimate future load.
	BFit_s10, 20, 30	
	FFit_StUp	
	FFit_s10, 20, 30	
	WFit_StUp	
	WFit_s10, 20, 30	

3.5 Load-aware strategies

An adequate load estimation could help to increase the efficiency of calls allocation. In the best case, the system can predict necessity of additional VNs exactly StUp time in advance, so that it has VM just in time when it needed to avoid reduction quality of service and increasing cost.

Previously, we describe how StUp time delay affects the call allocation (see Figure 4). Figure 5 shows an example of load prediction, the balancer estimates the load for the next PI (black dots lines), and if the prediction overpass the UT (red dots line) then balancer starts new VMs before the arriving calls

degrade the QoS. All prediction models are used to request new VMs; they allow balancer to estimate the number of VMs at each TS.

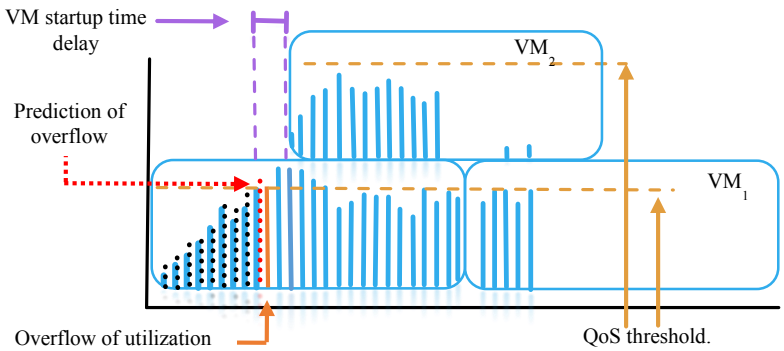

Figure 5. Calls allocation with prediction and startup time delay.

3.5.1 Rate of Change

Rate of Change – RoC (Campos and Scherson, 2000) is a dynamic distributed load balancing algorithm, it achieves the goal of minimizing processor idling times without incurring into unacceptably high load overheads. It calculates the change in the load between two TSs.

The TS is an adaptive parameter and its length may vary. RoC calculates the difference in load (Δ), and uses it as estimation on load for the PI. Δ allows allocating calls more efficiently. A finer sampling allows to improve the balance of the system, but it increases the overhead.

We use the concept of Δ as a mechanism to predict requests for new VMs, which can be provided after StUp time. Δ is used to estimate the number of VMs each PI. It permits to initialize VMs before the arriving calls degrade the QoS. Let $u_i(t)$ be the utilization of SNC_i at time t, and $k_i(t)$ the number of VMs running, then, the rate of load change during the TS is defined by:

$$\Delta_i(t) = (u_i(t) - u_i(t - TS))/k_i(t) \quad (8)$$

We estimate future load of the system by:

$$u_i(t + PI) = u_i(t) + \Delta_i(t) \quad (9)$$

Figure 6 shows RoC prediction scenario, the solid lines represent real utilization and dashed lines are estimated utilization. When the utilization is larger than UT then the system requests for a new VM. UT is an adjustable parameter that depends on the utilization threshold to guarantee QoS. For instance, at time T_4, the system initiates a new VM based on predicted utilization for T_5. This is an example of incorrect over-provisioning. At time T_5, the system predicts utilization at T_6 less than UT, and does not request new VM. However, the real utilization is more than UT causing QoS degradation (under-

provisioning). Finally, at time T_9, the system predicts utilization at T_{10} more that UT and requests new VM. This is an example of adequate VM provisioning.

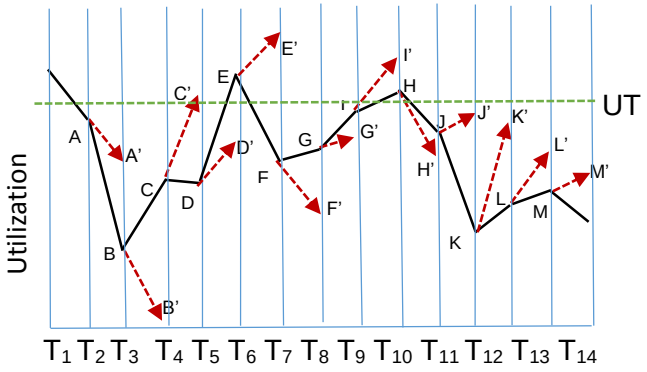

Figure 6. RoC prediction.

