Estudio y análisis de mecanismos orientados al robustecimiento de ANTop, utilizando ruteo proactivo

Cristian Gaston Tejia

26 de abril de 2010

Índice general

Capítulo 1

Introducción

Audiencia Este documento esta dirigido a lectores con conocimientos técnicos en el área de redes, desarrollo de software, herramientas de diseño de software, UML, conocimiento de protocolos de ruteo, conceptos de direccionamiento general TCP/IP, y protocolos de ruteo interno y externo.

En el presente capítulo se presenta una reseña general del presente trabajo de tesis, comenzando por una introducción breve a la temática que necesita ser desarrollada, el tratamiento que tendrá dentro del presente trabajo y como se organizan las simulaciones y presentación de resultados.

1.1. Objetivo

Tomando como punto de partida el trabajo de tesis realizado por Alejandro Marcu [2] sobre redes Ad-hoc, se continúa con el desarrollo del protocolo ANTop-1. En primer lugar se propone desarrollar el algoritmo de ruteo proactivo descripto en [1], el cual es apto para redes que no presenten grandes cambios en su topología a medida que el tiempo transcurre, y en segundo lugar se propone desarrollar los mecanismos necesarios para dotar de robustez al protocolo.

Se propone estudiar el comportamiento del protocolo ANTop-1 utilizando un protocolo de ruteo proactivo, presentando las simulaciones del mismo bajo 2 tipos diferentes de topologías de red, una considerando redes aleatorias y otra con una topología cuyas conexiones siguen una ley de potencias.

Una vez comprobado el funcionamiento del protocolo proactivo, se propone un algoritmo de tratamiento de fallas a fin de soportar la desconxión abrupta de nodos, mediante una reorganización de las tablas de ruteo de los nodos.

Por otro lado se discute una técnica para la distribución de información en la red a fin de proporcionar un servicio de localización y registro de nodos más ágil, veloz y robusto.

En consecuencia el objeto de la tesis consiste en la implementación y análisis de:

- Las modificaciones necesarias a la primer implementación del protocolo ANTop-1 para ofrecer una distribución más equilibrada de información en la red. Este problema fué tratado en la versión original del protocolo, pero a un nivel más básico.
- Las modificaciones necesarias al protocolo ANTop-1 para proveer la robustez necesaria para soportar la caída aleatoria de nodos e implementación de la solución. Este problema no esta resuelto en la versión original del protocolo.
- El algoritmo de ruteo necesario para mantener y distribuir información de ruteo en la red considerando una topología que no cambie drásticamente en el tiempo.

Las principales dificultades a resolver son:

1. Múltiples caídas: Soportar la caída de cualquier tipo de nodo de la red, principalmente aquellos que concentren grandes volúmenes de información propia de la red, y la redistribución de la información que los mismos contenían.
2. Distribución de información: Distribuir la información referente a la red de manera tal de no discriminar a ningún nodo y distribuir equitativamente la información a través de la red.
3. Escalabilidad: Mantener un volumen de información que permita hacer frente a la caída de nodos en la red y continuar enrutando paquetes en forma ágil, pero a su vez acotar el crecimiento de las tablas de ruteo a fin de economizar energía y disminuir el volumen de información a almacenar.

1.2. Organización del documento

El presente trabajo de tesis se organiza de la siguiente manera, comenzando por el capítulo actual donde se presenta una reseña general del trabajo a realizar y y una breve idea de la motivación existente.

En el segundo capítulo se desarrolla en mayor profundidad la motivación del trabajo de tesis y el estado actual del arte, donde se presentan los protocolos previos a ANTop-1, que conceptualmente proveen las bases y el punto de partida para el desarrollo del mismo, luego se introduce el protocolo ANTop-1, donde se analizan las problemáticas existentes en el mismo y el impacto que tienen.

En el tercer capítulo se presenta la solución desarrollada en ANTop-2 a cada uno de los puntos descriptos en el segundo capítulo junto con los algoritmos y paquetes que se agregan a ANTop-1.

En el cuarto capítulo se presentan las simulaciones correspondientes a los algoritmos agregados como solución en el presente trabajo de tesis, diferenciando las simulaciones y presentación de resultados para distintas topologías de red, en este caso redes aleatorias y redes sin escala.

