Asignación eficiente de recursos en centros de datos virtualizados abordando el compromiso entre tolerancia a fallos y consumo energético

Eder Samir Correa Acosta

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2016

Asignación eficiente de recursos en centros de datos virtualizados abordando el compromiso entre tolerancia a fallos y consumo energético

Eder Samir Correa Acosta

Tesis presentada como requisito para optar al título de:
Magister en Ingeniería de Telecomunicaciones

Director:
Ph.D. Juan Felipe Botero Vega

Grupo de Investigación en Telecomunicaciones Aplicadas, *GITA*
Línea de investigación en modelamiento de redes

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2016

Agradecimientos

El autor agradece al proyecto programático del CODI 2014-856 y a la estrategia de sostenibilidad CODI 2014-2015 de la Universidad de Antioquia, por el apoyo para el desarrollo de este trabajo.

Resumen

Los centros de datos han venido ganando mucho interés debido a su capacidad para almacenar una gran cantidad de información; como resultado se han convertido en el mayor soporte para la computación en la nube. Actualmente, los centros de datos presentan problemas de rendimiento, como: calidad de servicio no garantizada, riesgos de seguridad, complejidad en la administración e inflexibilidad. Los centros de datos virtualizados son una herramienta emergente para enfrentar estos problemas. Pero su implementación trae incluso un reto mayor: ¿Cómo asignar óptimamente las demandas de los centros de datos virtuales sobre la infraestructura física, de tal manera que se reduzcan los costos de operación del Centro de Datos, se mejoren las ganancias y se garanticen los acuerdos de servicio con los usuarios finales? Este reto es conocido como "Virtual Data Center Embedding" (VDCE, siglas en inglés) [1]. Este manuscrito inicialmente presenta un estado del arte de la investigación en este problema, y también propone una clasificación taxonómica de las propuestas principales para resolverlo. Esta revisión nos ha permitido encontrar que no se ha propuesto una solución que estudie el comportamiento del proceso del VDCE cuando se busca reducir los recursos necesarios para la tolerancia a fallos y minimizar el consumo energético en forma coordinada.

En esta investigación se presenta una propuesta para abordar el VDCE, basados en un programa lineal entero mixto (MILP, siglas en inglés) multiobjetivo, que se soluciona utilizando el método de las ϵ-restricciones con el que se pueden obtener respuestas no dominadas. Nuestra propuesta soluciona el VDCE buscando el equilibrio entre la minimización del consumo energético y el garantizar supervivencia a fallos en los servidores de la red del centro de datos físico.

Para comprobar el desempeño implementamos nuestra propuesta de forma simulada en java utilizando las funciones proporcionadas por las herramientas ALEVIN y Gurobi. Los resultados del experimento demuestran que nuestro modelo tiene mejor desempeño cuando se le dio más importancia a minimizar los recursos de respaldo necesarios para garantizar una completa tolerancia a fallos. Los resultados también demuestran que el modelo propuesto en esta investigación tiene una aceptación de redes virtuales que supera el 8 % cuando se compara con un algoritmo existente.

Contenido

Lista de Figuras

Lista de Tablas

1 Introducción

Los centros de datos (DC, siglas en inglés) han venido adquiriendo un interés creciente gracias a su capacidad de almacenamiento de grandes cantidades de información, así como por su capacidad de procesar aplicaciones y servicios a gran escala. La facilidad con que un usuario puede obtener servicios y aplicaciones en la nube de parte de un proveedor, ha contribuido significativamente al incremento de la popularidad de los centros de datos utilizados para ofrecer recursos computacionales bajo demanda [2]. Antiguamente, los DCs usaban servidores dedicados para almacenar la información y ejecutar las aplicaciones (una solución poco efectiva en términos de costo operativo). Con el fin de optimizar el costo de operar la infraestructura física de los DCs, la virtualización de servidores emergió como la tecnología que permite instanciar varias máquinas virtuales en una sola máquina física (e.g. Vmware[1], Virtualbox[2], etc.). Por medio de la virtualización de servidores los costos operativos en los DCs se vieron drásticamente reducidos [3]; sin embargo, esta solución no es suficiente para resolver todas las limitaciones de las arquitecturas actuales de los DCs. Las topologías de los DCs, no sólo constan de servidores físicos, sino que también éstos están interconectados por medio de una red de comunicación. Las aplicaciones que corren en los centros de datos tienen parámetros de calidad de servicio que no solo dependen de los servidores, sino también de los elementos de red. La operación de los DCs que virtualizan su infraestructura de servidores pero no lo hacen en su infraestructura de red resulta en una serie de limitaciones como: calidad de servicio no garantizada para las aplicaciones de computación en la nube, riesgos en la seguridad de las aplicaciones [4], dificultad de migrar las aplicaciones existentes en los clientes hacia la nube, falta de flexibilidad para delegar parte del control de la red a los dueños de las aplicaciones y falta de soporte para innovación en la red [3].

Para solucionar estos inconvenientes, recientemente la comunidad científica ha propuesto la virtualización de los recursos involucrados en las redes de comunicación (conmutadores, enlaces y enrutadores) de los DCs así como de los servidores físicos [3]. La virtualización de redes, es una línea ampliamente investigada que se ha propuesto como un componente inseparable de la futura arquitectura del Internet [5, 6, 7, 8]; que consiste en permitir múltiples redes coexistir en una misma red física mediante la virtualización de los equipos de red. La virtualización de los centros de datos es una línea de investigación emergente que básicamente, pretende aplicar los conceptos de la virtualización en redes al entorno de los DCs, es decir, crear instancias virtuales individuales del DC físico permitiendo aplicar las ventajas

[1]VMware, http://www.vmware.com

[2]Virtual Box, https://www.virtualbox.org/

provenientes de la virtualización: un aislamiento lógico y total flexibilidad para el manejo de la red. Al estar las redes de DCs lógicamente separadas es posible introducir aislamiento entre las instancias creando un nivel de seguridad superior a las arquitecturas tradicionales de centros de datos. El aislamiento además facilita la aplicación de calidad del servicio, pues se puede aplicar un tratamiento individual a cada uno de los centros de datos virtuales. El despliegue de nuevas aplicaciones y servicios en los centros de datos virtualizados es facilitado por protocolos personalizados, lo que permite una constante innovación en la red [3].
El implementar la virtualización de los recursos de los DCs trae consigo un nuevo reto: ¿Cómo asignar de manera óptima las demandas de recursos de los DCs virtuales sobre la infraestructura física de tal forma que se reduzcan los costos de operación del DC, mejoren los ingresos para los proveedores de infraestructura y se cumplan los acuerdos de calidad de servicio pactados con los usuarios? Este reto se conoce como problema de asignación de recursos en centros de datos virtualizados (VDCE, singlas en inglés).
Si un servicio no tiene la capacidad de sobrevivir a un fallo se corre el riesgo de ofrecer un servicio intermitente; lo que incumpliría los acuerdos de servicio arriesgando sanciones al proveedor y causándoles dificultades a los clientes, pues muchas aplicaciones y servicios que se ejecutan en un DC requieren estar siempre activas. Los costos producidos por energía consumida en los DCs son altos, y constituyen gran parte de los costos de operación [9, 3], por lo tanto es conveniente implementar estrategias que permitan mejorar la eficiencia energética. Según lo mejor de nuestro conocimiento hasta ahora no se ha propuesto una solución que estudie el proceso de asignación de recursos que de una idea del costo energético que hay que sacrificar para garantizar la supervivencia y que funcione en forma coordinada. Es por esto que en esta tesis se presenta una propuesta para solucionar el problema de VDCE mediante la implementación de un programa lineal entero mixto (MILP, siglas en inglés) multiobjetivo, que propone un método solución de las ϵ-restricciones con el que se pueden obtener respuestas no dominadas (conjunto de soluciones que son mejores a las demás en todas funciones objetivos evaluadas). En nuestra tesis se evalúa el equilibrio entre la minimización del consumo energético y la tolerancia a fallos en los servidores de la red de un centro de datos físico, para esto se escogen tres modos de operación que se definen de acuerdo a los resultados de la evaluación del modelo en la simulación.

1.1. Máximas contribuciones

En esta investigación, se presenta un método para la asignación de recursos para centros de datos. A continuación nuestras contribuciones:

- Se construyó un estado del arte, que estudia en detalle las propuestas que resuelven el VDCE.
- Se propone una taxonomía de clasificación y se presentan las posibles líneas de investigación emergentes consideradas por la comunidad académica.

- Se introduce un modelo para abordar el problema de VDCE, basados en un programa lineal entero mixto (MILP, siglas en inglés) multiobjetivo, que se soluciona utilizando el método de las ϵ-restricciones.
- Se investiga el equilibrio entre la minimización del consumo energético y garantizar supervivencia a fallos en los servidores de la red del centro de datos físico.
- En nuestra propuesta, el VDCE es resuelto en una forma coordinada: Los recursos de respaldo y la petición original son asignadas simultáneamente asegurando una solución óptima.

1.2. Organización de la Tesis

El resto de la tesis está organizada de la siguiente manera: El capítulo 2 introduce el VDCE en centros de datos, se propone una taxonomía para la clasificación de las diferentes propuestas que tratan de resolver este problema y también las posibles tendencias para futuras líneas de investigación; el capítulo 3 define el modelo matemático que representa la operación de un DC cuando se optimizan los recursos activos y de tolerancia al mismo tiempo; el capítulo 4 es dedicado a la simulación del modelo y el análisis de resultados; finalmente se concluye esta tesis en el capítulo 5.

2 Marco teórico y trabajo relacionado

En este capítulo se realiza una revisión exhaustiva de aquellas propuestas que buscan superar este reto en forma óptima apuntando a diferentes objetivos y se aborda el problema en sus diferentes vertientes. Así mismo se propone una taxonomía de clasificación y se presentan las posibles líneas de investigación emergentes consideradas por la comunidad académica.

2.1. Marco teórico

Un DC es una estructura diseñada para almacenar grandes cantidades de información. Su arquitectura tradicional está conformada principalmente por servidores, conmutadores, enrutadores y enlaces. Grandes empresas como Google[1], Facebook[2] y Amazon[3] utilizan DCs para almacenar sus datos y soportar sus aplicaciones.

2.1.1. Topologías

A continuación se presentan las topologías más utilizadas por la comunidad académica para modelar la infraestructura de los DCs.

VL2 [10]

Es una arquitectura cuyo objetivo es la construcción de una red que simula un conmutador central único de alta capacidad conectando todos los servidores. VL2 está formada por capas extensas de conmutadores de bajo costo que forman una topología tipo CLOS [11] que se puede ampliar para agregar más capacidad y resistencia a fallos, y que proporciona una alta diversidad de caminos entre los servidores. Una topología tipo CLOS consiste en varias capas de conmutadores donde cada conmutador está conectado a todos los conmutadores de las capas adyacentes (ver Fig. **2-1**). VL2 busca ampliar los niveles superiores de la red de manera que el impacto de los fallos en los conmutadores sea pequeño. El gran número de caminos entre dos conmutadores en las capas de distribución implica que si hay n conmutadores intermediarios, el fallo de uno de estos reduce el ancho de banda sólo por $1/n$.

[1]Google, Centros de Datos, http://www.google.com/intl/es-419/about/datacenters/
[2]Facebook, Prineville Data Center, https://www.facebook.com/PrinevilleDataCenter
[3]Amazon Elastic Compute Cloud, http://aws.amazon.com/es/ec2/

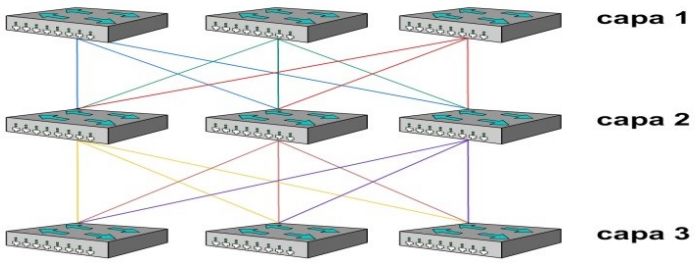

Figura 2-1: Topología tipo CLOS con 3 capas.

FatTree [12]

Es un tipo especial de topología CLOS que está organizada en una estructura de árbol. Para la construcción de una red Fat-Tree tipo k se deben tener conmutadores con capacidad de k puertos, y se diseñan cápsulas de k conmutadores; cada una de las capsulas contiene dos capas de conmutadores, una capa inferior de $k/2$ conmutadores de frontera (conectan a los servidores) y otra superior de $k/2$ conmutadores de adición; cada conmutador de frontera conecta a $k/2$ servidores, los dos puertos restantes de estos conmutadores son conectados a los conmutadores de la capa superior de la capsula [13]. De acuerdo a lo expuesto anteriormente, en la Fig. **2-2** se construye una red FatTree con $k = 4$, es por esto que se tienen conmutadores de 4 puertos y 4 cápsulas, donde cada capa inferior tiene 2 conmutadores al igual que cada capa superior; cada uno de los conmutadores de la capa inferior son conectados a 2 servidores. Una red FatTree tipo k tiene $(k/2)^2$ conmutadores de núcleo de k puertos; cada conmutador de núcleo tiene un puerto conectado a cada capsula de k conmutadores. El i-avo puerto de cada conmutador de núcleo es conectado a la capsula i, de tal manera que los puertos consecutivos de los conmutadores de la capa superior están conectados a los conmutadores de núcleo en $k/2$ caminos [13]. Volviendo a la Fig. **2-2**, podemos observar que existe un total de 4 conmutadores de núcleo y cada uno de estos se conecta a un conmutador de la capa superior de cada cápsula; a su vez, cada conmutador de la capa de adición se conecta a 2 conmutadores de núcleo. En general una red FatTree construida con conmutadores de k puertos puede soportar $k^3/4$ servidores.

Bcube [14]

Bcube es una topología definida recursivamente. Un Bcube_0 es construido con un conmutador de n puertos conectando n servidores. Un Bcube_1 es construido de n Bcube_0s y n conmutadores de n puertos. En general un Bcube_k $(k \geq 1)$ es construido de n $\text{Bcube}_{(k-1)}$s y n^k conmutadores de n puertos. Cada servidor en un Bcube_k tiene $k+1$ puertos enumerados del nivel 0 al nivel k. Un Bcube_k tiene $n^{(k+1)}$ servidores y $k+1$ niveles de conmutadores, donde cada nivel tiene n^k conmutadores de n puertos. En la Fig. **2-3**, se muestra una red

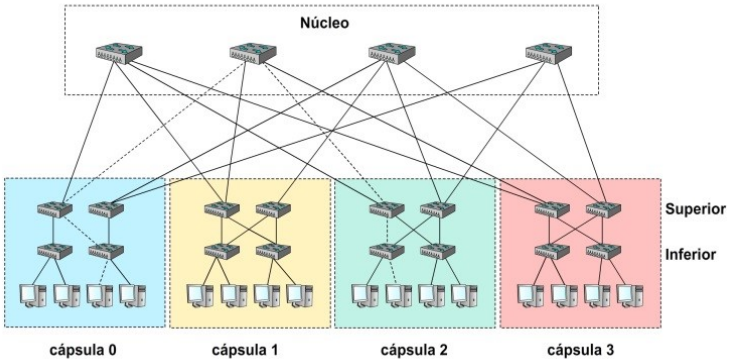

Figura 2-2: Topología Fat Tree, k=4.

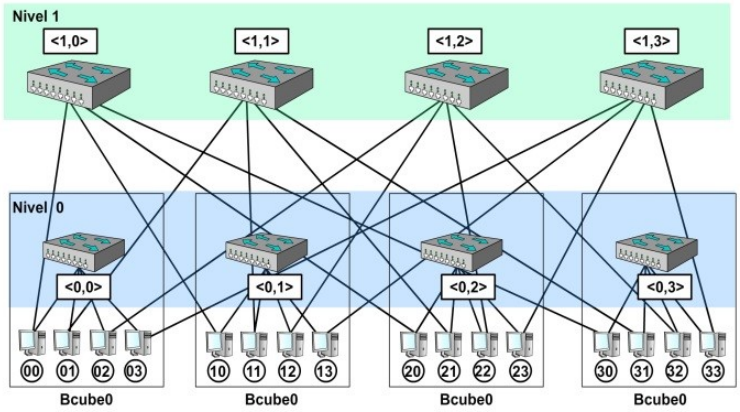

Figura 2-3: Centro de datos con topología Bcube con n = 4 y k = 1

Bcube_1 $n = 4$ y $k = 1$, primero se determina el Bcube_0, que en este caso corresponde a un conmutador de 4 puertos conectado a 4 servidores. Luego se consolida el Bcube_1, que se construye con 4 Bcube_0s que son conectados a 4 conmutadores que forman el nivel 1; los conmutadores dentro de la estructura de Bcube_0 forman la capa 0, esta estructura Bcube puede soportar 16 servidores $n^{(k+1)}$. Para las conexiones de Bcube_k se siguen los siguientes pasos: Se nombran los n $\text{Bcube}_{(k-1)}$ s de 0 hasta $n-1$, y los servidores en cada $\text{Bcube}_{(k-1)}$ de 0 a $n^k - 1$. Se conecta el puerto del nivel k de el i-ésimo servidor ($i \in [0; n^k - 1]$) en el j-ésimo $\text{Bcube}_{(k-1)}$, ($j \in [0; n-1]$) al j-ésimo puerto del i-ésimo conmutador del nivel k.