3.5.2 Neural Networks

Neural Networks (NNs) are widely used for prediction, classification or control, object recognition, etc. Figure 7 shows the NN architecture for call prediction. Nodes represent artificial neurons, and arrows characterize connections between them. An interconnected group of nodes (the output of one neuron can be the input of another) forms the network.

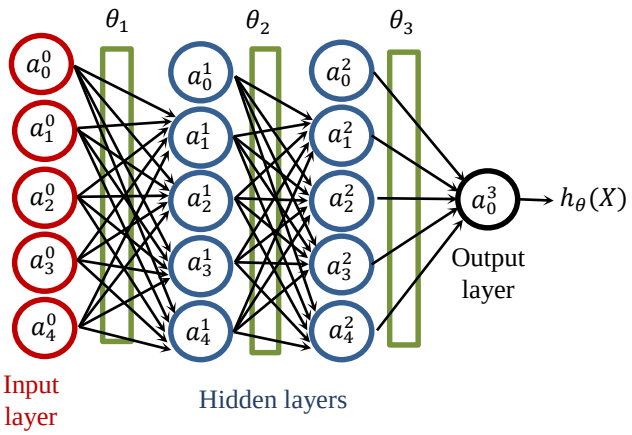

Figure 7. Neural network architecture.

The input data to the neural network is $X = [a_0^0, a_1^0, a_2^0, a_3^0, a_4^0]$. θ_1, θ_2, and θ_3 are the weight factors between layers 1, 2, and 3, respectively, and $h_\theta(X) = a_0^3$ is the output. In our case, $a_0^3 = u_i\ (t + PI)$. The inputs a_1^0, a_2^0, a_3^0 and a_4^0 are defined by: $a_1^0 = u_i(t)$, $a_2^0 = u_i(t - TS)$, $a_3^0 = u_i(t - 2 * TS)$ and $a_4^0 = t$.

The elements a_0^0, a_0^1, and a_0^2 define the bias at each layer. The neural network considers a sigmoid (logistic) activation function defined by (10). The activation of unit i (for i >0) in layer j (for j >0) is defined by (11) where $lon(\theta_i)$ is the number of weights on θ_i.

$$g(x) = \frac{1}{1 + e^{-z}} \tag{10}$$

$$a_j^i = g\left(\sum_{j=0}^{lon(\theta_i)} a_j^{i-1} * \theta_{i,j}\right) \tag{11}$$

The evaluation function is defined by (12). The first term is the sum-of-logarithmic error term. For training set, $T = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, where m is the number of elements on T, $y^{(i)}$ is the result value of the i-esim element, and $\mathrm{h}_\theta(x^{(i)})$ is the predicted value of $x^{(i)}$ inputs. The second term is a regularization term, where λ controls the relative importance of the two terms and L is the number of the layer in the neural network, see (13).

$$J(T) = \frac{-1}{m}\left[\sum_{i=1}^{m} \begin{matrix} y^{(i)} \log\left(\mathrm{h}_\theta(x^{(i)})\right) + \\ (1 - y^{(i)}) \log\left(1 - \mathrm{h}_\theta(x^{(i)})\right) \end{matrix}\right] + \frac{\lambda}{2m}\sum_{j=1}^{L-1} \theta_j^2 \tag{12}$$

$$\theta_i^2 = \sum_{j=1}^{lon(\theta_i)} (\theta_{i,j})^2 \tag{13}$$

3.5.3 History based prediction

To compare the efficiency of calls predictors previously described, we developed a historical predictor that consider the actual utilization and historical information about utilization. At time t, the historical predictor estimates $u_i\,(t + PI)$ based on: $u_i\,(t)$ and $\widehat{u_i}(t + PI)$, where $\widehat{u_i}(t)$ defines the historical standard deviation of utilization at time t. We estimate future load of the system by:

$$u_i\,(t + PI) = u_i(t) \pm \left(x * \widehat{u_i}(t + PI)\right) \tag{14}$$

Where x is a single uniformly distributed random number in the interval [0, 1]. Additionally, we generate a random value to define if the utilization is increased or decreased ($\pm$). Figure 8 shows an example of 85 days of utilization with an interval of 315 seconds.