Finalmente en el quinto capítulo se desarrollan las conclusiones a las que se llega luego de ejecutar las simulaciones del protocolo, por último se presentan las mejoras y correcciones elaboradas en ANTop-2.

Como sección agregada en el Anexo 1 se encuentran todas aquellas figuras que no fueron agregadas en el capítulo de simulaciones por cuestiones de simplicidad, siendo que no agregaban información al momento de presentar los resultados, aunque pueden llegar a ser útiles en casos particulares.

Capítulo 2

Motivación

En el presente capítulo se ofrecen una reseña y breve introducción al conocimiento y la información necesaria para comprender todos los conceptos desarrollados en el presente trabajo de Tesis.

En primer lugar se presenta una reseña del 'estado del arte' referente a los diferentes protocolos de ruteo ad-hoc existentes, describiendo el funcionamiento de aquellos protocolos que fueron precursores en el tema y que ofrecieron las base para el desarrollo del protocolo ANTop-1, como ser *peernet* o *lplus*.

Una vez realizada la introducción general a los protocolos de ruteo ad-hoc procedemos con una introducción a la primer versión del protocolo ANTop, ANTop-1 de aquí en más, para lo cuál se hace referencia a la motivación de construcción y desarrollo del mismo, concluyendo con la solución propuesta por ANTop-1.

Como punto final se introduce a la problemática existente en el protocolo ANTop-1, su deficiencia en cuanto a la reutilización de direcciones, la imposibilidad de recuperar información una vez que un nodo abandona repentinamente de la red, y se discute la necesidad de una correcta política de distribución de información de localización.

2.1. Estado del arte

En la presente sección se presenta una introducción a los protocolos de ruteo Ad-hoc existentes, la solución propuesta por los mismos, sus limitaciones, ventajas y desventajas. Luego de haber efectuado la reseña a los protocolos existentes se introduce ANTop-1, junto con una descripción breve de su mecánica de funcionamiento y la manera en que resuelve el ruteo.

2.1.1. Protocolos de ruteo Ad-hoc

En la presente sección haremos una breve reseña de aquellos protocolos claves en el desarrollo de protocolos de ruteo Ad-hoc, que fueron utilizados como punto de partida para el estudio y desarrollo de las soluciones propuestas en el presente trabajo de tesis.

Chord

Comenzaremos nuestra reseña introduciendo el protocolo *Chord* [4], el cuál plantea un esquema para distribuir en forma consistente y determinista objetos de información dispersos en la red, a los cuales el trabajo refiere como claves *chord key*, dependiendo de la implementación una *chord key* puede almacenar información de cualquier tipo. *Chord* hace uso de mecanismos consistentes de *hash* para lograr encontrar el destino donde almacenar y localizar dichas claves. Una vez distribuida la información en la red plantea mecanismos para delegar la información sobre las claves en aquel caso en que algún nodo decidiera abandonar la red.

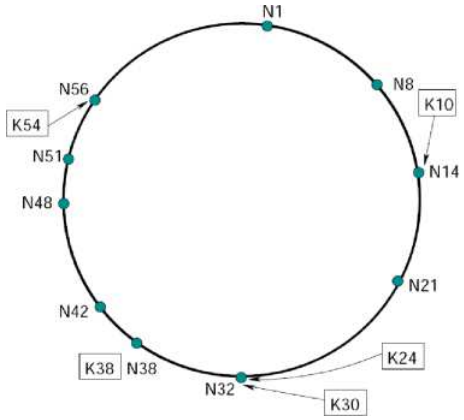
El mecanismo de distribución de claves planteado por *Chord* será luego utilizado en los conceptos de ruteo indirecto. *Chord* se basa en un esquema de ruteo simétrico, donde se asume que si A puede rutear hasta B, entonces B puede rutear hasta A, haciendo uso de IP como su capa de red, no propone esquema de direccionamiento ni asignación de direcciones, tampoco transporte o ruteo, por lo cuál *Chord* no ofrece por completo las herramientas para establecer una red AD-Hoc de la manera en que lo hace ANTop-1, simplemente permite una vez que un nodo se asocia a la red *Chord*, almacenar alguna clave en algún nodo de la red que luego pueda ser localizada . Al hacer uso de IP como capa de red no permite que el espacio de direcciones sea gestionado en forma automática por la red, sino que la asignación de direcciones se hace en forma estática o en forma dinámica por algún nodo bien conocido en la red, dando en esta ultima alternativa una gran responsabilidad a un solo nodo/s y convirtiéndolos en potenciales cuellos de botella o puntos de falla.