Estrella

Es aquella donde existe un conmutador central conectado a los demás dispositivos; el conmutador debe tener n puertos para poder soportar n servidores. Generalmente se usan para representar las peticiones virtuales y no para construir una red física que soporta un centro

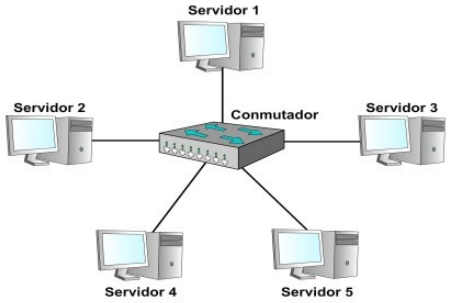

Figura 2-4: Topología tipo estrella

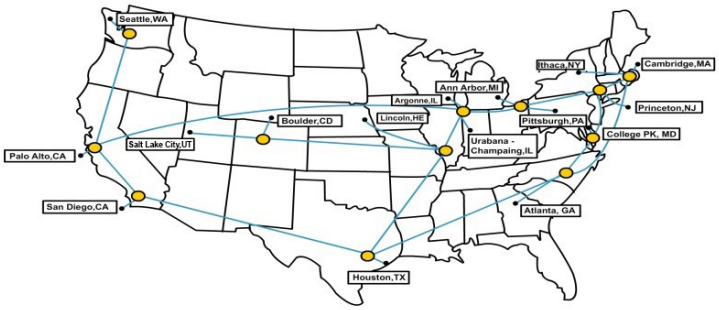

Figura 2-5: NSFNet

de datos. En la Fig. **2-4** se muestra la topología estrella para $n = 5$.

NSFNET

Aunque no es topología inherente para los DCs, es una topología de red que puede ser útil cuando se tiene una infraestructura de centros de datos conectados por una red central. NSFNET es una red troncal ubicada en los Estados Unidos que conecta varias instituciones en diferentes ciudades, que fue creada principalmente para promover la educación e investigación por parte del programa NSF[4]. La forma como están conectados los diferentes nodos se muestra en la Fig. **2-5**.

2.1.2. Virtualización en los DCs

Un centro de datos virtualizado es un centro de datos donde algunos o todos los componentes de hardware (servidores, conmutadores, enrutadores o enlaces) tienen capacidad de virtualización. Un centro de datos virtual (siglas en inglés VDC), es una colección de recursos virtuales (servidores, enrutadores y conmutadores virtuales) conectados por medio de enlaces

[4]The National Science Foundation Network, http://www.nsf.gov

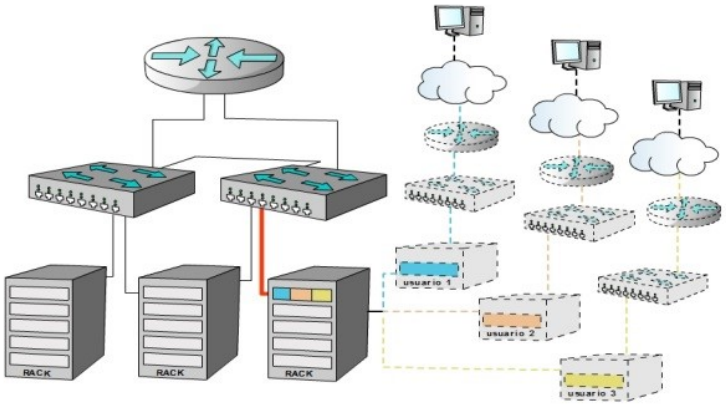

Figura 2-6: VDCs en un centro de datos virtualizado

virtuales. Un VDC es una instancia de un centro de datos a la que se le han aplicado técnicas de virtualización [15]. En un centro de datos virtualizado conviven varios VDCs que comparten su infraestructura física; cada uno de éstos está lógicamente separado de los demás. Un ejemplo de un centro de datos virtualizado con tres centros de datos virtuales asignados a él se puede observar en la Fig. **2-6**, vemos que de un solo servidor, enlace y conmutador físicos; se han creado tres VDCs independientes. Debido al aislamiento en cuanto al acceso y uso de físico recursos, un VDC da la ilusión de ser un centro de datos dedicado [16]. Una petición de VDC es un conjunto de requerimientos que un cliente quiere reservar para ejecutar uno o varios servicios/aplicaciones. Mediante peticiones de VDCs un cliente reserva una porción de un DC (e.g. una petición puede requerir un número determinado de servidores virtuales con especificaciones como el ancho de banda de conexión entre ellas y un sistema operativo para cada una). En la ver Fig. **2-7** podemos observar dos redes de VDCs construidas sobre dos DCs físicos diferentes; los usuarios 1 y 2 operan en un VDC con la ilusión de operar en un DC físico dedicado. El usuario 3 trabaja en un VDC independiente sin ningún tipo de interferencia con el VDC 1, a pesar de estar alojado en la misma infraestructura física, inclusive compartiendo algunos servidores, conmutadores y enrutadores.

Para que la implementación de las técnicas de virtualización sea posible en un ambiente de computación en la nube, se necesitan conmutadores y enrutadores que soporten la división de sus recursos por virtualización. Algunos ejemplos son: el enrutador "Cisco Cloud Services Router 1000V"[5], el conmutador "Cisco Nexus 1000V" [6], software virtuales implementados en Linux (e.g. open vSwitch[7]) o las soluciones ofrecidas por Juniper [8].

[5]Cisco Cloud Services Router 1000V, http://www.cisco.com/en/US/products/ps12559/
[6]Cisco Nexus 1000V Series Switches for VMware, http://www.cisco.com/en/US/products/ps9902/
[7]Open vSwitch, http://openvswitch.org/
[8]Juniper Networks, http://www.juniper.net/us/en/

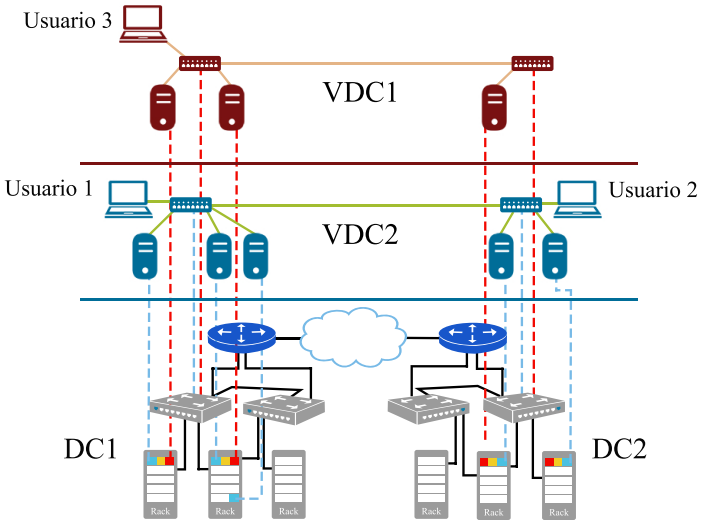

Figura 2-7: Asignación de VDCs en dos centros de datos físicos

2.2. Descripción del problema.

A continuación se describen los conceptos introductorios para el problema de VDCE y se propone una taxonomía para la clasificación de las diferentes propuestas que lo abordan.

2.2.1. Descripción

El problema de asignar los recursos necesarios demandados por un VDC sobre una infraestructura física se conoce como problema de asignación de recursos en DCs virtualizados. Las peticiones para asignar un nuevo VDC pueden llegar en distintos momentos y duran un tiempo finito en el sistema. Es por esto que se considera el problema de asignar recursos como un problema en línea y una característica que deben tener los algoritmos diseñados para resolverlo, es que cuando se realiza una asignación de un VDC, el centro de datos virtualizado debe quedar en un estado que facilite la inserción de nuevas VDCs, es decir, que se maximice el número de VDCs asignadas mientras que se optimizan los recursos restantes para mapear las VDCs subsiguientes. Además se debe considerar que un nodo virtual solo puede ser asignado a un nodo físico, de forma más específica, un servidor virtual solo puede ser asignado a un servidor físico (ver Fig. **2-7**); el mismo razonamiento aplica para los conmutadores y enrutadores. Un enlace virtual es asignado a un conjunto de enlaces físicos (i.e. camino) que comunican los dos nodos físicos, en donde están alojados los dos nodos virtuales que se quieren conectar.

Aunque ya se han estudiado modelos para resolver el problema de la asignación de las redes virtuales en las redes de datos tradicionales [17], éstos no pueden ser aplicados directamente

por varias razones; mientras una red virtual puede estar hecha en su mayoría por decenas de nodos (mayormente enrutadores virtuales y sus enlaces), un VDC puede comprender nodos virtuales de diferentes tipos (servidores, dispositivos de almacenamiento, enrutadores y conmutadores) [3]. El problema de asignar recursos en redes de datos es NP-completo [17], es decir, el tiempo de convergencia para hallar la solución óptima es muy alto. Como el VNE es un caso particular del VDCE, entonces éste también es NP-completo [18].

2.2.2. Principales métricas de desempeño

Para evaluar el desempeño de los algoritmos que resuelven el VDCE, es necesario definir un conjunto de métricas que permitan realizar una comparación objetiva. A continuación se muestran las más relevantes.

Porcentaje de aceptación

Esta métrica mide la capacidad de un algoritmo de aceptar las peticiones de VDCs. Su resultado se obtiene dividiendo la cantidad de VDCs aceptadas sobre la cantidad total de VDCs recibidas.

Utilización de la red

Se refiere a la cantidad de ancho de banda consumido por los VDCs asignados, dividido por la capacidad de ancho de banda total en el DC.

Utilización recurso de cómputo

Se refiere a la cantidad de un recurso de cómputo (i.e. CPU, memoria RAM) consumido por los VDCs asignados, dividido por la capacidad total de ese mismo recurso en el DC.

Ingresos

Es la medida de las ganancias obtenidas por el proveedor de infraestructura. Tiene una relación directa con el número de recursos asignados a los clientes.

Costos

Es la medida de los costos necesarios por el proveedor de infraestructura para asignar y mantener los VDCs.

Energía consumida

Mide el consumo energético del DC en cierto instante del tiempo. Este consumo va a depender principalmente del número de dispositivos encendidos en el DC.

2.2.3. Objetivos

El VDCE es un problema de optimización. Los principales objetivos que se han propuesto en la literatura son: 1) maximizar el número de VDCs asignadas en un centro de datos físicos, 2) disminuir los costos de asignar un VDC, 3) disminuir los costos de comunicación en la red del DC, 4) minimizar el consumo energético, asignar recursos de tal forma que los VDCs asignados sean tolerantes a fallos.

2.2.4. Complejidad de los algoritmos

Como se había mencionado anteriormente el VDCE es NP-completo. Con base en esto se pueden clasificar los algoritmos propuestos para resolver el problema desde tres enfoques diferentes: algoritmos exactos, algoritmos heurísticos y algoritmos metaheurísticos.

Algoritmos exactos

Estos algoritmos se implementan por medio de técnicas de programación matemática [19]. Aunque proporcionan la solución óptima al problema, se vuelven intratables a medida que el tamaño de sus instancias aumenta. Por lo tanto, estos algoritmos son utilizados para solucionar pequeñas instancias del VDCE en tiempos razonables.

Algoritmos heurísticos

Mapear las VDCs entrantes en un centro de datos de manera rápida es esencial para su buen desempeño. Si intentamos aplicar un algoritmo exacto en una red de centros de datos nos encontramos que los tiempos para encontrar la solución óptima son exageradamente altos. Es por esto que es necesario implementar algoritmos con un equilibrio entre la solución óptima y el tiempo de ejecución para asignar el recurso. Los algoritmos heurísticos permiten solucionar el problema con tiempos de ejecución cortos (incluso para grandes instancias), pero no aseguran que la solución encontrada sea la óptima.

Algoritmos metaheurísticos

Son algoritmos iterativos que van mejorando una solución inicial en cada iteración. Al alcanzar algún parámetro de convergencia definido se escoge la mejor entre las soluciones propuestas. Aunque tardan un tiempo superior a los algoritmos heurísticos, muchas veces logran acercarse más al óptimo [20].

2.2.5. Parámetros

Desde el punto de vista de la asignación de recursos las capacidades de los elementos de los DCs son fragmentadas para obtener las instancias individuales que conformaran los VDC

individuales. Los elementos que generalmente se tienen en cuenta son los servidores, enrutadores, conmutadores y los enlaces.

Servidores

Los servidores se fragmentan en instancias muy conocidas llamadas máquinas virtuales, las capacidades que comúnmente se tienen en cuenta para los procesamientos de asignación de recursos son: unidad central de procesamiento (CPU), capacidad de disco duro, memoria de acceso aleatorio (RAM).

Conmutadores y enrutadores

Al igual que los servidores, los conmutadores tienen capacidades de hardware que pueden ser fragmentadas. En los algoritmos de VDCE se están empezando a tener en cuenta estas restricciones. Estos pueden ser el número de puertos virtuales y la capacidad de buffer de cada puerto; las capacidades de procesamiento (CPU) y de memoria (RAM) también pueden ser consideradas.

Enlaces

Se tiene en cuenta el ancho de banda y en algunos casos el retraso y el jitter.

2.3. Estado del arte

2.3.1. Taxonomía para el VDCE

A continuación se propone una taxonomía para la clasificación de las diferentes propuestas que abordan el VDCE. Para cada grupo que se defina a continuación, las dos características son mutuamente excluyentes.

Dinámico vs estático

Análogamente a las redes virtuales, los DCs tienen que ser abordados como un problema en línea. En las redes virtualizadas los requerimientos no se conocen con antelación, además llegan en forma dinámica y se quedan en el sistema por una cantidad de tiempo arbitraria. A medida que el tiempo pasa, los VDCs que van dejando el sistema están continuamente liberando recursos. Definimos la diferencia entre una aproximación dinámica y otra estática en que la última no considera volver a asignar los recursos una vez mapeados los VDCs, y la primera considerará reasignarlos. La principal razón es que el cambio en las condiciones de la red del DC podrá reformar la asignación de recursos en una forma aún mejor de lo que en un principio el algoritmo de solución lo consideró.

Individual vs distribuido

Se considera un VDCE individual si el centro de datos se asigna a un sólo centro de datos físico. Un VDCE distribuido divide el VDC en varia partes, y asigna cada una de ellas a un centro de datos físico. Este procedimiento se realiza de manera distribuida.

Conciso vs redundante

Si una petición de VDCs reserva recursos adicionales para recuperarse de posibles fallos se considera redundante. Si en cambio, solo reserva lo que necesita para su funcionamiento se considera conciso.

2.3.2. Notación esquemática para la taxonomía

De acuerdo a la taxonomía presentada anteriormente un algoritmo de solución al problema de VDCE debe estar clasificado por tres características de los ítems presentados anteriormente. Así se define la siguiente sintaxis:
$[D|E]/[I|D]/[C|R]$
La primera letra denotara si es Dinámico o Estático, la segunda si es Individual o Distribuido y la tercera si es Conciso o Redundante. Así por ejemplo: $D/I/R$ será un algoritmo Distribuido, Individual y Redundante. Esta notación se tomó de [17].

2.3.3. Clasificación

A continuación se presentan los trabajos que han abordado la asignación de recursos centros de datos virtualizados junto con una tabla comparativa usando la taxonomía anteriormente propuesta.