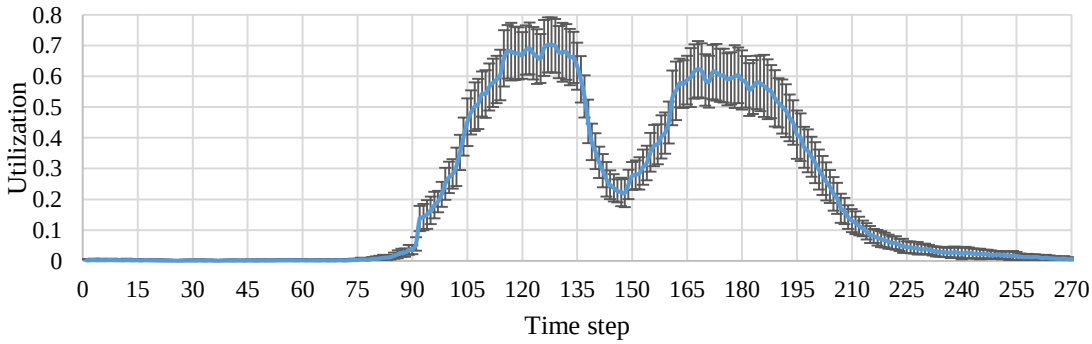

Figure 8. History based prediction with 315 sec. of TS.

4. Experimental setup

We perform experiments using standard trace based simulator CloudSim (Calheiros et al., 2011) extended by our algorithms, supporting dynamic calls arrival, VM startup delays, statistical analysis, and predictions models.

4.1 Workload

We use traces of real VoIP service (Simionovici et al., 2015) that include phone calls with the following information: Index of the call, ID of the user who makes the call, IP of the local phone, destination of the call, destination country name, telecommunications service provider; beginning of the call (timestamp); duration of the call (in seconds), duration of a paid call, cost per minute, etc.

Number of calls per day and calls duration are presented in Table 7 and Table 8. The histogram of the number of calls per hour during a day shows that the load is typical for business clients with two peaks in 10-12 and 14-16 hours (Simionovici et al., 2015).

Table 7. Number of calls per day.

Day	Total	Average
Monday	131,443	21,906
Tuesday	129,379	21,563
Wednesday	131,460	21,910
Thursday	130,439	21,739
Friday	120,999	20,166

Table 8. Call duration.

Time (min.)	Number of calls
0 - 1	310,602
1 - 2	136,211
2 - 3	68,988
3 - 4	39,392
4 - 5	23,397
…	…
19 - 20	721

4.2 Experimental analysis

The VoIP providers VMs on an hourly base. When the VM rental time is finished, the VM can be turned off only if VM is not processing calls. In any other case, this VM continue running for one hour more. In the first scenario, we analyze strategies with zero StUp, similar to a perfect prediction of the load. We study the behavior of allocation strategies under two UTs, with 0.7 and 1.

4.2.1 First scenario

Table 9 presents the simulation setup for the first scenario. Table 10 presents the average $\bar{b}$ during 24 weeks for both UTs. For UT = 0.7, BFit/FFit are the best strategies and WFit/MinFTFit are the worst ones. RR_05, RR_10, RR_15 have a better performance than RR. Similarly, WFit_05, WFit_10, and WFit_15 are better that WFit. The difference between the best strategy BFit and worst one (MinFTFit) is about 111 billing hours per week, on average. Additionally, Table 10 shows that 17 strategies improve the

performance of Round Robin currently used for VoIP service. The reduction of $\bar{b}$ is between 25.96% - 1.22% and only two strategies have worse performance.

Table 9. Simulation setup for the first scenario.