Para distribuir las claves el protocolo hace uso de un anillo circular, ver figura 2.1, ordenado en orden creciente según un ID numérico asignado a cada nodo, dado que utiliza IP como su capa de red, la asignación de direcciones y el ruteo son provistos por fuera del protocolo, y la conectividad entre los nodos fuera del anillo lógico es independiente de la topología y posición de nodos en la red.

Grid

Por otro lado encontramos los servicios de localización provistos por *Grid* [5] donde se introduce el concepto de servicios de localización para ruteo geográfico, en este esquema se hace uso de ruteo geográfico para localizar la posición de los nodos, cada uno de ellos debe contar con algún mecanismo para detectar cuál es su posición física (GPS) y de alguna manera poder informarla a los demás nodos. Cada nodo tendrá básicamente 2 formas de identificarse, una dirección

Figura 2.1: Esquema de distribución de claves utilizado en *Chord*. Fuente figura 2 de [4]



universal y una dirección física.

La información de localización de alguna manera debe ser almacenada y consultada, *Grid* utiliza una técnica similar a la desarrollada en *Chord* [4] a fin de almacenar la dirección geográfica de cada nodo. El concepto de *consistent hashing* es utilizado para localizar en forma determinista uno o más lugares a partir de la dirección universal de un nodo, los nodos registran su posición geográfica en múltiples lugares, y un nodo que quiera comunicarse con otro primero debe consultar su posición geográfica, para lo cual debe ir a buscarla en los mismos lugares donde el nodo destino previamente utilizó para almacenar su dirección física.

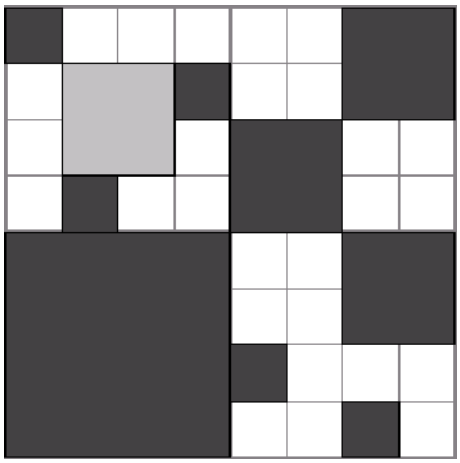
Una vez localizado un nodo en *Grid*, se hace uso de su posición geográfica para comunicarse con él, para lo cual cada nodo en base a su posición y la de sus vecinos inmediatos, decide cuál de ellos esta físicamente más cerca del nodo destino, esta decisión puede llevar a *dead ends*, donde la información puede llegar a un nodo que no tenga ningún vecino más cercano al destino que él mismo.

Grid necesita un esquema que provea direccionamiento y ruteo, para lo cual hace uso de ruteo geográfico, el problema del ruteo geográfico es su dependencia con la densidad de la red, siendo necesario tener una red densa para disminuir la posibilidad de formación de *Dead ends*, a diferencia de ANTop-1 donde mientras el nodo tenga una dirección de red que sea correcta en su entorno y dicha dirección se encuentre actualizada en los respectivos servidores de localización, logra cursar correctamente cualquier tipo de tráfico.

Una vez enunciado los principios básicos de funcionamiento es necesario interiorizarse en detalles que hacen poco práctico su uso, *Grid* propone una red organizada en sectores de diferente orden, siendo el particionamiento en sectores conocido para todos los nodos al momento de iniciarse el protocolo. Un nodo cualquiera al determinar los nodos que serán los encargados de almacenar su dirección geográfica debe conocer en que tipo de sección se encuentra, cuantos

tipos de sectores hay y quien se encuentra dentro, lo cual es un grave problema ya que implica el consenso de numerosos equipos sobre la jerarquía existente y división en sectores de la misma, ver figura 2.2.