Guo et al. [16]

Los autores proponen una arquitectura llamada SecondNet para implementar un centro de datos virtualizado, que se enfoca en garantizar ancho de banda y diferenciación de servicios. Dentro de esta arquitectura se incluye un algoritmo para la asignación de servidores y enlaces virtuales que constituye el primer intento para resolver el problema VDCE.
El algoritmo de asignación considera la migración de los servidores para casos de fallas o para mejorar la capacidad de la red del centro de datos, también permite aumentar o disminuir el uso de recursos que inicialmente se habían reservado por una petición. El algoritmo de asignación consta de tres etapas principales. En la primera etapa de la asignación se agrupan los servidores físicos en grupos de distintos tamaños, gracias a esto se logran bajos tiempos de convergencia pues en vez de buscar en cada servidor para cada una de las peticiones, se buscan los grupos apropiados. Esto además reduce los costos de comunicación debido a que un VDC debe quedar asignado a un solo grupo, que se caracteriza porque los servidores

que lo conforman están ubicados a un límite de saltos. En una segunda etapa se asignan los servidores virtuales en el grupo escogido y en la tercera etapa se asignan los enlaces virtuales comenzando por los de mayor demanda de ancho de banda utilizando el método del camino más corto.
Se realizaron simulaciones en las topologías Bcube, Fat-tree y VL2 logrando resultados como: tiempos de asignación bajos, la migración de los servidores virtuales mejora la utilización de la red y tiempos rápidos de convergencia para la expansión de recursos de los VDCs y el manejo de fallos.

Ballani et al. [2]

"oktopus" es una arquitectura propuesta por los autores para la implementación de VDCs que busca mantener un equilibrio entre las garantías de desempeño ofrecida a los clientes, sus costos e ingresos del proveedor. En esta propuesta se exponen dos topologías para las peticiones de los VDCs llamadas abstracciones. La primera denominada "virtual cluster" que es similar a topología estrella y la segunda "virtual oversuscribed cluster" , una estructura más compleja que se puede aproximar a varios "virtual clusters" interconectados.
En Oktopus se emplean dos algoritmos diferentes para asignarlas abstracciones propuestas en la red física, y se emplea uno dependiendo de la abstracción. El primer algoritmo trabaja para topologías físicas tipo árbol donde la capacidad de cada servidor es representada como el número de "espacios disponibles", que equivale al número de servidores virtuales que puede alojar. El objetivo del algoritmo es asignar cada servidor virtual en alguno de los espacios disponibles. Para el caso de los virtual clusters, se busca determinar subárboles en la estructura física del DC, que puedan soportar los requerimientos de las peticiones. El "over suscribed cluster" es la combinación de varios virtual clusters, por lo tanto, el algoritmo consiste en la asignación repetitiva de varios virtual clusters.
Para evaluar el desempeño del algoritmo de asignación, se implementa una simulación en donde se utiliza una topología tipo árbol para la red física; se emplea un algoritmo de referencia que asigna los servidores virtuales tan cerca como sea posible. Se evalúan varios aspectos, pero referente a la asignación de recursos el algoritmo propuesto logra un buen desempeño en número de VDCs aceptadas y en mejora de los ingresos del proveedor.

Xin et al. [21]

En esta propuesta se estudia el problema de asignación de recursos en una red de proveedores de servicios de computación en la nube (soportados por DCs), conectados por una red de tránsito central administrada por varios proveedores.
El algoritmo propuesto recibe una petición de VDC, luego realiza la división de ésta para asignar cada parte en los DCs disponibles, también asigna los enlaces necesarios para conectar las particiones usando la red de tránsito. El algoritmo selecciona la opción del menor costo; el costo para asignar una VDC está directamente relacionado con el balanceo de cargas

entre los DCs y el número de proveedores de tránsito que atraviesa (cuando la petición es dividida). El algoritmo se enfoca en la división y asignación de recursos en la red central, la asignación en los DCs locales no es considerada (se asume que los proveedores individuales completan el proceso de asignación local). En la representación de la red física se considera la capacidad de los DCs en términos de servidores virtuales y la capacidad de los proveedores de redes de tránsito representada por el número y ancho de banda de los enlaces disponibles. El algoritmo es implementado en ORCA (Open Resource Control Architecture)[9]. ORCA es un software para el manejo de infraestructuras físicas compartidas que puede contener cualquier combinación de servidores, dispositivos de almacenamiento, redes entre otros componentes. En esta propuesta se no se evalúa el desempeño del algoritmo de asignación, sino que simplemente se presenta su implementación.

Rabbani et al. [18]

En esta propuesta se busca mejorar la aceptación de VDCs generando incrementos en los ingresos del proveedor. Aquí, los autores introducen en el proceso de asignación los conmutadores virtuales y dispositivos de almacenamiento que no se habían tenido en cuenta en anteriores propuestas ([2], [16]), también se pueden definir diferentes clases de recursos para un mismo dispositivo virtual.
El algoritmo de asignación consta de tres fases. En una primera fase se asignan los servidores virtuales a los servidores físicos con capacidad de soportarlos; teniendo en cuenta como principal criterio la proximidad entre ellos. En la segunda fase se asignan los conmutadores virtuales a los físicos tomando en consideración la asignación previa de servidores virtuales. Finalmente en la fase 3 se asignan los enlaces virtuales entre los nodos virtuales ya asignados. El desempeño de este algoritmo es comparado con un algoritmo de referencia, que asigna todos los elementos de la petición en forma aleatoria. Tanto el algoritmo propuesto como el de referencia son evaluados mediante simulaciones en topologías físicas VL2; las peticiones virtuales, tienen la forma de VL2, estrella o combinaciones de éstas. Los objetivos logrados por el algoritmo propuesto con respecto al aleatorizado son: la reducción de la fragmentación en los servidores, minimización de los costos, aumento del ancho de banda residual de red aumentado y carga balanceada en la red física del DC.

Xu et al. [22]

Es una propuesta caracterizada por dos objetivos principales: a) permitir la reserva de recursos adicionales para permitir sobreponerse a un fallo y b) reducir el número de servidores activos (consolidación). Se proponen dos algoritmos para la solución del VDCE cuyo fin es la asignación de servidores virtuales, enlaces virtuales y los elementos de respaldo. El primero consiste en una subrutina heurística para la asignación de los servidores virtuales y otra para los enlaces virtuales, esta última es caracterizada por un algoritmo exacto con una

[9]Open Resource Control Architecture, https://geni-orca.renci.org/trac/wiki/orca-introduction

complejidad computacional baja.
Una segunda propuesta considera un algoritmo único para resolver los dos problemas en concurrencia. Una importante consideración es que un servidor virtual y su respaldo no deben estar en el mismo servidor físico. Para generar un algoritmo de referencia que permita evaluarlos resultados de la propuesta se emplea el método de primer ajuste decreciente para la asignación de los servidores virtuales, que consiste en asignarlos en orden decreciente respecto a sus requerimientos de hardware; para la asignación de enlaces se utiliza el método del camino más corto. Las dos propuestas son evaluadas en las topologías físicas Fat-Tree y VL2, con resultados que permiten reducir el ancho de banda reservado por los VDCs, ofreciendo el mismo rendimiento en número de servidores activos, lo que implica que los VDCs asignados consumen menor ancho de banda con el mismo número de máquinas activas. Los datos para realizar la simulación son tomados del centro de datos ecológico de la universidad de Syracusse[10].

Zhani et al. [23]

Es un modelo dinámico que busca reducir los costos de energía. Al igual que [18], además de la asignación de servidores y enlaces virtuales también se tienen en cuenta los conmutadores como parte de este proceso. Se propone un algoritmo heurístico que considera técnicas y costos de migración de los servidores virtuales.
En esta propuesta dinámica se permite aumentar ó disminuir el uso de recursos que inicialmente se habían reservado por una petición; además, se tiene en cuenta el hecho de que los VDCs están continuamente entrando y saliendo del sistema; por esta razón se considera la posibilidad de ser reasignar los VDCs. De este modelo se derivan ciertas ventajas como lograr ingresar nuevas VDCs cuando el sistema esta congestionado (ajustando previas VDCs para aceptar las nuevas) o en tiempos de baja congestión es posible consolidar los VDCs en algunos nodos de la red física y lograr minimizar el costo energético. El desempeño del algoritmo se evalúa simulando el algoritmo en una topología VL2 comparándolo con un algoritmo similar al diseñado en SecondNet [16]; en el que además, no se considera la migración de los servidores virtuales ni el proceso de consolidación. Los resultados demuestran una evidente mejoría en porcentaje de aceptación, ganancias y el número de máquinas activas.

Amokrane et al. [15]

"Greenhead" es una propuesta que considera varios DCs distribuidos geográficamente a través de una red central. Sus objetivos principales son: a) reducir la carga en la red central (básicamente ubicando las máquinas virtuales cuya comunicación consume gran ancho de banda en un mismo DC); b) reducir las emisiones de huella de carbono (mediante la utilización preferente de fuentes de energías renovables) y c) maximizar los ingresos del proveedor (mediante una asignación eficiente de recursos que permita incrustar la mayoría de VDCs

[10]Green data center at syracusse university, http://syr.edu/greendatacenter/

posibles). También vela por la reducción de costos de energía teniendo en cuenta la preferencia a tarifas regionales de energía y prefiriendo a los DCs que hagan uso eficiente de la misma.
El algoritmo heurístico de asignación consta de un controlador central encargado de dividir cada VDC entrante, monitorear todos los parámetros del DC, asignar las particiones a los DCs locales y asignar los enlaces en la red central. También consta de los controladores locales que se encargan de la asignación de las divisiones en forma local, usando diferentes clases de algoritmos para la asignación de recursos (e.g. [23], [16], [2]).
Para evaluar Greenhead, se simula un algoritmo de referencia que no divide las solicitudes de VDC, sino que considera cada servidor virtual como una partición. La topología NSFNET es la escogida para la red central que conecta 4 centros de datos distribuidos en 4 ciudades diferentes (con diferentes estados climatológicos y diferentes precios por consumo de energía). Bajo una cantidad considerable de variables, el algoritmo obtiene resultados como: solución cercana a la óptima con un tiempo computacional razonable para pequeñas instancias del problema, mejora de la utilización de la red central, incremento de porcentaje de aceptación de VDCs, alto uso de energía renovable y minimización de emisión de huella de carbono por VDC.

Rabbani et al. [24]

Con esta propuesta se busca garantizar una alta disponibilidad de los VDCs, al tiempo que se minimizan los costos de operación. Se considera la heterogeneidad de fallos en la red que implica una probabilidad de fallo distinta en los dispositivos físicos y se plantea una técnica para calcular la disponibilidad de un VDC que depende de la frecuencia de fallos de los equipos donde se aloja y de los recursos de respaldo asignados.
El algoritmo de asignación es efectuado en dos etapas. En la primera etapa se asignan los servidores virtuales en donde se busca preferiblemente asignar los servidores físicos ya activos y con mayor disponibilidad, de esta forma se busca reducir el número de máquinas activas en la red y disminuir el número de recursos de respaldo. En este proceso también se asignan sistemáticamente los recursos requeridos de respaldo para satisfacer la disponibilidad. En la siguiente etapa se realiza la asignación de enlaces y conmutadores en forma conjunta, buscando reducir los costos de comunicación.
Para evaluar este modelo la asignación es probada en las topologías VL2 (red física) y estrella (peticiones de VDC). Se plantea un algoritmo de referencia en el que los servidores virtuales son esparcidos en los servidores físicos para mejorarla disponibilidad y se asignan los recursos de respaldo en los servidores físicos aleatoriamente. Se hacen comparaciones en términos de ganancias, de número de VDCs aceptados, número de máquinas activas, costo de los recursos de respaldo y utilización de recursos por los VDCs, donde el algoritmo propuesto consigue un mejor desempeño en todos los aspectos.

Zhang et al. [25]

Los autores proponen un sistema de asignación llamado venice que está orientado a facilitar VDCs confiables. venice proporciona una técnica para calcularla disponibilidad de un VDC basado en características de fallo de los dispositivos físicos y de la dependencias entre los elementos que lo conforman. Considera la migración de servidores para mejorar la calidad del proceso de asignación. El objetivo de venice es minimizar el número de máquinas activas (costos de energía), los costos de migración de los servidores virtuales y los costos por no disponibilidad.
En el algoritmo de asignación se distinguen dos procesos; en uno se realiza la asignación de VDCs, y en el otro la consolidación de los VDCs. En la etapa de asignación por cada VDC se busca un equilibrio entre el uso del número de máquinas activas y la mejora de la disponibilidad por el uso de máquinas diferentes, así como la formación de grupos de réplica (grupos de respaldo a los fallos) para mejorar la disponibilidad al nivel requerido. A medida que los VDCs salen y entran del sistema, la red se desfragmenta y la asignación se aleja del óptimo, es por esto que en la segunda etapa y apoyados en la migración, se mejora la disponibilidad de los VDCs y se consolidan los servidores en caso de ser posible, para ahorrar energía y ancho de banda.
Para evaluar el desempeño del algoritmo, se toma como algoritmo de referencia el implementado en [23], que no utiliza información de disponibilidad. Los algoritmos son implementados en la topología VL2 (para la red física). Se realizan dos tipos de comparaciones un caso sin migración y sin consolidación y otro caso con migración y consolidación. Se implementa una primera comparación en términos de disponibilidad, donde se observa que venice tiene mejor desempeño en los dos casos. La disponibilidad en el primer caso es mejor que en el segundo, pero ésto se debe a que cuando se implementa migración y consolidación se aceptan más VDCs. También se comparan en términos de penalizaciones por incumplimiento de servicios y ganancias, observando un mejor desempeño por parte de venice.

2.3.4. Tabla comparativa

La tabla 2-1 resume las principales características de cada una de las propuestas anteriormente descritas. Se muestra la referencia del artículo publicado, tipo de algoritmos implementados para resolver el modelo de optimización, las topologías para VDCs incluidas y finalmente la principal contribución.

2.4. Direcciones emergentes

A continuación se hace una discusión de algunas de las tendencias y potenciales líneas de investigación detectadas a partir del análisis de las propuestas mencionadas en la sección 2.3.

Tabla 2-1: Propuestas que consideran resolver el problema de VDCE

Referencia	Categoría	Optimización	Topologías	Principal contribución
Rabbani et al.	E/I/C	Heurística	VL2, estrella	Introduce en el proceso la asignación de los conmutadores virtuales y diferentes clases de recursos para un mismo dispositivo virtual.
Zhani et al.	D/I/C	Heurística	VL2	Considera escalamiento de las VDCs y consolidación
Amokrane et al.	E/D/C	Heurística	NSFNET	Considera una red de DCS distribuidos y el uso de fuentes de energías renovables.
Xu et al.	E/I/R	Heurística	FatTree,VL2	Tolerancia a fallos y consolidación simultáneamente
Xin et al.	E/D/C	Heurística	N/A	Primera propuesta DCs distribuidos
Guo et al.	D/I/C	Heurística	Fattree, Bcube, VL2	Primera propuesta para resolver el problema de VDCE
Ballani et al.	E/I/C	Heurística	Árbol	Propuesta pionera en la VDCE. Manejo de abstracciones
Zhang et al.	E/I/R	Heurística	VL2	Provee VDCs confiables y consolidación.
Rabbani et al.	D/I/R	Heurística	VL2	Tolerancia a fallos y Consolidación para los VDCs

2.4.1. Tipo de algoritmos

Según la revisión realizada, todas las propuestas existentes para resolver el VDCE se han orientado a proveer soluciones con algoritmos heurísticos al problema de la asignación de recursos, con el fin de lograr tiempos de convergencia normalmente cortos. Hasta ahora solo [22] considera un algoritmo exacto para resolver una parte del problema (la asignación de enlaces), que busca garantizar suficiente ancho de banda en casos de fallos obteniendo el mínimo de ancho reservado por cada VDC. Nos parece que una interesante rama de investigación es el plantear algoritmos exactos que permitan evaluar el desempeño de las heurísticas planteadas para pequeñas instancias del problema; en [15] se compara la función objetivo con la obtenida por un algoritmo exacto para una instancia pequeña del problema.
Hasta ahora no se han considerado algoritmos metaheurísticos para resolver el problema de VDCE. Investigar sobre qué tipos de metaheurísticas son aplicables a la asignación de recursos en los DCs como por ejemplo recocido simulado [20], en donde se estudie el tiempo de convergencia y la calidad de la solución para diferentes instancias del problema, es una línea de investigación que proporcionará una mejora en el desempeño de estos algoritmos. Necesariamente se debe evaluar el desempeño con respecto a las soluciones heurísticas existentes, para determinar si la complejidad introducida por las soluciones metaheurísticas compensa la calidad de la soluciones obtenidas.