Name	Values
Super Node Cluster	1
Call allocation strategies	20
Number of Workloads	24
Workload length	168 hours
Number of codecs	4
Number of calls	2,526,595

We perform a joint analysis of two metrics according to the mean degradation methodology and obtain a set of compromise solutions that represent a good approximation to the Pareto front. In Table 11, we present the average $\bar{b}$ degradation, average $\bar{q}$ degradation and their means. The last four columns of the table contain the ranking of each strategy regard to the provider cost, quality, and their means. Rank $\bar{b}$ is based on the $\bar{b}$ degradation. Rank $\bar{q}$ refers to the position in relation to the $\bar{q}$ degradation. Rank mean is the position based on the averaging two ranking. Rank is the relative position to all strategies.

Table 10. Average weekly billing hours for different UTs.

Strategy \ UT	0.7	1.0
BFit	252.08	200.35
FFit	252.42	200.79
MaxFTFit	254.71	204.12
FFit_05	259.46	206.79
FFit_15	261.75	209.28
FFit_10	261.79	208.95
BFit_05	266.13	213.69
BFit_15	269.21	216.22
BFit_10	273.25	217.16
MidFTFit	276.08	221.23
RR_15	279.79	223.97
WFit_15	283.79	227.25
RR_10	289.00	231.59
WFit_10	290.42	232.93
RR_05	306.88	245.84
WFit_05	311.29	248.98
Rand	336.29	263.96
RR	340.46	266.13
WFit	351.08	272.12
MinFTFit	363.96	279.83
Average	288.99	229.56

BFit that allocates calls based on Best Fit strategy is the best strategy for $\bar{b}$. However, it is the worst strategy for $\bar{q}$. It tends to increase utilization, and reduce quality. The best strategy for $\bar{q}$ is WFit, it tends to underutilize VMs keeping the quality, but increasing VM number and renting cost. The strategy with the best compromise between two metrics is WFit_15. It allocates the call using Worst Fit strategy and limiting the assignation of calls 15 minutes before the VM reaches its renting time.

To solve the general bi-objective problem, we want to obtain a set of compromise solutions that represent a good approximation to the Pareto front. This is not formally the Pareto front as an exhaustive search of all possible solutions is not carried out, but rather serves as a practical approximation of a Pareto front.

Figure 9 shows the solution sets for the twenty strategies obtained based on 109 days of workload. This two-dimensional solution space represents a feasible set of solutions that satisfy the problem constraints. The solution space covers a range of values of $\bar{b}$ from 0 to 0.65, whereas values of $\bar{q}$ are in the range from 0 to 0.26.

We see that the solution space is divided into three groups located on the right lower side, left upper side and in the middle. BFit, FFit, and MaxFTFit are located in the lower right side being among the best solutions in terms of $\bar{b}$. They outperform other strategies, like RR, that are in current use for VoIP service. WFit is located in the left side being among the best solutions in terms of $\bar{q}$. The three versions of WFit with different RTs (WFit_05, WFit_10, WFit_15) have a good behavior.

Table 11. Degradations and ranking.

Strategy	$\bar{b}$	$\bar{q}$	Mean	Rank $\bar{b}$	Rank $\bar{q}$	Rank mean	Rank
BFit	0.263	21.099	10.681	**1**	20	6	4
BFit_05	6.911	17.930	12.420	7	16	12	6
BFit_10	8.405	17.342	12.874	9	13	15	5
BFit_15	8.091	17.240	12.665	8	12	14	**3**
FFit	0.535	21.031	10.783	**2**	19	7	4
FFit_05	3.483	18.758	11.120	4	18	8	5
FFit_10	4.638	18.069	11.354	5	17	10	5
FFit_15	4.812	17.819	11.315	6	14	9	**3**
MaxFTFit	2.096	17.835	9.965	**3**	15	4	**1**
MidFTFit	10.536	16.442	13.489	10	11	16	4
MinFTFit	39.147	12.800	25.973	20	10	20	7
Rand	31.263	1.094	16.178	17	4	17	4
RR	32.681	0.732	16.707	18	**2**	18	**3**
RR_05	22.542	2.144	12.343	15	5	11	**3**
RR_10	15.388	4.666	10.027	13	7	5	**3**
RR_15	11.772	6.980	9.376	11	9	**2**	**3**
WFit	35.435	0.000	17.718	19	**1**	19	**3**
WFit_05	23.904	1.085	12.494	16	**3**	13	**2**
WFit_10	16.606	3.301	9.954	14	6	**3**	**3**
WFit_15	13.292	5.428	9.360	12	8	**1**	**3**