Figura 2.2: Particionamiento del mundo según *Grid*. En color oscuro se muestran algunos cuadros de diferente orden, en color más claro se muestra una sección que no es una zona debido a su posición en la grilla. Fuente figura 2 de [5]



Si bien las ideas de ruteo geográfico y la distribución de información en forma escalada según regiones es interesante, el protocolo tiene la necesidad de recorrer toda el área para identificar tanto los niveles de las áreas como sus integrantes, y de esa manera encontrar quienes son los posibles servidores de localización con ID cercano al de cada nodo, claramente la solución no es escalable a grandes redes ni adecuada a situaciones donde los participantes entran y salen de la red constantemente.

DLM

El ruteo geográfico se basa la posición física de los nodos para enviar información de la mejor manera posible, aunque se basa fuertemente en la existencia de un servicio de localización eficiente y que escale correctamente, *DLM* [6] propone un sistema de localización jerárquico, donde la idea básicamente es similar a la propuesta por *Grid* [5], pero ofrece otro manera de seleccionar los posibles nodos donde almacenar la información de localización.

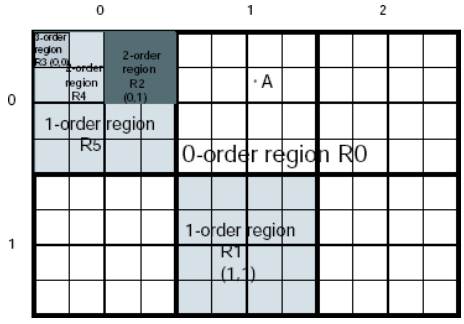
Al igual que *Grid* se basa en una topología física dividida en secciones o regiones, ver figura 2.3, dentro de las cuales serán localizados los nodos, se comienza seccionando la red en una grilla, donde el tamaño mínimo de una sección esta dado por el radio de cobertura de un nodo y la sección de máximo tamaño esta dada por la red completa. La asignación de direcciones es dependiente de la sección en la cual el nodo se encuentra, siendo esta dirección dependiente de la posición la que se almacena en los servidores de direcciones. Una vez que el nodo se mueve de una sección mínima a otra, su dirección cambia, pero solamente

cambia en parte, por lo cual tenemos 2 formas de la dirección, una absoluta con la resolución completa de la misma y una parcial donde indicamos a grandes rasgos donde se encuentra localizado el nodo. En el momento de almacenar la dirección y actualizarla en los diferentes servidores, se actualizará la dirección completa en aquellos servidores cercanos y solo se actualizará la parcial en los servidores lejanos cuando sea necesario, de esta manera los mecanismos de actualización no ocupan la red con mensajes no indispensables.

Al momento de localizar un nodo, aquellos que estén cercanos al destino tendrán cerca los servidores con la dirección absoluta por lo cual encontrarán directamente al destino, y aquellos que estén lejanos encontrarán primero la dirección parcial. Al llegar a las cercanías del nodo destino necesitarán hacer otra consulta a los servidores cercanos para obtener finalmente la dirección absoluta del nodo destino.

En el caso en que un nodo esté considerablemente lejos de su destino, es necesario consultar en forma iterativa su localización hasta obtener una dirección completa que identifique correctamente al nodo. Al igual que *Grid* hace uso de GPS como sistema de posicionamiento para el ruteo geográfico, lo cual implica ciertas restricciones como ser ahorro de energía y visibilidad de los satélites de posicionamiento, mientras que ANTop-1 necesita solamente estar dentro del radio de cobertura de algún nodo miembro de la red.

Figura 2.3: Particionamiento global de la red. Fuente figura 1 de [6]



Al utilizar un esquema de distribución de información basado en el dividir toda la red, encontramos la misma problemática encontrada en *Grid*, es necesario dividir el espacio físico donde se desarrolla la red en secciones o regiones, para luego posicionar al nodo en una u otra región, con lo cual se hace necesario conocer previamente la dimensión del espacio físico que será utilizado para el desarrollo de la red, a diferencia de ANTop-1 cuya topología no impone la necesidad de particionar la red en secciones o regiones.

