Mejorando la red de los servicios de motores de búsqueda a través de enrutamiento basado en la aplicación

Joe Carrion Jumbo

Mejorando la red de los servicios de motores de búsqueda a través de enrutamiento basado en la aplicación

por

Joe Carrión Jumbo

Memoria presentada para optar al grado de Doctor por la Universidad Autónoma de Barcelona. Este trabajo ha sido desarrollado en el Departamento de Arquitectura de Computadores y Sistemas Operativos bajo la dirección del Dr. Daniel Franco Puntes

Bellaterra, Mayo 2017

Declaración de Autoría

Tesis doctoral presentada por Joe Carrión Jumbo, para optar por el Grado de Doctor por la Universidad Autónoma de Barcelona.

El presente trabajo de investigación ha sido desarrollado en el Departamento de Arquitectura de Computadores y Sistemas Operativos de la Escuela de Ingeniería de la Universidad Autónoma de Barcelona, dentro del programa de Doctorado en Informática bajo la dirección y tutoría del Doctor Daniel Franco Puntes.

Bellaterra, Mayo del 2017.

Autor: Joe Luis Carrión Jumbo

Firma: _____

Director y Tutor: Dr. Daniel Franco Puntes Ph.D.

Firma: _____

UNIVERSITAT AUTÓNOMA DE BARCELONA

Resumen

Mejorando la red de los servicios de motores de búsqueda a través de enrutamiento basado en la aplicación

por Joe Carrión Jumbo

Los sistemas de cómputo complejos como los Servicios de Motores de Búsqueda proveen servicios a miles de usuarios, y su demanda puede cambiar repentinamente. Esta inestable demanda impacta significativamente a los componentes del servicio (como red de datos y nodos). El sistema debería ser capaz de gestionar escenarios inesperados, de otro modo, los usuarios se podrían ver forzados a dejar el sistema. Un motor de búsqueda tiene una típica arquitectura compuesta por un Front Service que procesa las solicitudes de usuarios, un Index Service que almacena la información recopilada de Internet y un Cache Service que gestiona el acceso eficiente a contenido de uso más frecuente. Los avances científicos que proveen estos servicios son en general tecnología emergente. Los servicios de red de un motor de búsqueda requieren de una planificación especializada; la presente investigación se lleva a cabo con el estudio del patrón de tráfico de un motor de búsqueda y el diseño de un modelo de enrutamiento de los mensajes entre los nodos de la red basado en las condiciones de flujo de datos del motor de búsqueda. El resultado esperado es un servicio de red especializado en el tráfico de un motor de búsqueda que asigne los recursos de red de forma eficiente según sea la demanda que soporta en tiempo real. La evaluación del patrón de tráfico permitió identificar condiciones de desbalance de la red de datos y congestión de mensajes, de modo que se diseñó un modelo que combina diferentes modelos de enrutamiento de la literatura y nuevos criterios basados en las condiciones específicas del tráfico del motor de búsqueda. Para el diseño de ésta propuesta ha sido necesario diseñar un modelo a escala del motor de búsqueda utilizando técnicas de simulación y se ha utilizado tráfico de un sistema real que ha permitido evaluar de forma precisa el modelo propuesto y compararlo con modelos de enrutamiento actualmente disponibles en la literatura y tecnología actual. Los resultados obtenidos demuestran que el modelo propuesto mejora el rendimiento de la red del motor de búsqueda en términos de latencia y throughput de la red.

UNIVERSITAT AUTÓNOMA DE BARCELONA

Abstract

Improving the network of a search engine services through application-driven routing

by Joe Carrión Jumbo

Large-scale computer systems like Search Engines provide services to thousands of users, and their user demand can change suddenly. This unstable demand impacts sensitively to the service components (like network and hosts). The system should be able to address unexpected scenarios; otherwise, users would be forced to leave the service. A search engine has a typical architecture consisting of a Front Service, that processes the requests of users, an Index Service that stores the information collected from the internet and a Cache Service that manages the efficient access to content frequently used. The scientific advances that provide these services are in general emergent technology. The network services of a search engine require specialized planning; This research is carried out by studying the traffic pattern of a Search Engine and designing a routing model for messages between network nodes based on the data flow conditions of the Search Engine Service. The expected result is a network service specialized in the traffic of a Search Engine that allocates network resources efficiently according to demand it supports in real time. The evaluation of the traffic pattern allowed us to identify conditions of unbalance of the network and congestion of messages. Therefore model designed combines different routing models of the literature and a new criteria based on the specific conditions of the traffic of the Search Engine. For the design of this proposal it has been necessary to design a scale model of a Search Engine using simulation techniques and It has has used traffic from a real system that allowed us to accurately evaluate the proposed model and compare it with currently available routing models in the literature and technology. The results show that the proposed model improves the performance of the Search Engine network in terms of latency and network throughput.

UNIVERSITAT AUTÓNOMA DE BARCELONA

Resum

La millora de la xarxa d'uns serveis de cerca a través d'enrutament impulsat per les aplicacions

by Joe Carrión Jumbo

Els sistemes de còmput complexos com els Serveis de Motors de Cerca proveeixen serveis a milions d'usuaris, i la seva demanda pot canviar sobtadament. Aquesta inestable demanda impacta significativament als components del servei (com xarxa de dades i nodes). El sistema hauria de ser capaç de gestionar escenaris inesperats, d'altra manera, els usuaris es podrien veure forçats a deixar el sistema. Un motor de cerca té una típica arquitectura composta per un Front Service que processa les sol·licituds dels usuaris, un Index Service que emmagatzema la informació recopilada d'Internet i un Cache Service que gestiona l'accés eficient al contingut d'ús més freqüent. Els avanços científics que proveeixen aquests serveis són en general tecnologia emergent. Els serveis de xarxa d'un motor de cerca requereixen una planificació especialitzada; la present investigació es desenvolupa amb l'estudi del patró de tràfic d'un motor de cerca i el disseny d'un model de enrutament dels missatges entre els nodes de la xarxa basats en les condicions de fluxos de dades del motor de cerca. El resultat esperat és un servei de xarxa especialitzat en el tràfic d'un motor de cerca que assigni els recursos de la xarxa de forma eficient segons sigui la demanda que suporta en temps real. La verificació del patró de tràfic va permetre identificar condicions de desequilibri de la xarxa de dades i congestió de missatges, així que es va dissenyar un model que combina diferents models de enrutament de la literatura i nous criteris basats en les condicions específiques del tràfic del motor de cerca. Per al disseny d'aquesta proposta ha estat necessari dissenyar un model a escala del motor de cerca utilitzant tècniques de simulació i s'ha usat tràfic d'un sistema real que ha permès verificar de forma precisa el model proposat i comparar-lo amb models de enrutament actualment disponibles a la literatura i tecnologia actual. Els resultats obtinguts demostren que el model proposat millora el rendiment de la xarxa del motor de cerca en termes de latència i throughput de la xarxa.

Agradecimientos

El presente trabajo no hubiera podido ser desarrollado sin el soporte de varias personas que a lo largo de cuatro años del desarrollo del programa de Doctorado en Informática me han dado continuo soporte en lo personal como profesional.

En primer lugar al Doctor Daniel Franco, Director del proyecto por su soporte metodológico, académico y acompañamiento profesional a fin de llevar a cumplimiento las tareas relacionadas a los objetivos de la investigación. Además por su comprensión en temas de interés personal y familiar fundamentales para cumplir con mis objetivos.

Al Doctor Emilio Luque Fadón, y la Doctora Dolores Rexachs por su valiosa retroalimentación en las sesiones técnicas permanentes y por sus comentarios y retroalimentación para afrontar los retos de la investigación con los criterios metodológicos y técnicos apropiados. Su apoyo ha sido fundamental en asuntos familiares y personales.

A la Doctora Verónica Gil-Costa, Doctor Mauricio Marín y Centro de Biotecnología y Bioinformática bajo Proyecto Basal, por su soporte profesional durante la investigación.

Al equipo grupo de trabajo de cada semana Pilar, Laura, Alvaro, Jorge, Carlos, Hai quienes han estado siempre en predisposición de comentar y aportar los resultados y avances de mi trabajo. Al equipo que durante estos cuatro años formó parte de las reuniones, Marcela, Hugo, Xavier, Aprigio, Carlos Nuñez.

Al personal académico del Departamento, quienes han presenciado los informes anuales y han expresado sus valiosos comentarios y observaciones tanto como parte del Tribunal de Evaluación Anual como parte de la audiencia de seguimiento.

A los compañeros y amigos con quienes inicié este programa de Doctorado, por apoyo y solidaridad en diferentes actividades académicas, personales y familiares, Liu, Josefina, César, Eva, Francisco, Albert, Gemma. Al personal administrativo del Departamento, Gemma, Daniel con quienes se ha compartido momentos del día a día y asuntos de soporte y logística para el desarrollo de la investigación.

A la distancia con emoción por volverlos a ver, a mi familia y amigos en Ecuador quienes siempre han estado en contacto conmigo y mi familia.

Al Ministerio de Economía, Industria y Competitividad (MINECO) de España bajo contrato TIN2014-53172-P. A la Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) de Ecuador bajo contrato 2013-AR7L335.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Abreviaturas

SES	**S**earch **E**ngine **S**ervice
FS	**F**ront **S**ervice
CS	**C**ache **S**ervice
IS	**I**ndex **S**ervice
CSR	**C**ache **S**ervice **R**eplica
ISR	**I**ndex **S**ervice **R**eplica
QL	**Q**uery **L**atency
T	**T**hroughput
VSE	**V**ertical Search Engine

"Con inmenso amor dedicado a Paula, Eduardo, Cecilia.
A mis padres Carlota y Luis; hermanos Ronny, Henry,
Paulo, José y Byron"
Joe

Capítulo 1

Introducción

1.1 Motivación

Los servicios de motores de búsqueda (Search Engine Services SES) son utilizados por millones de usuarios diariamente. Algunos proveedores de éstos servicios son Google Inc., Yahoo, Baidu, Ask y aunque ellos prevalecen en el mercado [2], hay otros entornos donde los SES son una herramienta importante, como en redes sociales y portales corporativos. El alcance del servicio se extiende desde miles de individuos para uso personal o profesional hasta industrias multinacionales, empresas estatales y organizaciones internacionales. La capacidad de los centros de datos para estos servicios es crítica y el mantenimiento de sus operaciones requiere un nivel de planificación de gran alcance que evalúe diferentes escenarios de demanda. El diseño de centros de datos del futuro es un tópico recurrente por los requerimientos de elevada capacidad de ancho de banda y muy grande número de clientes [3].

El usuario de estos servicios puede ser una persona, que envía solicitudes de forma individual a razón de algunas peticiones por minuto, ó, puede ser un mecanismo automático (un robot) que utiliza el SES para proveer servicios especializados a

otras personas o sistemas, en éste caso la razón de las solicitudes puede ser del orden de cientos o miles de peticiones por minuto. El SES puede ser utilizado para buscar contenido heterogéneo, como documentos de texto, imágenes, audio, vídeo. Las necesidades por las que se utiliza además son diversas, puede ser triviales, para buscar el significado de una palabra, o críticas, para buscar una noticia importante o un reporte financiero. Características de búsquedas han sido publicadas en [4].

La operación de un SES demanda de un centro de datos planificado y diseñado con la capacidad de soportar la demanda estimada de consultas que procesa y de datos que almacena. Una característica esperada es la capacidad mantener su rendimiento con bajos y altos periodos de carga. Estos cambios pueden resultar impredecibles ya que dependen del volumen de usuarios y de condiciones externas, como tendencias a buscar el mismo contenido por miles de usuarios y horarios en los que se producen. Este comportamiento lanza una inesperado tráfico a los recursos de la red, por lo tanto se espera que la asignación de recursos (switches, cables) para resolver la demanda sea eficiente. Con éste tipo de demanda planteamos un supuesto inicial, en relación a como elegir la tecnología apropiada de red para una aplicación específica, y, en como afinar tal tecnología para el tráfico que un SES genera. Por lo tanto una estrategia de gestión del tráfico es necesaria [5], [6].

La relación que se establece entre *quién* o *qué* lo utiliza, el *contenido* y el *porqué* se utiliza, convierte a un SES en un servicio que se espera esté disponible, sin interrupciones y que la respuesta se obtenga en un periodo de tiempo razonable. El criterio razonable puede explicarse como la capacidad de responder antes de que el mismo solicitante pueda enviar otra solicitud. El SES por lo tanto es un sistema crítico al proveer información que es necesaria para otros servicios o sistemas en periodos cortos de tiempo.

Desde el punto de vista del proveedor del servicio, la *capacidad* operativa y técnica para mantenerlo crece proporcionalmente al nivel de usuarios y volumen de datos almacenados. Corporaciones con cientos de empleados proveen servicios de búsqueda

en portales corporativos (Intranet) con centros de datos de pequeña escala con redes de área local (LAN) o redes de área amplia (WAN). Mientras que multinacionales con miles de empleados necesitan centros de datos con réplicas a nivel regional (por país) o mundial (por continente). Finalmente servicios de alcance global requieren de grandes centros de datos a nivel mundial que distribuyen geográficamente la demanda para mantener el contenido accesible a cualquier usuario en el mundo. "Una precisa predicción de la carga de trabajo ayudará a los diseñadores a planificar los recursos de hardware y software" [7].

Al criterio de capacidad hay que agregar la *continuidad* del servicio que en pequeña escala puede admitir horarios de disponibilidad según calendarios locales mientras que para otros requiere horarios más extensos con criterios de cierre de operaciones de fin de semana; hasta servicios de uso continuo de 24 horas del día durante todo el año.

Los párrafos anteriores perfilan la complejidad de un motor de búsqueda. Esta complejidad nos conduce a las interrogantes de: ¿cómo planificar la operación de un sistema de tales características?, ¿como se puede prever su capacidad de operación?. La respuesta nos conduce a la necesidad de disponer de herramientas y técnicas para planificar su operación y prever su rendimiento. Además para las interrogantes planteadas es necesario analizar globalmente el problema que plantean y definir una estrategia que nos conduzca a abordar el problema con objetividad. Por lo tanto, la presente investigación ha sido conducida con el soporte de información de un motor de búsqueda real que nos permite definir el alcance del estudio con un caso práctico y con el avance de la investigación diseñar criterios que a futuro puedan ser generalizados para problemas relacionados con otros motores de búsqueda o aplicaciones.

Existen tres fases en la operación del sistema; la primera proviene de la petición (la entrada) que se expresa como "un usuario", sin embargo tal demanda puede ser de miles de peticiones diferentes, la gestión de las solicitudes, el procesamiento

en el sistema remoto; la segunda que es la preparación de los resultados, que demanda la selección de los documentos relevantes para la solicitud; y finalmente la tercera que es el proceso de recolección previa (o simultánea) de contenido para proveer resultados. Las tres fases requieren del servicio eficiente de una red de comunicación, que procese el tráfico generado de forma especializada.

Cada fase resumida en el párrafo anterior demanda de un sistema complejo que requiere un análisis y planificación detallada. La primera fase, de entrada, ésta definida por el volumen de usuarios que utilizan el servicio que en el caso de servicios globales está en el orden de miles de consultas por día y por hora. Esta demanda es gestionada con el suministro de servicios geográficamente distribuidos por país o región continental y para reducir el volumen de la demanda y procesarla en centros de datos y sistemas más pequeños.

La segunda fase, correspondiente al tratamiento de la solicitudes de usuario que son gestionadas por el servicio de "Front Service" (FS), además se requiere de una técnica de clasificación del contenido en, contenido inusual y en generalmente consultado. El contenido inusual es gestionado por el servicio de "Index Service" (IS), mientras que el contenido más frecuente es gestionado por el servicio de "Cache Service" (CS). La tercera fase, corresponde al proceso de recolección de documentos de Internet para alimentar la base de datos del motor de búsqueda. Durante esta fase la información es organizada en diferentes particiones y réplicas que permitan mejorar el rendimiento y fiabilidad del sistema.

Los tres componentes de software. Front Service (FS), Index Service (IS) y Cache Service (CS) corresponden a instancias ejecutándose en diferentes nodos de procesamiento desplegados en una red con topología de árbol. Las consultas de usuario arriban al FS, el FS divide la consulta en mensajes que se envían a diferentes CS. Cuando el CS no resuelve la consulta retorna un error y los mensajes son enviados al IS para resolver la consulta. Los mensajes entre los nodos someten a la red de datos a una carga de trabajo de comportamiento no predecible o relacionada a

patrón de comunicaciones conocido. Los mensajes portan un identificador de la consulta a la que pertenecen, esto permite mantener una independencia entre ellos durante su tratamiento en la red.

El volumen de usuarios del sistema es impredecible desde el punto de vista del nivel de heterogeneidad del contenido (contenido diverso, usuarios diversos). El volumen de usuarios simultáneos enviando solicitudes al sistema es también un factor determinante de la capacidad del sistema.

Actualmente el mercado de grandes centros es predominado por Infiniband, 10 Gigabit Ethernet [8]. Esta tecnología puede soportar una variedad de aplicaciones, es decir es una caja que se puede adaptar y ha sido diseñada con un amplio conjunto de funciones. Sin embargo para un servicio especializado que únicamente ejecuta una aplicación distribuida, se puede obtener ventaja si se especializa la red para responder a los requerimientos particulares de un motor de búsqueda. Esto se puede lograr estudiando la carga generada por un SES y configurar la red para ser eficiente con el patrón de comunicaciones que la aplicación genera.

Un SES es una aplicación distribuida y compleja como se ha delineado en párrafos anteriores, esto nos conduce al uso de técnicas de simulación para abordar el problema en un ambiente seguro, de bajo coste y dinámico que permita modelar el funcionamiento real del sistema y de la interacción entre sus componentes (hardware y software).

1.2 Objetivos

1.3 Objetivo General

La presente investigación propone diseñar un mecanismo de gestión eficiente del tráfico de la red basado en el estudio del tráfico y carga de trabajo que genera un

Servicio de Motor de Búsqueda.

1.4 Objetivos Específicos

Proponer una metodología para analizar el rendimiento de la red de comunicaciones del servicio de un motor de búsqueda que permita analizar el impacto de diferentes configuraciones del sistema.

Analizar el comportamiento de tráfico real de datos de un motor de búsqueda como caso de estudio y caracterizar el sistema, con el fin de contribuir en un ámbito relevante y crítico que requiera del uso de tecnología de emergente.

Diseñar una plataforma de simulación para experimentación de diferentes configuraciones de red, topología y algoritmos de enrutamiento.

Diseñar un algoritmo de enrutamiento de paquetes que gestione el tráfico basado en la reglas de flujo de datos del motor de búsqueda.

Evaluar el rendimiento de la red utilizando el algoritmo propuesto, comparar el rendimiento con mecanismos actualmente utilizados y existentes en la literatura.

1.5 Marco de la Tesis

El proyecto pertenece a la línea de investigación de Inteconnection Networks del Departamento de Arquitectura de Computadores y Sistemas Operativos.

La investigación inicia con el estudio de algoritmos creados en la línea de investigación, en éste sentido los algoritmos actuales de Enrutamiento Distribuido

y Balanceado (DRB)[1] y Enrutamiento Predictivo Distribuido y Balanceado (PR-DRB)[2], son una base de inicio para la creación de algoritmos de acuerdo al estudio del patrón de comunicaciones de un motor de búsqueda. Los resultados preliminares de este estudio se publicaron en [9].

El proyecto se ha desarrollado utilizando datos reales de proyectos de investigación relacionados con el estudio de motores de búsqueda y simulación de su comportamiento, los cuales han sido publicados en [10], [11], [12] y corresponden a trazas reales obtenidas por Yahoo Search Engine en el año 2005. Esto nos ha permitido estudiar la arquitectura de un sistema real que nos permita validar los resultados de la investigación.

Este trabajo coordinado ha permitido extender los resultados de la investigación con la publicación de un modelo del simulador en [9] y el algoritmo de enrutamiento diseñado en [9].

1.6 Principales Contribuciones

Las principales contribuciones del presente trabajo se resumen en:

- **1.** Política de enrutamiento basada en la aplicación de un SES.
- **2.** Modelamiento del tráfico de un motor de búsqueda para evaluación del rendimiento de la red.
- **3.** Metodología de evaluación del tráfico de una aplicación distribuida.

El diagrama de la Figura 1.1 muestra la relación entre las fases de la investigación y los resultados obtenidos.

[1] DRB: Distributed routing balancing
[2] PR-DRB: Predictive Distributed routing balancing

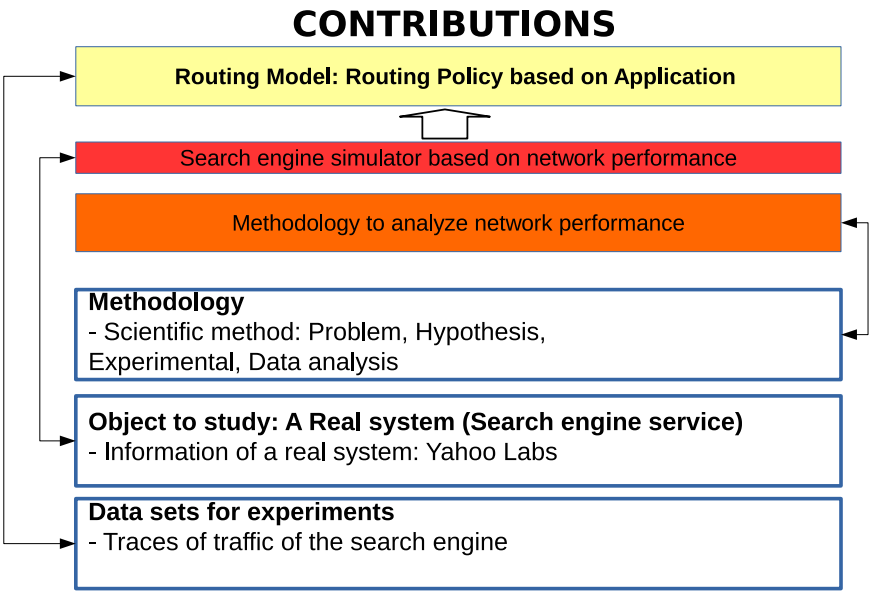


FIGURE 1.1: Fases de la investigación y principales contribuciones

1.7 Metodología

La presente tesis ha sido desarrollada tomando como base el método científico. Se ha desarrollado un estudio del estado del arte sobre de redes interconectadas y modelos de enrutamiento relacionados a la topología de red utilizada en el caso de estudio de la presente investigación.

A continuación se ha desarrollado una fase de observación que está soportada por el estudio del tráfico de red de un sistema real y el estudio de la arquitectura del motor de búsqueda.

Se ha evaluado el comportamiento de la red y los factores que impactan en su rendimiento a fin de proponer una hipótesis basada en la gestión del tráfico teniendo en cuenta la arquitectura de la aplicación.

La hipótesis propuesta nos permite diseñar un modelo de gestión del tráfico que ha sido evaluado mediante el uso de técnicas de simulación. El modelo propuesto se ha evaluado mediante el uso de datos reales y se ha seguido un criterio evolutivo

del modelo a fin de obtener los mejores resultados en términos de rendimiento de la red.

1.7.1 Caso de Estudio

Mediante el análisis de un caso de estudio [13], se ha diseñado una metodología que conduce a diseñar y construir un algoritmo de enrutamiento adaptado al tráfico de un motor de búsqueda en estudio.

La metodología propone los siguientes pasos:

- **1.** Análisis de la arquitectura de la aplicación.
- **2.** Análisis de tráfico de la aplicación.
- **3.** Análisis de la infraestructura y topología.
- **4.** Análisis del patrón de comunicaciones.
- **5.** Diseño de un modelo de gestión del tráfico.

La metodología se representa por el siguiente diagrama que define el flujo de procesos a seguir.

1.7.2 Análisis de la Arquitectura

La arquitectura de la aplicación se analiza identificando los componentes distribuidos que la conforman. Cada componente es independiente en su funcionamiento y genera solicitudes para otro componente. Un componente que recibe solicitudes, las procesa de forma independiente y genera resultados para otro componente de la aplicación.