WFit is the best for $\bar{q}$ degradation. The range of the $\bar{b}$ degradations is from 0.16 to 0.56. WFit_05 increases $\bar{q}$ degradation to 0.017, but reduces $\bar{b}$ up to 0.05. For WFit_10, the $\bar{q}$ degradation raises from 0.017 to 0.06, but $\bar{b}$ reduces to 0.023.

Finally, WFit_15 has a wide range of solution for $\bar{q}$ degradation (from 0.019 to 0.099) but only the 20% of their solutions are over the 20% of $\bar{b}$ degradation. WFit versions cover different sectors in the

Pareto front, and they show the best compromise between both objectives for the twenty strategies. The MaxFTFit solution space is in the same range for cost as BFit and FFit. It overcomes $\bar{q}$ of both strategies.

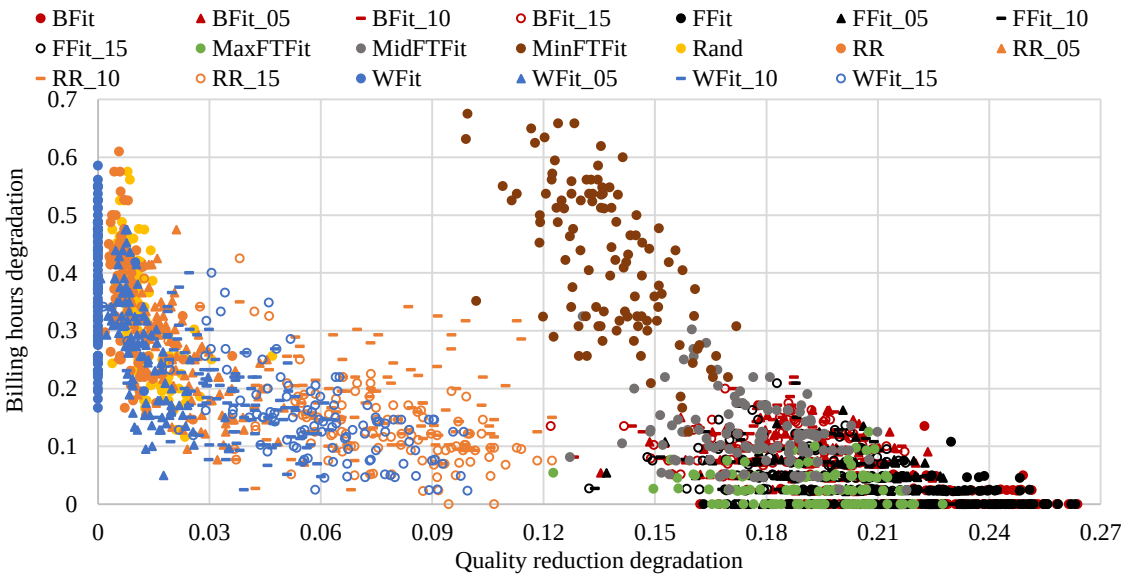

Figure 9. The solution space.

WFit_05, WFit_10, WFit_15 and MaxFTFit strategies cover better the solution space and Pareto front. They are good options for the VoIP providers. Figure 10 shows the twenty approximations of Pareto fronts generated by the studied strategies.