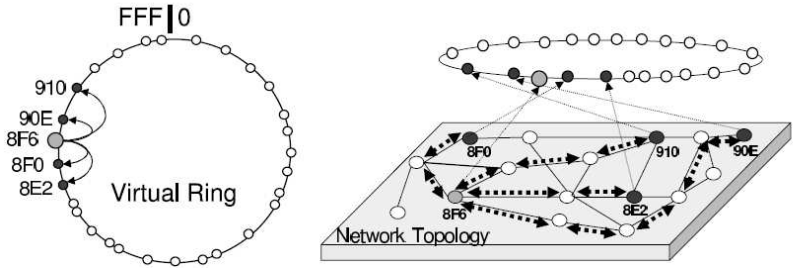
Virtual Ring Routing

Virtual Ring Routing fue presentado en la conferencia *Sigcomm 2006* en [42] y plantea un esquema similar a los protocolos hasta aqui mencionados, comienza

planteando la necesidad de diferenciar a los nodos en forma física y lógica para lo cual hace uso de enteros aleatorios para identificar a cada nodo en la red, por ejemplo partiendo desde 0 a FFFF en una red con 65535 nodos.

Una vez asignados los identificadores, se forma un anillo circular desde 0 a FFFF, donde al ser un anillo circular el nodo FFFF es el nodo anterior a 0. El anillo circular lógico es completamente independiente de la topología física, y 2 nodos que en el anillo circular son contiguos pueden no tener ningún contacto físico dentro de la red. Cada nodo mantiene una tabla de ruteo con 2 tipos de entradas, por un lado tiene R entradas correspondientes al anillo lógico, conteniendo $R/2$ vecinos lógicos del anillo a su derecha y $R/2$ vecinos lógicos del anillo circular a su izquierda; y por otro lado contiene a todos sus vecinos físicos, la figura 2.4 muestra el esquema de funcionamiento de protocolo.

Figura 2.4: Relación entre el anillo lógico y la topología física. Fuente figura 1 de [42]



Al momento de rutear paquetes lo hace mediante el anillo lógico y mediante las entradas correspondientes a los vecinos físicos, si el destino es un nodo que existe en su tabla de ruteo el paquete será ruteado directamente al destino, y si la entrada no está en su tabla de ruteo, será enviado al vecino numéricamente más cercano, siendo este último el que continúe la secuencia de ruteo. Por otro lado, el trayecto entre 2 vecinos adyacentes en el anillo lógico es potencialmente a través de múltiples nodos, por lo cual cuando un anillo esta siendo ruteado mediante el anillo lógico, el paquete puede ser redirigido por algún nodo intermedio que conozca una mejor ruta.

El esquema de utilizar números aleatorios para identificar nodos entre 0 y un límite superior finito y conocido, implica la necesidad de contar con un mecanismo de consenso para asignar los números a cada nodo o la existencia de un nodo o entidad que sea encargada de asignar los identificadores. Dicha necesidad impone restricciones en cuanto a la escalabilidad del protocolo.

El esquema de ruteo utilizado no hace un uso eficiente de la topología física sobre la cual está desplegada la red, cada nodo solamente hace uso de sus vecinos físicos directamente conectados, y la decisión inicial de ruteo se basa en la conectividad sobre el anillo lógico. El algoritmo se basa en la probabilidad que tiene un paquete de encontrar en su camino algún nodo que tenga una mejor ruta que la provista por el anillo lógico. Dado que se basa en la probabilidad de que el paquete encuentre algún nodo que ofrezca un buen camino, existe la probabilidad

de penalizar otros recorridos, forzándolos a usar el camino provisto por el anillo lógico, lo cuál resulta en una desición de ruteo ineficiente para dichos recorridos.

Landmark

El sistema introducido en *Landmark* y *L+* [7] mantiene activamente una jerarquía para proveer ruteo en una red cambiante. Los nodos en este tipo de redes tienen un ID único y permanente que no se utiliza directamente para rutear. Cada nodo también posee una dirección dinámica, que consiste en una lista de direcciones formada por los nodos localizados entre él mismo y la raíz de la red. Dicha dirección dinámica puede ser utilizada directamente para rutear, nuevamente se hace uso de un sistema que mapea $ID's$ a direcciones dinámicas.

Una vez que un nodo X debe actualizar su posición actual, mediante un *hash* obtiene la dirección del nodo Y que deberá almacenar la dirección dinámica de X, si algún nodo tiene la dirección Y será el encargado de almacenar la información, en cambio si ningún nodo tiene la dirección Y, entonces el nodo más cercano a Y almacenará la información. El mismo mecanismo es utilizado cuando un nodo necesita resolver cual es la dirección dinámica del nodo X.