1.7.3 Análisis de Tráfico de la Aplicación

Es propósito de esta fase es, identificar la interacción entre los componentes de la aplicación, definir qué componentes inician una secuencia de mensajes y, hacia qué componentes los envían. El resultado es un secuencia de flujo de mensajes entre componentes.

1.7.4 Análisis de la Infraestructura y Topología

Esta etapa de la metodología se enfoca en el estudio de la forma de organización de los nodos del sistema, el esquema de conexión (enlaces) entre cada nodo y equipos de comunicaciones.

1.7.5 Análisis del Patrón de Comunicaciones

Esta fase analiza el tráfico real de la aplicación para identificar a lo largo del tiempo que nodos se comunican de forma repetitiva y cómo se relacionan con la ubicación en la topología.

1.7.6 Diseño de un Modelo de Gestión del Tráfico

Esta fase de la metodología, depende de la aplicación que se analiza. El objetivo es diseñar un mecanismo que altere el comportamiento por defecto de los equipos de comunicaciones para comportarse de acuerdo a la demanda de la aplicación.

1.8 Organización de la Tesis

En el presente capítulo se ha expuesto una visión general de los servicios de motores de búsqueda, la información que procesan, los resultados que proveen, los problemas que afrontan, los sistemas que los soportan y porque motivan un estudio de su ambiente de comunicaciones. A fin de proveer soluciones se ha propuesto un objetivo general que se apoya en un conjunto de objetivos específicos. Se presenta la metodología utilizada a los largo de la investigación.

El capítulo dos, aborda los fundamentos teóricos para la evaluación del rendimiento de la red de un sistema, topología y patrones de tráfico. Además se expone las bases teóricas del motor de búsqueda en estudio, un análisis de la topología de red que utiliza. Se definen los criterios relevantes para la evaluación de la red y se resumen las políticas de enrutamiento actualmente existentes en la literatura.

El capítulo tres, plantea el problema en estudio con un análisis de la arquitectura del motor de búsqueda y el patrón de comunicaciones que genera. se evalúa el impacto de la aplicación en la red de comunicaciones.

El capítulo cuatro, presenta un modelo de enrutamiento basado en el tráfico de la aplicación, se detalla el diseño del modelo y su implementación. Adicionalmente se expone el modelo de simulación del motor de búsqueda diseñado como herramienta de experimentación para la investigación.

El capítulo cinco presenta el diseño de los experimentos llevados a cabo para validar tanto el modelo de simulación del motor de búsqueda como el algoritmo de enrutamiento propuesto.

Finalmente el capítulo seis presenta las conclusiones de la presente investigación, las líneas abiertas para investigación en el futuro y las publicaciones realizadas.

Capítulo 2

Marco Teórico

2.1 Fundamentos de Redes Interconectadas

Redes interconectadas se refiere al área de las ciencias de la computación que estudia los mecanismos de intercambio de mensajes entre dos elementos de cómputo (nodos). Cada nodo, para comunicarse con otro requiere de un medio de comunicación. La forma en que los nodos se enlazan se refiere a la topología [1], [14]. Durante el proceso de comunicación existen condiciones que demandan que los nodos cumplan con reglas básicas para asegurar que los mensajes se entregan correctamente. Las condiciones se refieren al cumplimiento de reglas en la gestión del envío de mensajes, como tamaño de los mensajes, ó mecanismos para decidir cuando se presentan situaciones de transmisión simultánea de mensajes que generan competencia por el uso de un medio de comunicación.

Los desarrollos científicos hasta la actualidad permiten que el uso de redes interconectadas sea de gran escala y casi omnipresente en la provisión de servicios electrónicos. "La mayoría de las aplicaciones almacenadas en los grandes centros de datos están soportadas por entornos de programación paralela" [15]. En

el presente trabajo nos centramos en el ámbito de redes interconectadas para la provisión de servicios de motores de búsqueda.

2.1.1 Topología

La topología de red define el esquema de interconexión de nodos. Para el presente estudio nos centramos en una topología de árbol (Fat-tree) [16] que es la que se utiliza en el sistema real en nuestro objeto de estudio.

2.1.2 Patrón de Tráfico

La operación y rendimiento de la red, está relacionado con los mensajes que debe procesar. Por lo tanto el estudio del flujo de mensajes en la red es necesario para evaluar su rendimiento. El análisis del patrón de comunicaciones permite analizar el comportamiento de la red frente a diferentes situaciones de operación que permite diagnosticar posibles condiciones de reducción de su capacidad por gestión ineficiente de los recursos de red. Existen dos métodos para la evaluación de la red mediante los patrones de tráfico. El primero es la inyección de cargas de datos generadas por modelos matemáticos que someten a la red a condiciones de sobrecarga de toda la red, ó un segmento de ella. El segundo método consiste en el uso de cargas reales de mensajes que son inyectados a la red para simular el funcionamiento de la red con mayor precisión respecto a la aplicación que la red soporta.

En la Figura 2.1 exponen tres patrones de tráfico representados gráficamente (por la relación (Fuente-destino) correspondientes a tráfico: (a) Uniforme, (b) Shuffle y (c) Bit Complemento. Además en se expone en (d) un patrón de tráfico real para la aplicación de Servicio de Motor de Busqueda.

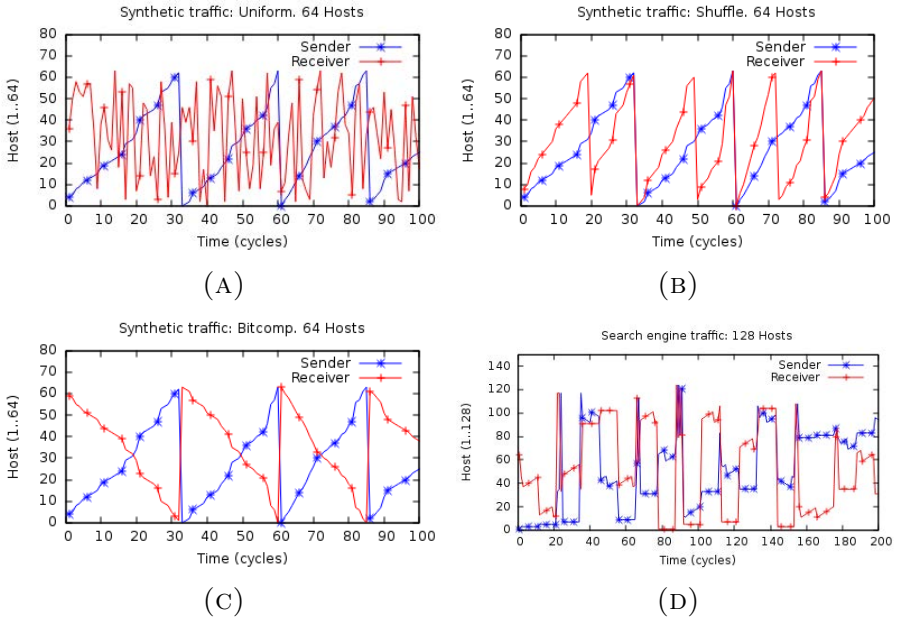


FIGURE 2.1: Representación gráfica de patrones de tráfico sintético, basada en [1]

Los patrones sintéticos crean situaciones hipotéticas que permiten analizar como los dispositivos de red y su capacidad responderán ante la presencia de tráfico de mensajes. Sin embargo un patrón real es obtenido de un sistema en producción (o simulado y calibrado como el sistema real) y que se utiliza para someter a un modelo a condiciones más cercanas a la realidad.

2.1.3 Mapa de Distribución de Carga de Red

La evaluación del rendimiento de la red, además se puede realizar mediante el mapeo de la carga que procesa cada par Fuente-Destino, este mecanismo permite identificar en que segmentos de la red hay sobrecarga de tráfico o ausencia de tráfico. Este análisis permite diagnosticar si el tráfico de la red se esta distribuyen uniformemente entre todos los recursos. Pero además si la distribución es eficiente para la demanda de la aplicación. No necesariamente la alta o baja concentración de tráfico significa una gestión deficiente del tráfico. La conclusión de uso eficiente

o ineficiente, debe ir ligada al rendimiento de la aplicación que la red soporta y al uso de los recursos de la red.

En la Figura 2.2 se observa un mapa del volumen de datos procesados por cada par Fuente (Eje X) y Destino (Eje Y). La carga procesada corresponde a tráfico sintético (intencionalmente creado) generado con un patrón Uniforme, es decir, donde todos los nodos tengan igual probabilidad de enviar datos. Se observa que el tráfico se ha distribuido por todo el espacio de direcciones. Existen zonas y pares que no han enviado datos, sin embargo este ausencia de tráfico está distribuida a lo largo de todos los nodos. La barra lateral izquierda, indica el volumen de datos que cada par ha intercambiado, se observa también que en general el *volúmen* de tráfico Uniforme y se ha distribuido entre todos los nodos de la red.

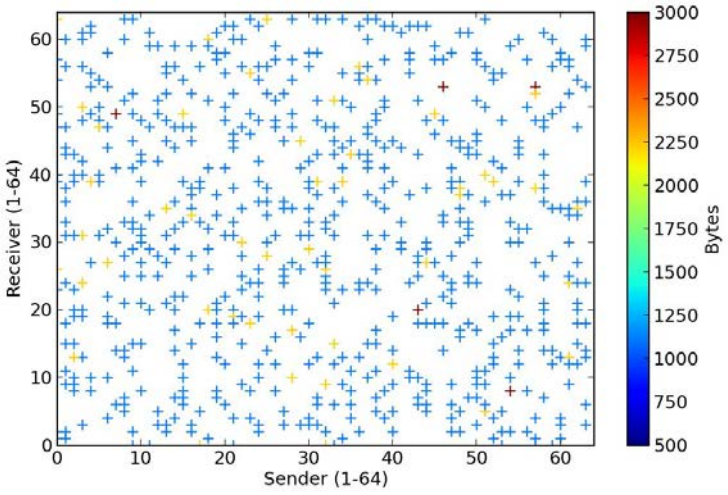


FIGURE 2.2: Mapa de carga de la red para tráfico Uniforme

2.2 Algoritmos de Enrutamiento

En [14] y [1] se expone algunos criterios para clasificar los algoritmos de enrutamiento en Redes Interconectadas. A continuación se expone un resumen de

estos criterios, a fin de ser considerados en la investigación actual para el diseño de un modelo de enrutamientos para un SES.

- **Determinista**: Utiliza una estructura da datos como tabla de enrutamiento que define que canal utilizar sean sea el destino de un mensaje.
- **Adaptativo**: Las decisiones sobre el enlace a utilizar cambian en bases a condiciones en tiempo real. Entre los criterios utilizados pueden estar entre otros: la carga de trabajo o la disponibilidad de un recurso. La decisión para elegir un canal de salida puede involucrar información local del switch, es decir sin conocimiento del estado global (o parcial) de la red. El criterio Adaptativo además puede requerir de información de otro elemento (centralizad).

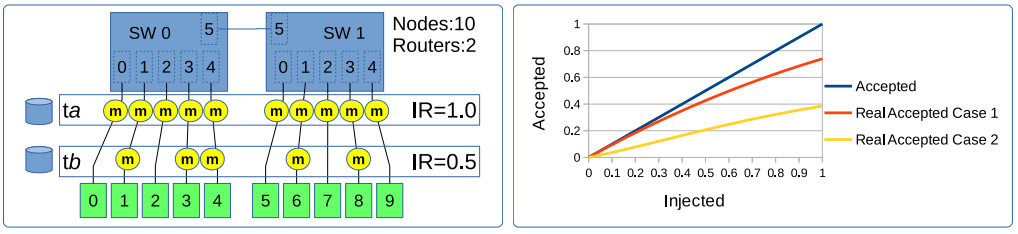
  Los mecanismos Determinista y Adaptativo, basan las decisiones en el uso (nivel de uso) de los recursos de la red.
- **Por su implementación**: Este criterio tiene que ver con la forma como se construye un modelo de enrutamiento, las decisiones de enrutamiento se pueden basar en el uso de los recursos de la red o el tráfico de la aplicación (traffic pattern).

2.3 Evaluación del Rendimiento de la Red

El rendimiento de la red, en general se realiza en base a dos indicadores: Latencia y Throughput [1], [14]. Un criterio generalmente aceptado es evaluar el comportamiento de la red comparándola con la demanda que ha recibido. La demanda es el conjunto de solicitudes que recibe la red desde un *Origen* para transmitir mensajes hacia un *Destino* en un periodo de tiempo determinado. Estas métricas también han sido utilizadas como base para definir medidas mas detalladas y específicas a la topología como se propone en [17].

Por ejemplo, en la Figura 2.3(a), se expone un conjunto de nodos (0 al 9) conectados por medio dos switches (SW0 y SW1). Se expone dos condiciones: (1) que los nodos envien mensajes en el tiempo *ta* y (2) que envíen en el tiempo *tb*. Si todos los nodos envían mensajes simultáneamente en el tiempo *ta* la demanda será del 100% de la capacidad de la red. El siguiente caso es que únicamente cinco nodos envíen mensajes, en este caso la demanda seria del 50% de la capacidad.

En la Figura 2.3(b), se analiza el rendimiento de la red estableciendo una relación entre la Demanda (eje X: Injected) y la Entrega (eje Y: Accepted). La mejor condición de rendimiento se presenta con la línea "Accepted", donde la relación Injected/Accepted es igual a 1, que define que la red Acepta la misma demanda que ha recibido en un periodo de tiempo. "Accepted" se refiere a entregado en el destino. Sin embargo por condiciones de competencia de mensajes por los mismos recursos de red (Contention) es posible que la relación de Injected/Accepted sea inferior a 1, en tal caso la relación se presente en la Figura como "Real Accepted Case 1" o "Real Accepted Case 2". "Real Accepted Case 1" evidencia una reducción en los mensajes entregados, lo cual significa un decrecimiento del rendimiento a partir de un volumen de entrada (es decir a partir de una tasa de entrada). "Real Accepted Case 2" muestra que la reducción del rendimiento fue desde el punto inicial, es decir desde la tasa de entrada más baja el rendimiento se redujo.



(A) Tasa de inyección (B) Medida de rendimiento

FIGURE 2.3: Esquema de manejo tasa de inyección por fracción de capacidad y representación de medida de rendimiento.

2.3.1 Latencia

Latencia de la red (Network Latency) según se define en [14] (p. 217), es igual al tiempo transcurrido desde que la cabecera de un mensaje (head) emerge del nodo origen para entrar a la red hasta que la cola (tail) del mensaje sale de la red para entregarse al nodo destino. En [1] (p.55) se define Latencia en términos de paquetes, y se introduce el concepto de Latencia de cabecera (latency head Th) y latencia de serialización (serialization latency)

2.3.2 Throughput (T)

El Throughput de la red, se define para un periodo de tiempo de operación (intervalo). Define el rendimiento de la red en comparación con la demanda que recibe, es decir es la proporción que existe en un periodo de tiempo entre la demanda y la entrega de mensajes.

En [1] (p.21) al *tráfico aceptado* se define como "la tasa de tráfico entregado en los terminales de destino".

2.4 Servicios de Motores de Búsqueda

2.4.1 Visión Global de un Servicio de Motor de Búsqueda

Una vista de alto nivel de un servicio de motor de búsqueda es, un usuario enviando una petición (expresada en una cadena de caracteres) hacia un sistema remoto que retornará una lista de enlaces relevantes a la solicitud; el sistema dispone de una base de información previamente recopilada para proveer los resultados mas relevantes a la petición. El criterio para definir la relevancia ha sido abordado en [18],[19],[20], [21] entre otros. En general si se trata de un motor público de

búsqueda en Internet la recolección se basa en un método de web-crawling [22],[23]. Otra vista del sistema es en términos funcionales de sus componentes, hay un sistema entrada de las solicitudes, un sistema de almacenamiento del contenido y un sistema lógico de clasificación y búsqueda de resultados relevantes. Ésta vista funcional se complementa con un sistema de interconexión de los componentes (red de datos). Solo asuntos relacionados al diseño de la red demanda gran interés científico [24],[25],[26],[27],[28]. Además un SES puede perfilarse mediante los siguientes atributos de operación, número de documentos almacenados, volumen de información almacenada, cantidad de nodos que almacenan la información, capacidad de los nodos de cómputo, capacidad de la red de interconexión y topología de interconexión de la red. En el caso de motores de búsqueda de alcance global o regional bien se puede hablar de miles de nodos interconectados y en el caso de SES para empresas multinacionales se hablaría de centros de datos con cientos de nodos. Estudios específicos sobre el tamaño de estos centros de datos han sido publicados en [29], [30]. Estos atributos numéricos conducen a analizar el sistema en términos de la capacidad de la red que soporta la comunicación entre sus elementos (hardware y software) y la eficiencia en la gestión de los recursos disponibles.

2.4.2 Componentes Principales de un Servicio de Motor de Búsqueda

Un típico motor de búsqueda tiene tres principales componentes: Front Service (FS), Index Service (IS) y Cache Service (CS) [11]. El SES procesa las consultas (Q) de los usuarios por medio del Front Service. El Front Service distribuye las consultas a un conjunto de nodos Cache Service. Cada nodo CS verifica si la consulta ha sido resuelta previamente. El Cache Services retorna un éxito o fallo hacia el Front Service. En caso de falló de la consulta, el Front Service envía la consulta Q hacia el Index Service. El Index Service busca en la base de datos de enlaces a

documentos y genera una lista con los documentos relevantes a Q (la lista generada se conoce como Top K) y la envía hacia el Front Service. Finalmente el Front Service envía la lista Top K hacia el usuario. En el caso de Google la arquitectura se compone de los siguientes tres componentes principales: Web Server, Index Server y Document Server [31]. Autores de [32] consideran los siguientes componentes: Crawler and Indexer Query Analyzer, Ranking Function User Interface. SPIRIT [33] utiliza básicamente: User Interface, Indexes, Metadata, otras publicaciones sobre la arquitectura y componentes se han publicado en: [34], [35], [36], [37].

2.5 Trabajos Relacionados

Investigaciones relacionadas a redes interconectadas proponen adaptar los recursos de red a la demanda, este enfoque regresa hasta la propuesta de [38], acerca de Redes Activas (Active Network) donde los routers ejecuten operaciones con los mensajes que fluyen sobre ellos. Estudios como [1] describen como el patrón de tráfico de las aplicaciones tienen impacto en el estado de la red y como los algoritmos Adaptativos pueden tomar decisiones basadas en el estado de la red.

Algunas técnicas de gestión de los recurso de la red han sido propuestos. Autores en [39] describen Generic-Adaptive-Resource-Control (GARC) como un mecanismo de control de a conectividad para mejorar el rendimiento global, GARC es un mediador entre las aplicaciones y la red.

Técnicas mas específicas como algoritmos de enrutamiento permite balancear la caga de la red. [40] presenta Application-Specific Routing Algorithm (APSRA) para modelar la aplicación utilizando grafos y el resultado del modelamiento es una tabla de enrutamiento estático.

Como se ha mencionado en la sección 1.5 la presente investigación se enmarca en la línea de investigación de Interconnection Networks, en la cual se ha desarrollado

algoritmos de enrutamiento como DRB (Distributed Routing Balanced) y PR-DRB (PR: Predictive Routing) que se han publicado en [41],[42],[43]. DRB permite mejorar el rendimiento de una red por medio de un mecanismo que distribuye la carga de trabajo y flujo de datos de forma equitativa entre los posibles caminos para mantener una baja latencia. PR-DRB, incorpora una característica que mantiene una lista de las mejores rutas de forma histórica para ser las primeras en utilizarse cuando un paquete busca el mismo destino. Por un lado DRB tiene un enfoque que no considera el patrón de comunicaciones de la aplicación mientras PR-DRB extiende la propuesta con enfoque en el patrón de comunicaciones. Ambos métodos han sido experimentados con aplicaciones científicas en ambiente de HPC

Los autores en [44] exponen la importancia de los algoritmos de enrutamiento para el rendimiento y proponen un algoritmo uniformemente balanceado (UGAL: Universal Globally-Adaptive Load-balanced). En la misma linea de investigación en [45] se propone el algoritmo enrutamiento adaptativo indirecto (IAR: Indirect Adaptative Routing), UGA y IAR se basan en utilizar información que no necesariamente está disponible localmente a un router, sino que mantienen información del estado global de la red introduciendo mensajes adicionales al tráfico para actualizar el estado de la red.

En relación a la arquitectura de un SES se han publicado algunas investigaciones acerca la planificación de la capacidad de motores de búsqueda como se expone en [11]. Además en [10] se propone una metodología de simulación de la operación de un motor de búsqueda y los autores detallan la arquitectura del sistema.

En la industria, la tecnología disponible para redes de datos de gran escala, como Infiniband [46], en la proveedores como Mellanox [47] soportan modelos de enrutamiento como Min-Hop (que realiza un proceso de cálculo de tablas de enrutamiento de camino mínimo), UPDN (basado en la detección de nodos raíz y análisis de información histórica del tráfico, con versiones modificadas para

topología Fa-tree). LASH (LAyered SHortest Path Routing) método determinista, que calcula la ruta más corta entre dos nodos y la almacena una tabla de enrutamiento, estos modelos están documentados en [48]. Los autores de [49], [50]. [51], [52] realizan una comparación de modelos de enrutamiento en Infiniband.

Acerca del modelamiento de un SES, una metodología fue introducida por [10], donde se propone simular completamente el funcionamiento de un SES, los autores proponen el modelamiento de los servicios con computación paralela y utilizan benchmarks para medir el costo de los servicios (Index Service, Cache Service, Front Service) y la red. Los autores en [12] presentan la especificación de un SES mediante Simulación de Eventos Discretos. El modelo simula las consultas de los usuarios como un mensaje moviéndose en diferentes escenarios para calcular el tiempo que toma resolver la consulta. Los autores compararon los resultados con una traza de ejecución de un motor de búsqueda real. La precisión de los experimentos de aquellas simulaciones permiten generar trazas para comparar con el sistema real. Adicionalmente, los resultados permiten extender los escenarios de prueba para ejecutar nuevos experimentos relacionados a otros componentes del sistema, como la red del centro de datos.

Para la simulación de la red, los autores de [1] proponen un simulador de red como una herramienta para analizar y evaluar el rendimiento de la red, se remarca la importancia de los requerimientos de la red para su diseño. En sistemas complejos como un SES el rendimiento esta basado en el número de consultas resueltas por unidad de tiempo, lo cual en otras palabras significa, en términos de rendimiento de la aplicación. Por lo tanto el modelamiento con tráfico real es fundamental.

Existen aspectos importantes relacionados al tráfico de una aplicación real, como la dependencia entre mensajes, los autores de [53] abordan este problema mediante la codificación de las dependencias en dos fases, la primera es una simulación completa para detectar y codificar las dependencias y la segunda fase es una nueva

simulación basada en le ejecución de ventanas de tiempo para asegurar que los mensajes relacionados son procesados en orden.

Capítulo 3

Planteamiento del Problema

3.1 Antecedentes

Un SES requiere del soporte de una red de comunicaciones eficiente, sin embargo para enfocarnos en mejorar los servicios de la red, es necesario conocer detalladamente los aspectos del tráfico que soporta la red, a fin de identificar los aspectos críticos que influyen en su rendimiento; por lo tanto en las siguientes secciones se expone el estudio de los aspectos más relevantes de un motor de búsqueda y que están relacionados con la presente investigación.

En la sección 1.1 se planteó dos interrogantes acerca de la planificación de un SES y como prever su capacidad de operación. Se ha delineado en términos generales la necesidad de contar con herramientas y técnicas que provean información relevante para tomar decisiones y definir la estrategia a seguir.

Con la primera fase de la investigación relacionada con: el estudio de la literatura, de la base teórica sobre redes interconectadas y trabajos relacionados, que se han resumido en el Capítulo 2 se ha sentado las bases acerca de los criterios e indicadores del rendimiento de redes, lo que permite abordar un fenómeno con el análisis en los criterios relevantes para la presente investigación.