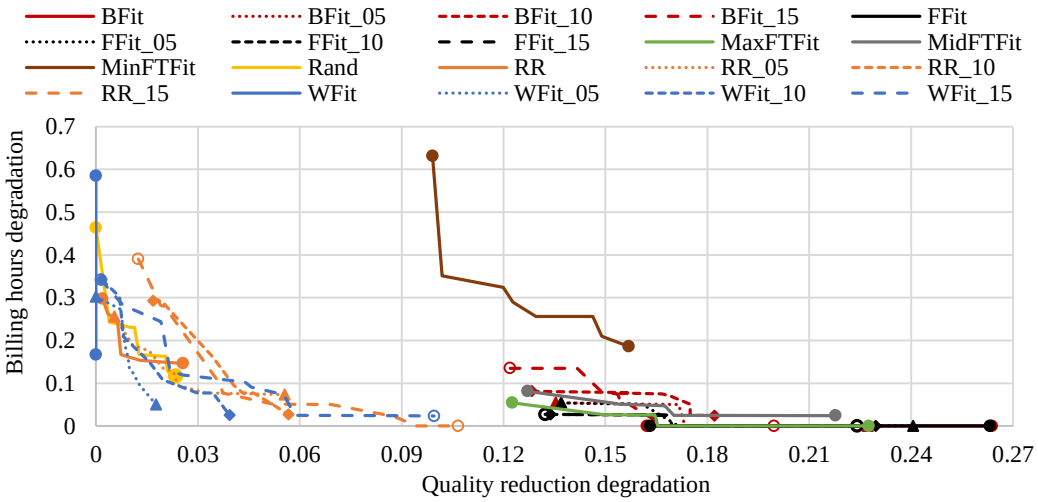

Figure 10. Pareto fronts.

Two sets of non-dominated solutions can be compared using the set coverage metric. The rows of the Table 12 show the values $SC(A,B)$ for the dominance of strategy A over strategy B. The columns indicate $SC(B,A)$, that is, dominance of B over A. The last two columns show the average of $SC(A,B)$ for row A over column B, and ranking based on the average dominance. Similarly, the last two rows show average dominance B over A, and rank of the strategy in each column.

The ranking of the strategies is based on the coverage percentage. A higher ranking implies a better front. Table 12 reports the SC results for each of the twenty Pareto fronts. According to the set coverage metric, the strategy that has the best compromise between $\bar{b}$ and $\bar{q}$ is WFit_15, followed by MaxFTFit, RR_15 and WFit_05.

We see that MaxFTFit dominates the fronts of other strategies in the range 0-60%, with 21.8% in average occupying the second rank. $SC(A, MaxFTFit)$ shows that MaxFTFit is dominated by other fronts on 6.9% in average. Meanwhile, WFit_05 and MaxFTFit with the second and third ranks are dominated by other strategies on 19.5% and 21.9%, on average, respectively. They are dominated for other strategies on 6.2% and 6.9%.

However, we should not consider only Pareto fronts, when many solutions are outside the Pareto optimal solutions. This is the case of BFit_xx, FFit_xx, RR_xx: although the Pareto fronts are of good quality, many of the generated solutions are quite far from it, and, hence, a single run of the algorithm may produce significantly worse results.

Table 12. Set coverage and ranking (days).