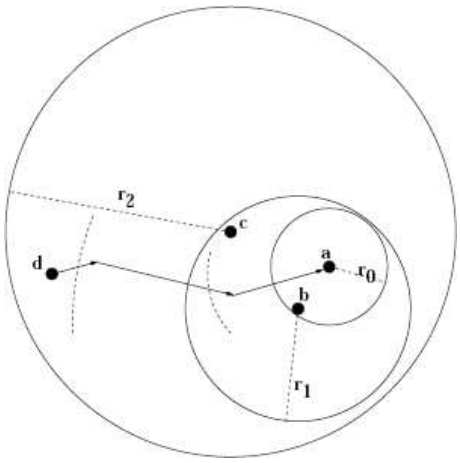
El protocolo hace uso de nodos específicos en cada nivel/jerarquía a los que llama *landmarks*, donde por cada nivel o jerarquía debe elegirse un nodo como el representativo del nivel/jerarquía, este mecanismo impone la necesidad de contar con algún método de consenso para determinar quien será el nodo elegido. Una vez elegido dicho nodo, se propaga su identidad dentro un determinado radio de cobertura (2 veces el radio de cobertura de un nivel), la información propagada es almacenada por un protocolo de tipo *Distance Vector* en los nodos que reciben la información, la identidad y posición de dicho nodo es utilizada luego para el ruteo de paquetes, ver figura 2.5.

La dirección que se utiliza para localizar a cada nodo consiste en un listado de *landmarks* desde el nodo mismo hasta la raíz de la red, dicho listado es utilizado al momento de rutear tráfico. El listado de marcas es descompuesto en todas sus componentes, se localiza la más cercana al nodo de la cual se tenga conocimiento en la tabla interna de ruteo y el paquete es enviado hacia dicha marca. Dado que todos los nodos que componen el nivel tienen en su base interna información de ruteo correspondiente a 2 niveles, el paquete puede ser reenrutado hacia el próximo *landmark* del listado antes de llegar al *landmark* anterior.

Tribe

Tribe [9] hace uso de muchos de los mecanismos y técnicas utilizadas en ANTop, pero con diferencias fundamentales en el esquema de ruteo que utiliza. *Tribe* es un sistema distribuido sin ningún tipo de control centralizado, donde todos los nodos tienen la misma responsabilidad y todas las comunicaciones son simétricas.

Figura 2.5: Ejemplo de jerarquía en *Landmark*. El nodo *a* es una marca de nivel 0. El nodo *b* es una marca de nivel 1, dentro de un radio *ro* de *a*. El nodo *c* es una marca de nivel 2, dentro de un radio *r1* de *b*. La dirección del nodo *a* es *a.b.c*. Las curvas en línea punteada representan los límites de distribución de información de ruteo desde *b* y *a*. Un paquete que *b* envía a *a.b.c* es enviado según el camino representado por la secuencia de flechas. Fuente figura 1 de [7]



Los nodos construyen una infraestructura de red que describe la posisción relativa del nodo en base a la vecindad actual del mismo. Cada nodo tiene un identificador global y se le asigna un identificador temporal acorde a su posición relativa en la red. *Tribe* utiliza mecanismos de *hashing* para distribuir información y lograr así un control distribuido escalable y tolerante a fallas. También hace uso de funciones de *hash* para traducir la dirección universal de cada nodo en una dirección virtual válida dentro de un espacio virtual de direcciones. La función de *hash* debe seleccionarse de manera de minimizar la probabilidad de que dos nodos obtengan la misma dirección virtual. Utilizando la dirección virtual se distribuye la información de localizacion del nodo, el mismo tipo de mecanismo utilizado por ANTop.

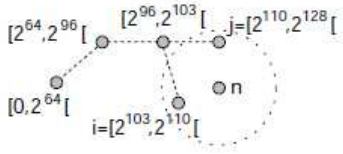
Tribe utiliza un esquema indirecto de ruteo. Al igual que ANTop cada nodo en *Tribe* posee tres identificadores, su dirección universal U, su dirección virtual V y su dirección relativa E. El mecanismo de ruteo indirecto será descripto mas adeltante al momento de ser utilizado por ANTop.

Durante el proceso de conexión a cada nodo se le asigna una región, la cual es cedida por sus vecinos. De esta manera cada nodo conoce la región de sus vecinos lo cual utilizará para el proceso de ruteo.