2.4.2. Tolerancia a fallos y ahorro de energía

Una prometedora rama de investigación futura es la que trata de asignar los VDCs buscando optimizar el compromiso entre consumo energético y tolerancia a fallos. Muchas de las aplicaciones y servicios que se ejecutan en un centro de datos virtualizado requieren una gran fiabilidad, por lo que un fallo en la red conllevaría un daño irreversible. A su vez, motivados por los altos costos producidos por la energía consumida, los administradores de los DCs han tratado de buscar estrategias que les permitan operar sus aplicaciones y servicios tratando de minimizar al máximo el consumo energético. En tiempos de baja carga es posible agrupar las VDC en pocos elementos físicos del centro de datos, de esta forma podemos apagar los que quedan sin funcionar (consolidación) reduciendo el consumo de energía, esto implica ventajas que son tanto ecológicas como financieras. Implementar tolerancia a fallos implica reservar recursos extras lo que aumenta los costos operacionales, es por esto, que es necesario investigar métodos que traten de optimizar el compromiso entre tolerancia a fallos y costos operacionales.
Hasta ahora, las soluciones propuestas en [22], [24] y [25] consideran la tolerancia a fallos y la consolidación pero no dan una idea del costo energético que hay que sacrificar para proveer tolerancia a los fallos simultáneamente. El trabajo propuesto en [23] considera la consolidación, sin embargo, no estudian la protección a posibles fallos en su propuesta. Nuestra investigación mediante simulaciones y formulando un modelo matemático de optimización estudia el desempeño en la minimización del consumo energético, y la supervivencia al fallo de un servidor de la red de un centro de datos físico; en donde el problema de VDCE es resuelto en forma coordinada, es decir, los recursos de respaldo y la petición original son asignadas simultáneamente asegurando una solución óptima, a diferencia de las demás propuestas similares en que la asignación es hecha en dos fases.

2.4.3. Asignación de recursos en centros de datos distribuidos

El VDCE distribuido hace referencia a la asignación de un VDC en varios centros de datos virtualizados. La complejidad del problema de asignación de recursos aumenta considerablemente con ellos; pero al mismo tiempo genera oportunidades para lograr mejores objetivos de desempeño. Greenhead [15] es un excelente modelo que considera el problema VDCE distribuido buscando objetivos como aumento de ingresos, optimización del uso de la red, eficiencia en energía y técnicas ecológicas. Otro trabajo que considera centros de datos distribuidos es [26] que busca distribuir la carga de operación entre los diferentes DCs. Como podemos notar pocos trabajos abordan el VDCE en DCs distribuidos. La complejidad para asignar recursos es notablemente alta debido a que el número de variables implicadas es mucho mayor que cuando se considera un centro de datos individual. Nuevas oportunidades de investigación se pueden encontrar al considerar DCs distribuidos enfocados en satisfacer un objetivo o varios de ellos.

2.4.4. Asignación de recursos en centros de datos ecológicos

Existe una gran presión a nivel mundial para la creación de DCs que sean amigables con el medio ambiente. Una investigación reciente reportó que en el año 2012 la huella de carbono de los DCs constituía alrededor del 0,25 % de la emisión mundial, lo cual constituye el 10 % de las emisiones de las tecnologías de información y telecomunicaciones[11]. Es por esto que compañías como Google[12] han empezado a emplear los centros de datos virtuales ecológicos. Es posible entonces asignar los VDC o parte de ellos en aquellas zonas que se considere generaría un impacto menos nocivo al medio ambiente. Por ejemplo, al tener un DC ubicado en zona cercana al ártico que pertenece a una compañía con centros distribuidos, se pueden implementar en su algoritmo de VDCE preferencias de asignación de nodos a este centro de datos. Greenhead [15] es la única de las propuestas que estudia el uso de energías renovables en DCs y da un uso preferente a las asignaciones en aquellos donde esté la disponibilidad de fuentes renovables.

[11] Toolkit on Environmental Sustainability for the ICT Sector (ESS), http://www.itu.int/ITU-T/climatechange/ess/index.html

[12] Google Green Data Centers, http://www.google.com/green/efficiency/

3 Formulación del problema

El enfoque de nuestro modelo de optimización trata de balancear el equilibrio entre tolerancia a fallos y minimización del consumo de energía. Si bien, mientras se trata de garantizar la tolerancia esta tiende a distribuir los nodos virtuales en la red de un DC (ver Fig. **3-2**), la consolidación (ahorro de energía encendiendo menos nodos) tiende a agruparlos (ver Fig. **3-1**). En consecuencia, en este capítulo se define el modelo matemático de optimización que representa la operación de un DC cuando se optimizan los recursos activos y de tolerancia al mismo tiempo. Este dilema lleva a una optimización multi-objetivo.

3.1. Descripción del modelo

En nuestra propuesta, una red de un DC físico (ver Fig. **3-1**) se describe como un conjunto de nodos físicos conectados por un conjunto de enlaces físicos. Donde los nodos son servidores físicos y conmutadores. Cada nodo físico tiene una capacidad residual de CPU y cada enlace físico tiene una capacidad residual de ancho de banda.
Las peticiones de VDC están limitadas a la topologías tipo Star (ver Fig. **3-1**), donde hay un conmutador central conectado a todos los servidores, de manera similar a como se ha hecho en previas propuestas [2, 24, 18].
En la Fig. **3-1**, dos peticiones de VDC están siendo asignadas en el DC. La figura muestra como el proceso de asignación ahorra energía, gracias a la consolidación de nodos y enlaces virtuales en un conjunto reducido de nodos y enlaces físicos, permitiendo así apagar o poner en estado inactivo el resto de equipos. Sin embargo, la solución propuesta en la figura no provee supervivencia a fallos; si un servidor físico falla, los servidores virtuales ya asignados en este servidor fallarán; y como en este escenario no se ha reservado espacio extra en la red del DC físico en caso de fallo, la red virtual también fallará.
Para garantizar supervivencia en la red, nuestra solución implementa un grafo auxiliar llamado "grafo de tolerancia" que consiste de un conmutador conectado por dos enlaces a un servidor; estos elementos virtuales representan los recursos extras para la supervivencia. Al conmutador del grafo de tolerancia no se le requiere asignar recursos, pues en este trabajo de investigación no consideramos fallos de conmutadores (será considerado para trabajo futuro), de esta forma los recursos del conmutador de tolerancia ya son asignados en la petición original. Además de esto el conmutador del grafo de tolerancia debe ser asignado en el mismo lugar donde el conmutador de la petición original se asigna, esto es porque estos dos conmutadores son en realidad el mismo y representa la intersección de los dos grafos.

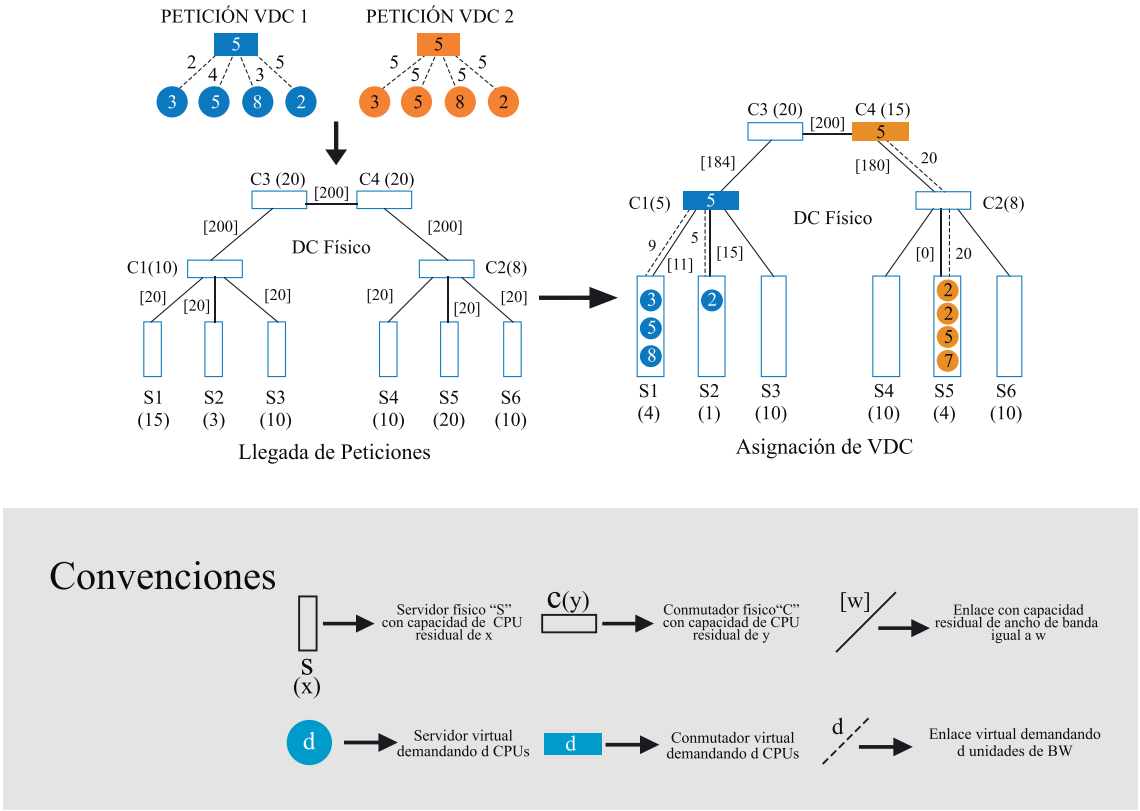

Figura 3-1: Asignación en un DC

Lo anterior es necesario para coordinar los dos grafos, de esta forma el grafo de tolerancia y la petición de VDC pueden ser asignados simultáneamente dando como resultado el grafo aumentado. En la Fig. **3-2** vemos como se coordina la asignación del grafo de tolerancia y de la petición original de un VDC; en el segundo recuadro se produce un grafo aumentado en el que el conmutador de respaldo y el conmutador de la petición original son fusionados en uno. En este punto aún no se sabe la capacidad de tolerancia del nodo de respaldo ni de los enlaces de respaldo. El nodo de respaldo y los servidores de la petición original se conectan al conmutador del grafo aumentado cuya demanda es igual a la demanda del conmutador de la petición original de VDC; el nodo de respaldo se conecta por medio de dos enlaces de respaldo (uno de subida y otro de bajada), los servidores virtuales por medio de las conexiones originales. El tercer recuadro representa la solución de un proceso de asignación realizada a un DC físico como resultado de aplicar el modelo de optimización. De esta forma, se propone extender la petición de VDC, con un nuevo nodo, para garantizar supervivencia a fallos.
El nodo de respaldo puede soportar todos los nodos virtuales que podrían fallar durante un problema inesperado de un nodo físico. Vemos en la Fig. **3-2** que si un servidor físico falla, e.g. el "S2", la capacidad reservada en el servidor S3 (8 núcleos de CPU reservadas

para esta petición) para supervivencia a fallos es suficiente para alojar los 7 núcleos de CPU que se necesitan para reestablecer el VDC. Los enlaces de respaldo también deben estar en capacidad de suplir las conexiones originales que se interrumpen por el fallo. Para simplificar el modelo se asume que solo un nodo físico falla en un instante del tiempo y solo consideraremos recursos de CPUs en los nodos físicos y virtuales como medida única de capacidad.

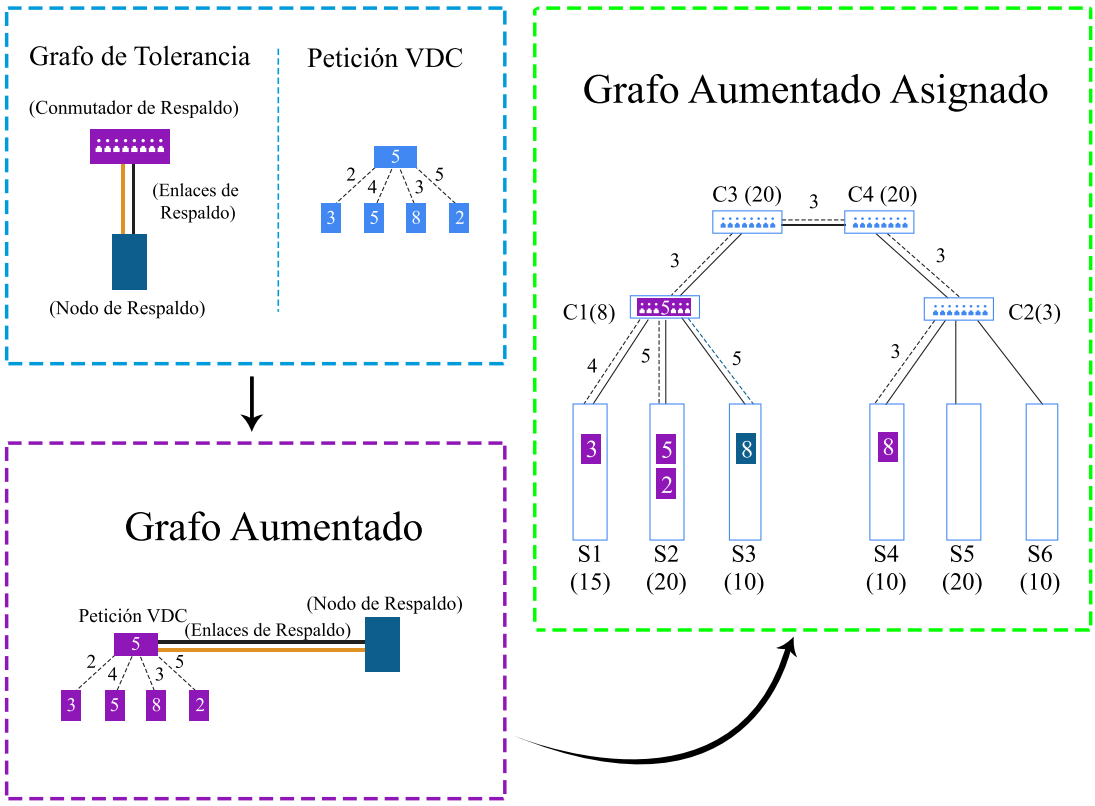

Figura 3-2: Tolerancia a fallos

3.2. Asignación coordinada

Las propuestas existentes [22, 24, 25] que resuelven el VDCE proveyendo tolerancia a fallos, solucionan el problema en dos pasos; 1) Asignan recursos para la petición original y 2) asignan los recursos de respaldo. Este proceso no garantiza la asignación óptima porque el primer y el segundo paso no están coordinados. En nuestra propuesta, el VDCE es resuelto en una forma coordinada: Los recursos de respaldo y la petición original son asignadas simultáneamente asegurando una solución óptima con respecto a un objetivo predeterminado. Nuestro objetivo

básico es minimizar el consumo de energía reduciendo los recursos físicos usados para asignar tanto la petición como los recursos de respaldo virtuales.
La función del nodo de respaldo es alojar todos los servidores virtuales de la petición de VDC cuya asignación no sea funcional en caso que un solo servidor físico falle. El servidor físico escogido en el proceso de asignación para alojar la máxima capacidad virtual de CPU para servidores de una petición, determina la capacidad virtual del nodo de respaldo, pues consideramos que uno o más nodos físicos no pueden fallar simultáneamente. En el ejemplo mostrado en la Fig. **3-2** y Fig. **3-3**, se puede ver como la capacidad máxima de CPU está localizada en el nodo físico $S4$ con una capacidad virtual alojada de 8, entonces la capacidad del nodo virtual debe ser de 8, para prevenir el peor caso de falla (e.g. fallo del nodo físico S4). Lo mismo aplica para la demanda de BW para el enlace de respaldo; en este caso, se determina por el servidor que usa la máxima cantidad de ancho de banda demandado (Ver Fig. **3-3**). Una cuestión importante es que el nodo de respaldo no puede ser asignado al mismo servidor físico que está soportando los servidores virtuales de la petición de VDC. La contribución más importante de nuestra propuesta es que asigna la petición original y, concurrentemente, dimensiona y asigna los recursos de respaldo de forma tal que, mientras se garantiza una tolerancia a fallos del 100 %, se trata de minimizar el consumo energético de la red.
En esta investigación, se desarrolla un modelo de optimización multi-objetivo MIP (siglas en inglés, Mixed Integer Programming) para resolver el VDCE. Una solución factible en un modelo multi-objetivo es un punto eficiente (óptimo de pareto) si ninguna otra solución factible logra mejor desempeño en todas las funciones objetivos y mejor en una, la frontera eficiente (frontera de Pareto) de los modelos multi-objetivo es la colección de puntos eficientes para el modelo [27].