El proyecto de investigación cuenta con el soporte de información de un motor de búsqueda real, según se ha expuesto en el marco de la tesis en el apartado 1.5, lo cual permite estudiar la arquitectura de un SES a fin de proponer posibles soluciones que puedan ser validadas con el uso de tráfico real.

3.2 Metodología

La presente sección expone la metodología aplicada para el estudio del problema de gestión del tráfico del SES. Ésta metodología nos conduce hacia el diseño de una propuesta con la función y capacidad apropiada para gestionar el tráfico de un sistema distribuido en particular.

La metodología propone los siguientes pasos:

- **1.** Análisis de la arquitectura de la aplicación.
- **2.** Análisis de tráfico de la aplicación.
- **3.** Análisis de la infraestructura y topología.
- **4.** Análisis del patrón de comunicaciones.
- **5.** Diseño de un modelo de gestión del tráfico y de un simulador del SES.
- **6.** Construir el modelo de gestión del tráfico.
- **7.** Evaluar el modelo de gestión del tráfico.

La metodología se representa la Figura 3.1, muestra relación entre cada fase y los recursos de entrada utilizados.

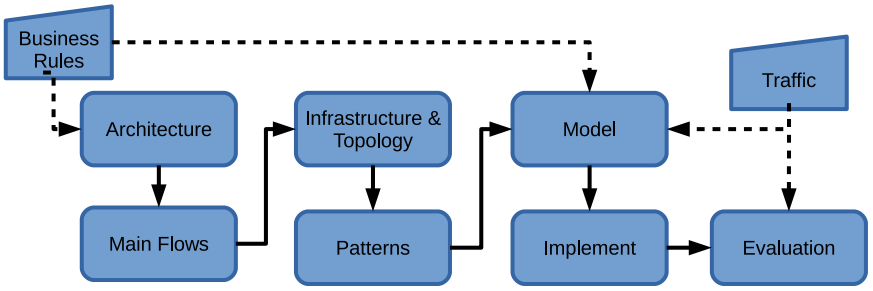


FIGURE 3.1: Resumen de la metodología aplicada para la evaluación del tráfico y modelamiento del SES enfocado en la evaluación de la red.

Las dos primeras fases: analizar la arquitectura y analizar los principales el tráfico de la aplicación están estrechamente relacionadas y toman como entrada el flujo de datos de la aplicación (Business rules). El estudio de la topología se realiza tomando en cuenta el patrón de tráfico que procesa y el análisis de la arquitectura.

El diseño del modelo toma como entrada el análisis de la topología y arquitectura de la aplicación. El modelo requiere estar basado en el flujo de la aplicación, se construye en a base a la arquitectura de la aplicación y se evalúa con tráfico real.

El fenómeno observado para la presente investigación se aplica como un caso de estudio y se presenta en la Sección 3.3.

La fase de análisis del patrón de comunicaciones se soporta con la observación del tráfico del caso de estudio. Se utiliza información de tráfico de un sistema real según se expuso en el marco de la tesis, Sección 1.5.

La sexta fase tiene dos aportaciones, primero el modelado del SES como herramienta de análisis y experimentación y segundo, un modelo de enrutamiento basado en la aplicación y la construcción de la política de enrutamiento.

Finalmente la séptima fase consiste en la experimentación y evaluación de las contribuciones.

3.3 Presentación de Caso de Estudio

En la Sección 1.7 se expuso la metodología utilizada en le presente investigación, la cual considera el estudio de un caso de estudio [13] que represente un sistema complejo, por lo cual se ha considerado una aplicación distribuida, como es el Servicio de Motor de Búsqueda (Search Engine Service).

3.3.1 Justificación del Caso de Estudio

La aplicación distribuida se elige al tratarse de un problema de redes interconectadas (Interconnection networks), lo cual demanda un conjunto de nodos interconectados que trabajan de forma coordinada para un propósito común. Este tipo de problemas ha sido abordado en el grupo de investigación en trabajos previos, que se han enfocado tanto en los fundamentos teóricos de redes, algoritmos de enrutamiento y tolerancia a fallos. El ámbito ha estado centrado en cómputo de altas prestaciones con aplicaciones científicas. Los problemas se han abordado con datos de experimentación que generan patrones de tráfico artificial generado por problemas comunes (operaciones con matrices), operaciones con bits (bit reversal) o modelos matemáticos.

En enfoque del presente proyecto es avanzar hacia la resolución de un problema con aplicaciones distribuidas, de cómputo de altas prestaciones en el ámbito industrial. Un sistema distribuido de amplio alcance típicamente provee el servicio a cientos o miles de usuarios e infraestructuras de decenas de nodos interconectados. Los servicios en línea son sistemas distribuidos con éstas características y un motor de búsqueda es un punto de acceso a los servicios en línea que permite a los usuarios encontrar un recurso.

El proyecto de investigación además propone utilizar recursos de una aplicación real, sin comprometer sistemas en producción o sistemas críticos que signifiquen un riesgo para organizaciones o personas.

El proyecto cuenta con el soporte de un grupo de investigación que aborda problemas relacionados con motores de búsqueda y ha provisto al proyecto trazas para experimentación anónimas. Ver Sección 1.5.

Los antecedentes expuestos permiten al proyecto tomar como caso de estudio un motor de búsqueda como un caso de estudio relevante par aplicar la metodología propuesta.

3.3.1.1 Delimitación del caso de estudio. Motor de Búsqueda Vertical (VSE: Vertical Search Engine)

Un motor de búsqueda puede ser de propósito general (aloja contenido de fuentes y origen diferente sin importar la localización geográfica) o puede definirse como vertical, alojando información de acuerdo a la ubicación geográfica o clasificada por un contenido en particular (comercial, científico, cultural, etc.).

El proyecto se enfoca en el estudio de un motor de búsqueda vertical, y utilizará información relacionada a topologías de tamaño de 64, 128, 256 nodos.

Los servicios del motor de búsqueda vertical están desplegados sobre una red física organizada con un esquema de árbol. Ésta topología tiene tres niveles (Core, Agregation, Edge). Los nodos de cómputo están enlazados al nivel tres como se propone en [16].

3.3.1.2 Requerimientos

Existen tres requerimientos fundamentales en este servicio de motor de búsqueda:

- **1.** El tiempo de respuesta de una consulta debe mantenerse bajo una cota determinada. (Latencia de Consultas)
- **2.** Cada nodo debe manejar una carga de trabajo determinada que permita al sistema en general manejar incrementos inesperados de solicitudes.
- **3.** La relación entre solicitudes recibidas y solicitudes resueltas debe mantenerse equilibrada (Throughput)

3.4 Arquitectura del Servicio de Motor de Búsqueda

La presente sección describe la arquitectura de los componentes de software del SES, plataforma y topología de red para la provisión del servicio. Además se expone un análisis del tráfico que la aplicación genera.

3.5 Arquitectura del Componentes de Software

El SES es una aplicación distribuida soportada por tres componentes principales: Front Service, Cache Service e Index Service. Paralelamente a estos servicio se ejecuta un componente adicional que se encarga de recopilar información de Internet. La Figura 3.2 muestra una visión global de la arquitectura del motor de búsqueda. Los detalles acerca de la arquitectura del motor de búsqueda han sido sobre la base de las investigaciones publicadas en [10] sobre la organización de un motor de búsqueda y flujo de datos y [11] sobre la arquitectura del motor de búsqueda.

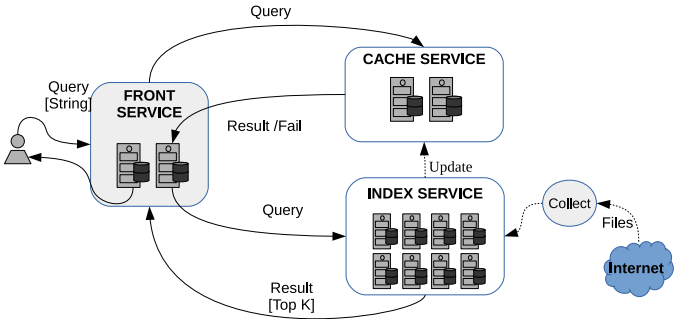


FIGURE 3.2: Principales componentes del Servicio de Motor de Búsqueda

Los componentes del sistema proveen los siguientes servicios, Front Service, recibe solicitudes de usuarios y entrega resultados. El Index Service, busca resultados a consultas en el contenido completo de datos alojados en el sistema. El Cache Service mantiene contenido dinámico sobre las consultas resueltas más frecuentes. El proceso de recolectar el contenido de Internet (en el diagrama: Collect) no es considerado en el presente estudio al ser un servicio gestionado paralelamente al SES y no estar relacionado directamente con la gestión de de la búsqueda.

Un SES esta soportado por un conjunto de nodos de cómputo de modesta capacidad interconectados entre sí por medio de una red de datos de alta velocidad. Cada nodo tiene un rol, el cuál puede ser FS, CS o IS. El Rol se define por la instancia de software que ejecuta (puede ser instancia de FS, IS o CS). La cantidad de nodos por cada tipo de servicio está previamente definida como configuración del sistema y ha sido calculada en el sistema real por medio de benchmarks y experimentos con cargas de datos reales.

Con el fin de reducir el tiempo de respuesta de los consultas y mantener una tasa de respuesta aceptable, los nodos de IS y FS se organizan en arreglos de PxD nodos de procesamiento (P: Particiones y D: réplicas). P indica el nivel de particionamiento de los datos y D es el nivel de replicación de cada servicio. La uso de réplicas además provee la capacidad de tolerancia a fallos de los nodos. En la Figura 3.3 se presenta un esquema básico de esta configuración.

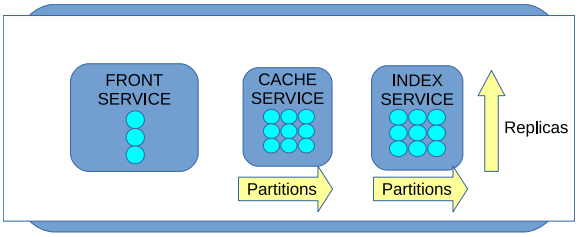


FIGURE 3.3: Esquema de particionamiento de réplicas de FS, IS, y CS

Los nodos ejecutado instancias de replica de Index Service y de Cache Servise se conocen como Index Server Replica (ISR) y como Cache Service Replica (CSR).

3.5.1 Arquitectura de la Red del Servicio de Motor de Búsqueda

Los nodos del SES están interconectados por medio de una topología Fat-tree y utilizan el esquema de direccionamiento que se propone en [16]. Los switches de la red están organizados en n niveles (Nivel 0 hasta Nivel n-1). Los nodos se conectan al Nivel *n-1* que es el nivel de borde (Edge). Las instancias de FS, IS y CS se despliegan de acuerdo a un esquema de mapeo previamente definido y calculado. La Figura 3.4 muestra un diagrama de como las instancias de FS, IS y CS se despliegan en la red. Los nodos se agrupan en PODs (Point of Delivery) y se genera un POD por cada switch de Nivel 0 de la topología.

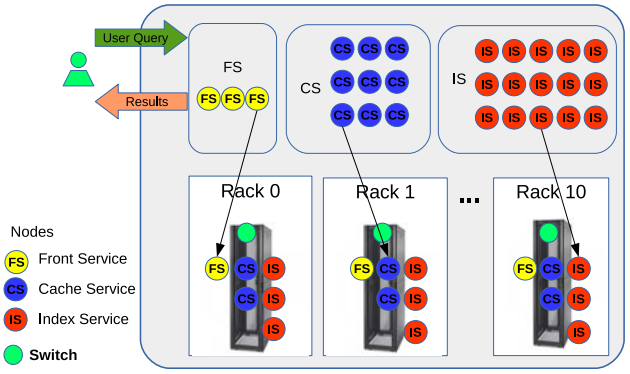
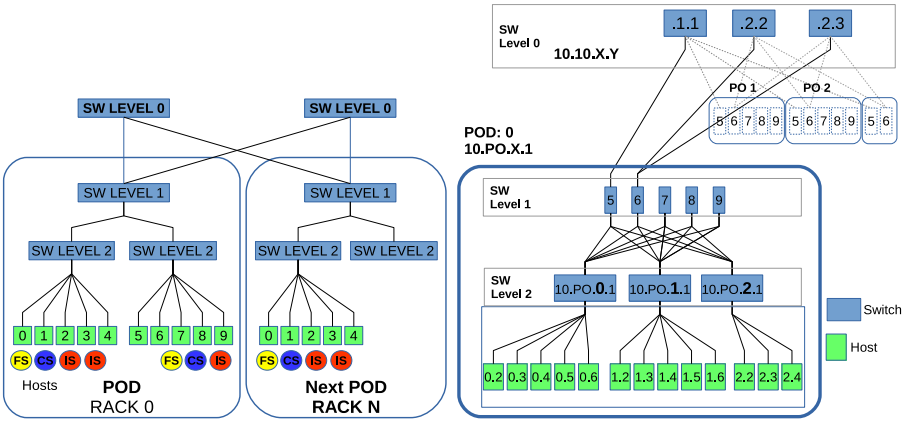


FIGURE 3.4: Esquema de despliegue de las instancias de FS, IS y CS

Un típico mapeo distribuye las instancias de IS de forma equilibrada entre todos los switches de borde, de modo que cada switch de borde dispone de una instancia de IS. Los CS y FS se distribuyen también de forma equilibrada, sin embargo puede darse el caso de que existen switches sin instancias de FS o CS.

Los nodos y switches se organización en diferentes POD y se conectan por medio del Nivel 0 de la topología. La figura 3.5a muestra un esquema de la organización de la topología y la Figura 3.5b muestra un esquema del direccionamiento.



(A) Organización switches (B) Esquema de direccionamiento

FIGURE 3.5: Esquema básico de organización de los switches y direccionamiento de la topología del SES

Por ejemplo, para el caso de una configuración de 128 nodos de tamaño de la red, cada switch de Nivel 0 (Core) forma un Punto de Entrega (Poinf of delivery), al cual se conectan los restantes dispositivos requeridos. En la Figura 3.6 se muestra un esquema para este ejemplo.

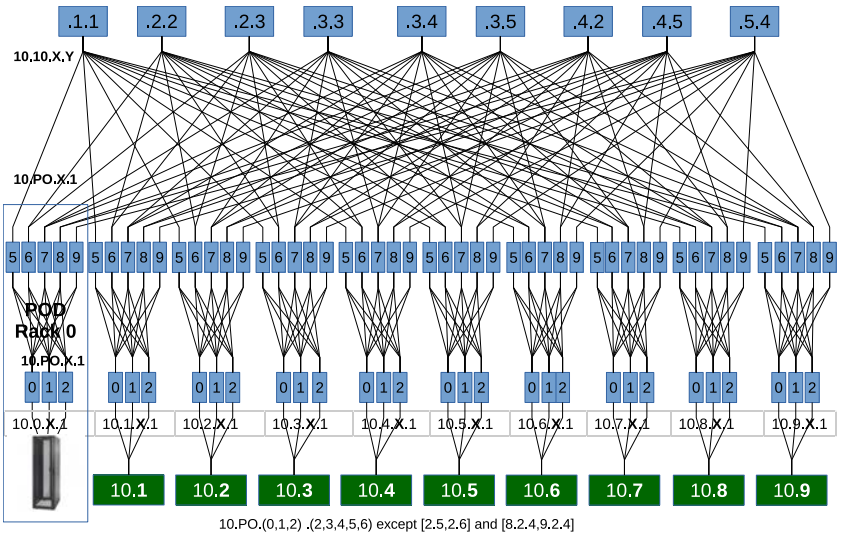


FIGURE 3.6: Esquema de conexión y direccionamiento para una configuración con 128 nodos

Al nivel de borde (Nivel 2) de la topología, a cada POD se conectan los nodos siguiendo el esquema de direccionamiento mencionado al inicio de ésta sección. Un esquema detallado se muestra en la Figura 3.7.

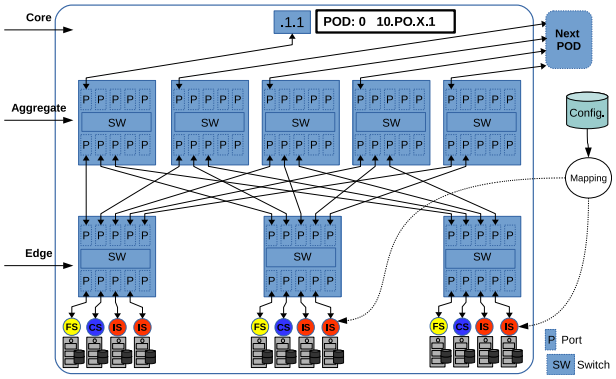


FIGURE 3.7: Esquema de conexión detallado al nivel de borde para una configuración de 128 nodos

3.5.2 Front Service

El FS acepta una consulta Q del usuario y después de ser procesada con el soporte del CS o IS se retorna una lista de documentos relevantes a la consulta. Una

consulta es una cadena de caracteres que esta compuesta por términos clave. El FS es un grupo de nodos FS (cluster FS). La consulta es un mensaje que contiene los siguientes atributos: la cadena de caracteres, un identificador único de consulta, un atributo de latencia de la consulta y un estado de la consulta.

Las consultas durante el procesamiento en el SES atraviesan por cuatro estados: *new*, *hit*, *no_hit* y *done*, que se describen a continuación. El FS cuando recibe la consulta, le asigna un identificador, inicializa el atributo de latencia en zero y le asigna el estado *new*, a continuación la envía a diferentes CS. Los nodos CS actualizan (incrementan) la latencia de la consulta y devuelven la consultas con estado *hit* o *no_hit*. El FS espera por la respuesta de todos los CS y verifica si alguno ha retornado la consulta en estado *hit*. En caso de existir un estado *hit*, los resultados (Top K) se devuelven al usuario, caso contrario la consulta se envía al IS, antes del envio el FS actualiza (incrementa) el atributo de latencia. El FS distribuye las consultas basado en algoritmos (como Round Robin) para balancear la carga entre diferentes nodos IS. El IS realiza la búsqueda y actualiza la latencia de la consulta. La Figura 3.13 muestra un esquema del flujo de una consulta entre FS, IS y CS.

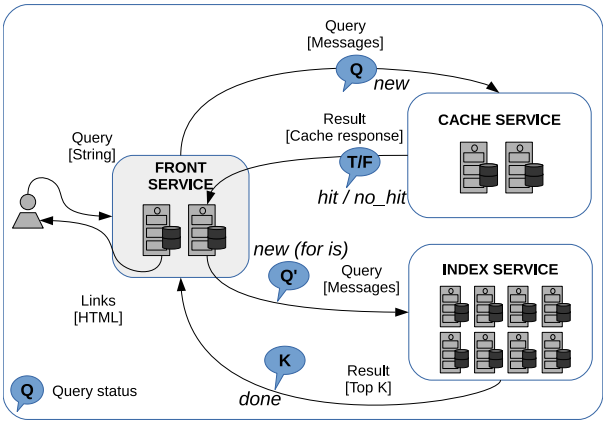


FIGURE 3.8: Esquema del flujo de una consulta en un SES. FS envía una consulta al CS. CS retorna Exito o Fallo. En caso de Fallo la consulta es envíada hacia el IS.

3.5.3 Cache Service

El CS mantiene un conjunto de las más frecuentes consultas y sus resultados, que permite reducir la carga del SES al resolver las consultas mediante en envío de resultados previos al usuario. Existen dos tareas básicamente del CS, la primera es el procesamiento de nuevas consultas y la segunda, es mantener actualizado su contenido para resolver consultas más rápidamente. La operación del CS involucra el tiempo para balancear la carga entre los nodos. Simultáneamente el CS procesa resultados previos aplicando políticas de actualización del CS. CS actualiza el estado de la consulta de *new* hacia *hit* o *no_hit* e incrementa la latencia de la consulta (QL). El CS es un grupo de nodos mas pequeño que el IS que requiere de menos carga de la red de datos.

3.5.4 Index Service

El IS se encarga de generar una lista de los Top K documentos relevantes para un consulta Q ingresando al contenido completo del SES. Por lo tanto existe un proceso paralelo que recopila documentos desde Internet para crear una colección de documentos. Cada documento está asociado con un identificador único. Los documentos son explorados para extraer una lista de términos relevantes. El resultado de ésta exploración es una estructura de datos llamada "Inverted Index". Esta estructura puede llegar a ser enorme (medida en términos GB), por lo tanto es necesario distribuirla entre los hosts IS. El IS utiliza el Inverted Index para recuperar documentos relevantes a una consulta y los resultados pueden estar almacenados en distintos nodos. Para la búsqueda se aplica un algoritmo de ranking para generar la lista de los Top K documentos relevantes a la consulta. Top K es una lista con los enlaces hacia la ubicación del documento original. El IS actualiza el estado de la consulta desde *no_hit* hacia *done* e incrementa la latencia de la consulta (QL).

3.5.5 Análisis del Patrón de Tráfico

Esta sección presenta el análisis del tráfico real de la aplicación para obtener una vista global de la carga de la red y definir los requerimiento para la gestión del tráfico.

Inicialmente se identifica a lo largo del tiempo que nodos se comunican de forma repetitiva y cómo se relacionan con la ubicación en la topología. El mecanismo utilizado para representar la repetitividad es, asignar un número único a cada nodo de la red, por ejemplo para una red de 128 nodos, cada nodo se numera del 1 al 128. La repetitividad se presenta cuando a lo largo del tiempo un subconjunto de nodos se vuelve a comunicar con un mismo subconjunto de nodos.

Tradicionalmente el estudio del rendimiento de la red se evalúa con el análisis de tráfico artificial o sintético (generado por un modelo matemático). El tráfico sintético se puede representar gráficamente y permite evidenciar un patrón de repetición. Algunos los patrones sintéticos se han comentado y representado la Sección 2.1.1 y Figura 2.1.

Ahora nos enfocamos en el análisis del tráfico del SES. En la Figura 3.13 se presenta un experimento que captura en un periodo de tiempo los pares de nodos del SES que se comunican, en el eje X, se presenta el tiempo (en ciclos de simulación) y en el eje Y los nodos que se comunican. Los nodos fuente y destino están enumerados desde el 1 hasta el 128. Se puede observar que para una secuencia de nodos Fuente los nodos Destino se desplazan verticalmente (nodos fuente con identificación menos a 10 y nodos destino desde 1 hasta 128), se observa que siguen un patrón de comunicación que se desplaza desde los nodos numerados más bajos hasta más altos, lo cual significa que la comunicación ha fluido solo entre un segmento reducido de la red para un periodo de tiempo específico. A continuación se observa que los nodos Fuente, se desplazan hacia otro segmento de la red (entre los nodos 10 y 20), y los identificadores de Destino nuevamente se desplazan desde

abajo hacia arriba. Este comportamiento se repite mas adelante (a la derecha del gráfico), pero intercambiando los nodos Fuente y Destino. El comportamiento descrito define una tendencia de uso de la red que se concentra en un segmento de la red utilizada a la vez. Por lo que se puede decir que no existe una distribución de uso extendido hacia toda la red.

Por lo expuesto en el párrafo anterior, se puede concluir que el uso de la red está focalizado hacia un segmento de la red y por lo tanto hay criterios para decir que únicamente ciertos recursos son utilizados simultáneamente y hay recursos de la red que no tienen la misma carga de trabajo.

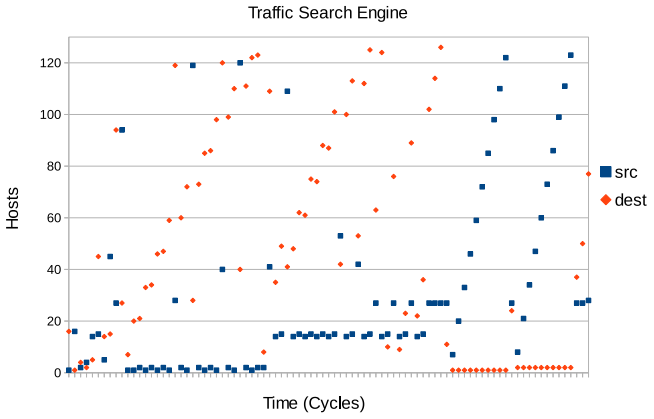


FIGURE 3.9: Vista de pares Fuente-Destino durante el envío de mensajes a lo largo de un periodo de tiempo con una configuración de 128 nodos.