A \ B	BFit	BFit_05	BFit_10	BFit_15	FFit	FFit_05	FFit_10	FFit_15	MaxFTFit	MidFTFit	MinFTFit	Rand	RR	RR_05	RR_10	RR_15	WFit	WFit_05	WFit_10	WFit_15	Mean	
BFit	1	0	0	0	0.30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.065	18
BFit_05	0.07	1	0.06	0.02	0.08	0.17	0.05	0.01	0.01	0.01	0	0	0	0	0	0	0	0	0	0	0.074	17
BFit_10	0.03	0.17	1	0.07	0.02	0.11	0.10	0.05	0.02	0.02	0	0	0	0	0	0	0	0	0	0	0.079	16
BFit_15	0.03	0.23	0.28	1	0.05	0.10	0.11	0.10	0.03	0.01	0	0	0	0	0	0	0	0	0	0	0.096	14
FFit	0.21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.061	19
FFit_05	0.33	0.06	0.04	0.02	0.30	1	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0.088	15
FFit_10	0.18	0.17	0.09	0.06	0.18	0.38	1	0.06	0.04	0.03	0	0	0	0	0	0	0	0	0	0	0.109	11
FFit_15	0.21	0.29	0.19	0.12	0.21	0.33	0.34	1	0.03	0.06	0	0	0	0	0	0	0	0	0	0	0.139	8
MaxFTFit	0.43	0.42	0.28	0.24	0.46	0.60	0.47	0.38	1	0.09	0	0	0	0	0	0	0	0	0	0	0.218	**2**
MidFTFit	0.02	0.20	0.31	0.18	0	0.14	0.13	0.08	0.03	1	0	0	0	0	0	0	0	0	0	0	0.105	13
MinFTFit	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0.050	20
Rand	0	0	0	0	0	0	0	0	0	0.01	0.94	1	0.01	0.09	0.03	0	0	0.05	0.01	0.02	0.107	12
RR	0	0	0	0	0	0	0	0.01	0	0	0.90	0.52	1	0.10	0.02	0.03	0	0.07	0.00	0.02	0.133	10
RR_05	0	0.04	0.04	0.04	0	0	0	0.01	0.01	0.08	0.97	0.03	0.02	1	0.21	0.11	0.01	0.01	0.19	0.14	0.145	7
RR_10	0.01	0.13	0.20	0.17	0	0.06	0.06	0.07	0.04	0.28	0.99	0.02	0	0.02	1	0.31	0	0	0.01	0.21	0.178	6
RR_15	0.02	0.32	0.39	0.39	0.02	0.11	0.17	0.16	0.06	0.52	0.98	0	0	0.02	0.01	1	0	0	0	0.02	0.210	**3**
WFit	0	0	0	0	0	0	0	0	0	0	0.84	0.27	0.41	0.08	0.01	0.02	1	0.09	0.03	0.01	0.138	9
WFit_05	0	0.04	0.02	0.02	0	0.02	0	0.01	0	0.03	0.96	0.42	0.15	0.51	0.20	0.15	0.01	1	0.22	0.15	0.195	4
WFit_10	0	0.09	0.15	0.17	0.01	0.05	0.06	0.06	0.03	0.25	0.97	0.01	0.04	0.06	0.38	0.28	0	0.02	1	0.28	0.194	5
WFit_15	0.02	0.26	0.32	0.32	0.01	0.14	0.16	0.14	0.08	0.45	0.98	0	0.01	0.02	0.06	0.39	0	0	0.02	1	0.219	**1**
Mean	0.128	0.171	0.169	0.140	0.132	0.160	0.132	0.106	0.069	0.142	0.477	0.113	0.082	0.095	0.096	0.114	0.051	0.062	0.074	0.092		
Rank	12	19	18	15	14	17	13	9	**3**	16	20	10	5	7	8	11	**1**	**2**	4	6		

4.2.2 Second scenario

In the second scenario, we incorporate eight StUp delays: 0, 45, 90, 135, 180, 225, 270, and 315 seconds (Mao and Humphreyet, 2012; Hoffman, 2017) for VMs as test cases and ten PSs: 10, 20, 30, StUp (PS= StUp). StUp = 0 represents the perfect predictor because the system starts VMs when it needs, the

predictor estimates ideally the load in the previous PI, and it is used as a reference to evaluate the performance of other strategies under several StUps.

The main goal is to analysis the quality of service degradation inherent to StUp and the reduction of quality degradation due to the prediction of the load. Table 13 presents the simulation setup for the second scenario. It considers 30 workloads of 24 hours without weekends and holiday.

Table 13. Simulation setup for second scenario.

Name	Value
Super Node Cluster	1
Total call allocation strategies:	128
Without prediction	32
With prediction	96
Number of Workloads	30
Workload length	24 hours
Number of codecs	4
Number of calls	643,718
VMs StUp	7

We do not present voice quality reduction ($\bar{q}$) due to the difference between the best and worst strategy is about 9.292×10^{-3}. We analyze number of calls placed on hold $\bar{c}$ as a representation of QoS reduction.We perform a joint analysis of two metrics according to the mean degradation methodology. First, we present the analysis of rented VMs ($\bar{b}$) and the number of call on hold ($\bar{c}$) separately. Then, we find the strategy that generates the best compromise between them.