3.3. Formulación multiobjetivo MIP del VDCE

En esta sección, la red física, las peticiones de VDC y el grafo de tolerancia son formalmente descritos. Además, el modelo de optimización MIP para resolver el VDCE es presentado. Este modelo apunta a estudiar el equilibrio entre ahorro de energético y la tolerancia a fallos. Como se detalló anteriormente, de la topología tipo estrella, un nuevo grafo aumentado, con un nodo de respaldo adicionado (protegiendo los servidores virtuales del VDC), es creado para garantizar la supervivencia. Este nodo de respaldo es conectado al conmutador del VDC. Para simplificar el modelo, se asume que solo un nodo físico falla en un instante del tiempo. Todas las entradas y variables usadas en este modelo son explicadas en detalle en la tabla **3-1**.

Término	Descripción
Entradas DC físico	

Continúa en la página siguiente.

Termino	Descripción
$G^p(N^p, E^p)$	Grafo dirigido; N^p y E^p representan nodos físicos y enlaces físicos respectivamente.
$N^p = S^p \cup J^p$	Conjunto de nodos físicos, donde S^p es el conjunto de servidores físicos y J^p es el conjunto de conmutadores físicos.
c_{n^p}	Capacidad residual de CPU del nodo físico $n^p \in N^p$.
b_{e^p}	Capacidad residual de BW del enlace físico $e^p \in E^p$.
$u_{n^p e^p}$	Parámetro booleano que indica si un nodo físico $n^p \in N^p$ es la fuente de un enlace físico $e^p \in E^p$.
$d_{n^p e^p}$	Parámetro booleano que indica si un nodo físico $n^p \in N^p$ es el destino de un enlace físico $e^p \in E^p$.
Entradas Petición VDC	
$G^v(N^v, E^v)$	Grafo dirigido, donde N^v es el conjunto de nodos virtuales y E^vel conjunto de enlaces virtuales.
$N^v = S^v \cup J^v$	conjunto de nodos virtuales donde S^v es el conjunto de servidores virtuales y J^v es el conmutador virtual (El índice 0 es reservado).
c_{n^v}	Demanda de CPU del nodo virtual. $n^v \in N^v$
b_{e^v}	Demanda de BW del enlace virtual. $e^v \in E^v$
$u_{n^v e^v}$	Parámetro booleano que indica si un nodo virtual $n^v \in N^v$ es la fuente de un enlace virtual $e^v \in E^v$.
$d_{n^v e^v}$	Parámetro booleano que indica si un nodo virtual $n^v \in N^v$ es el destino de un enlace virtual $e^v \in E^v$
Entradas grafo de tolerancia	
$G^t(N^t, E^t)$	Grafo dirigido, donde N^t son el conjunto de nodos de respaldo; para el servidor de respaldo el índice 1 es reservado y para el conmutador de respaldo el índice 0 es reservado; E^t son los enlaces de respaldo (conectan el conmutador virtual con el nodo de respaldo)
$u_{n^t e^t}$	Parámetro booleano que indica si un nodo de respaldo $n^t \in N^t$ es la fuente del enlace de respaldo $e^t \in E^t$
$d_{n^t e^t}$	Parámetro booleano que indica si un nodo de respaldo $n^t \in N^t$ es el destino del enlace de respaldo $e^t \in E^t$
Variables	
$x_{n^p n^v}$	Variable booleana que indica si un nodo físico $n^p \in N^p$ tiene asignado el nodo virtual $n^v \in N^v$.
$f_{e^p e^v}$	Variable booleana que indica si un enlace físico $e^p \in E^p$ tiene asignado el enlace virtual $e^v \in E^v$.
$g_{e^p e^t}$	Variable booleana que indica si un enlace físico $e^p \in E^t$ tiene asignado el enlace de respaldo $e^t \in E^t$.
$w_{j^p n^v}$	Variable booleana que indica si un conmutador físico $j^p \in J^p$ es usado para asignar el enlace virtual que conecta el conmutador virtual a el nodo virtual.
$\bar{w}_{j^p n^t}$	Variable booleana que indica si un conmutador físico $j^p \in J^p$ es usado para asignar el enlace virtual que conecta el conmutador virtual a el nodo de respaldo.
y_{n^p}	Variable booleana que indica si la maquina $n^p \in N^p$ esta activa.

Continúa en la página siguiente.

Termino	Descripción
$z_{n^p n^t}$	Variable booleana que indica si el nodo físico $n^p \in N^p$ tiene asignado el nodo de respaldo $n^t \in N^t$.
$ctol$	Variable real positiva que indica la cantidad de CPU que debe ser reservada para el nodo de respaldo con índice 1.
tbw_0, tbw_1	Variable real positiva que indica la cantidad de BW que debe ser reservada para los enlaces de respaldo con índice 0 e índice 1 respectivamente.
aux_{n^p},$aux0_{e^p}$ y $aux1_{e^p}$	Variables reales positivas que ayudan a evitar las no linealidades del problema.

Tabla 3-1: Entradas y variables del modelo matemático

3.3.1. Restricciones

$$\sum_{n^p \in N^p} x_{n^p n^v} = 1, \forall n^v \in N^v \tag{3-1}$$

$$\sum_{n^p \in N^p} z_{n^p n^t} = 1; \forall n^t \in N^t \tag{3-2}$$

Las restricciones (3.1) y (3.2) garantizan que los nodos y enlaces virtuales son asignados en un solo servidor físico.

$$\sum_{n^v \in N^v} c_{n^v} x_{n^p n^v} \leq c_{n^p}, \forall n^p \in N^p \tag{3-3}$$

$$\sum_{e^v \in E^v} b_{e^v} f_{e^p e^v} \leq b_{e^p}, \forall e^p \in E^p \tag{3-4}$$

Las ecuaciones (3.3) y (3.4) son las restricciones de capacidad en términos de CPU y BW.

$$\begin{aligned} w_{j^p n^v} \leq & \sum_{e^p}\sum_{e^v} u_{n^v e^v} f_{e^p e^v} u_{j^p e^p} + \sum_{e^p}\sum_{e^v} u_{n^v e^v} f_{e^p e^v} d_{j^p e^p} + \\ & \sum_{e^p}\sum_{e^v} d_{n^v e^v} f_{e^p e^v} u_{j^p e^p} + \sum_{e^p}\sum_{e^v} d_{n^v e^v} f_{e^p e^v} d_{j^p e^p} \end{aligned} \tag{3-5}$$

$$\begin{aligned} K \cdot w_{j^p n^v} \geq & \\ \sum_{e^p}\sum_{e^v} u_{n^v e^v} f_{e^p e^v} u_{j^p e^p} + \sum_{e^p}\sum_{e^v} u_{n^v e^v} f_{e^p e^v} d_{j^p e^p} + & \\ \sum_{e^p}\sum_{e^v} v_{n^v e^v} f_{e^p e^v} u_{j^p e^p} + \sum_{e^p}\sum_{e^v} d_{n^v e^v} f_{e^p e^v} d_{j^p e^p} & \\ \forall j^p \in J^p; n^v \in N^v & \end{aligned} \tag{3-6}$$

$$\bar{w}_{j^p n^t} \leq \sum_{e^p} \sum_{e^t} u_{n^t e^t} f_{e^p e^v} u_{j^p e^p} + \sum_{e^p} \sum_{e^t} u_{n^t e^t} f_{e^p e^t} d_{j^p e^p} + \sum_{e^p} \sum_{e^t} d_{n^t e^t} f_{e^p e^t} u_{j^p e^p} + \sum_{e^p} \sum_{e^t} d_{n^t e^t} f_{e^p e^t} d_{j^p e^p} \tag{3-7}$$

$$K \cdot \bar{w}_{j^p n^t} \geq \sum_{e^p} \sum_{e^t} u_{n^t e^t} f_{e^p e^t} u_{j^p e^p} + \sum_{e^p} \sum_{e^t} u_{n^t e^t} f_{e^p e^t} d_{j^p e^p} + \sum_{e^p} \sum_{e^t} v_{n^t e^t} f_{e^p e^t} u_{j^p e^p} + \sum_{e^p} \sum_{e^t} d_{n^t e^t} f_{e^p e^t} d_{j^p e^p} \quad \forall j^p \in J^p; n^t \in N^t \tag{3-8}$$

Las variables $w_{j^p n^v}$ y $\bar{w}_{j^p n^t}$ que están definidas en las ecuaciones (3.5) a la (3.8) son usadas para ayudar a definir cuando un dispositivo está apagado o encendido; está encendido cuando contiene un nodo virtual o cuando sirve como puente entre dos nodos virtuales.

$$\sum_{e^p \in e^p} u_{n^p e^p} f_{e^p e^v} - \sum_{e^p \in e^p} d_{n^p e^p} f_{e^p e^v} = \sum_{n^v \in n^V} u_{n^v e^v} x_{n^p n^v} - \sum_{n^v \in n^v} d_{n^v e^v} x_{n^p n^v}, \forall, e^v \in E^v, n^p \in N^p \tag{3-9}$$

Eq. (3.9) es la restricción de conservación de flujo para la petición de VDC y asegura que todos los enlaces virtuales son asignados.

Dos nuevos parámetros son definidos aquí. Uno indica si un nodo físico está en capacidad de soportar un nodo virtual (de la petición), y otro define si un nodo físico está en capacidad de soportar un nodo de respaldo. Así $\bar{x}_{n^p n^v}$ es 1 si $n^p \in N^p$ puede alojar un nodo virtual $n^v \in N^v$ y $\bar{z}_{n^p n^v}$ es 1 si $n^p \in N^p$ puede alojar un nodo de respaldo $n^v \in N^v$. Con estos parámetros se pueden formular las restricciones (3.10) y (3.11).

$$x_{n^p n^v} \leq \bar{x}_{n^p n^v}; \forall n^p \in N^p, n^v \in N^v \tag{3-10}$$

$$z_{n^p n^t} \leq \bar{z}_{n^p n^t}; \forall n^p \in N^p, n^t \in N^t \tag{3-11}$$

$$y_{n^p} \geq x_{n^p n^v}; \forall n^p \in N^p, n^v \in N^v \tag{3-12}$$

$$y_{n^p} \geq z_{n^p n^t}; \forall n^p \in N^p, n^v \in N^V \tag{3-13}$$

$$y_{n^p} \geq w_{n^p n^v}; \forall n^p \in J^p, n^v \in N^v \tag{3-14}$$

$$y_{n^p} \geq \bar{w}_{n^p n^t}; \forall n^p \in J^p, n^t \in N^t \tag{3-15}$$

Para definir cuando una maquina física está activa, debemos implementar las restricciones desde la (3.12) hasta la (15). Las Eqs. (3.12) y (3.13) activan un nodo físico si este es usado para alojar uno o más nodos virtuales o de respaldo. Las Eqs. (3.13) and (3.14) activan los conmutadores que son parte de un camino que aloja un enlace virtual o de respaldo.

$$z_{n^p 1} + x_{n^p n^v} \leq 1; \forall n^p \in N^p, n^v \in N^v \tag{3-16}$$

La restricción (3.16) garantiza que el nodo de respaldo es asignado en un lugar diferente al de los nodos virtuales, así se asegura un 100 % de tolerancia a fallos cuando un servidor físico falla.

$$\sum_{n^v \in N^v} c_{n^v} x_{n^p n^v} \leq ctol, \forall n^p \in N^p \tag{3-17}$$

$$aux_{n^p} \leq c_{n^p} \cdot z_{n^p 1}, \forall n^p \in N^p \tag{3-18}$$

$$\sum_{n^p \in N^p} aux_{n^p} = ctol \tag{3-19}$$

Las ecuaciones (3.17), (3.18) y (3.19) determinan la capacidad de CPU del servidor de respaldo y aseguran que el servidor físico donde es asignado, está en la capacidad de alojarlo. Las mencionadas restricciones reemplazan las no linealidades de la restricción: $ctol \cdot z_{n^p 1} \leq c_{n^p}, \forall n^p \in N^p$

$$z_{n^p 0} = x_{n^p 0}; \forall n^p \in N^p \tag{3-20}$$

En (3.20), el conmutador de respaldo en el grafo de tolerancia y el conmutador virtual en la petición de VDC son el mismo, y ambos comparten el mismo índice 0 en sus propios grafos.

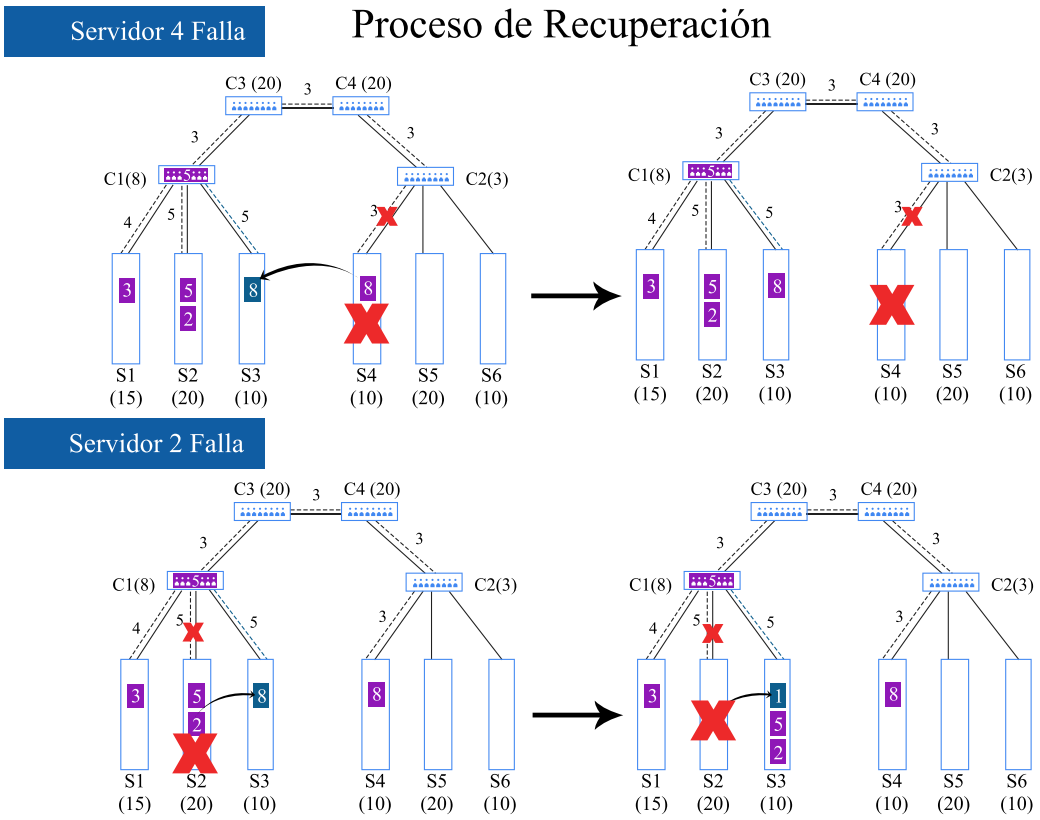

Figura 3-3: Proceso de supervivencia a fallos

La diferencia estará en que el conmutador de respaldo no consumirá recursos de CPU. Esto será útil cuando los enlaces de respaldo sean asignados. En la Fig. **3-3** se puede ver este proceso; los nodos que están en color morado representan la petición original y los nodos azules los recursos de respaldo para esta petición, además se puede ver cuál es el proceso de recuperación cuando fallan dos servidores diferentes y como dicho servidor de respaldo es capaz de soportar dos situaciones de fallo diferentes. Cuando falla el servidor S4, el nodo virtual que tenía asignado es transferido al servidor S3. Cuando falla el servidor S2 los dos nodos virtuales asignados a este son transferidos al servidor S3, que es quien soporta el recurso de respaldo.

$$\sum_{e^p \in E^p} u_{n^p e^t} g_{e^p e^t} - \sum_{e^p \in E^p} d_{n^p e^t} g_{e^p e^t} = \sum_{n^t \in N^t} u_{n^t e^t} z_{n^p n^t} - \sum_{n^t \in N^t} d_{n^t e^t} z_{n^p n^t}, \forall, e^t \in E^t, n^p \in N^p \tag{3-21}$$

La ecuación (3.21) representa la restricción de conservación de flujo para el grafo de toleran-

cia, asegurando que todos los enlaces de respaldo sean asignados.

$$g_{e^p e^t} + f_{e^p e^v} \leq 1; \forall e^p \in E^p, e^v \in E^v e^t \in E^t \tag{3-22}$$

La restricción (3.22) garantiza enlaces disjuntos entre los enlaces de la petición de VDC y los enlaces de respaldo.

$$\sum_{e^p \in E^p} \sum_{e^v \in e^v} u_{n^p e^p} \cdot f_{e^p e^v} \cdot b_{e^v} \leq tbw_0, \forall, n^p \in N^p \tag{3-23}$$

$$aux0_{e^P} \leq sumbw0 \cdot g_{e^p 0}, \forall e^p \in E^p \tag{3-24}$$

$$aux0_{e^P} \leq tbw_0, \forall e^p \in E^p \tag{3-25}$$

$$aux0_{e^p} \geq tbw_0 - sumbw0(1 - g_{e^p 0}), \forall e^p \in E^p \tag{3-26}$$

Las restricciones (3.23)- (3.26) determinan la capacidad del enlace de subida de respaldo. $sumbw0$ es un parámetro que indica la suma de todos los BW de los enlaces de subida en toda la petición de un VDC. Las restricciones (21)-(24) reemplazan las no linealidades de las restricciones de capacidad: $tbw_0 \cdot g_{e^p 0} \leq b_{e^p}, \forall, e^p \in E^p$.

$$\sum_{e^p \in E^P} \sum_{e^v \in e^V} d_{n^p e^p} \cdot f_{e^p e^v} \cdot b_{e^v} \leq tbw_1, \forall, n \in N \tag{3-27}$$

$$aux1_{e^P} \leq sumbw1 \cdot g_{e^p 1}, \forall e^p \in E^p \tag{3-28}$$

$$aux1_{e^P} \leq tbw_1, \forall e^p \in E^p \tag{3-29}$$

$$aux1_{e^p} \geq tbw_1 - sumbw1(1 - g_{e^p 1}), \forall e^p \in E^p \tag{3-30}$$

Las restricciones (3.27)- (3.30) determinan la capacidad del enlace de bajada de respaldo. $sumbw1$ es un parámetro que indica la suma de todos los BW de los enlaces de bajada en toda la petición de VDC. Las restricciones (25)-(28) reemplazan las no linealidades de las restricciones de capacidad: $tbw_1 \cdot g_{e^p 1} \leq b_{e^p}, \forall, e^p \in E^p$.