La Figura 3.10 complementa el análisis anterior, con la presentación en forma de mapa de comunicación de los pares Fuente-Destino. Se puede observar que un conjunto reducido de nodos se comunican y que algunos pares no intercambian mensajes (definido por las áreas vacías). Este comportamiento se puede deducir también desde la arquitectura del SES, ya que los nodos IS y CS no intercambien mensajes. Al menos no lo hacen como parte del funcionamiento del buscador durante la resolución de consultas ya que podrían comunicarse por medio de otro proceso o canal de comunicación que actualiza del contenido del CS (según sean las reglas de actualización del CS). El volumen de comunicación entre los pares

también depende de donde fueron resueltas las consultas (por ejemplo algunas son resueltas por el CS y otras requieren ir al IS). Se observa también bloques de nodos que son secuenciales, es decir hay segmentos de la red donde se concentra el intercambio de mensajes (es un comportamiento que define localidad); en este caso los nodos podrían estar conectados al mismo POD o al mismo switch. Para los nodos que están en el mismo POD o Switch el recorrido que realizan es mas cercano y la ubicación y distancia que recorre un mensaje están directamente relacionadas con el tiempo que toma su entrega, ya que el recorrido es más corto.

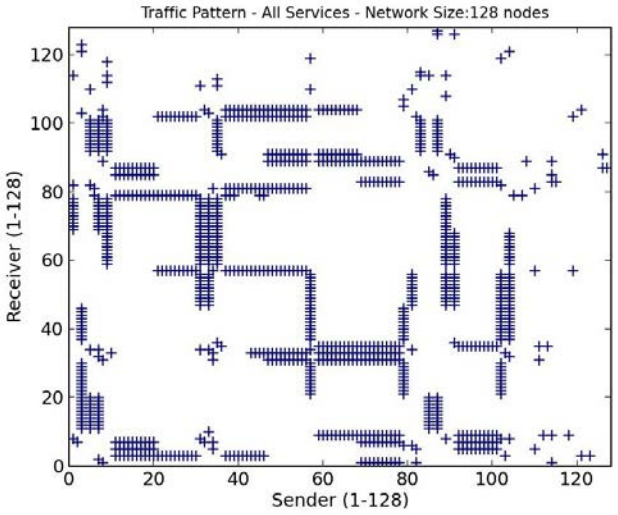


FIGURE 3.10: Mapa de los pares Fuente-Destino que se comunican acumulado en un periodo de tiempo para una configuración de 128 nodos.

Hasta el momento se ha analizado los mensajes solo como Fuente y Destino, sin analizar la aplicación. Para una revisión más detallada el tráfico se analiza por el servicio al que pertenece. En este caso es necesario tomar un periodo más corto de tiempo para clasificar el contenido en FS, IS y CS. Utilizamos el mismo tipo de gráfica y los resultados se muestran en la Figura 3.13, se observa el tipo de servicio de cada mensaje, la gráfica muestra que los nodos de FS generan carga para un segmento de la red en cuánto a nodo destino se refiere, sin embargo el nodo Fuente corresponde a una distribución a lo largo de toda la red. Lo mismo ocurre para los mensajes de CS, pero en sentido contrario. Y, finalmente el tráfico de nodos

IS se despliega con mayor cobertura de la red. Sin embargo hay segmentos con mayor concentración de tráfico y otros segmentos no tienen ocupación.

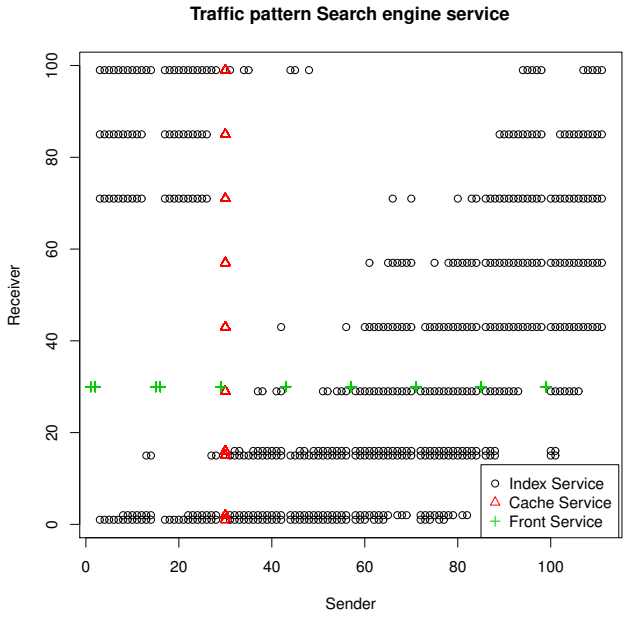
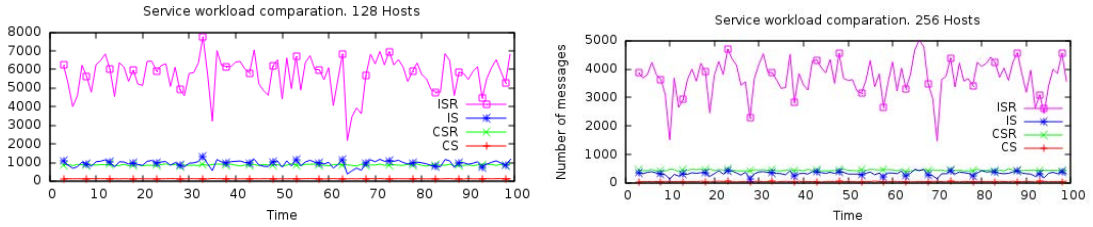


FIGURE 3.11: Mapa de los pares Fuente-Destino que se comunican acumulado en un periodo de tiempo separado por servicio: FS, CS, IS.

A continuación se expone la comparación del tráfico analizado por el volumen de carga de cada servicio. En la Figura 3.12a se observa la carga generada para una red de 128 nodos. Se observa que el tráfico de Index Service predomina el uso de red (incluido el tráfico de IS Réplica). Este comportamiento se repite con una red de 256 nodos, según se expone en la Figura 3.12b.



(A) Configuración: 128 Nodos (B) Configuración: 256 Nodos

FIGURE 3.12: Comparación de carga por el número de mensajes inyectados a la red clasificado por tipo de servicio IS, ISR, CS, CSR

3.5.5.1 Análisis de balanceo de carga

El tráfico generado por el motor de búsqueda genera una carga en los recursos de red, específicamente a nivel de la ocupación de los buffers y que a su vez, define que canales son los más ocupados y por lo tanto permiten analizar la demanda y la carga de la red.

El análisis de balanceo de carga se realiza por medio del estudio de utilización de los canales de cada switch (Buffer = B). Un mensaje se divide en paquetes que van de la fuente hacia el destino (S a D). Cuando un paquete llega a un switch (R), es asignado a un buffer para esperar que se le asigne un canal de salida. Los buffers tiene un tamaño (BS), y su tamaño es calculado de acuerdo a la carga de trabajo de la aplicación.

Este comportamiento se representa del siguiente modo. En el eje X se muestran los buffer de los switch de la red y se diferencia por color los switch de los niveles 0, 1 y 2 de la red. El nivel de uso (Ocupación del Buffer BO) se presenta en el eje Y.

La ocupación se analiza contabilizando los eventos en entrada de datos al buffer. Cuando la cantidad de buffers ocupados sobrepasa un umbral se incrementa un contador de ocupación. Se han definido tres niveles de ocupación: Bajo (L), Medio (M) y alto (H). Bajo cuando la ocupación está entre el 25% y el 50%. Medio cuando está entre 50% y 75% y Alto cuando la ocupación es mayor que 75%.

Cuando se analiza la ocupación de los buffers del tráfico del SES, se puede observar el desequilibro en el uso de los switches de la red. Ver Figura 3.13.

Un acercamiento hacia la ocupación que supera el umbral M, donde se observa el tráfico concentrado solo en un grupo reducido de switches.

El tráfico del SES se concentra solo en determinadas áreas de la red produciendo un desbalance de la carga de la red.

La asignación de recursos de la red debería ser equilibrada (Balanceo de carga) para utilizar de forma eficiente los recursos, sin que haya reducción en el rendimiento.

La red soporta una aplicación con un mismo propósito, aunque los mensajes pertenezcan a diferentes nodos con diferentes instancias, es decir el intercambio de mensajes debería responder a las reglas de flujo de mensajes definida en la Figura 3.13 que son las reglas básicas de flujo de datos entre el FS, IS y CS.

Por lo tanto la asignación de buffers y canales de salida podría considerar el tipo de servicio (componente) en la asignación de recursos. Otro criterio a tener en cuenta es el volumen de carga. Entonces la asignación de recursos debería tener en cuenta el volumen de cada servicio en la decisión de asignación del canal de salida.

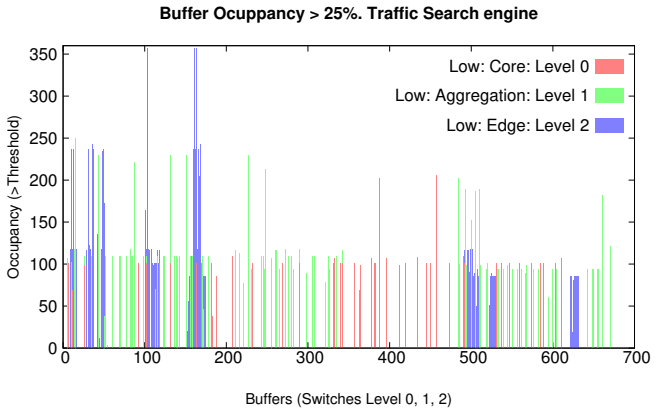


FIGURE 3.13: Ocupación de buffers en los tres niveles de la topología. Eventos que superan el 25% de la capacidad de los buffers en un periodo determinado.

3.6 Simulación del Motor de Búsqueda

La complejidad en la operación del motor de búsqueda nos conduce al uso de técnicas de simulación para abordar el problema en un ambiente seguro, de bajo coste y dinámico que permita modelar el funcionamiento real del sistema y de la

interacción entre sus componentes (hardware y software). Puesto que el enfoque de la presente investigación es en el rendimiento de la red, el modelo que se propone hace énfasis en la interacción del SES con la red, más que con los detalles internos de como cada componente procesa internamente la consulta.

La creación de un ambiente de simulación del sistema y la carga de demanda real puede aportar valiosas información para evaluar ambientes de configuración prometedores para el sistema real.

Una típica estrategia en la evaluación del rendimiento de la red está basada en el uso de cargas de datos generadas con modelos matemáticos y que se introducen en un simulador para evaluar el impacto en una configuración determinada. Este modelo es presentado en la Figura 3.14. El modelo basado en cargas sintéticas permite someter a carga controlada de la red y a aplicar condiciones de sobrecarga para evaluar una configuración de la red y como ésta puede responder a situaciones hipotéticas.

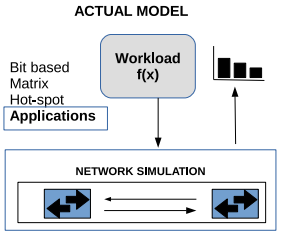


FIGURE 3.14: Modelo actual de simulación y evaluación del tráfico con inyección de patrones de carga generados con modelos matemáticos.

A fin de evaluar la aplicación de SES el enfoque mas preciso se espera conseguir con la utilización de tráfico real, mediante el uso de cargas de datos controladas por la aplicación (application driven).

Con la evaluación de la red del motor de búsqueda se espera modelar las condiciones más cercanas de tráfico real. Según se plantea en [14](p. 479) modelar una aplicación, "es una tarea compleja ya que el comportamiento de la red difiere de una arquitectura a otra y de una aplicación a otra". Por lo tanto aún en el

caso de utilizar modelos que generen alta demanda a la red (por la demanda de usuarios, capacidad de cada uno de los nodos o características de componentes de software), el uso de tráfico de la aplicación real es necesario para evaluar y calcular la capacidad de la red necesitada con precisión.

3.6.1 Recursos para el Diseño del Simulador del SES

Para el diseño del simulador simulador la información disponible incluye, conocer la demanda a la que será sometido el sistema (trazas reales). Se cuenta con la topología del sistema real (mapeo de nodos y ubicación en la topología) y tamaño de la red (número de nodos en diferentes configuraciones del sistema), rol que cumple cada nodo desplegado en la red (FS, IS y CS). Esta información ha sido expuesta en la Sección 3.4 junto con la arquitectura de software.

3.6.2 Características Básicas del Simulador del SES

Para la simulación se requiere diseñar un modelo de simulación que explote todos los recursos disponibles. Es decir los datos reales, la topología y la funcionalidad de la aplicación.

La simulación debería por lo tanto incluir;

- 1. Simulación de la la topología real del SES.
- 2. La entrega de mensajes entre los componentes de FS, IS y CS.
- 3. La simulación del tiempo procesamiento en IS, FS, y CS.

El nivel de detalle de la simulación se realiza con el uso de una versión modificada del simulador Booksim, reconocido en la comunidad científica y que ha sido documentado en [1] y ha evolucionado y incorporado nuevas funciones publicadas en [54].

Si bien el simulador tiene una nivel de detalle y precisión a nivel de Flit, que permite analizar la latencia de paquetes, no considera la interacción con una aplicación, por lo que ha sido necesario incorporar funciones para simular la aplicación de SES. Los cambios incorporados al simulador permiten la evaluación de la red del servicio de motor de búsqueda considerando tres aspectos básicos:

- 1. Cargas reales tráfico de red.
- 2. Modelamiento de Tiempo de servicios (FS, IS, CS) basado en tiempo real de trazas
- 3. Interacción de los servicios FS, IS y CS.
- 4. Dependencia de mensajes.
- 5 Análisis de resultados para reconfiguración de la red.

Una vista de alto nivel de la simulación del SES se muestra en la Figura 3.15. En primer lugar se utiliza trazas reales que consisten en mensajes entre cada instancia de FS, IS y CS. Las trazas generan cargas de trabajo de enviadas hacia el CS (W_{cs}) y cargas de trabajo enviadas hacia el IS (W_{is}). El simulador genera métricas de rendimiento (Latencia y Throughput) de la traza. Finalmente los resultados se evalúan para reconfigurar la red.

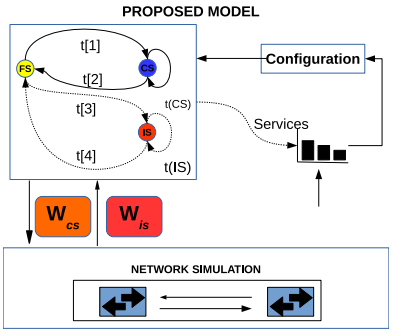


FIGURE 3.15: Modelo propuesto de simulación y evaluación del tráfico de red basado en una aplicación.

A nivel de la simulación de red se realiza un análisis de cada mensaje para simular la interacción de la aplicación y se generan las respuestas analizando la dependencia de mensajes correspondientes al mismo servicio.

3.6.3 Capa de la Aplicación

La simulación de la aplicación se realiza mediante el análisis de los datos de cada mensaje para identificar a que servicio pertenece y simular el tiempo de procesamiento de la consulta antes de ser enviada una respuesta hacia el siguiente servicio. Este modelo se representa en la Figura 3.16.

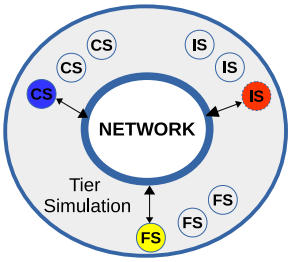


FIGURE 3.16: Modelo básico de alto nivel para la simulación del Search Engine Services con enfoque en el rendimiento de la red.

En la capa interna, el simulador procesa los mensajes de cada servicio, y se realiza la simulación del intercambio de mensajes a nivel de los equipos de red (switches, buffers).

3.6.4 Simulación de la Aplicación

La simulación de la aplicación, es una capa que se incorpora al simulador con el procesamiento de una traza. A continuación cada mensaje es asignado a una cola de inyección en cada nodo con el tiempo de simulación obtenido de la traza.

La Figura 3.17 muestra una vista mas detallada del proceso de simulación. Para la entrada se utiliza trazas reales, que se someten a un proceso de análisis (trace

parcing) para crear la topología en base a un archivo de configuración del simulador. A continuación se simula la aplicación por la inyección de los mensajes hacia el simulador de red (Simulate services). El simulador de red procesa los mensajes (m) y genera respuestas (ḿ) para nuevamente simular los servicios (FS, IS y CS). Finalmente en base a la respuesta del simulador nuevos mensajes son inyectados a la red.

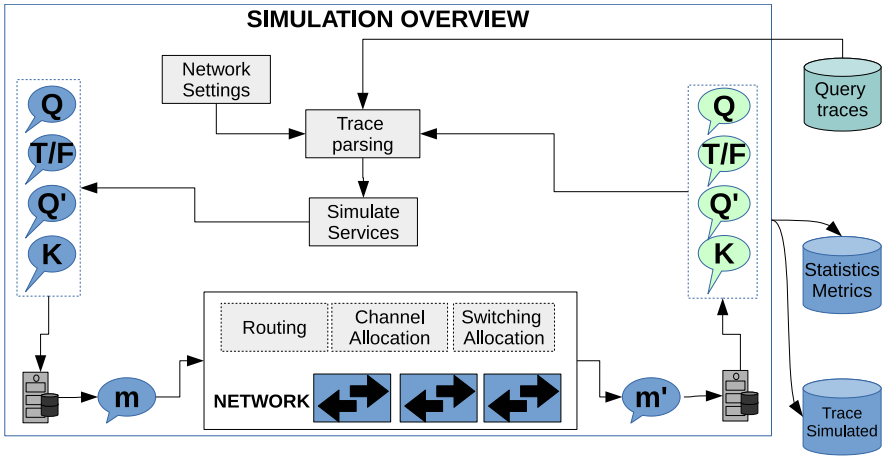


FIGURE 3.17: Diagrama del proceso de simulación de la aplicación y del simulador de red.

Durante el proceso de inyección de tráfico simultáneamente se genera estadísticas para evaluación de las métricas (Statistics Metrics) de rendimiento de la red. El simulador además genera una traza simulada para ser comparada con la traza real (Trace simulated).

La traza (Query traces) tiene dos tipos de eventos del SES. Un evento es *procesamiento en los nodos* y el segundo tipo de evento es *mensaje entre los nodos*. La lista de atributos de cada evento se describen en la Tabla 3.1.

La Figura 3.18 expone una muestra de un mensaje enviado del Front Service hacia el Cache Service y la respuesta correspondiente. La columna *Data* contiene un *identificador "q1"* de la consulta. En la figura (para facilitar la lectura) se ha agregado dos columnas *Service* indicar el rol que ejecuta cada *Fuente* (Src) y *Destino* (Des).

TABLE 3.1: Lista de atributos de cada mensaje de la traza

Timestamp	Indica la hora en que ocurre el evento
Src	Identificador único del host en la red. En la traza en una dirección IP.
Event	Cadena de texto. Contiene "MSG" si trata de un evento de envío de mensaje. "CPU" para indicar tiempo de procesamiento del servicio
Size	Tamaño del mensaje en bytes
Des	Identificador único del host en la red. En la traza en una dirección IP.
CPU	Duración del evento para el caso de tiempo de procesamiento del mensaje (Eventos tipo "CPU").
Data	Cadena de texto. Contiene un número que es el identificador de la consulta a la que pertenece el mensaje.

Time	Src	Service	Event	Size	Des	Service	CPU	Data
ms	IP			Bytes	IP		ms	string
11500	**1**	**"FS1"**	**"MSG"**	8	30	**"CS"**	0	q1
13200	30	"CS"	"CPU"	0	30	"CS"	110460	q1
121960	1	"FS1"	"CPU"	0	1	"FS1"	11500	q1
13200	**30**	**"CS"**	**"MSG"**	5	1	**"FS1"**	0	q1

FIGURE 3.18: Muestra de traza con un mensaje generado por FS para el CS para una consulta de ejemplo Q1 enviada únicamente al Cache Service.

Una vez que el mensaje es movido a la capa del simulador, los mensajes de respuesta de cada nodo son obtenidos de una cola de mensajes retirados que genera el simulador. La cola de mensajes removidos indica que se debe leer el mensaje de respuesta correspondiente de la cola de inyección de servidor (host) del cual ha sido removido.

Una secuencia de ejecución se muestra en la Figura 3.19. La secuencia de simulación tiene cuatro fases. La primera fase, corresponde a un mensaje de FS hacia CS, La segunda fase es la respuesta de CS a FS, la tercera fase es de FS hacia IS y la cuarta fase corresponde a la respuesta de IS a FS.

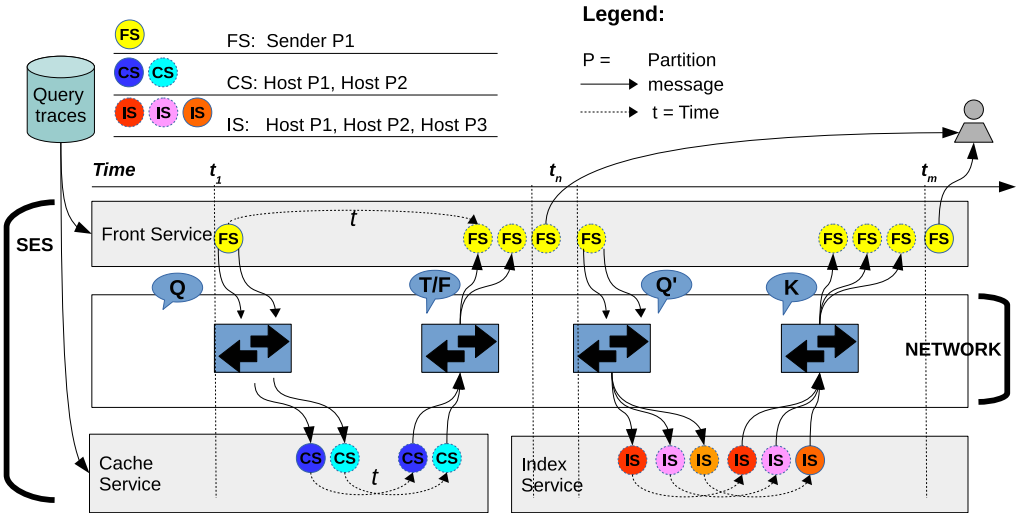


FIGURE 3.19: Secuencia de simulación de los servicios y el simulador de red.

La primera fase, inicia con la lectura de un mensaje de la traza, (corresponde a un mensaje del FS), el mensaje es movido a la capa de red. El mensaje en la capa de red sigue el proceso de simulación de red normal, hasta ser colocado en la una cola de mensajes retirados.

En la segunda fase, cuando el mensaje inicial FS está en la cola de retirados, se identifica a que nodo de destino perteneció. De la cola del nodo de destino se lee el mensaje correspondiente a la respuesta (el mensaje correspondiente es un mensaje de CS que pertenece a la misma consulta). El mensaje de respuesta no es procesado hasta que hayan llegado todos los mensajes de la misma consulta. Cuando todos los mensajes han llegado, se simula el tiempo de procesamiento del servicio (FS, IS, CS) mediante la comparación con el tiempo de simulación, una vez cumplido el tiempo de procesamiento los mensajes se inyectan a la red como respuesta.

La tercera y cuarta fase se realizan solo en el caso de haber mensajes en las colas de los nodos que se envían hacia el IS y respuestas de IS. Se sigue el mismo proceso de inyección a la red.

Las cuatro fase se ejecutan mientras hay mensajes en una traza o mientras no se alcance el numero de máximo muestras de datos a simular definido en la configuración del simulador.

Para la capa de red, el simulador ha sido modificado para inyectar una traza y validar la dependencia de mensajes que pertenezcan a la misma consulta. En la Figura 3.20 se muestra un diagrama de flujo del simulador.