Table 14 presents the average degradation of $\bar{b}$, $\bar{c}$ and their means for the best strategies. The last four columns of the table contain the ranking of each strategy regarding the provider cost, quality, means, and ranking between strategies. Rank $\bar{b}$ is based on the billing hours degradation. Rank $\bar{c}$ refers to the position in relation to the calls on hold. Rank mean is the position based on the averaging two ranking. Rank is the relative position of all strategies, it considers the mean of Rank $\bar{b}$ and Rank $\bar{c}$.

The best strategy for $\bar{b}$ is NN_FFit_stUp that allocates calls based on first fit strategy with prediction time is equal to StUp. However, it is the second worst strategy for $\bar{c}$. It tends to increase utilization, and reduce quality. NN_BFit_stUp and NN_MaxFTFit_stUp are the second and third best strategies (with respect to $\bar{b}$). All strategies use neural networks to predict future calls arrival.

The best strategies for $\bar{c}$ are RoC_MinFTFit_s30 (where we put the calls into the VM where its rental time is the nearest to finish), RoC_Rand_s30, RoC_RR_s30, and RoC_WFit_s30. They use rate of change to predict the workload and the prediction time is 30 seconds. It tends to underutilize VMs reducing $\bar{c}$, but increasing $\bar{b}$.

Finally, 28 strategies improve the performance of Round Robin considering the average degradation of $\bar{b}$ and $\bar{c}$. The rank of Round Robin is 32, with respect to the relative position of all the strategies, it means that at least 52 strategies have a performance equal or better than this strategy.

Table 14. Degradations and ranking for the best allocation strategies.

Strategy	Deg $\bar{b}$	Deg $\bar{c}$	Mean	Rank $\bar{b}$	Rank $\bar{c}$	Rank mean	Rank
NN_FFit_stUp	0.0000	4.1342	2.0671	**1**	100	127	32
NN_BFit_stUp	0.0084	3.9974	2.0029	**2**	99	126	32
NN_MaxFTFit_stUp	0.0157	4.4158	2.2158	**3**	101	128	34
RoC_WFit_s30	0.4455	0.0000	0.2227	105	**1**	30	36
RoC_MinFTFit_s30	0.5118	0.0000	0.2559	118	**1**	36	49
RoC_Rand_s30	0.3918	0.0000	0.1959	85	**1**	21	18
RoC_RR_s30	0.4056	0.0000	0.2028	91	**1**	22	23
Roc_MinFTFit_s10	0.5253	0.0132	0.2692	121	**2**	41	53
RoC_WFit_s10	0.4543	0.0132	0.2338	106	**2**	33	38
RoC_RR_s10	0.4192	0.0158	0.2175	95	**3**	27	29
Rand_15	0.1731	0.0579	0.1155	64	11	**1**	12
RoC_FFit_s30	0.0464	0.2026	0.1245	15	28	**2**	**2**
RoC_BFit_s30	0.0517	0.2026	0.1272	20	28	**3**	**3**
RoC_FFit_s10	0.0425	0.2500	0.1462	9	33	9	**1**
RoC_FFit_s20	0.0427	0.2474	0.1451	10	32	8	**1**
RR	0.4222	0.0184	0.2203	**97**	**4**	**29**	**32**

5. Conclusions

We formulate and study scheduling problems addressing cloud-based VoIP load-aware scheduling strategies with VM startup prediction. The bi-objective model considers provider cost, contributed by billing hours for used VMs, and quality of service, affected by call processing and calls on hold.

We conduct comprehensive simulation on real data of on-line non-clairvoyant scheduling strategies with fixed threshold of utilization to request VMs, and strategies with dynamic prediction of the load. Experimental results indicate that the proposed algorithms can be efficiently used in a VoIP cloud environment.

We show that our load-aware with predictions strategies outperform the known ones providing suitable quality of service and lower cost. The robustness of these strategies is also analyzed varying VM startup time delays to deal with realistic VoIP cloud environments.