Función objetivo

El objetivo de nuestro trabajo es minimizar el número de nodos físicos activos y reducir la cantidad de recursos usados para garantizar la supervivencia de la red; estas dos primeras ecuaciones están representadas por los términos $\sum_{n^p \in N^p} y_{n^p}$ y $ctol$ respectivamente. Las variables de ancho de banda $g_{e^p e^t}$, tbw_0, $f_{e^p e^v}$ y tbw_1 están incluidas para garantizar que el modelo de optimización escoja el camino más corto disponible por cada petición; estos términos son multiplicados por pequeño factor porque no son nuestro principal objetivo.

$$\begin{aligned} Minimizar(\sum_{n^p \in N^p} y_{n^p} + ctol + 0{,}001 \cdot (\sum_{e^p \in E^p} \sum_{e^t \in E^t} g_{e^p e^t} \\ + \sum_{e^p \in E^p} \sum_{e^v \in E^v} f_{e^p e^v} + tbw_0 + tbw_1)) \end{aligned} \tag{3-31}$$

Cuando se tiene un problema multi-objetivo, la función objetivo consta de más de un resultado que se puede catalogar como no dominado (otro resultado no es mejor que el no dominado). Al obtenerse varios resultados se debe escoger un resultado como mejor bajo ciertos criterios predeterminados; la técnica más popular es el método de los pesos, que consiste en multiplicar cada función objetivo por un factor de importancia; el problema con este método radica en que escoger el valor de ese factor de importancia se convierte en un nuevo problema de optimización. Es por esto que para resolver el modelo formulado usaremos el método de las ϵ-restricciones. Esta técnica consiste en convertir un problema multi-objetivo en uno mono-objetivo, seleccionando una función objetivo como primario y las otras como restricciones asociadas con un conjunto de parámetros llamados épsilon. En nuestro experimento la tolerancia será nuestro máximo objetivo y las maquinas activas las nuevas restricciones. En el próximo capítulo se ampliará este concepto y de cómo se aplicó para obtener los resultados finales.

4 Evaluación de desempeño

En este capítulo se estudiara el modelo planteado anteriormente utilizando simulaciones basadas en el entorno ALEVIN [28] y generando un escenario de un DC.

4.1. Condiciones experimentales y selección de parámetros

En esta sección se definen los parámetros básicos para realizar el análisis del modelo, planteando un escenario de un DC como una instancia pequeña del problema VDCE, pues el modelo de optimización que se evalúa es MIP, es por esto que los algoritmos que resuelven dicho modelo no son escalables para instancias grandes del problema; por lo tanto, se harán las simulaciones con una instancia pequeña que permita obtener resultados en tiempos razonables.

4.1.1. Parámetros DC físico

La topología escogida para realizar la simulación de un DC físico fue la distribución tipo VL2 [10] (Ver sección 2.1.1), pues es usada en la mayoría de propuestas referenciadas en el capítulo 1 que estudian el problema de VDCE, así se facilita realizar la evaluación del desempeño. VL2 mejora la redundancia en las conexiones (enlaces) que deben existir en un DC físico. La forma como están conectados los nodos físicos de la red se muestra en la Fig. **4-1**; cabe aclarar que cada línea que conecta dos nodos representa un enlace de subida y otro de bajada. El problema VDCE es NP-completo [18], así para grandes instancias del problema el tiempo de ejecución del modelo puede darse en el orden de semanas o incluso meses. Por lo tanto, en este proyecto se plantea una instancia pequeña del problema que consiste en tres capas de conmutadores. En el primer nivel están los servidores conectados a los conmutadores TOR (Siglas en inglés, Top of Rack), a cada TOR se conectan 4 servidores para un total de 16 servidores físicos en toda la red del DC físico; en el segundo nivel están los conmutadores intermedios, cada TOR está conectado a dos conmutadores intermedios y en el último nivel están 2 conmutadores de núcleo que interconectan los 4 conmutadores intermedios (ver Fig. **4-1**). En resumen se tiene un total de 10 conmutadores y 16 servidores que conforman la red y que están conectados por enlaces de alta capacidad (1000 Mbps de BW). Cada servidor físico y conmutador tiene una capacidad de 8 núcleos de CPU. Los

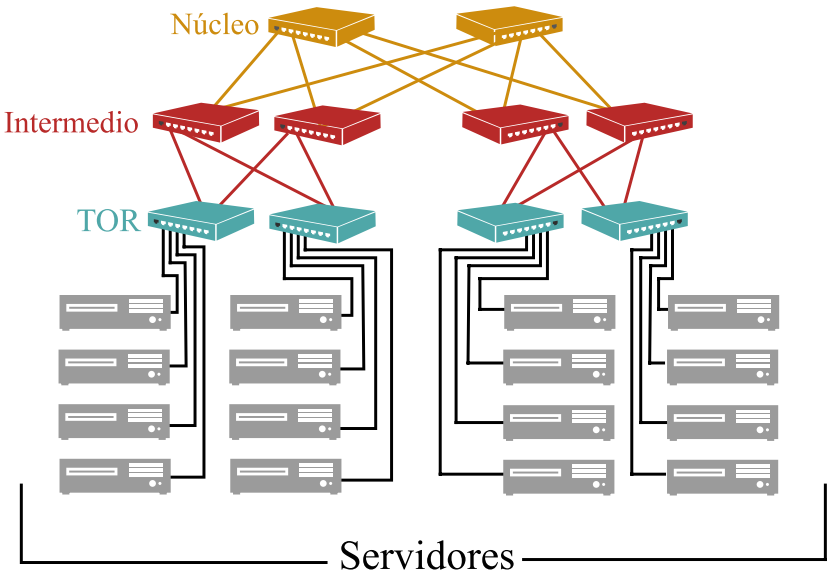

Figura 4-1: Tolerancia a fallos

parámetros y distribución de nodos y enlaces del DC físico, fueron basados en trabajos previos cuyos objetivos de optimización son similares a nuestro modelo como [25] y [24] .

4.1.2. Petición VDC

Para las peticiones de VDC se usa una topología estrella, donde hay un conmutador central conectado a todos los servidores, similar a como se ha hecho en previas propuestas [2, 24, 18]; las capacidades están definidas de la siguiente forma: cada petición tiene de 1 a 3 servidores virtuales aleatoriamente conectados a un conmutador virtual, la máxima capacidad de CPU demandada es de 4 núcleos y la mínima de 1 núcleo; los enlaces mínimo de 10 BW y máximo de 100 BW, estas capacidades también se asignan de forma aleatoria. Al igual que en la distribución del DC físico los valores asignados a la petición de VDC están basados en la literatura previa [25] y [24].

4.1.3. Número de Redes virtuales y repeticiones

Para el análisis del experimento se tendrán de 1 hasta 27 peticiones de VDC, así podremos analizar el estado de las métricas de desempeño cuando el DC está en baja, media y

alta carga. Cada experimento se repetirá 20 veces en promedio para incrementar su validez estadística; los resultados se grafican con un intervalo de confianza del 95 %.

4.2. Método de solución: ϵ-restricciones en VDCE

En un problema multi-objetivo, la función objetivo arroja varios resultados posibles, es así como se debe determinar qué conjunto de valores de las variables del problema es el mejor, por lo que se deben establecer criterios claros; es por esto que para resolver el modelo formulado usaremos el método de las ϵ-restricciones propuesto por Haimes en 1971 . Esta técnica consiste en convertir un problema multi-objetivo en uno mono-objetivo, seleccionando una función objetivo como primaria y las otras como restricciones asociadas con un conjunto de parámetros llamados épsilon [29]. Como se había mencionado en el capítulo anterior la tolerancia será nuestro objetivo principal y las máquinas activas las nuevas restricciones. En las ecuaciones (4.1) y (4.2) f_1x representa a la cantidad de CPU empleada para proveer recursos de respaldo y f_2x el número de máquinas (servidores y conmutadores) activos en toda la red. El valor de épsilon representa el valor al que se restringe f_2x (máquinas activas), ϵ_2 consiste en una serie de niveles y con cada nivel se obtiene una solución al problema multiobjetivo. En la Fig. **4-2** vemos un ejemplo en la que cada punto representa una solución obtenida por cada variación de los valores representados por ϵ_2; el eje y representa los recursos de CPU utilizados para soportar supervivencia a fallos y el eje x las máquinas activas.
Transformar:

$$Minimizar(f_1x, f_2x)\forall x \in X \tag{4-1}$$

En:

$$\begin{aligned} &Minimizar(f_1x) \\ &\qquad f_2x \leq \epsilon_2 \\ &\epsilon_2 \in R; \forall x \in X \end{aligned} \tag{4-2}$$

En la Fig. **4-3** se representan los niveles de ϵ_2; cada nivel está representado por un superíndice, i.e. $\epsilon_2^1 = 1$ representa el valor con que inicialmente se restringe f_2x; con cada nivel se debe correr el modelo; por lo tanto, con cada nivel se obtiene una posible solución.Es por esta razón que la función máquinas activas es mejor opción para ser transformada a restricción, pues el máximo número de niveles de ϵ_2 posibles es 26.
Con el método de las ϵ-restricciones se pueden obtener soluciones fuertemente eficientes o débilmente eficientes. En la Fig. **4-4** las soluciones rodeadas de color rojo representan las soluciones fuertes y las rodeadas de blanco las débiles, así en las soluciones fuértemente eficientes (o soluciones no dominadas) no existe una solución mejor que ella en el conjunto de soluciones obtenidas por este método; las débiles son soluciones óptimas para un nivel o valor de épsilon, pero que son dominadas por otras soluciones. Por ejemplo, la solución

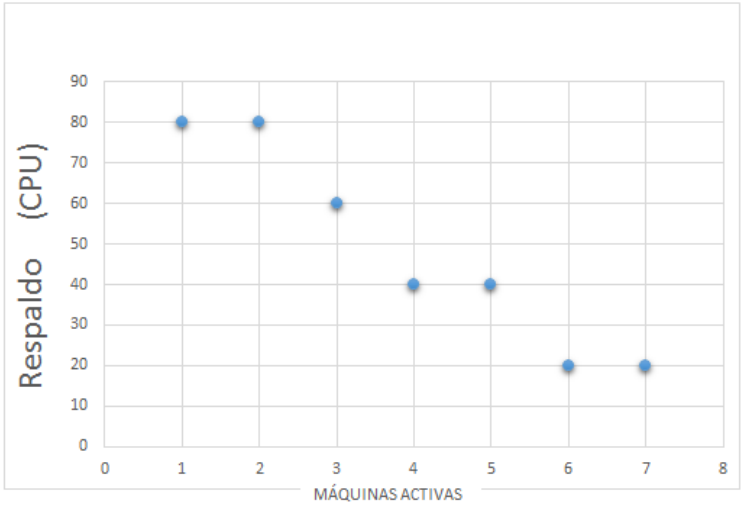

Figura 4-2: Máquinas Activas vs Recursos de respaldo (CPU)

(Máquinas Activas=5 ; Recursos de respaldo = 40) es una solución débil, puesto que está dominada por la solución (Máquinas Activas=4 ; Recursos de respaldo = 40), pues aunque esta última solución tiene igual desempeño en ahorro de CPU, supera a la primera solución en la función objetivo máquinas activas.

4.2.1. Selección de la respuesta

En nuestro modelo, los objetivos se encuentran en conflicto, lo que implica que si se mejora la eficiencia energética (ahorro energético apagando el mayor número de máquinas posibles), se tiene un peor desempeño en ahorro de recursos de respaldo para la supervivencia. Como se ha mencionado este método provee una serie de soluciones no dominadas llamado conjunto de Pareto, que son las mejores soluciones al modelo de optimización, en las que si inicialmente no hay criterio de selección, no se puede decidir qué respuesta se debe escoger o que respuesta es mejor que otra. Es por esto que en esta investigación se designa el parámetro de escogencia denominado "porcentaje". Si se establece este parámetro en el valor 0 quiere decir que se le va a dar más importancia a reducir el número de máquinas activas, es decir, el modelo escogerá como respuesta única, la solución no dominada que menos máquinas activas utilice. Si se establece este parámetro en el valor 100 quiere decir que se le va a dar más importancia al ahorro de recursos para la supervivencia, es decir, se escogerá como mejor respuesta la solución no dominada que menos recursos de respaldo utilice; si en cambio, se le asigna a este parámetro el valor 50 entonces se le da igual importancia a los dos ítems antes mencionados. Este parámetro puede tomar cualquier valor entre 0 y 100, definiendo así la importancia que se le quiera dar a estas dos funciones objetivo definidas. Para efectos de estudiar cómo se comporta el sistema, el experimento propuesto en esta investigación va a utilizar los valores

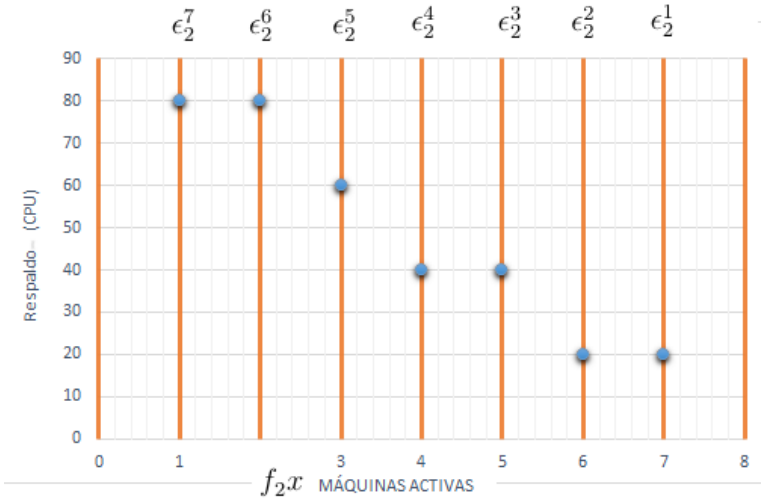

Figura 4-3: Cortes de ϵ

Tabla 4-1: Selección de respuesta

Porcentaje	Descripción
0	Escoger mejor solución no dominada que tenga mejor ahorro de máquinas activas
50	Solución intermedia
100	Escoger mejor solución no dominada que tenga mejor ahorro de recursos de respaldo

de 0, 50 y 100, así obtendrán las gráficas y resultados de los comportamientos del DC (ver Tabla **4-1**). Vemos en la Fig. **4-5** que la solución coloreada de negro se escogió cuando se le dio más importancia a las máquinas activas (porcentaje=0), la verde cuando se le dio más importancia a la tolerancia (porcentaje=100) y la amarilla cuando se le dio igual importancia a ambas (porcentaje=50).