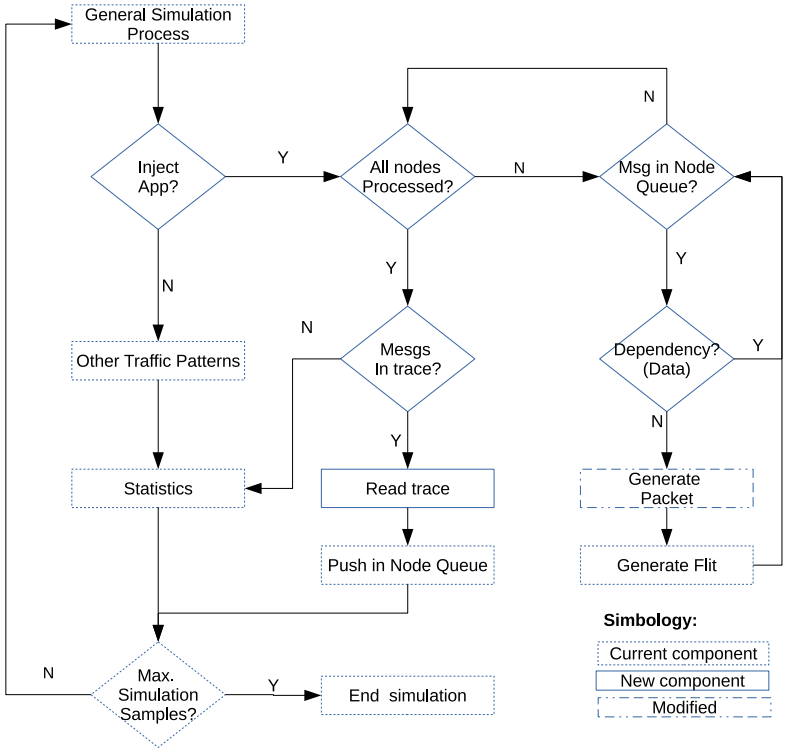


FIGURE 3.20: Diagrama de flujo de la simulación de la aplicación en la versión modificada de Booksim. El bloque [*Dependency? (data)*] es detallado en la Figura 3.24.

Los componentes nuevos corresponden a la lectura de la traza y validación de la dependencia de mensajes de la misma consulta. Los componentes modificados corresponden a la generación de paquetes para incorporar en el atributo de *datos* el *identificador* de las consultas del SES. Los atributos de los mensajes se muestran en la Tabla 3.1.

3.6.5 Dependencia de Mensajes

El procesamiento de las consultas se realiza mediante un mecanismo de distribución de la consulta en diferentes mensajes hacia los nodos CS o hacia el IS cuando la consulta no ha sido resuelta por el CS. La Figura 3.21 muestra la secuencia de procesamiento de las consultas por el FS. En el paso (1) la consulta es generada por el usuario y está formada por los Datos (secuencia de caracteres) que el usuario solicita al buscador. En el paso (2) el FS asigna un identificador (*IDQ*) a la consulta para ser identificada de forma única durante todo el proceso de búsqueda. En los pasos (3) y (4) se crean diferentes mensajes m_i, m_j para distribuir la consulta hacia los nodos CS.

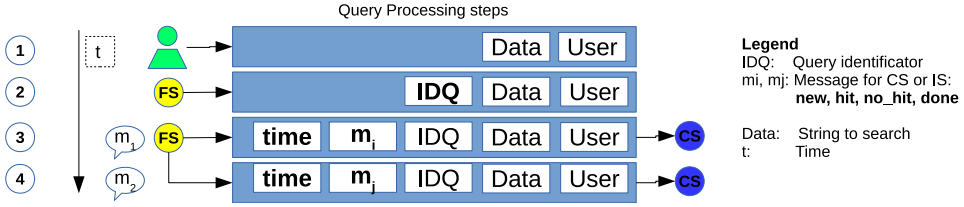


FIGURE 3.21: Estructura de datos de la consulta y secuencia de procesamiento en el FS. El FS asigna un Identificador, luego genera mensajes para cada CS o IS.

La relación entre los mensajes m_i, m_j es un identificador único de la consulta (*IDQ*). Los mensajes envíados al CS son retornados en estado *hit* o *no_hit*. Si no hay estados *hit* la consulta no ha sido resuelta por el CS. Las consultas no resueltas (*no_hit*) son enviadas hacia el IS. Finalmente una consulta se considera resuelta por el IS cuando un mensaje ha cambiado a estatus *done*.

En la Figura 3.22 se muestra un diagrama de la secuencia de envío de mensajes del FS hacia el CS. Se puede observar que los mensajes m_1, m_2, m_3 (generados por el FS), se envían paralelamente hacia el CS en el tiempo t. Por condiciones de la red podrían llegar a los nodos CS en tiempos diferentes: t_1, t_2, t_3. Después de ser procesados por los nodos CS, cada CS genera una respuesta para el FS. Los resultados pueden regresar en tiempos diferentes t_4, t_5, t_6. Si los mensajes tienen

estado *hit*, la consulta ha sido resuelta por el CS y se puede entregar los resultados al usuario. Caso contrario si el estado es *no_hit*, la consulta es enviada en t_7 hacia el IS.

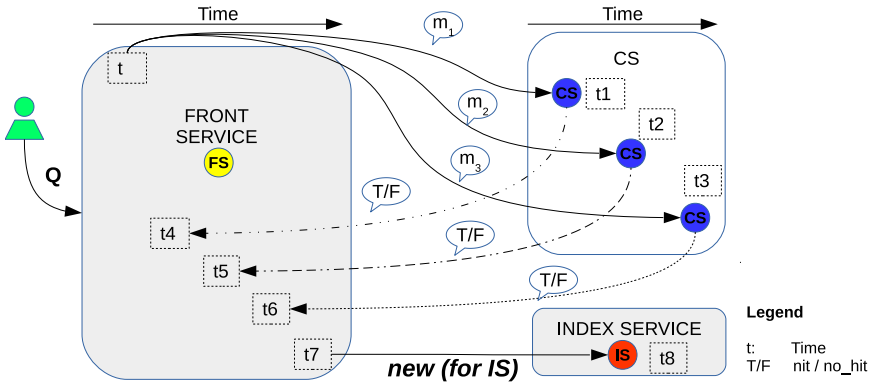


FIGURE 3.22: Secuencia de procesamiento de las consultas en el FS durante el envío de la consulta hacia el CS y respuesta desde el CS.

La simulación del proceso de dependencia de mensajes se realiza por medio del lectura de la traza y acceso a la sección de datos *data* de la estructura de la consulta, ver Figura 3.21, pasos (3) y (4). En el proceso de inyección se lee la consulta de la traza (Paso *Read trace*) en la Figura 3.20.

Después de leer la traza se realiza la simulación de la dependencia de mensajes. Este proceso se ilustra en la Figura 3.23. Después de la inyección de consultas, se realiza la simulación de los servicios. Seguidamente se verifica la dependencia de mensajes que pertenecen a la misma consulta. Se lee la consulta y se inyectan los datos al simulador de red. El simulador almacena un registro para analizar métricas de rendimiento como Latencia de mensajes (L) y throughput (T). El registro permite además generar una traza simulada.

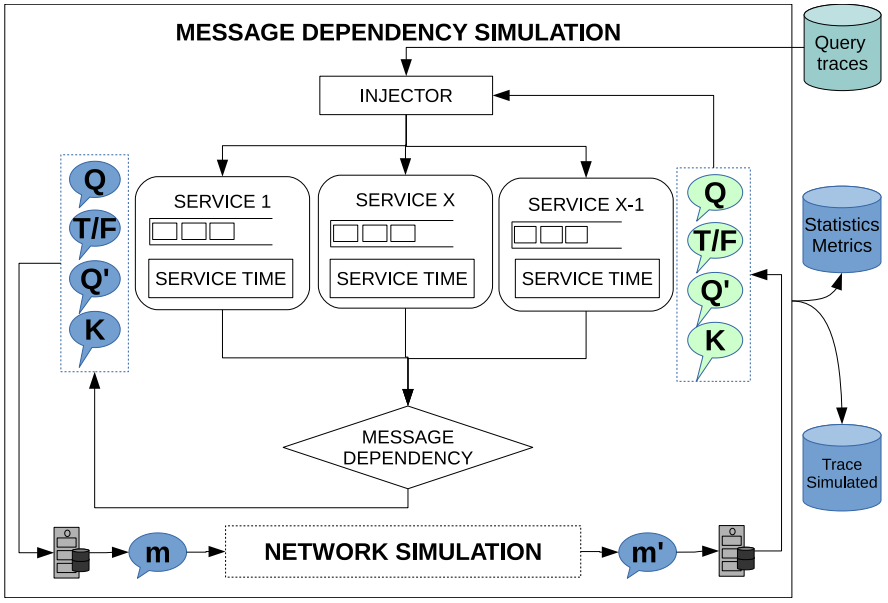


FIGURE 3.23: Simulación de dependencia de mensajes de la aplicación en relación a la misma consulta.

El diagrama de flujo de la verificación de dependencia de mensajes se muestra en la Figura 3.24. El proceso se aplica para todos los nodos. Se verifica si hay mensajes en una cola de mensajes retirados de cada nodo. Se verifica si ha llegado un mensaje anterior para la misma consulta. Si ha llegado el mensaje anterior, la consulta continua el flujo normal en el servicio, caso contrario permanece en cola de espera por los mensajes restantes.

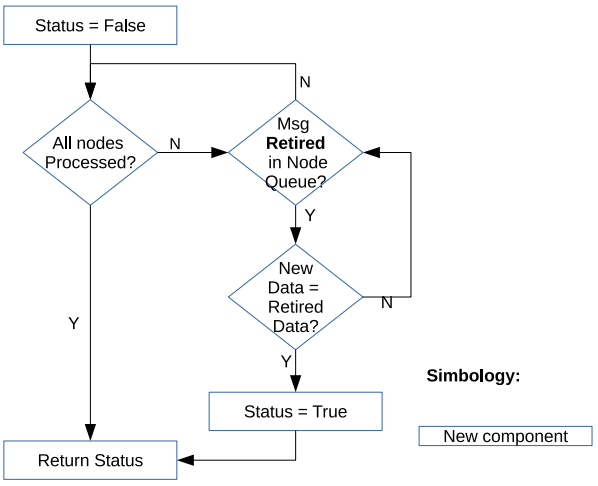


FIGURE 3.24: Diagrama de flujo para verificar la dependencia de mensajes para cada nodo. Este proceso es parte del diagrama mostrado en la Figura 3.20.

3.6.6 Inyección de Consultas y Cálculo de Latencia

La Latencia de consultas se realiza por medio del cálculo del tiempo en que se generó el primer mensaje en el FS para el CS. Si la consulta es resuelta por el CS, el cálculo termina con la recepción del último mensaje del CS en el FS. Cuando la consulta es resuelta por el CS la llamamos QL. Si la consulta es resuelta por el IS la llamamos QL'. En el Caso de QL' el cálculo termina con el último mensaje generado por el IS es recibido por el FS. La latencia además es afectada por el tiempo de tránsito de los mensajes en la red. La Figura 3.25 muestra un grafo que ilustra el cálculo de la latencia de la consulta.

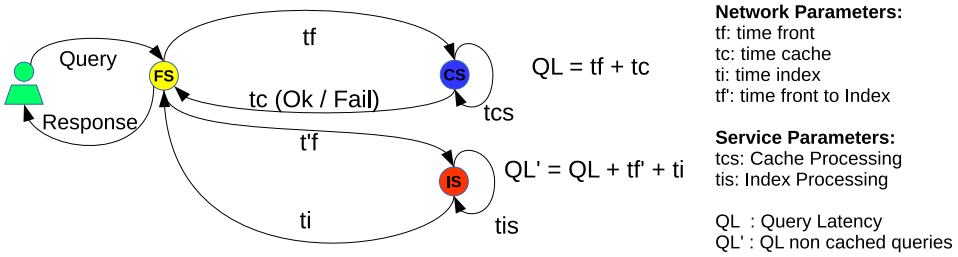


FIGURE 3.25: Cálculo de la Latencia de las consultas. **QL** cuando son resueltas por el Cache Service y **QL'** para resueltas por el Index Service.

Como se observa en la Figura 3.26, cuando se procesa una traza de tráfico real en el simulador el cálculo de la latencia se realiza mediante el *tiempo t1* de inyección del primer mensaje en el FS (Paso 1) hasta la recepción del último mensaje en el FS (Paso 2).

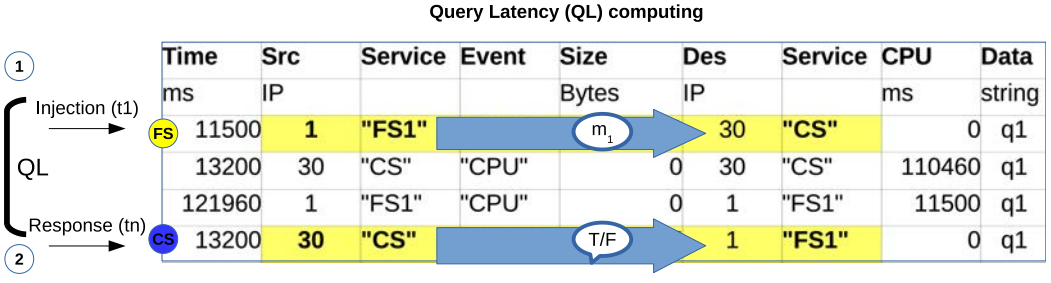


FIGURE 3.26: Muestra de consulta resuelta por el Cache Service.

3.6.7 Evaluación de la Simulación

Para la evaluación del simulador, se ha utilizado una traza correspondiente a una configuración de 128 nodos. La topología de la red es Fat-tree. La topología se construyó mediante la lectura de la traza y se genera un archivo que se importa utilizando la función *anynet* del simulador. Primero se procesa la traza y se construye la topología en el formato requerido por el simulador y se envía al simulador mediante el parámetro *network_file*.

Uno de los elementos a analizar relacionado al rendimiento de la red es la Ocupación de los enlaces. La figura 3.28 muestra el comportamiento de los buffers en una ventana de tiempo de ejecución del simulador.

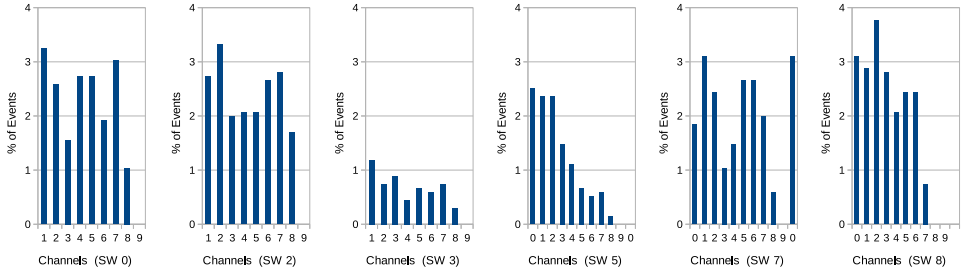


FIGURE 3.27: Ocupación de buffers para una ventana definida de tiempo en los switches de Nivel 0. El eje X presenta los canales y el eje Y presenta el porcentaje de eventos donde la Ocupación fue más alta que el umbral de 25% de la capacidad. (Los switches 1, 4 y 6 reportaron 0 eventos)

La Figura 3.28 muestra la Latencia de las consultas (QL) de una traza real y la Latencia de las consultas generada por el simulador. Se utiliza una secuencia de 480 consultas. En la simulación se obtiene una Latencia de consultas (QL) de 38.8ms versus 34.8ms de la traza real, lo cual significa el 97.1% de precisión.

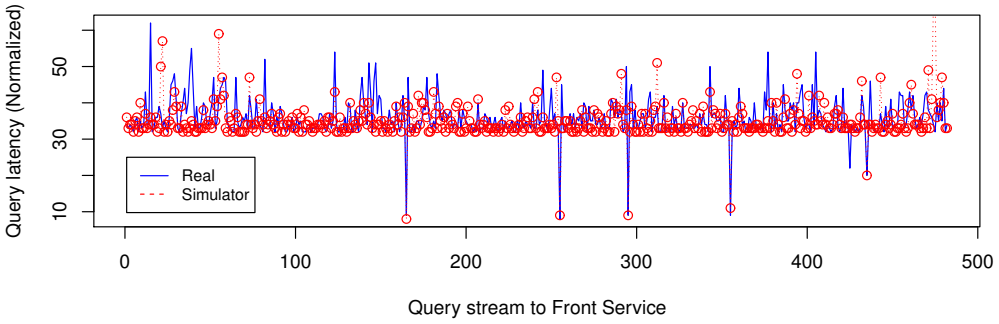
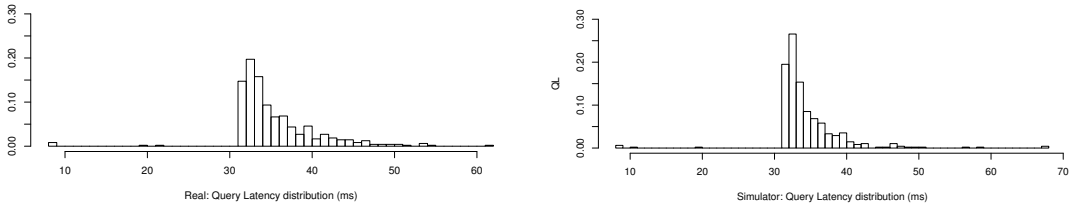


FIGURE 3.28: Comparación normalizada de tiempo real y simulado de la Latencia de consultas para una secuencia de 480 consultas utilizando una configuración de 128 nodos.

En la Figura 3.29 se muestra la distribución de la latencia *Real* y *Simulada*, se observa que la tendencia se concentra en la misma zona de tiempo tanto para QL real como simulado.



(A) Distribución de QL Real real

(B) Distribución de QL simulada.

FIGURE 3.29: Comparación de la distribución de la Latencia de consultas real y simulada

Capítulo 4

Modelo de Enrutamiento Basado en la Aplicación

El propósito del SES es la resolución de las consultas del usuario, para lo cual se utiliza una aplicación distribuida (definida por la arquitectura de los componentes de la aplicación) que se resumió en la Sección 3.5 y, se utiliza un servicio de comunicaciones, según se expuso en la Sección 3.5.1.

Se ha visto que, de acuerdo a la arquitectura del sistema, que el SES es un sistema complejo y requiere el uso de componentes de software (FS, CS, y IS) especializados para provisión del servicios. Así mismo requiere de una plataforma de red especializada para la gestión de los mensajes de sus componentes.

La presente investigación en enfoca en el estudio de los componentes de red para la gestión de un SES eficiente. Por tal motivo el presente Capítulo en enfoca en analizar como la aplicación se comporta en términos de demanda de servicios de red y como gestionar tal demanda con un servicio de red especializado en los requerimientos de la aplicación distribuida (SES). Se espera proponer un modelo de gestión de tráfico que basado en la aplicación que mejore el rendimiento del

modelo actualmente utilizado en el sistema real, así como otros modelos de la literatura.

4.1 Criterios Básicos para el Modelo

Entre los atributos principales para el diseño de un modelo de enrutamiento según se expuso en la Sección 2.2 es necesario tener en cuenta que, para el SES, el tamaño de la red puede crecer según sea el número de nodos de servicios, que la topología está predefinida por la configuración del sistema, las instancias (FS, IS, CS) pueden alojarse en cualquiera de los nodos disponibles y que existen una número diverso de enlaces para conectar dos nodos. Con esto criterios al algoritmo de enrutamiento esperado debería utilizar tres elementos básicos: (1) Los componentes de la aplicación. (2) la topología y (3) los enlaces disponibles. La Figura 4.1 resume estos tres criterios.

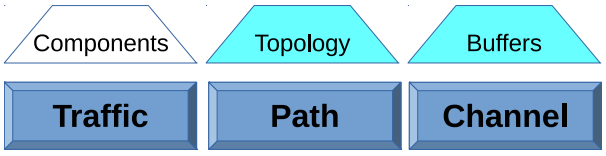


FIGURE 4.1: Criterios básicos para el modelo de enrutamiento. Modelos actuales utilizan los recursos de la red, el modelo propuesto se basa en los componentes de la aplicación.

Con estos criterios se espera utilizar únicamente información local del switch o de un switch directamente conectado (vecino), para reducir la complejidad del algoritmo de enrutamaiento y por lo tanto la sobrecarga (overhead) introducida por operaciones de enrutamiento.

4.1.1 Introducción de la Topología para el Enrutamiento

Los nodos FS, IS y CS intercambian mensajes que se mueven entre los componentes de la red. Tal movimiento depende de la ubicación de cada nodo en la red y depende de la topología de red utilizada. En el caso del SES en estudio se utiliza topología Fat-tree, una muestra de la topología específica utilizada en el SES se muestra en 3.6. Durante el envío de mensajes, cada par de Fuente-Destino puede estar ubicado en: (a) el mismo switch, (b) en diferentes switch y en el mismo POD o (c) en diferentes switches y en diferente POD. La Figura 4.2 muestra un esquema sobre los saltos (hops) que un mensaje recorre según sea la ubicación de los nodos FS, IS y CS.

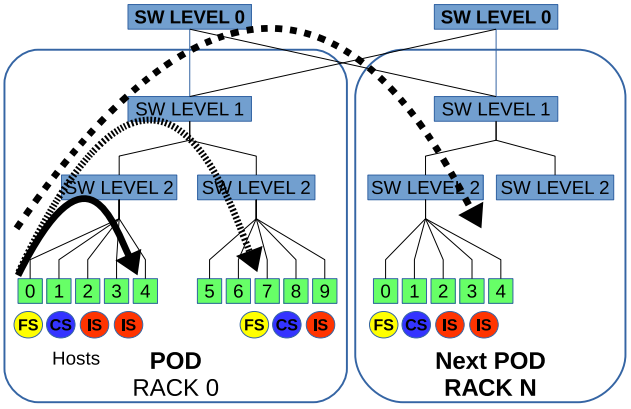


FIGURE 4.2: Diagrama de cantidad de saltos (Hops) que deben recorrer los mensajes según se la ubicación de las instancias de FS, IS y CS que lo generan.

El tiempo de entrega de un mensaje generado por una instancia de FS para llegar hacia una instancia de CS o IS, está definido por: (1) el tiempo que le tome en ser colocado en una cola de salida en el nodo origen, más (2) el tiempo que le toma en recorrer el medio físico que conecta el nodo con el switch, más (3) el tiempo que le toma en atravesar el switch, más (4) el tiempo que para recorrer el medio físico que conecta el switch con el nodo destino, y finalmente (5) el tiempo que le tome estar en cola de espera en el nodo destino. Este recorrido define la Latencia de serialización de un mensaje. La Latencia de un mensaje se incrementa

en condiciones de competencia (Contention) por un recurso cuando hay más de un mensaje en la red y más nodos enviando mensajes.

Existen dos casos posibles en la resolución de consultas. (1) Cuando una consulta es resuelta por el CS y (2) cuando la consulta es resuelta por el IS. En los dos casos la ruta requiere diferente recorrido de los mensajes.

- (1) Consulta es resuelta por el CS: El par Fuente - Destino (FS, CS) están desplegados en el mismo switch y POD.
- (2) Consulta es resuelta por el IS: El par Fuente - Destino (FS, IS) están desplegados en diferente switch y POD.

La Tabla 4.1 muestra las nueve posibles de saltos que dan los mensajes en la resolución de la consula.

TABLE 4.1: Posibles combinaciones de recorrido de los mensajes según sean resueltas por el CS o sean resueltas por IS. En total todas las consultas resueltas por el FS previamente han pasado por uno de los posibles recorridos del CS. En total son nueve combinaciones de saltos (hops)

Switches	1			3			5		
Solved by CS	2h			4h			5h		
Solved by CS+IS	2h	4h	6h	2h	4h	6h	2h	4h	6h

El cálculo de la Latencia de la Consulta (QL) según se expuso en la sección 3.6.6, se calcula desde que la salida del primer mensajes del FS hasta la llegada del último mensaje hacia el FS con el estado de consulta resuelta: *done*. El tiempo requerido para el procesamiento de la consulta internamente en el IS o CS, está fuera del alcance del presente estudio. Por lo que el tiempo que compete analizar es el tiempo que le toma a la consulta ser procesada por los elementos de la red. Para el usuario hay un periodo de tiempo adicional que es el tiempo que le tomó a la solicitud arribar desde la estación de trabajo del usuario hasta el nodo FS.