Utiliza un protocolo de control distribuido que permite a los nodos manejar regiones de un espacio virtual. Los nodos que esten cercanos fisicamente manejaran regiones virtuales también cercanas. Cada nodo que llega a la red manejará una region virtual, donde definimos una region virtual R como un set de números enteros en el rango $[0, 2^m - 1]$. El límite inferior de la región identifica al nodo univocamente en la topología y es su dirección relativa E. Un

ejemplo de la topología propuesta por *Tribe* y la asignación de regiones puede verse en la figura 2.6.

Figura 2.6: Topología de *Tribe*, fuente figura 2 de [9]



Tribe asume la existencia de algún mecanismo de inicio que permite a los nodos nuevos conocer a sus vecinos en la red. Luego del cuál se obtiene un listado de todos los espacios manejados por todos los vecinos, de todos las regiones vecinas se elige alguna de ellas, [9] propone utilizar la de mayor tamaño. Al elegir la región de un vecino, se toma la mitad superior para sí mismo, luego el nodo necesita registrarse en la red, para lo cuál calcula mediante una función de *hash* la región del nodo que será encargado de almacenar su información de localizacion. Cuando un nodo delega parte de su region administrada y asigna la mitad superior a un nuevo vecino, envia un mensaje a todos sus vecinos indicando que actualizen la informacion sobre la region que administra.

El mecanismo de ruteo es sencillo, para rutear un mensaje un nodo simplemente los envía al vecino que tenga su región mas cercana a la del nodo destino. El único aspecto no tratado por el esquema de ruteo de *Tribe* es la propagación de información de ruteo, ANTop es muy diferente en este aspecto como será visto mas adelante.

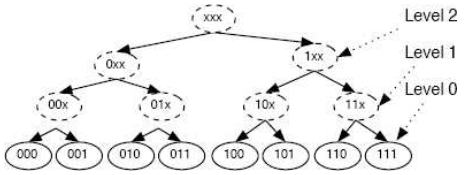
Tribe presenta problemas en lo que refiere a la partida de nodos, en primer lugar requiere que un nodo avise previamente si va a abandonar la red, no soporta la partida abrupta de un nodo. Una vez que un nodo avisa de su desconexión, si la región que manejaba el hijo forma un espacio continuo con el espacio del padre la transición es inmediata y no requiere ningún tipo de comunicación con los vecinos del nodo. Si el espacio no es contínuo con el del padre se producen problemas en la estructura de ruteo que tienen que ser corregidos con la ejecución de un algoritmo de reasignación de regiones que establece caminos alternos para mantener la consitencia del algoritmo de ruteo.

Peernet

Por último haremos una reseña de *peernet* [3], siendo este protocolo el que nos proveerá de los últimos detalles a tener en cuenta a la hora de hablar de ANTop-1.
En primera instancia *Peernet* se diferencia de aquellos protocolos que usan IP como su capa de red, siendo la dirección IP en realidad un identificador dependiente de la posición física del nodo en la red, impidiendo pensar en movilidad

de nodos, ya que al moverse un nodo potencialmente estaría entrando hacia otra red la cual muy probablemente utilice un prefijo diferente para direccionamiento.

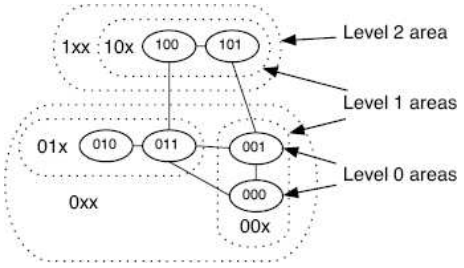
Figura 2.7: Arbol binario en un espacio de direcciones de 3 bits. Los nodos físicos existen solamente a nivel de hoja. Fuente figura 2 de [3]



Luego *peernet* plantea una fuerte diferencia entre dirección e identificador de un nodo, siendo la dirección la manera de reflejar su posición actual dentro de la red en todo momento. Este concepto plantea 2 puntos importantes, la necesidad de poseer un mecanismo de localización y la necesidad de actualizar en forma dinámica y constante la dirección de los nodos mientras se mueven. Como esquema de asignación de direcciones hace uso de un árbol binario con una longitud de dirección fija, ver figura 2.7, donde el primer nodo de la red posee la dirección 00000....00, siendo la raíz de la red, luego al ir uniéndose más nodos a la red se ubican en las hojas del árbol, por ejemplo en un esquema de 3 bits de longitud, el nodo 000 y 001 son hojas de nivel 0, donde el prefijo 00x es el representante de nivel 1 (solo el prefijo, los nodos reales son hojas en el esquema de árbol binario), y el prefijo 0xx es el representante del nivel 2, ver figura 2.8.