4.2.2. Simulación e Instrumentos

Para demostrar el funcionamiento del modelo formulado, este se implementó en lenguaje de programación JAVA utilizando las librerías y funciones proporcionadas por la herramienta de software ALEVIN [28]. El modelo se solucionó con las librerías proporcionadas por el software Gurobi en su versión académica. Para resolver el problema se debe invocar a las funciones de Gurobi varias veces, ya que el método ϵ maneja un proceso iterativo en la que cada vez que se elija una restricción para la función objetivo, se debe resolver el modelo. La ejecución del modelo de optimización para las 27 peticiones de VDC redes duró alrededor de 56 horas.

Para realizar las simulaciones se utilizó un computador con las siguientes características: Sistema Operativo Windows 7 de 64 bits, Memoria RAM DE 8 GB, Procesador INTEL

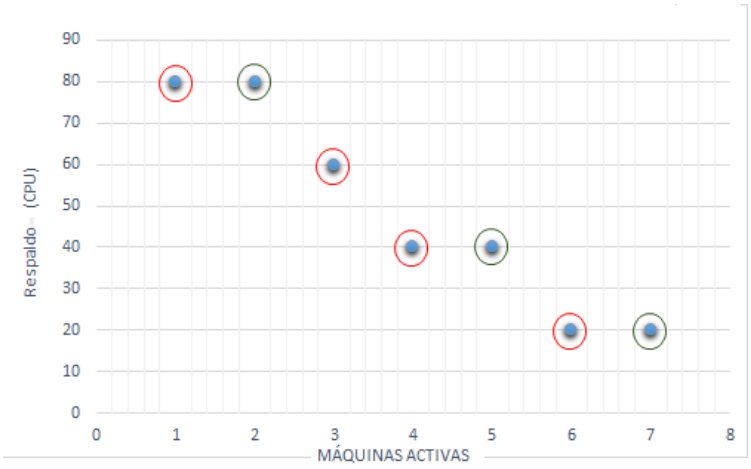

Figura 4-4: Soluciones fuertemente y débilmente eficientes

CORE I-7. Se utilizó el IDE Eclipse en su versión Luna (Service Release 2 (4.4.2)) en el que se instalaron las librerías necesarias para que el código de ALEVIN pudiera ser ejecutado.

4.2.3. Análisis de resultados

A continuación se presentan los resultados de la operación del modelo simulado, bajo ciertas métricas de desempeño que se definen a continuación. Se comparó el modelo en tres modos de operación distintos definidos por el parámetro porcentaje 0, 50 y 100.

Porcentaje de VDCs aceptados

Como se había mencionado en el capítulo 2, esta métrica mide la efectividad de un algoritmo para aceptar las peticiones de VDCs y su resultado se obtiene dividiendo la cantidad de VDCs aceptadas sobre la cantidad total de VDCs recibidas. Este resultado es muy importante básicamente porque más VDCs asignadas implica mayores ganancias para el proveedor de servicios. Se puede observar que cuando se ajustó el parámetro porcentaje a 100, el modelo tuvo mejor aceptación (61,5 % cuando la carga de peticiones de VDCs es mayor) de Redes Virtuales, que cuando porcentaje fue configurado en 0 y 50 (en estas dos configuraciones la aceptación osciló en valores alrededor del 56 %); esto se debe a que cuando se escogen soluciones no dominadas que buscan ahorrar recursos de respaldo, el modelo tiende a distribuir los nodos virtuales a través de la red, lo que facilita la acomodación de más redes virtuales (se reducen los cuellos de botella); otra razón importante es que cuando se busca ahorrar recursos quedan más espacios libres para acomodar más VDCs. Cuando se ajustó el porcentaje igual a 50 hubo un punto medio de operación entre porcentaje 0 y 100, lo que corresponde a lo que se esperaba cuando se le da igual importancia a ambas funciones

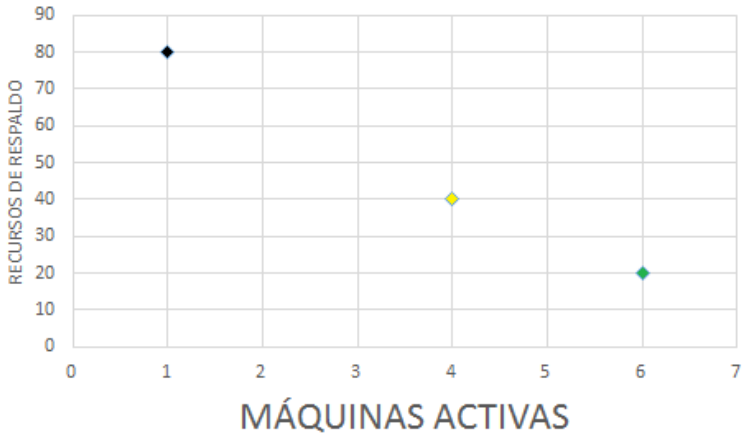

Figura 4-5: Selección de solución

objetivo. En promedio y bajo las condiciones dadas el algoritmo presentaba alrededor de 3 soluciones no dominadas (ver Fig. **4-6**).

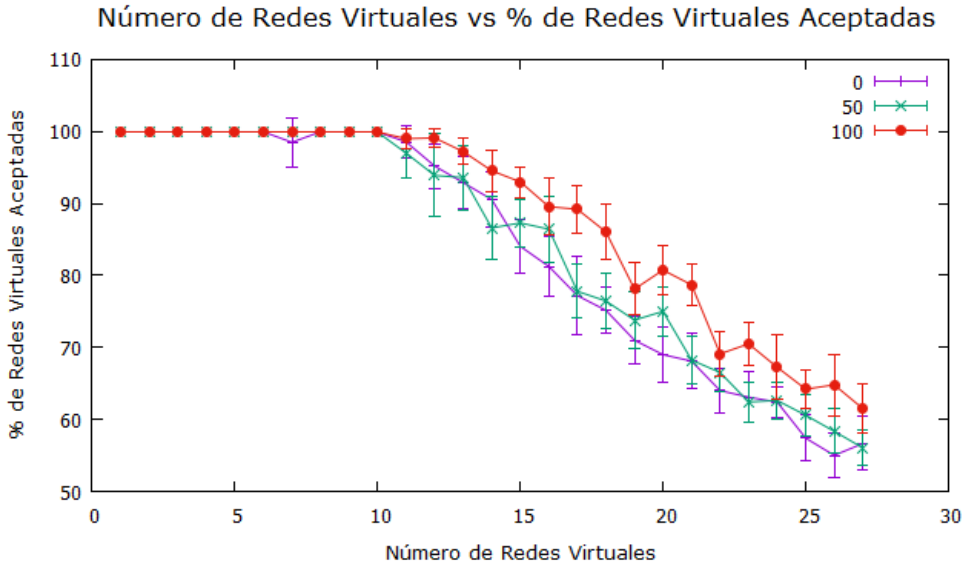

Figura 4-6: Porcentaje VDCs Aceptadas

Máquinas activas

El número de máquinas activas es una métrica que define la eficiencia de un algoritmo con respecto al consumo de energía, los proveedores de servicios de los DCs buscan constantemente estrategias que les permitan operar minimizando al máximo el consumo energético;

esto por el ahorro de costos y para contribuir a un mejor medio ambiente. Si se comparan los tres valores predeterminados de porcentaje solo hay un ahorro de 1 máquina activa en baja carga de peticiones de VDCs, cuando la métrica porcentaje es 0, es decir, cuando se escogen las soluciones con mejor desempeño en consolidación de máquinas activas; esto es debido a que la instancia del problema es muy pequeña (el modelo aplicado no es escalable para largas instancias del problema con las herramientas de hardware disponibles para el experimento), pero se espera a que a instancias más grandes el ahorro termine siendo mucho mejor para el porcentaje 0; por lo tanto este puede ser el modo de operación del DC cuando hay baja carga de peticiones de VDCs. Cuando hay 6 peticiones el ahorro en consumo energético es de 15 máquinas activas (ver Fig. **4-7**).

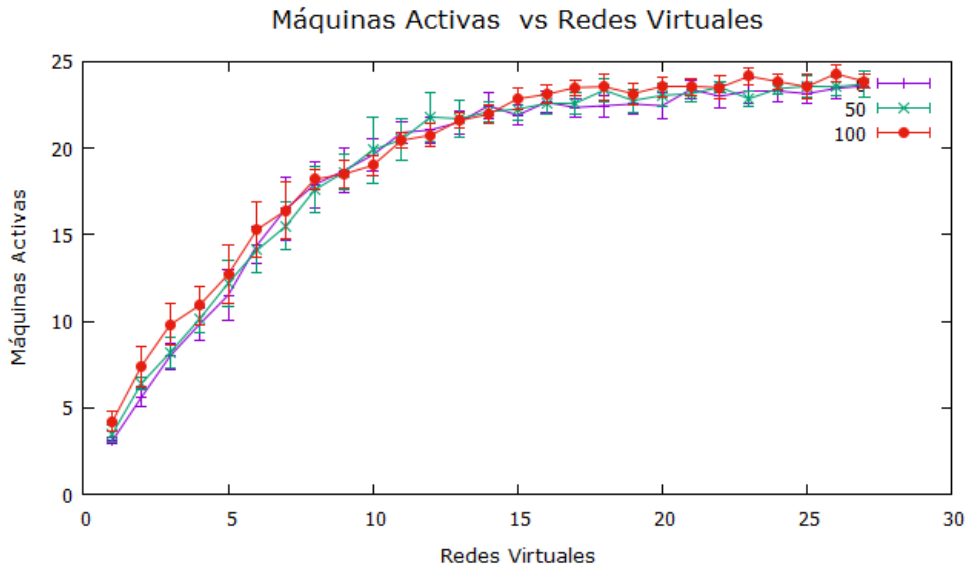

Figura 4-7: Máquinas activas

Ahorro en los recursos de respaldo CPU

La métrica ahorro en recursos de respaldo se relaciona directamente con los costos de operación, pues ocupar espacio extra de CPU implica gastos extras; también es capacidad que podria ser utilizadas para futuras asignaciones de VDCs. En el experimento se genera un ahorro de aproximadamente 10 CPUs cuando se escoge el parámetro porcentaje en el valor 100, con respecto a porcentaje configurado entre 0 y 50; esto hace que cuando se trabaja bajo esta configuración el modelo tenga mejor aceptación, pues se ahorra espacio para futuras VDCs (ver Fig. **4-8**).

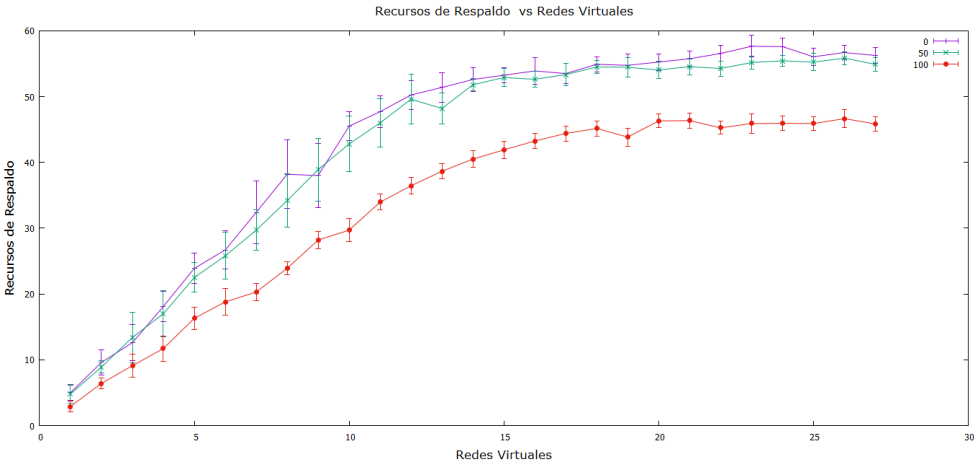

Figura 4-8: Ahorro recursos de respaldo

CPU Asignada

Esta métrica muestra cuantos recursos de CPU está consumiendo el proceso de asignación en la red del DC. Cuando el porcentaje se ha ajustado al valor 100, el proceso de asignación tiende a consumir alrededor de 10 núcleos menos de CPU en carga media de peticiones de VDCs, lo que se explica por el ahorro en recursos de respaldo en este modo de operación. Cuando la carga es alta en la red del DC, los tres casos del parámetro porcentaje operan igual, pues la red se congestiona y se agotan los recursos (ver la Fig. 4-9). Aunque hay que tener en cuenta que cuando se ajustó el porcentaje 100 existió la ventaja de que se asignaron más redes virtuales.

4.2.4. Comparación con un algoritmo de referencia

Hasta donde se pudo conocer, no existe una propuesta previa que aborde el proceso de equilibrio entre tolerancia a fallos y ahorro en consumo energético. Pero para efectos de evaluar el desempeño, se toma el algoritmo de asignación de referencia basado en el propuesto en [24] pero sin tener en cuenta el parámetro de disponibilidad implementado originalmente en este algoritmo y asignando los conmutadores virtuales preferiblemente en los conmutadores ya activos. En este algoritmo de referencia primero se asignan los servidores virtuales y el conmutador virtual en donde se busca preferiblemente asignar los nodos físicos ya activos; al mismo tiempo se asignan los recursos requeridos de respaldo para garantizar la supervivencia. En la siguiente etapa se realiza la asignación de enlaces utilizando el método del camino más corto.

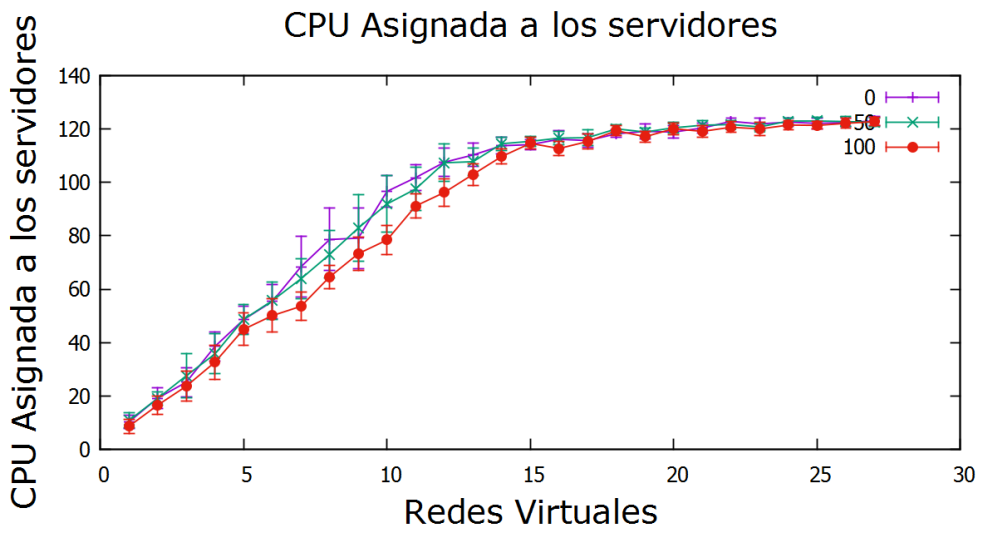

Figura 4-9: CPU asignada en los servidores

Porcentaje VDCs aceptados

Se había determinado en la sección anterior que el parámetro porcentaje configurado a 100 obtenía mejor desempeño de aceptación de VDCs; así mismo cuando se compara esta configuración con el algoritmo de referencia se obtiene un 8 %de mejora en porcentaje de aceptación de redes cuando la carga de peticiones de VDCs es alta en la red del DC; de igual forma en cargas bajas y medias el algoritmo propuesto en esta investigación tiene siempre rendimiento superior en aceptación de redes. Esto se debe a que el modelo de asignación de recursos propuesto en nuestro trabajo de investigación resuelve el problema VDCE de manera coordinada, manteniendo un equilibrio entre el ahorro de recursos para la tolerancia y el ahorro en el consumo energético (ver Fig. **4-10**).