En la Figura 4.3 se observa como se canaliza la consulta y que operaciones de red se necesita mejorar para reducir la latencia de las consultas. En el diagrama se observa con numeral **(1) la secuencia real** de ejecución de una consulta, las variables *tf*, *tc* y *ti* son las que se espera influir para mejorar los servicios de red. En el numeral **(2) secuencia esperada**, se observa la **secuencia esperada** y las variables **tc'** y **ti'** son las que se espera reducir.

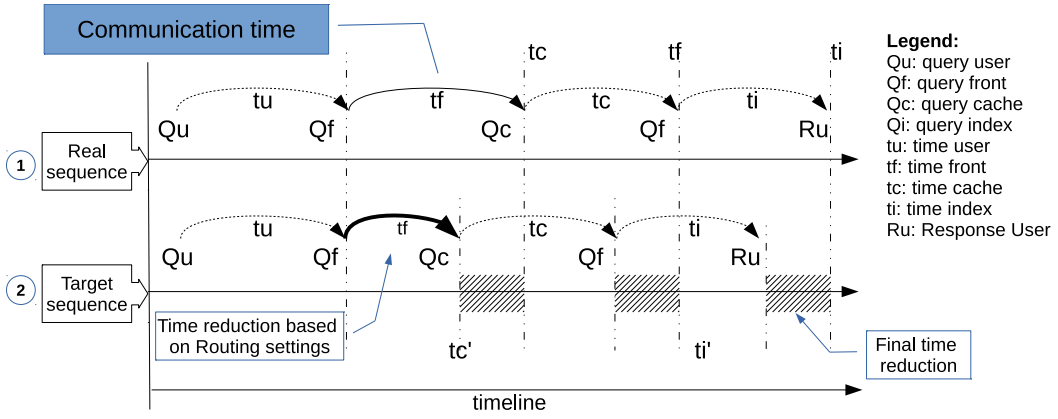


FIGURE 4.3: Diagrama de secuencia de calculo de la Latencia de Consultas real y esperada con la mejora de las condiciones de la red

4.1.2 Introducción de la Ocupación de Canales para el Enrutamiento

El análisis de la ocupación de los canales (por medio de los buffers del switch) nos permite identificar el impacto de la competencia de los mensajes por canales de salida en los switches.

La ocupación de buffers es un criterio a tener en cuenta para mejorar la asignación de recursos de red para todos los servicios. Sin embargo es necesario considerar cual es el criterio que mejor rendimiento produzca en la entrega de mensajes.

Al analizar la ocupación de los buffers, el algoritmo de enrutamiento debería incluir un mecanismo de balanceo de carga basado en la ocupación de los enlaces.

La Figura 4.13 expone el mecanismo que se considera para el enrutamiento de mensajes. Primero se mantiene un monitoreo de la Ocupación durante una ventana de tiempo y se registran los eventos de ocupación. Śegundo, para la decisión de balanceo de carga por ocupación se definen umbrales en porcentaje de ocupación. Tercero un canal será escogido si no ha alcanzado su limite de ocupación.

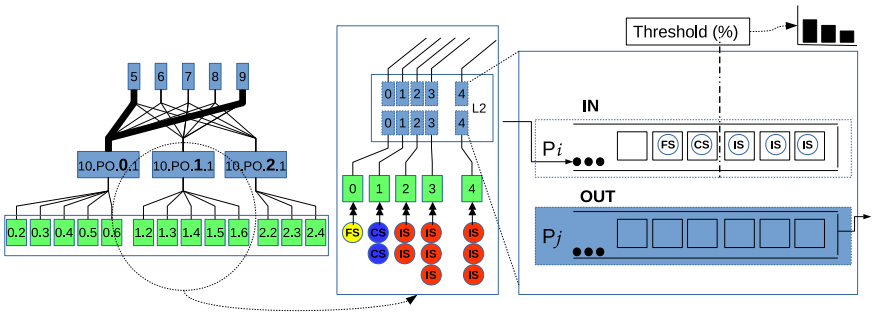
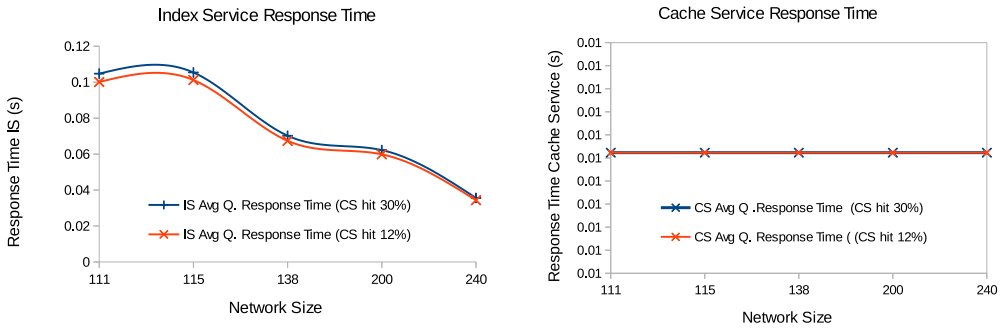


FIGURE 4.4: Monitoreo de ocupación de buffer para elección de ruta alternativa mediante el monitoreo de la ocupación en una ventana de tiempo.

4.1.3 Criterio Basado en la Aplicación

A fin de utilizar el criterio de la aplicación en el enrutamiento, se analiza el tiempo que requiere una consulta para ser procesado tanto por el IS como por el CS. Este tiempo es obtenido de las trazas del sistema real. En la traza los tiempos de procesamiento de la consulta en el IS son mas altos que los que requiere el CS.

La Figura 4.5a muestra el Tiempo Promedio de Procesamiento de la Consulta por el IS para diferentes configuraciones de red (desde 111 nodos hasta 240 nodos), se observa que el tiempo se reduce según se aumenta el tamaño de la red. La Figura 4.5b muestra al Tiempo Promedio de Procesamiento de la Consulta por el CS, se observa que el tiempo no cambia con el tamaño de la red. Aún con un tamaño de red de 240 nodos, el tiempo del IS es considerablemente más alto que el tiempo de CS.



(A) Tiempo de Respuesta del Index Service (B) Tiempo de Respuesta del Cache Service.

FIGURE 4.5: Comparación la duración del procesamiento de las consultas del IS y CS para diferentes configuraciones de tamaño de red.

En la Figura 4.6 se muestra la relación entre el tiempo de resolución de las consultas por el IS y por el CS. Se observa aún con una configuración de 240 nodos, el tiempo de procesamiento del IS es tres veces más alto que el CS. Con este comportamiento, es necesario considerar que mientras se procesan consultas por el IS, se pueden resolver consultas en el CS. La ocupación de la red según se expuso en la Sección 4.1.2 acerca de la Ocupación de los Canales, es predominada por mensajes del IS, sin embargo el procesamiento de éstos mensajes toma más tiempo que procesar mensajes del CS. Por lo tanto es necesario gestionar los mensajes para priorizar los mensajes que pueden resultar en respuestas para el usuario final.

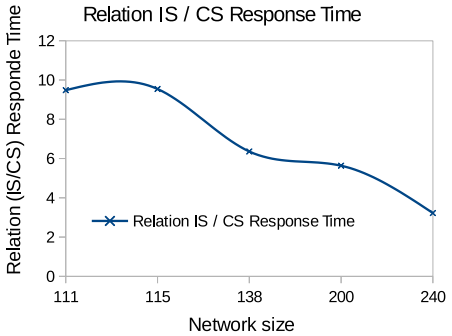


FIGURE 4.6: Relación del tiempo de procesamiento de las consultas entre el Index Service y el Cache Service para diferentes configuraciones de tamaño de red.

Por lo expuesto, se propone un mecanismo de gestión del tráfico que divida la carga en la red en tráfico que **puede esperar** mientras se procesan todos los mensajes de la misma consulta en el IS y mensajes del CS que probablemente serán procesados en menor tiempo. La Figura 4.13 muestra un diagrama de éste análisis. El tráfico se clasifica en carga de trabajo (W_{cs}) y carga de trabajo (W_{is}). Se propone balancear el tráfico de forma **justa** entre el IS y CS para asignar los recursos de la red de forma más eficiente.

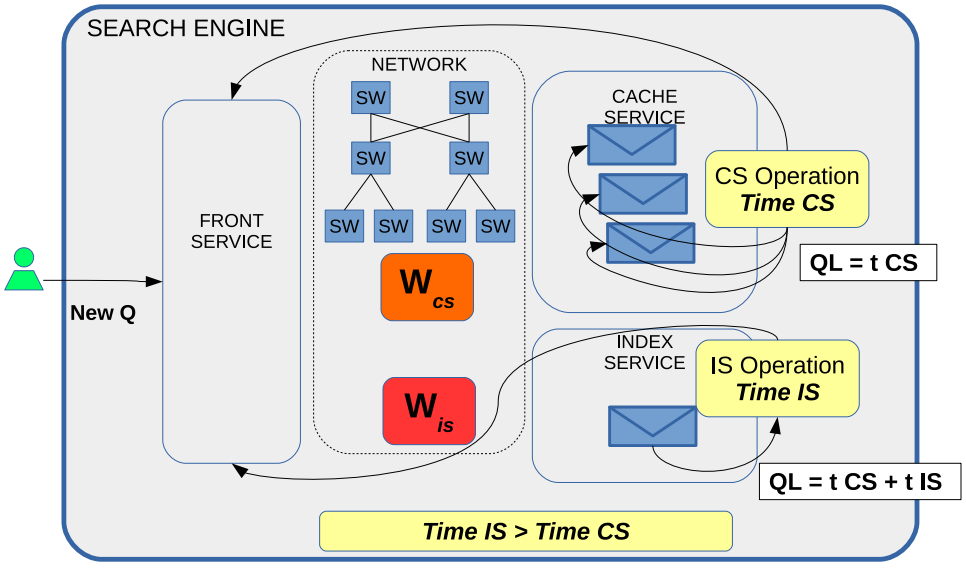


FIGURE 4.7: Clasificación de la carga por el tiempo de procesamiento de las consultas en IS y CS.

Puesto que el flujo de mensajes del SES, según se expuso en la sección 3.5.2 y la Figura 3.13, los mensajes generados por el CS son potenciales respuestas para el usuario final, se espera que reduzcan la carga en la red. Por lo tanto una estrategia a seguir es dar preferencia a los mensajes del CS. La Figura 4.13 expone la estrategia a seguir en la gestión del tráfico, bajo dos criterios, (1) El tráfico del CS reduce la carga en la red y (2) el tráfico IS preserva la carga de la red y necesita más tiempo en los nodos de procesamiento. Por lo tanto, mediante la clasificación del tráfico en **Primario** para el CS y **Secundario** para el IS, se gestiona los recursos de la red de forma justa y en base a las reglas del flujo de mensajes del SES.

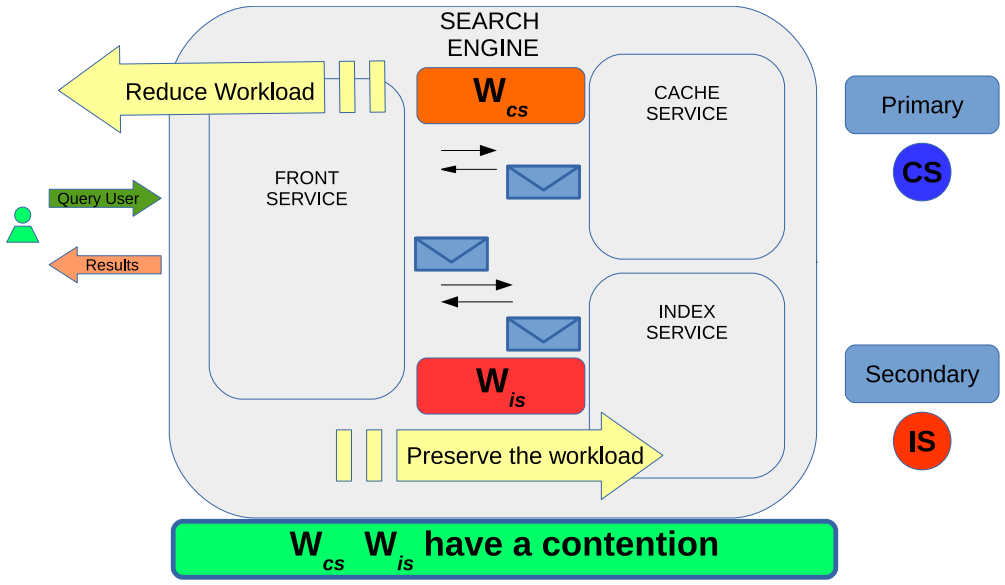


FIGURE 4.8: Clasificación del tráfico basada en el Rol de la aplicación. El tráfico CS es una potencial descarga de contenido hacia el usuario y descarga de tráfico en la red.

4.2 Modelos de Gestión de Tráfico Actuales

La presente sección resume modelos actuales de enrutamiento, a fin de tomar una línea de base en el diseño del modelo de enrutamiento basado en la aplicación. En la literatura, según se expuso en la Sección 2.2 acerca de algoritmos de enrutamiento, los modelos se basan en la gestión de un recurso de la red para tomar decisiones en el enrutamiento de mensajes. Cada modelo de enrutamiento, toma una **entrada** y ejecuta un proceso sobre un **recurso** de la red. En la Figura 4.9 se expone tres modelos de enrutamiento tomados como línea de base:

- **Estático**: Es el modelo actualmente utilizado en SES del nuestro caso de estudio utiliza como entrada una tabla de enrutamiento predefinida. El recurso que utiliza en básicamente la topología de red.
- **Diversity**: Es el modelo que utiliza una propiedad de robustez de la red (Path Diversity) según [1] que define la existencia de mas de un camino

mínimo para llegar a un destino. En este modelo se utiliza como entrada un algoritmo que calcula el canal que se debe utilizar. El recurso utilizado es la topología de red. El cálculo del canal puede hacerse mediante un mecanismo aleatorio o de uso uniforme de todos los enlaces de forma secuencial.

- **Balance**: Este modelo utiliza como entrada la ocupación de los canales. El proceso que ejecuta es la distribución de la carga entre los enlaces disponibles. El recurso que utiliza son los buffers.

La Figura 4.9 muestra un diagrama que resume las características de los modelos de enrutamiento tomados como linea de base. Se observa la *entrada*, el *algoritmo* y el *recurso* que utiliza cada uno.

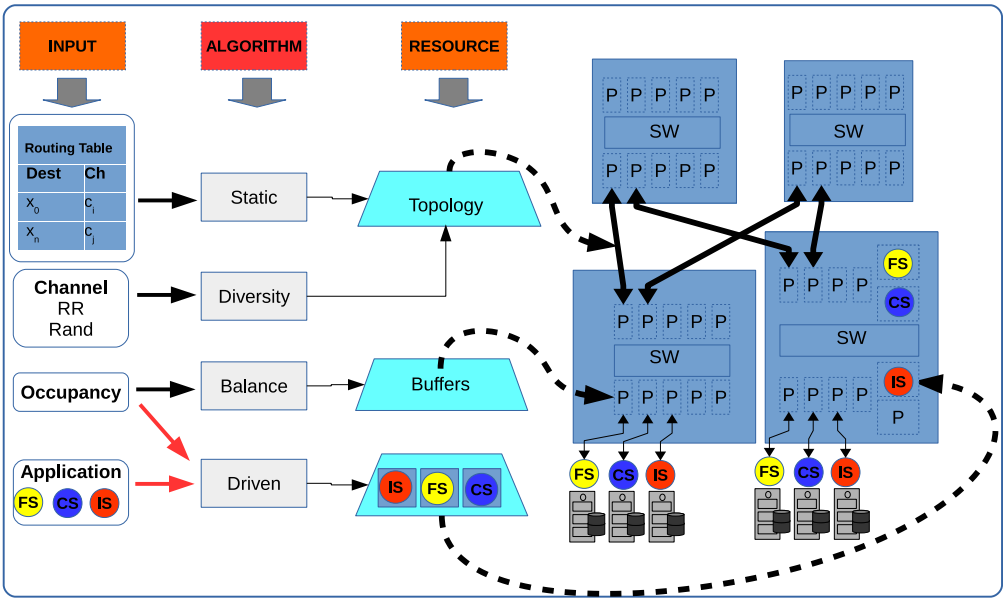
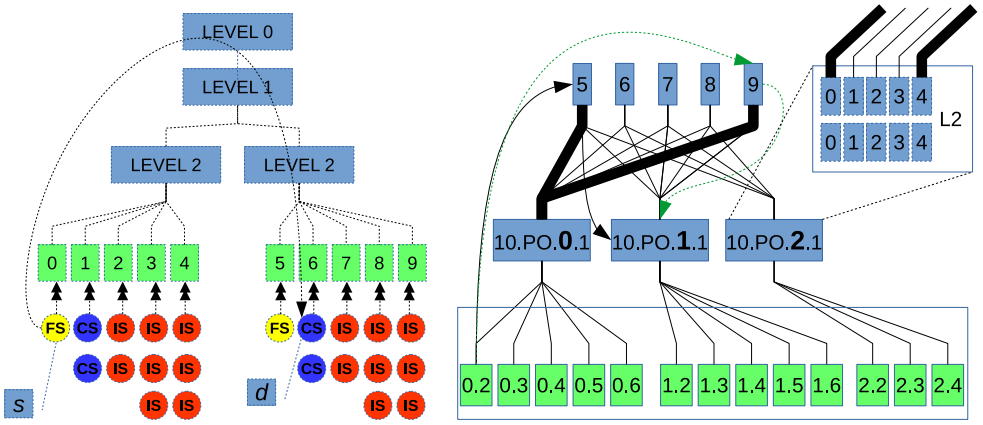


FIGURE 4.9: Diagrama comparativo de diferentes modelos de enrutamiento basado en la entrada y el recurso que gestionan.

En la Figura 4.9 se expone un primer acercamiento al modelo de routing propuesto en el presente trabajo. El modelo está basado (controlado) por la aplicación, toma como entrada dos elementos, (1) la ocupación de los canales y (2) los componentes de la aplicación. El recurso que utiliza son los buffers y actúa sobre las colas de espera de los mensajes.

4.3 Modelo de Gestión de Tráfico Propuesto

En base al análisis presentado en las secciones anteriores, se ha diseñado un modelo que incluye el flujo de tráfico (priorizado), la topología (balanceo de carga) y la carga en los routers (Buffer Occupancy). La Figura 4.10 muestra el modelo de gestión de tráfico y los criterios que se utiliza. En primer lugar se considera que el intercambio de mensajes entre nodos se envía por rutas alternativas a fin de balancear el tráfico entre diferentes canales. Por ejemplo se observa que, si el nodo *0.2* envía un mensaje hacia le nodo *1.2*, el mensaje puede ser enviado, por medio del **switch 5** o por medio del **switch 9**.



(A) Secuencia de flujo S-D (B) Uso de rutas alternativas

FIGURE 4.10: Diagrama para enrutamiento basado en la existencia de diferentes caminos mínimos para enrutamiento (Diversity Path).

En la configuración por defecto del switch, los mensajes son asignados a los canales de entrada y se colocan en una lista de espera mientras se asigna su canal de salida. En la Figura 4.11 se observa un esquema del procesamiento de los mensajes en los switches de nivel dos (Edge). Se observa que cada nodo inyecta un mensaje a la red, a continuación los mensajes son colocados en lista de espera según sea el orden de llegada. En la figura se observa que los mensajes se colocan en la lista de espera **Pj** y cuando se ha asignado un canal de salida los mensajes se colocan en la lista **Pi**.

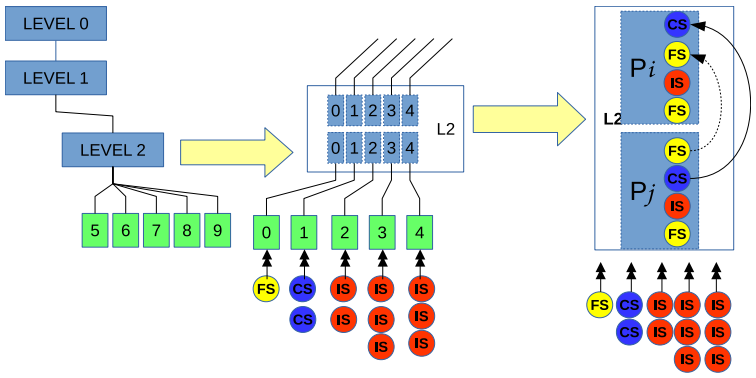


FIGURE 4.11: Diagrama de ocupación de buffers por mensajes de diferentes servicios.

El criterio para la elección del tráfico a priorizar se realiza mediante una tabla de mapeo de los servicios alojada en el switch que define los nodos que ejecutan instancias del CS. Cuando un mensaje pertenece a un nodo CS, el mensaje es movido hacia un canal con menos Ocupación. En la Figura 4.12 se bserva que un mensaje de CS si ha llegado al final de la lista de espera *Pj*, es movido hacia una lista sin ocupación *Pk*.

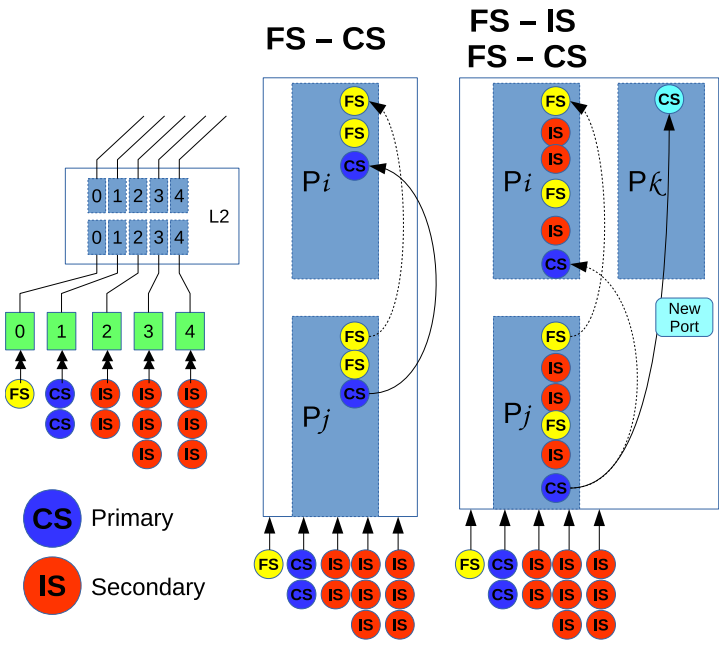


FIGURE 4.12: Criterio para elección de tráfico con baja carga de las operaciones de enrutamiento.

4.3.1 Arquitectura del Router

En el diseño del algoritmo de enrutamiento se introduce tres cambios (módulos) en la arquitectura del router. Un módulo de Monitoreo de la Ocupación del Buffer (BOM), el segundo es un mecanismo de Inspección de paquetes (DPI) y finalmente un mecanismo de redirección de paquetes (DM). Se ha definido este algoritmo como: Application-Driven Routing (ADR).

El monitoreo de la Ocupación, se realiza en tiempo de ejecución y consiste en hacer un seguimiento de los eventos de escritura en el buffer de cada router. El seguimiento se realiza por medio de tres contadores, uno por cada umbral definido (Alto, Medio o Bajo). La información recopilada por BOM permite a la política cambiar el mecanismo por defecto del sistema de asignación de canal de salida. Los contadores pueden reinicializarce después de un período de tiempo. El criterio para reinicio de contadores es necesario que no implique cálculos complejos, por lo tanto un criterio de período de tiempo (ciclos) configurable es suficiente.

BOM mantiene información histórica de la Ocupación del buffer. DPI permite identificar a que servicio pertenece un paquete específico. Finalmente el DM utiliza la información de BOM y de DPI para redirigir el tráfico asignándole recursos dinámicamente y en proporción a la carga de trabajo del servicio. De este modo la carga de trabajo es enviada hacia los recursos menos utilizados de la red. En la Figura 4.10b se muestra un diagrama de uso de diferentes enlaces.

El siguiente componente es el DPI, que mantiene como entrada un mapeo de Host-Servicio de la aplicación. El segundo parámetro contiene información acerca de la carga de trabajo que genera el servicio. Cuando un paquete llega a un router la política combina el mapeo, el servicio y la carga de trabajo. El resultado lo convierte al paquete en un candidato a ser redirigido. Este paquete es enviado el DM que se encarga de asignarlo a un canal de salida. Los paquetes restantes son reenviados utilizando la configuración por defecto que tenga el router.

Finalmente el DM, toma como entrada la Ocupación y el paquete candidato y lo envía hacia el canal de salida con menor Ocupación. En la Figura 4.12 se muestra un diagrama sobre la asignación de un mensaje hacia un canal menos ocupado.

En la Figura 4.13 se muestra un diagrama de la arquitectura del router propuesta.

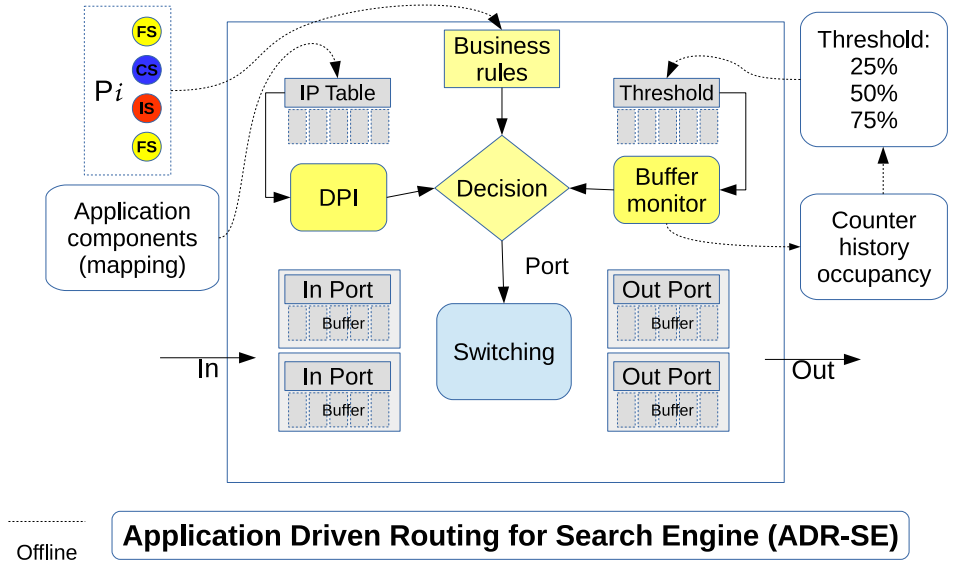


FIGURE 4.13: Diagrama de arquitectura de un router para aplicación del Modelo de Routing Basado en la Aplicación.

En la Figura 4.14 se muestra un diagrama de flujo del algoritmo propuesto. Las etiquetas numeras (1), (2) (3) y (4) corresponden al diagrama de la Figura 3.20 que muestra le proceso de inyección de mensajes.

El modelo de enrutamiento ADR inicia con la lectura de los mensajes generados para cada uno de los nodos. Mediante la verificación en una estructura de datos **Table Services** se verifica (Check Data) si el mensaje corresponde a un Cache Service . En caso afirmativo, se realiza la verificación de la Ocupación de los canales en la estructura **Buffer Monitor** y se obtiene el puerto de salida del mensaje (Compute Outport). El siguiente paso consiste en actualizar la Ocupación del puerto asignado (Write Occupancy). Finalmente el proceso de continúa con la etiqueta (2).

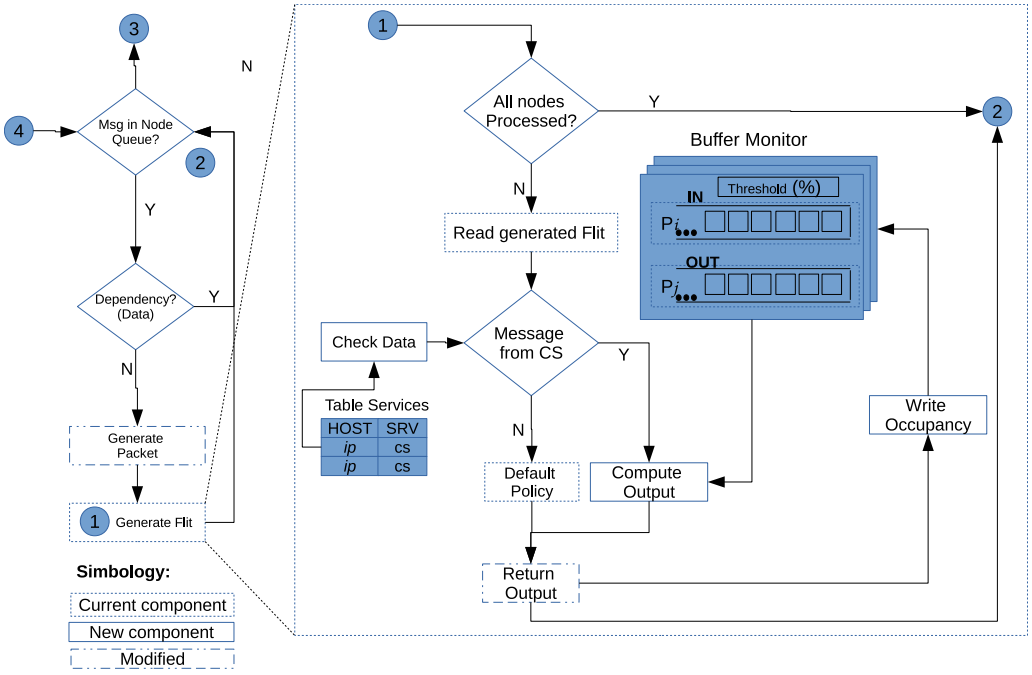


FIGURE 4.14: Diagrama de Flujo del Modelo de Enrutamiento Propuesto.

Capítulo 5

Validación Experimental

La presente sección describe un conjunto de experimentos diseñados para evaluar el comportamiento del modelo de enrutamiento propuesto. Según se expuso en la sección 1.5 se utilizará trazas de tráfico real utilizados por autores de [10]. Los experimentos pretenden demostrar en que casos de funcionamiento de la red que soporta el SES puede mejorarse aplicando la política de routing diseñada. Se realiza una comparación con métodos de enrutamiento de la literatura a fin de cuantificar la contribución. A continuación se detalla la metodología y los datos de experimentación utilizados.

5.1 Metodología de Experimentación

Los experimentos están basados es introducir una carga de datos reales al simulador de red para evaluar su rendimiento en términos de Latencia y Throughput.

A continuación se evalúa el rendimiento para condiciones diferentes de carga por medio de la variación (incremento) paulatina de la la tasa de inyección. Cada tasa de inyección constituye un experimento el cual es utilizado sobre una configuración de red (por ejemplo 115 nodos).

En cada experimento se monitorea y registra las variables de Latencia y Throughput a fin de compararlas con otros experimentos y diferente configuración de red. Cada experimento es repetido con diferentes algoritmos de enrutamiento.

5.2 Datos de Experimentación

Se utiliza dos conjuntos de datos correspondientes a diferentes configuraciones del SES. La configuración del SES está definida por los atributos expuestos en la Tabla 5.1

Table 5.1: Parámetros de configuración del SES

PARAMETRO
Cantidad de Front Service (nodos)
Cantidad de Cache Service (nodos)
Cantidad de Index Service (nodos)
Topologia de Red
Cantidad de Core Switches Core (Nivel 0)
Cantidad de Aggregation Switches (Nivel 1)
Cantidad de Edge Switches (Nivel 2)
Cantidad de PODs
Cantidad de nodos por POD
Máxima capacidad de nodos
Cantidad real de nodos utilizada (current)

Los grupos de datos corresponden a dos configuraciones de un SES con una topologia Fat-tree y un tamaño de red de 115 y 240 nodos respectivamente. La configuración detallada se expone en la Tabla 5.2.

TABLE 5.2: Resumen de configuraciones para el SES utilizando los dos conjuntos de parámetros.

Parameter	Trace A	Trace B
Front Service (nodes)	12	40
Cache Service (nodes)	3	20
Index Service (nodes)	100	180
Network Topology	3-Level Fat-Tree	3-Level Fat-Tree
Qty of Core Switches	16	25
Qty of Aggregation Switches	32	50
Qty of Edge Switches	32	50
Qty of Pods	8	10
Qty of nodes per Pod	16	25
Max. nodes capacity	128	250
Qty (current) hosts	115	240

El mapeo de nodos y tipo de servicio (FS, IS, CS) está definido por los atributos mostrados en la Tabla 5.3. Este archivo es diferente para cada configuración y detalla cada nodo, POD en el que esta desplegado e instancia que ejecuta. Además define si el nodo es replica y a que nodo corresponde la replica.

TABLE 5.3: Estructura de archivo de mapeo

ID	timestamp	Event	Src	Dest	Size	CPU	Query	Data

Las trazas de tráfico están definidas por los atributos mostrados en la Tabla 5.5.

TABLE 5.4: Estructura de archivo de traza

Host	POD	Rol	Replica	Host replica
IP	(0..10)	FS/CS/IS	R/NR	IP

5.3 Volumen de la Carga de Datos

Se utilizaron dos cargas de datos que corresponde a 60.000 consultas, con un aproximado del 31% de consultas que fueron resueltas por el Cache Service para el caso de una configuración de 115 nodos. La segunda carga de datos corresponda a 240 nodos tambien con una tasa de respuesta con acierto del CS de 30%. La Figura 5.1 muestra un ejemplo de una traza para una consulta Q1 resuelta por el IS.

id	timestamp	Event	IP_Src	IP_Des	Serv Src	Srev Dest	Size	time_cpu	Data
Number	*ms*	*string*	*ID*	*ID*	*string*	*string*	*bytes*	*ms*	*ID Query (string)*
1	0	CPU	1	1	FS1	FS1	0	0.00115	1
3	0.00115	MSG	1	30	**FS1**	**CS**	8	0	1
4	0.00115	CPU	30	30	CS	CS	0	0.011046	1
47	0.012196	MSG	30	1	CS	FS1	5	0	1
49	0.012196	CPU	1	1	FS1	FS1	0	0.00115	1
54	0.013346	MSG	1	3	**FS1**	**IS**	7	0	1
55	0.013346	MSG	1	17	FS1	IS	7	0	1
56	0.013346	MSG	1	7	FS1	ISR	7	0	1
57	0.013346	MSG	1	20	FS1	ISR	7	0	1
58	0.013346	MSG	1	21	FS1	ISR	7	0	1
59	0.013346	MSG	1	5	FS1	ISR	7	0	1
60	0.013346	MSG	1	6	FS1	ISR	7	0	1
61	0.013346	MSG	1	19	FS1	ISR	7	0	1
62	0.013346	MSG	1	4	FS1	ISR	7	0	1
63	0.013346	MSG	1	18	FS1	ISR	7	0	1
64	0.013346	CPU	4	4	ISR	ISR	0	0.002632	1
65	0.013346	CPU	3	3	IS	IS	0	0.002632	1
66	0.013346	CPU	5	5	ISR	ISR	0	0.002632	1
67	0.013346	CPU	7	7	ISR	ISR	0	0.002632	1
68	0.013346	CPU	6	6	ISR	ISR	0	0.002632	1
69	0.013346	CPU	18	18	ISR	ISR	0	0.002632	1
70	0.013346	CPU	19	19	ISR	ISR	0	0.002632	1
71	0.013346	CPU	20	20	ISR	ISR	0	0.002632	1
72	0.013346	CPU	17	17	IS	IS	0	0.002632	1
73	0.013346	CPU	21	21	ISR	ISR	0	0.002632	1
195	0.015978	MSG	3	1	**IS**	**FS1**	120	0	1
196	0.015978	MSG	4	1	ISR	FS1	120	0	1
197	0.015978	MSG	5	1	ISR	FS1	120	0	1
198	0.015978	MSG	6	1	ISR	FS1	120	0	1
199	0.015978	MSG	7	1	ISR	FS1	120	0	1
200	0.015978	MSG	19	1	ISR	FS1	120	0	1
201	0.015978	MSG	18	1	ISR	FS1	120	0	1
202	0.015978	MSG	20	1	ISR	FS1	120	0	1
203	0.015978	MSG	17	1	IS	FS1	120	0	1
204	0.015978	MSG	21	1	ISR	FS1	120	0	1
205	0.015978	CPU	1	1	FS1	FS1	0	0.023	1

FIGURE 5.1: Muestra de mensaje generado por FS para el CS para una consulta de ejemplo Q1. Los atributos *CPU* y *time_cpu* indican el tiempo de procesamiento en el nodo. La consulta Q1 es resuelta por el Index Service.

5.4 Algoritmos de Línea de Base

Los experimentos se han realizado utilizando tres algoritmos de línea de base y se han comparado con el método propuesto. El primer algoritmo utilizado es

el actualmente utilizado en el sistema real, lo llamaremos *Estático* (EST) y está basado en tablas de enrutamiento. El segundo algoritmo utiliza un mecanismo dinámico para calcular la ruta de salida de un mensaje,de modo que los mensajes son enviados por diferentes caminos, lo llamaremos *Diversity* (DIV). Finalmente el tercero es un mecanismo que balancea la carga tomando una decisión en la ocupación de las colas de salida, lo llamaremos *Balance* (BAL).

TABLE 5.5: Resumen de características de modelos de enrutamiento comparados.

Algoritmo	**Entrada**	**Proceso**	**Recurso**
Static	Tabla estática de enrutamiento.	Elegir de tabla canal de salida de la tabla en base a destino.	Topología
Diversity	Multiples canales disponibles y método de selección de canal.	Calcular canal disponible y elegir mediante modelo matemático.	Topología
Balance	Nivel de ocupación de canal.	Leer de nivel de ocupacióny elegir canal	Topología y buffers
Driven	Nivel de ocupación de canal. Aplicación	Leer de nivel de ocupación y elegir canal	Topología, buffers y datos de aplicación

Estos algoritmos utilizan un criterio acumulativo en el sentido en que utilizan más criterios para tomar decisiones en la elección de un canal del salida.

Los algoritmos utilizan información local, es decir información que tienen almacenada en sus propios espacios de memoria, que no permite ver el estado de la red en general. El método *estatico* utiliza únicamente información de la topología,

es decir de los enlaces disponibles y ha sido previamente definida para utilizar solo los canales explícitamente expresados en la tabla para alcanzar un destino, este mecanismo tiene la limitación de no utilizar otros caminos que existen en la topología.

A continuación, tenemos el método *diversity* que puede utilizar una tabla de enrutamiento con múltiples caminos para llegar a un destino; en este caso la cuestión es, que camino elegir de todos los disponibles. La elección del camino de salida puede ser basado en, primero, un modelo matemático que permita elegir el canal, segundo un mecanismo como de tomar el primero disponible y luego el siguiente, ó tercero, elegir un camino diferente en cada solicitud hasta elegir todos los caminos uniformemente de forma que la carga se distribuya entre todos los canales disponibles.

Los método hasta ahora explicados (static y divertity) han utilizado como recurso de la red: la topología y canales disponibles. Se puede ir un paso mas adelante y observar dentro de cada canal por su nivel de Ocupación a fin de tener otro criterio para la elección del canal. El método *balance* se basa en observar el nivel de Ocupación y elige el canal de salida menos ocupado cuando hay múltiples canales disponibles. Este mecanismo permite balancear no solo los canales, sino también los datos entre los canales menos ocupados, de forma que se distribuya la carga hacia canales menos utilizados.

El método propuesto en el Capítulo 4 se basa en un análisis de la aplicación, lo que implica ir hacia el contenido de los mensajes. La siguientes secciones comparan el rendimiento de éstos algoritmos con el modelo propuesto a fin de validar los resultados de enrutamiento con enfoque en la aplicación.

5.5 Análisis de Balanceo de Carga

A continuación se muestra el análisis comparativo de la ocupación de buffer en los tes niveles de la topología. Se utiliza la topología mostrada en la Figura 3.7. En el Nivel 0 existen nueve switches, en la Figura 5.2 se representan los buffers de cada switch. Se registra el número de eventos cuando la ocupación superó el 25%. La comparación se realiza sobre los tres algoritmos de línea de base. Se observa que por el *switch 1* únicamente BAl envío mensajes. Se observa que DIV gestiono los mensajes de forma equitativa entre todos los switches ya que la carga se distribuye entre todos (excepto el 1). En general la carga de la red fue procesada principalmente por los switches 7 y 8.

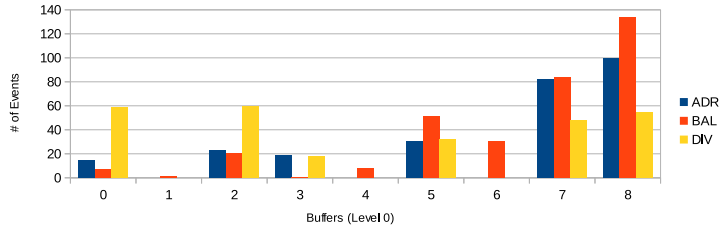


FIGURE 5.2: Comparación de la Ocupación de Buffers del Nivel 0 de la topología, con uso mayor al 25% de la capacidad del buffer para una configuración de 115 nodos.

La Figura 5.3 muestra los switches del Nivel 1 (numerados del 30 al 79). En este caso los swiches 75 al 75 no recibieron mensajes. Se presente nuevamente el caso que DIV distribuye de forma homogénea los mensajes. En el caso de BAL el comportamiento cambia y se producen algunos picos de uso. ADR tiene un comportamiento no uniforme de cada, sin embargo no llega a tener picos sobresalientes.

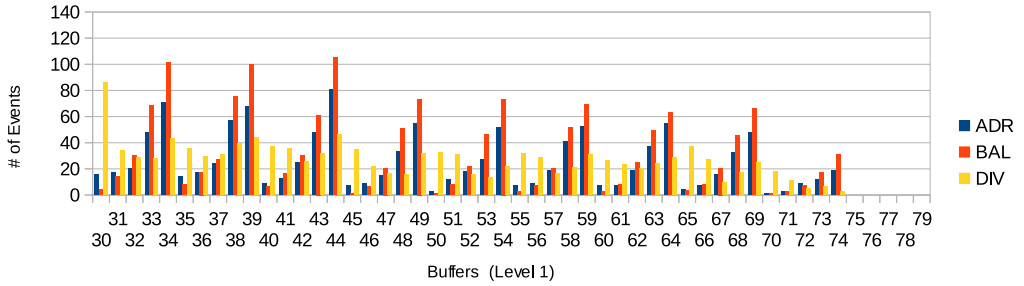


FIGURE 5.3: Comparación de la Ocupación de Buffers del Nivel 1 de la topología, con uso mayor al 25% de la capacidad del buffer para una configuración de 115 nodos.

La Figura 5.4 muestra los switches del Nivel 2 (numerados del 0 al 29). En este caso los swiches 27 al 29 no recibieron mensajes. Para los switches restantes se observa que los tres modelos de routing utilizaron los switches para la entrega de los mensajes y el comportamiento de carga entre ellos es similar, por ejemplo todos concentran carga en los switches 1, 2, 7, 10, 12, 18, 20 y 23. En el Nivel 2 que corresponde a la entrega a los nodos la carga es homogénea.

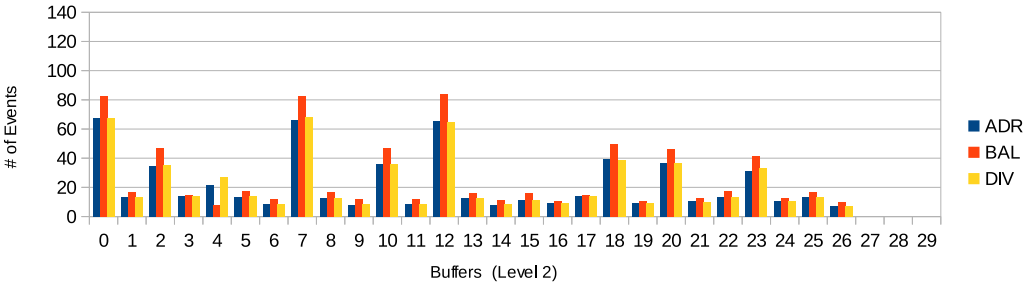


FIGURE 5.4: Comparación de la Ocupación de Buffers del Nivel 2 de la topología, con uso mayor al 25% de la capacidad del buffer para una configuración de 115 nodos.

5.6 Evaluación de la Latencia de la red

Para la evaluación de la latencia se ejecutó un conjunto de experimentos con diferentes tasas de inyección en cada ejecución en el rango de 10% de la capacidad hasta

100%. A continuación se exponen los resultados en los intervalos más relevantes de comparación y son entre el 15% y el 40%.

En la Figura 5.5 se observa que para el rango de inyección en rango 18% de la carga hasta el 25%, se observa una ligera reducción de la latencia para el enrutamiento ADR.

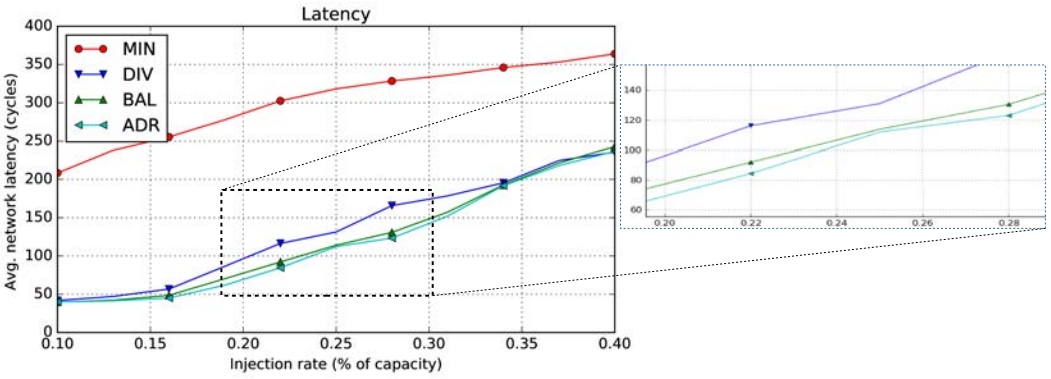
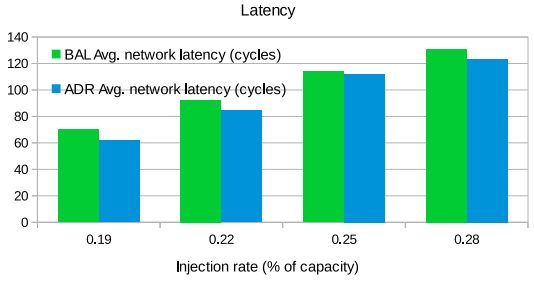


FIGURE 5.5: Comparación de Latencia de la red de diferentes algoritmos en enrutamiento y el propuesto para configuración de 115 nodos.

Se observa que, para éste rango si se compara los algoritmos BAL y ADR, la diferencia en promedio entre los dos algoritmos es del 6.2%. La reducción de la latencia con respecto a BAL el algoritmo ADR logra una reducción del 6%.2 de la Latencia de la red. Esto se presenta para una configuración de 115 nodos. La Figura 5.6 muestra los datos detallados de la reducción de a latencia.



IR	BAL	ADR	Diferencia
0.19	70.2	61.8	8.4
0.22	92.2	84.6	7.6
0.25	114.0	112.2	1.8
0.28	130.8	123.3	7.4
Avg.	101.8	95.5	6.3
115 nodos	Reducción:		**6.2%**

FIGURE 5.6: Análisis de diferencias en Latencia con algoritmos de enrutamiento, utilizando 115 nodos.

Para el caso de 240 nodos, según se muestra en la Figura 5.7, la menor latencia se obtiene en el rango del 20% hasta el 40% o más. Si se compara BAL con ADR, se observa que la reducción de la latencia en promedio alcanza el 13.4, y ésta reducción significa el 9.8% si se compara con BAL.