Número de máquinas activas

En esta métrica el desempeño del algoritmo de referencia es superior a la configuración porcentaje 100 de nuestro modelo de optimización; pero cabe destacar que para esta configuración del modelo, este termina aceptando un número mayor de redes virtuales, lo que implica que se debe utilizar un mayor número de máquinas activas, pues el número de máquinas virtuales es mayor en los casos en que el porcentaje de aceptación es mejor; así mismo el algoritmo de referencia tiene mucho énfasis en ahorrar consumo energético (ver Fig. **4-11**).

Ahorro en los recursos de respaldo de CPU

Vemos en esta métrica que se está generando 10 CPUs de ahorro en el modelo de optimización configurado a porcentaje igual a 100. Esta es la ventaja de trabajar en forma coordinada

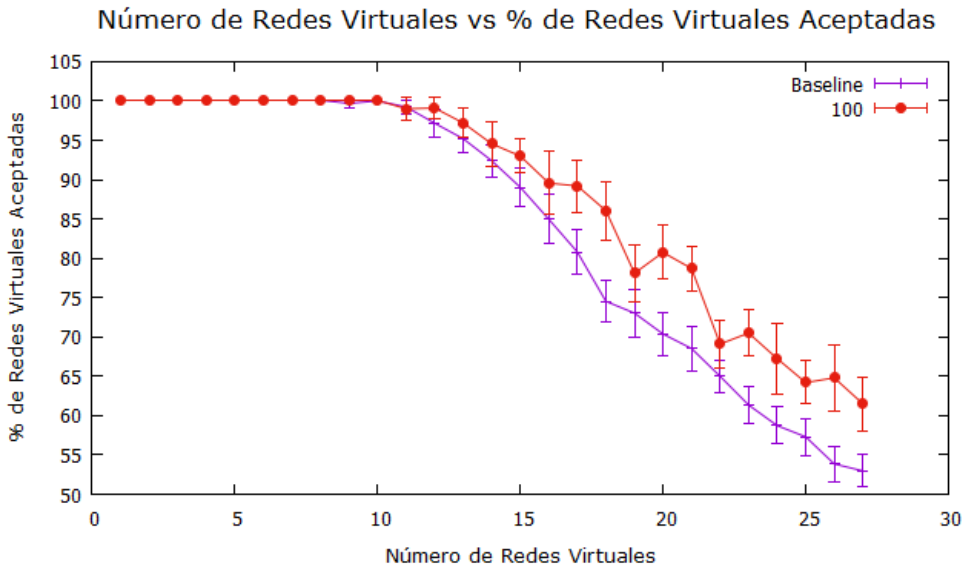

Figura 4-10: Porcentaje de VDCs aceptados en algoritmo de referencia

la supervivencia a fallos y la minimización del consumo energético, pues al no enfocarse solo en consumo energético (como el algoritmo de referencia), se permite ahorrar muchos recursos extras de CPU, además de evitar los cuellos de botella en la red; lo que permite generar mayores ganancias al proveedor de servicios por la utilización de menos recursos y la aceptación de más VDCs (ver Fig. **4-12**).

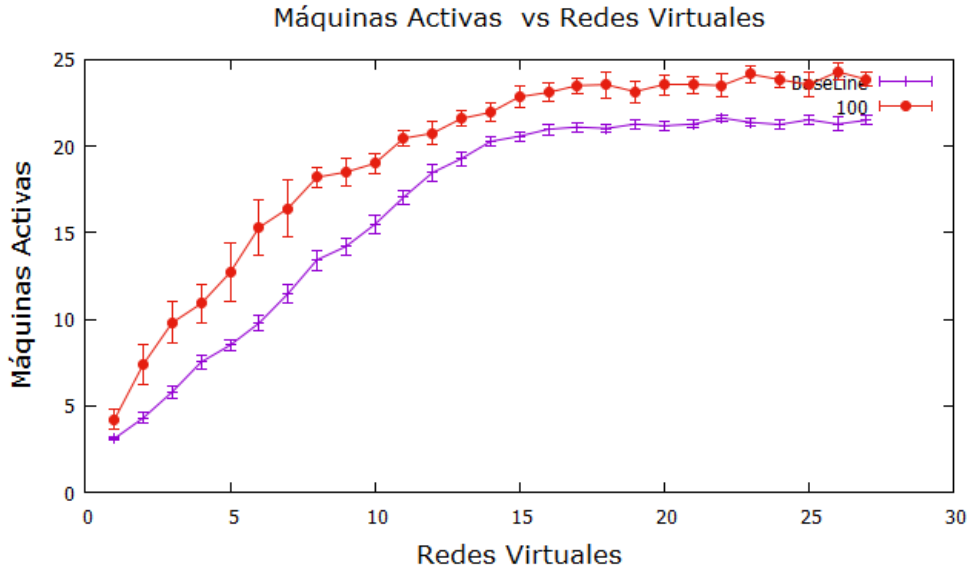

Figura 4-11: Máquinas activas en algoritmo de referencia

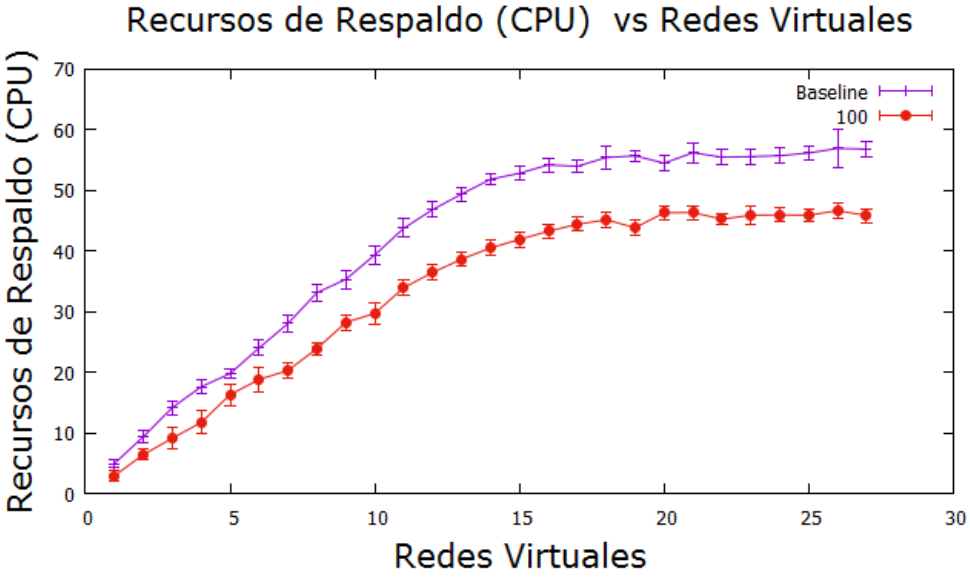

Figura 4-12: Ahorro Recursos Respaldo

5 Conclusiones y trabajo futuro

En este capítulo se presentarán las conclusiones principales del estudio de investigación realizado y se describirán las posibles ramas de trabajo futuro que surgen del desarrollo de este tema de investigación.

5.1. Conclusiones

El protagonismo de los DCs ha venido en aumento en los últimos años impulsado por el crecimiento de la computación en la nube. La comunidad científica ha demostrado que la virtualización de los servidores y de la red de los DCs es una alternativa para solucionar los problemas que estos atraviesan actualmente. La correcta asignación de recursos en DCs virtualizados es uno de los retos principales de la virtualización de DCs.
En esta tesis se han revisado aquellas propuestas que han estudiado el problema de asignación de recursos en centros de datos virtualizados, los objetivos perseguidos y los resultados de estos. Se ha propuesto una taxonomía que permite clasificar cualquier propuesta conducente a solucionar el problema VDCE y se han propuesto direcciones de investigación identificadas, a partir del análisis de las propuestas existentes.
Dentro de este trabajo de investigación propusimos una solución al problema VDCE basado en un modelo de optimización matemático que coordina el equilibrio entre la minimización del consumo energético y la provisión de tolerancia a fallos, lo que implica un problema de optimización multi-objetivo. Para resolverlo se usó el método de la épsilon-restricción que básicamente consiste en convertir un problema multi-objetivo en uno mono-objetivo; se seleccionó como función primaria del modelo, la provisión de recursos de respaldo para la supervivencia y como restricción las máquinas activas dentro de la red del DC.
Se designó el parámetro de escogencia denominado “porcentaje” para que cuando tomara el valor 0 dar más importancia a reducir el número de máquinas activas; si el valor asignado era 100 se le da más importancia al ahorro de recursos para la supervivencia; y si el parámetro toma el valor 50 entonces se le da igual importancia a las dos funciones objetivo. En general la configuración del parámetro 100 generó mejores desempeños en aceptación de redes y ahorro de recursos de respaldo. Aunque como era de esperarse el parámetro 0 fue superior en la métrica maquinas activas. Cabe aclarar que el comportamiento de las tres graficas es parecido porque se están escogiendo solo las respuestas no dominadas, por lo tanto, corresponden todas a configuraciones de alta calidad, y que se aproximan a un equilibrio óptimo; además es una instancia pequeña del problema la que se ha evaluado. Es conveniente entonces escoger la

configuración de 0 cuando se espera baja carga de peticiones de VDCs y de 100 cuando se espera alta congestión en la red. De esta manera se ahorra energía y a la vez se pueden acomodar más redes virtuales.
El desempeño de nuestro modelo de optimización también fue evaluado con un algoritmo de referencia basada en la literatura previa [24], en la que nuestro modelo fue superior en un 8 % en la aceptación de redes y ahorraba alrededor de 10 CPUs en el consumo de recursos para proveer supervivencia.
Finalmente durante el desarrollo de este proyecto de investigación se alcanzaron los objetivos planteados en la propuesta de investigación, pues se caracterizaron los modelos de VDCE de otros autores; se formuló un modelo de VDCE determinando los parámetros y métricas del mismo; se propuso un algoritmo que permitió solucionar el modelo formulado y finalmente se evaluó el desempeño.

5.2. Trabajo futuro

La asignación de recursos en centros de datos es un tema emergente y existen muchos campos por ser explorados. La implementación de algoritmos exactos y metaheurísticos, el ahorro de energía, la tolerancia a fallos, los centros de datos distribuidos y los centros de datos ecológicos son ramas prometedoras de investigación.
Como se ha mencionado constantemente en este trabajo de investigación el VDCE es NP-completo. Entonces, una posible rama de trabajo futuro consiste en investigar las clases de algoritmos heurísticos o metaheurísticos son aplicables para resolver el problema. En este sentido, se debe estudiar la convergencia y la calidad de soluciones para diferentes instancias del problema.
Otra posible rama de trabajo futuro se puede enfocar en aplicar los algoritmos de solución diseñados en escenarios reales, utilizando las Redes Definidas por Software [30] como herramienta para proveer la virtualización.

Bibliografía

[1] E.S. Correa, L.A. Fletscher, and J.F. Botero. Virtual data center embedding: A survey. *Latin America Transactions, IEEE (Revista IEEE America Latina)*, 13(5):1661–1670, May 2015.

[2] Hitesh Ballani, Paolo Costa, Thomas Karagiannis, and Antony IT Rowstron. Towards predictable datacenter networks. In *SIGCOMM*, volume 11, pages 242–253, 2011.

[3] M.F. Bari, R. Boutaba, R. Esteves, L.Z. Granville, M. Podlesny, M.G. Rabbani, Qi Zhang, and M.F. Zhani. Data center network virtualization: A survey. *Communications Surveys Tutorials, IEEE*, 15(2):909–928, 2013.

[4] Alan Shieh, Srikanth Kandula, Albert Greenberg, Changhoon Kim, and Bikas Saha. Sharing the data center network. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, pages 23–23. USENIX Association, 2011.

[5] Nick Feamster, Lixin Gao, and Jennifer Rexford. How to lease the internet in your spare time. *ACM SIGCOMM Computer Communication Review*, 37(1):61–64, 2007.

[6] NM Mosharaf Kabir Chowdhury and Raouf Boutaba. Network virtualization: state of the art and research challenges. *Communications Magazine, IEEE*, 47(7):20–26, 2009.

[7] Kurt Tutschku, Thomas Zinner, Akihiro Nakao, and Phuoc Tran-Gia. Network virtualization: Implementation steps towards the future internet. *Electronic Communications of the EASST*, 17, 2009.

[8] Andreas Berl, Andreas Fischer, and Hermann de Meer. Using system virtualization to create virtualized networks. *Electronic Communications of the EASST*, 17, 2009.

[9] Albert Greenberg, James Hamilton, David A Maltz, and Parveen Patel. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73, 2008.

[10] Albert Greenberg, James R Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A Maltz, Parveen Patel, and Sudipta Sengupta. Vl2: a scalable and flexible data center network. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 51–62. ACM, 2009.

[11] William Dally and Brian Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[12] C.E. Leiserson. Fat-trees: Universal networks for hardware-efficient supercomputing. *Computers, IEEE Transactions on*, C-34(10):892–901, Oct 1985.

[13] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 63–74. ACM, 2008.

[14] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. Bcube: a high performance, server-centric network architecture for modular data centers. *ACM SIGCOMM Computer Communication Review*, 39(4):63–74, 2009.

[15] A Amokrane, M.F. Zhani, R. Langar, R. Boutaba, and G. Pujolle. Greenhead: Virtual data center embedding across distributed infrastructures. *Cloud Computing, IEEE Transactions on*, 1(1):36–49, Jan 2013.

[16] Chuanxiong Guo, Guohan Lu, Helen J. Wang, Shuang Yang, Chao Kong, Peng Sun, Wenfei Wu, and Yongguang Zhang. Secondnet: a data center network virtualization architecture with bandwidth guarantees. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 15:1–15:12, New York, NY, USA, 2010. ACM.

[17] A. Fischer, J.F. Botero, M. Till Beck, H. de Meer, and X. Hesselbach. Virtual network embedding: A survey. *Communications Surveys Tutorials, IEEE*, 15(4):1888–1906, Fourth 2013.

[18] M.G. Rabbani, R. Pereira Esteves, M. Podlesny, G. Simon, L. Zambenedetti Granville, and R. Boutaba. On tackling virtual data center embedding problem. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 177–184, 2013.

[19] John Wiley. *Integer Programming*. 1998.

[20] El-Ghazali Talbi. Metaheuristics: from design to implementation. volume 74, 2009.

[21] Yufeng Xin, Ilia Baldine, Anirban Mandal, Chris Heermann, Jeff Chase, and Aydan Yumerefendi. Embedding virtual topologies in networked clouds. In *Proceedings of the 6th International Conference on Future Internet Technologies*, CFI '11, pages 26–29, New York, NY, USA, 2011.

[22] Jielong Xu, Jian Tang, K. Kwiat, Weiyi Zhang, and Guoliang Xue. Enhancing survivability in virtualized data centers: A service-aware approach. *Selected Areas in Communications, IEEE Journal on*, 31(12):2610–2619, December 2013.

[23] M.F. Zhani, Qi Zhang, G. Simon, and R. Boutaba. Vdc planner: Dynamic migration-aware virtual data center embedding for clouds. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 18–25, 2013.

[24] Md Golam Rabbani, Mohamed Faten Zhani, and Raouf Boutaba. On achieving high survivability in virtualized data centers. *IEICE Transactions on Communications*, 97(1):10–18, 2014.

[25] Qi Zhang, Mohamed Faten Zhani, Meyssa Jabri, and Raouf Boutaba. Venice: Reliable virtual data center embedding in clouds. *IEEE International Conference on Computer Communications (INFOCOM)*, 2014.

[26] Yufeng Xin, Ilia Baldine, Anirban Mandal, Chris Heermann, Jeff Chase, and Aydan Yumerefendi. Embedding virtual topologies in networked clouds. In *Proceedings of the 6th International Conference on Future Internet Technologies*, CFI '11, pages 26–29, New York, NY, USA, 2011. ACM.

[27] Ronald L Rardin. *Optimization in operations research*, volume 166. Prentice Hall New Jersey, 1998.

[28] Andreas Fischer, Juan Felipe Botero Vega, Michael Duelli, Daniel Schlosser, Xavier Hesselbach Serra, and Hermann De Meer. Alevin-a framework to develop, compare, and analyze virtual network embedding algorithms. pages 1–12, 2011.

[29] Matthias Ehrgott and Stefan Ruzika. Improved ε-constraint method for multiobjective programming. *Journal of Optimization Theory and Applications*, 138(3):375–396, 2008.

[30] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2015.