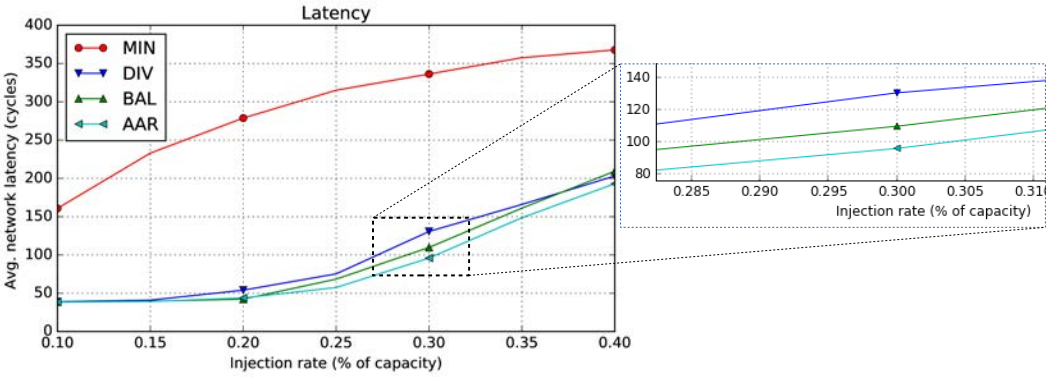
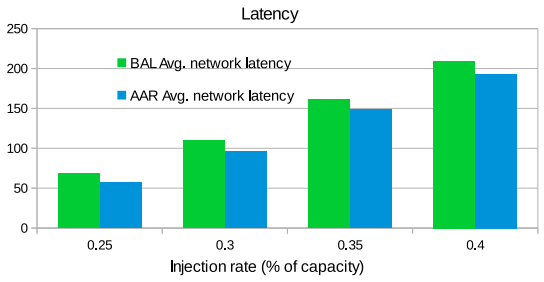


FIGURE 5.7: Comparación de Latencia de la red de diferentes algoritmos en enrutamiento y el propuesto para configuración de 240 nodos.

La Figura 5.8 muestra los datos detallados de la Latencia. Se observa una redución para todo el rango en análisis.



IR	BAL	ADR	Diferencia
0.25	68.4	57.5	11.0
0.3	109.8	95.9	13.9
0.35	161.1	148.6	12.5
0.4	209.4	193.1	16.3
Avg.	137.2	123.8	13.4
	Reducción:		**9.8%**

FIGURE 5.8: Análisis de diferencias en Latencia con algoritmos de enrutamiento, utilizando 240 nodos.

En resumen se reduce la Latencia en 6.2% para 115 nodos y en 9.8% para 240 nodos.

5.7 Evaluación del Rendimiento de la red

En relación al Throughput, se puede observar en la Figura 5.9 que el modelo ADR provee un mejor rendimiento para tasas de inyección en el rango 20% de la capacidad de la red al 35% de la capacidad. en este rango, si se compara con BAL, ADR incrementa el rendimiento en para una configuración de 115 nodos, tal como se muestra en la gráfica aumentada.

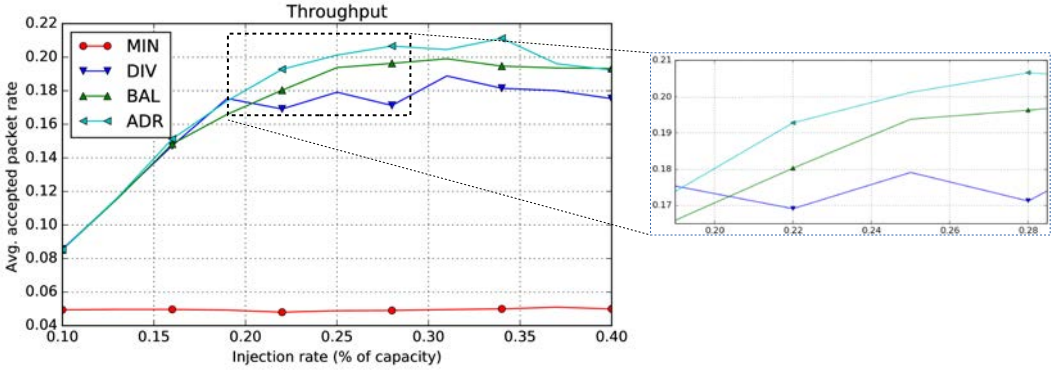
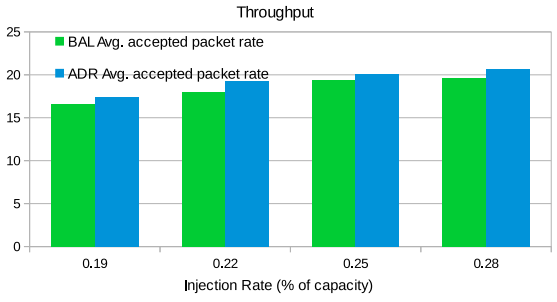


FIGURE 5.9: Comparación de Throughput de la red de diferentes algoritmos en enrutamiento y el propuesto para una configuración de 115 nodos.

En la Figura 5.10 se observa los resultados detallados de la evaluación. En promedio para el rango indicado, BAL mejora el rendimiento en el 1%, éste incremento que puede considerarse a primera vista reducido, es necesario compararlo que el estado actual de rendimiento que es de 18.4% y con ADR pasa a ser de 19.4%, lo que significa una mejora del 5.2% del estado actual aportado por BAL.



IR	BAL	ADR	Diferencia
0.19	16.6	17.4	0.8
0.22	18.0	19.3	1.3
0.25	19.4	20.1	0.7
0.28	19.6	20.7	1.0
Avg.	18.4	19.4	1.0
115 nodos	Incremento:		**5.2%**

FIGURE 5.10: Análisis de diferencias en Throughput con algoritmos de enrutamiento, utilizando 115 nodos.

La evaluación para una configuración de 240 nodos se observa en la Figura 5.11, gráficamente puede observarse los resultados de ADR sobre BAL.

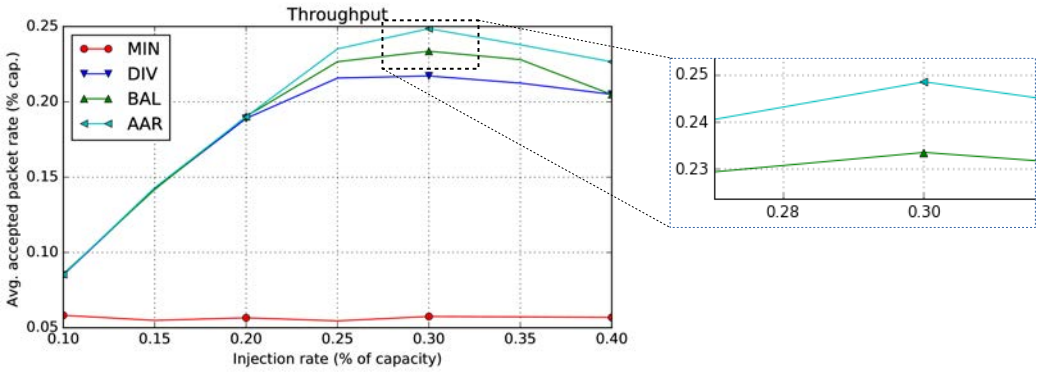
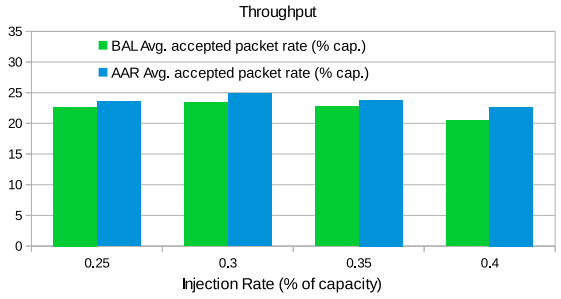


FIGURE 5.11: Comparación de Throughput de la red de diferentes algoritmos en enrutamiento y el propuesto para una configuración de 240 nodos.

En la Figura 5.12 se exponen los resultados detallados, y puede observarse que el rendimiento se incrementa en un promedio del 1.4%, sin embargo comparando con el rendimiento actual de BAL es que del 22.3% con ADR se obtiene el 23.7%, lo cal significa una mejora del 6.2% respecto al estado actual aportado por BAL.



IR	BAL	ADR	Diferencia
0.25	22.7	23.5	0.9
0.3	23.4	24.9	1.5
0.35	22.8	23.8	1.0
0.4	20.5	22.7	2.2
Avg.	22.3	23.7	1.4
	Incremento:		**6.2%**

FIGURE 5.12: Análisis de diferencias en Throughput con algoritmos de enrutamiento, utilizando 240 nodos.

En resumen se mejora el Throughput en 5.2% para 115 nodos y en 6.2% para 240 nodos.

Finalmente como resumen de Latencia y Throughput, después de la evaluación del modelo de enrutamiento, aplicado a dos configuraciones de red 115 nodos y

240 nodos, se ha podido demostrar que se reduce la latencia promedio de la red, mientras se mejora el rendimiento. Es decir se logra entregar mas mensajes en las terminales de destino mientras se reduce el tiempo de entrega.

Capítulo 6

Conclusiones y Líneas Abiertas

En base al objetivo general y específicos planteados al inicio de la investigación y replanteados según se ha avanzado en el proceso, y, tomando de base la motivación del presente trabajo, a continuación se presentan las conclusiones alcanzadas y líneas de investigación abiertas.

- (1) Se ha diseñado y propuesto una metodología de evaluación del rendimiento de la red tomando de base un caso de estudio de un sistema distribuido complejo como los Servicios de un Motor de Búsqueda, la metodología utilizó datos reales para evaluar el rendimiento del sistema. El estudio de la arquitectura de la aplicación permitió identificar condiciones de flujo de mensajes que tienen impacto en otros componentes del sistema que no están directamente relacionados con el software sino con los servicios de red. La metodología ha sido aplicada utilizando diferentes conjuntos de datos reales lo que permitió obtener datos más precisos para la evaluación del patrón de tráfico.

- (2) La metodología desarrollada permitió evaluar la red del SES, estudiar su arquitectura y patrón de tráfico que ha permitido proponer un modelo de simulación del sistema para ser utilizado como herramienta de investigación

de este sistema complejo. Este modelo provee información para la simulación y evaluación del SES, genera trazas de datos simuladas para experimentación y permite analizar el comportamiento del tráfico del SES a nivel de la red.

- (3) Se ha diseñado un modelo de enrutamiento basado en la aplicación tomando como línea de base modelos actuales y mejorando sus características al incorporar las condiciones y requerimientos aplicación real, que permite evaluar el rendimiento de la red de con mayor precisión. El modelo propuesto utiliza información local que permite que pueda ser utilizado con una tamaño de red mas grande sin que afecte el rendimiento de la red.

- (4) El modelo de enrutamiento propuesto ha sido comparado con otros modelos de enrutamiento mejorando las métricas de rendimiento de red que los modelos actuales proveen, agrega un componente de información a la toma de decisión del switch que son los mensajes de la aplicación, esto permite incorporar un criterio que provee información de los nodos que están distribuidos a lo largo de la red; de modo que sin aumentar la complejidad en la decisión o arquitectura del switch, se provee información global de la red.

- (5) Se ha diseñado e implementado el modelo de enrutamiento basado en la aplicación de SES que verificado mediante experimentación que permite obtener un mejor rendimiento de la red en términos de Latencia y Throughput.

Como líneas abiertas se ha visto la necesidad de ajustar el modelo de enrutamiento a la tecnología actual de utilizada en computo de altas prestaciones. Evaluar el modelo de enrutamiento con enfoque en la latencia de consultas del usuario.

Se ha visto que la latencia de las consulta esta predominado por el tiempo de procesamiento más que por el tiempo consumido en la red. Por lo que un aspecto

importante a analizar es como mejorar la red en términos de su capacidad o complejidad. Los atributos a analizar podrían ser cantidad de switches, capacidad de canales y buffers, enlaces o cantidad de nodos requeridos.

En relación a la metodología es necesario evaluarla mediante su aplicación en otro caso de estudio de un sistema complejo y distribuido.

6.1 Publicaciones

Publicación Realizada:
Improving the network of a search engine services through application-aware routing.
Estado: Aceptada
Resolución: 26-Abril-2017
Conferencia: Euro-Par 2017 - 23rd International European Conference on Parallel and Distributed Computing (EUROPAR).
País: España

Publicación Realizada:
Simulating a Search Engine Service focusing on Network Performance.
Estado: Aceptada
Resolución: 23-Marzo-2017
Conferencia: ICCS'17 - International Conference on Computational Science (ICCS).
País: Suiza

Publicación Realizada:
Application-Aware Routing Policy based on application pattern traffic.
Estado: Aceptada

Resolución: 30-Abril-2015

Conferencia: PDPTA'15 - The 21st International Conference on Parallel and Distributed Processing Techniques and Applications

País: EEUU

Publicación Realizada:

Evaluación de algoritmos de enrutamiento con enfoque en el patrón de comunicaciones de motores de búsqueda.

Estado: Aceptada.

Resolución: 2-Junio-2014

Conferencia: Jornadas de Paralelismo 2014

País: España

Bibliografía

[1] William James Dally and Brian Patrick Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

[2] Net Applications. Market Share Statistics for Internet Technologies, 2016. https://www.netmarketshare.com/.

[3] Min Chen, Hai Jin, Yonggang Wen, and Victor CM Leung. Enabling technologies for future data center networking: a primer. *Ieee Network*, 27 (4):8–15, 2013.

[4] Bernard J Jansen and Amanda Spink. How are we searching the world wide web? a comparison of nine search engine transaction logs. *Information processing & management*, 42(1):248–263, 2006.

[5] Kashif Bilal, Saif Ur Rehman Malik, Osman Khalid, Abdul Hameed, Enrique Alvarez, Vidura Wijaysekara, Rizwana Irfan, Sarjan Shrestha, Debjyoti Dwivedy, Mazhar Ali, et al. A taxonomy and survey on green data center networks. *Future Generation Computer Systems*, 36:189–208, 2014.

[6] Bruno Astuto A Nunes, Marc Mendonca, Xuan-Nam Nguyen, Katia Obraczka, and Thierry Turletti. A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials*, 16(3):1617–1634, 2014.

[7] B. Liu, Yinan Lin, and Y. Chen. Quantitative workload analysis and prediction using google cluster traces. In *2016 IEEE Conference on*

Computer Communications Workshops (INFOCOM WKSHPS), pages 935–940, April 2016. doi: 10.1109/INFCOMW.2016.7562213.

[8] TOP500 Supercomputer Sites. Top500 supercomputer sites. [online], 2016. https://www.top500.org/statistics/list/.

[9] Joe Carríon, Daniel Franco, and Emilio Luque. Application-aware routing policy based on application pattern traffic. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, page 142. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015.

[10] Mauricio Marin and Veronica Gil-Costa. Simulating search engines. *Computing in Science and Engineering*, 1:1–1, 2017.

[11] Veronica Gil-Costa, Jair Lobos, Alonso Inostrosa-Psijas, and Mauricio Marin. Capacity planning for vertical search engines: An approach based on coloured petri nets. In *International Conference on Application and Theory of Petri Nets and Concurrency*, pages 288–307. Springer, 2012.

[12] Alonso Inostrosa-Psijas, Gabriel Wainer, Veronica Gil-Costa, and Mauricio Marin. Devs modeling of large scale web search engines. In *Simulation Conference (WSC), 2014 Winter*, pages 3060–3071. IEEE, 2014.

[13] Pamela Baxter and Susan Jack. Qualitative case study methodology: Study design and implementation for novice researchers. *The qualitative report*, 13 (4):544–559, 2008.

[14] Jose Duato, Sudhakar Yalamanchili, and Lionel M Ni. *Interconnection networks: an engineering approach*. Morgan Kaufmann, 2003.

[15] Christoforos Kachris, Konstantinos Kanonakis, and Ioannis Tomkos. Optical interconnection networks in data centers: Recent trends and future challenges. *IEEE Communications Magazine*, 51(9):39–45, 2013.

[16] 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.

[17] Xin Yuan, Santosh Mahapatra, Michael Lang, and Scott Pakin. Lfti: A new performance metric for assessing interconnect designs for extreme-scale hpc systems. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 273–282. IEEE, 2014.

[18] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. URL http://ilpubs.stanford.edu:8090/422/. Previous number = SIDL-WP-1999-0120.

[19] Neelam Duhan, AK Sharma, and Komal Kumar Bhatia. Page ranking algorithms: a survey. In *Advance Computing Conference, 2009. IACC 2009. IEEE International*, pages 1530–1537. IEEE, 2009.

[20] Gyanendra Kumar, Neelam Duhan, and AK Sharma. Page ranking based on number of visits of links of web page. In *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on*, pages 11–14. IEEE, 2011.

[21] Amy N Langville and Carl D Meyer. *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2011.

[22] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[23] Andrei Z. Broder, Marc Najork, and Janet L. Wiener. Efficient url caching for world wide web crawling. In *Proceedings of the 12th International*

Conference on World Wide Web, WWW '03, pages 679–689, New York, NY, USA, 2003. ACM. ISBN 1-58113-680-3. doi: 10.1145/775152.775247. URL http://doi.acm.org/10.1145/775152.775247.

[24] Theophilus Benson, Aditya Akella, and David A. Maltz. Network traffic characteristics of data centers in the wild. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 267–280, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0483-2. doi: 10.1145/1879141.1879175. URL http://doi.acm.org/10.1145/1879141.1879175.

[25] Yantao Sun, Jing Chen, Q Lu, and Weiwei Fang. Diamond: an improved fat-tree architecture for large-scale data centers. *Journal of Communications*, 9(1):91–98, 2014.

[26] Theophilus Benson, Aditya Akella, and David A Maltz. Network traffic characteristics of data centers in the wild. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 267–280. ACM, 2010.

[27] Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subramanya, and Amin Vahdat. Portland: a scalable fault-tolerant layer 2 data center network fabric. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 39–50. ACM, 2009.

[28] Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The cost of a cloud: Research problems in data center networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73, December 2008. ISSN 0146-4833. doi: 10.1145/1496091.1496103. URL http://doi.acm.org/10.1145/1496091.1496103.

[29] Yingying Chen, Sourabh Jain, Vijay Kumar Adhikari, Zhi-Li Zhang, and Kuai Xu. A first look at inter-data center traffic characteristics via yahoo! datasets. In *INFOCOM, 2011 Proceedings IEEE*, pages 1620–1628. IEEE, 2011.

[30] Md Faizul Bari, Raouf Boutaba, Rafael Esteves, Lisandro Zambenedetti Granville, Maxim Podlesny, Md Golam Rabbani, Qi Zhang, and Mohamed Faten Zhani. Data center network virtualization: A survey. *IEEE Communications Surveys & Tutorials*, 15(2):909–928, 2013.

[31] Luiz André Barroso, Jeffrey Dean, and Urs Holzle. Web search for a planet: The google cluster architecture. *IEEE micro*, 23(2):22–28, 2003.

[32] Damon Horowitz and Sepandar D. Kamvar. The anatomy of a large-scale social search engine. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 431–440, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772735. URL http://doi.acm.org/10.1145/1772690.1772735.

[33] Christopher B Jones, Alia I Abdelmoty, David Finch, Gaihua Fu, and Subodh Vaid. The spirit spatial search engine: Architecture, ontologies and spatial indexing. In *International Conference on Geographic Information Science*, pages 125–139. Springer, 2004.

[34] Huajing Li, Isaac Councill, Wang-Chien Lee, and C. Lee Giles. Citeseerx: An architecture and web service design for an academic document search engine. In *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, pages 883–884, New York, NY, USA, 2006. ACM. ISBN 1-59593-323-9. doi: 10.1145/1135777.1135926. URL http://doi.acm.org/10.1145/1135777.1135926.

[35] Hai Dong, Farookh Khadeer Hussain, and Elizabeth Chang. A service search engine for the industrial digital ecosystems. *IEEE Transactions on Industrial Electronics*, 58(6):2183–2196, 2011.

[36] Robert C Seacord, Scott A Hissam, and Kurt C Wallnau. Agora: A search engine for software components. *IEEE Internet computing*, 2(6):62, 1998.

[37] Yilei Zhang, Zibin Zheng, and Michael R Lyu. Wsexpress: A qos-aware search engine for web services. In *Web Services (ICWS), 2010 IEEE International Conference on*, pages 91–98. IEEE, 2010.

[38] David L Tennenhouse, Jonathan M Smith, W David Sincoskie, David J Wetherall, and Gary J Minden. A survey of active network research. *IEEE communications Magazine*, 35(1):80–86, 1997.

[39] Julius Mueller and Thomas Magedanz. Towards a generic application aware network resource control function for next-generation-networks and beyond. In *Communications and Information Technologies (ISCIT), 2012 International Symposium on*, pages 877–882. IEEE, 2012.

[40] Maurizio Palesi, Rickard Holsmark, Shashi Kumar, and Vincenzo Catania. Application specific routing algorithms for networks on chip. *IEEE Transactions on Parallel and Distributed Systems*, 20(3):316–330, 2009.

[41] Diego Lugones, Daniel Franco, and Emilio Luque. Dynamic and distributed multipath routing policy for high-speed cluster networks. In *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*, pages 396–403. IEEE, 2009.

[42] Daniel Franco, I Garces, and Emilio Luque. A new method to make communication latency uniform: distributed routing balancing. In *Proceedings of the 13th international conference on Supercomputing*, pages 210–219. ACM, 1999.

[43] Carlos Núñez Castillo, Diego Lugones, Daniel Franco, Emilio Luque, and Martin Collier. Predictive and distributed routing balancing, an application-aware approach. *Procedia Computer Science*, 18:179–188, 2013.

[44] Arjun Singh. *Load-balanced routing in interconnection networks*. PhD thesis, Stanford University, 2005.

[45] Nan Jiang, John Kim, and William J Dally. Indirect adaptive routing on large scale interconnection networks. In *ACM SIGARCH Computer Architecture News*, volume 37, pages 220–231. ACM, 2009.

[46] InfiniBand® Trade Association (IBTA). Infiniband volume 1. release 1.3, nov 2015. [online], 2015. http://www.infinibandta.org.

[47] Mellanox Technologies. [online], 2016. http://www.mellanox.com.

[48] Mellanox Technologies. Mellanox ofed for linux user manual rev 3.30. [online], 2016. http://www.mellanox.com/related-docs/prod_software/Mellanox_OFED_Linux_User_Manual_v3.10.pdf.

[49] Torsten Hoefler, Timo Schneider, and Andrew Lumsdaine. Optimized routing for large-scale infiniband networks. In *High Performance Interconnects, 2009. HOTI 2009. 17th IEEE Symposium on*, pages 103–111. IEEE, 2009.

[50] Bartosz Bogdanski, Bjørn Dag Johnsen, Sven-Arne Reinemo, and Frank Olaf Sem-Jacobsen. Discovery and routing of degraded fat-trees. In *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2012 13th International Conference on*, pages 697–702. IEEE, 2012.

[51] Sadiq M Sait and Raed Al-Shaikh. Evaluating qlogic's dispersive routing on high performance clusters. In *Computer Modelling and Simulation (UKSim), 2012 UKSim 14th International Conference on*, pages 499–504. IEEE, 2012.

[52] Jared F Carr and Kenneth P Schmidt. Dnup routing algorithm description. Technical report, Pacific Northwest National Laboratory (PNNL), Richland, WA (US), 2011.

[53] Joel Hestness, Boris Grot, and Stephen W Keckler. Netrace: dependency-driven trace-based network-on-chip simulation. In *Proceedings of the Third International Workshop on Network on Chip Architectures*, pages 31–36. ACM, 2010.

[54] Nan Jiang, James Balfour, Daniel U Becker, Brian Towles, William J Dally, George Michelogiannakis, and John Kim. A detailed and flexible cycle-accurate network-on-chip simulator. In *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, pages 86–96. IEEE, 2013.