Al momento en que un nodo se une a la red obtiene una dirección entregada por los nodos que ya son parte de la red, luego se encarga de registrarla en un servidor de localización, mientras el nodo se mueve en la red, sus nuevos vecinos son los encargados de ofrecer nuevas direcciones acorde a su nueva posición en la red.

Figura 2.8: Red de 3 niveles. Las áreas de nivel 0 contienen solamente un nodo, las áreas de nivel n consisten de hasta 2 áreas de nivel *n-1*. Fuente figura 1 de [3]



El mecanismo de localización de información planteado por *peernet*, indica la necesidad de localizar un nodo tal que la relación 'Dirección de nodo' XOR 'identificador' sea mínima, lo cual implica un recorrido de todos los participantes de la red a fin de encontrar la menor relación para todas las direcciones, en consecuencia necesita un servicio de localización externo al protocolo. Dicha

restricción impone problemas en una red donde no se conoce la cantidad total de participantes o donde los participantes ingresan y salen continuamente de la red.

Por otro lado el esquema de ruteo que propone se basa en recorrer el árbol binario, donde no propaga información de ruteo como ANTop-1 sino que potencialmente todos los paquetes recorran la distancia que existe en el árbol binario utilizado para ofrecer las direcciones, no logra hacer uso de las conexiones intermedias entre nodos de la misma manera que ANTop-1.

2.1.2. Introducción al protocolo ANTop-1

El protocolo ANTop-1 hace uso de los conceptos introducidos en la sección anterior, cada uno de los protocolos antes mencionados presenta mecanismos adecuados para diferentes ocasiones, ANTop-1 toma lo mejor de cada uno de ellos y propone la estructura de hipercubo para desplegar el protocolo.

De los protocolos antes mencionados la influencia más visible pertenece a *peernet* en cuanto al mecanismo de asignación de direcciones y la separación entre dirección e identificador, del resto de los protocolos se toman los diferentes conceptos de distribución de información de localización y distribución de servidores.

Ruteo indirecto

El modelo de ruteo indirecto es materializado por un modelo de comunicación basado en la existencia de *rendezvous*, éste modelo es descrito en numerosos trabajos (ver por ejemplo: [17, 5, 6, 7, 8, 9]).

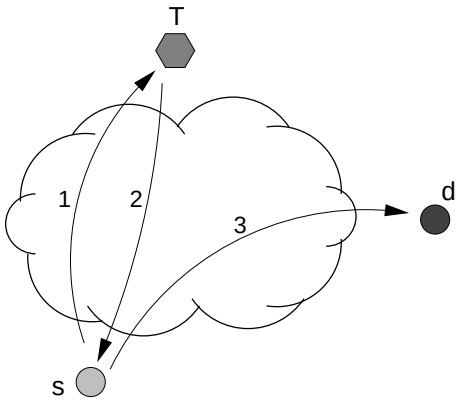
Nodos llamados rendesvouz son los responsables de almacenar la información de localización de otros nodos en la topología. El ruteo es realizado en forma indirecta a través de los nodos servidores de *rendezvous*, los cuales traducen un identificador universal de un nodo en su dirección de red dependiente de la topología. En esta tesis describiremos brevemente como se realiza el ruteo indirecto mediante el uso de la abstracción provista por tablas de *Hash* distribuidas, de aquí en más DHT (*Distributed Hash Tables*). Información detallada al respecto puede ser encontrada en [1].

El ruteo es realizado mediante una técnica basada directamente en la topología de la red. Cada nodo es identificado por su posición en la red, lo cual es traducido a una dirección dependiente de la topología. Es importante notar que la única manera de realizar ruteo es utilizando esta dirección.

En el caso general, cada nodo tiene 3 identificadores. El primero, es llamado identificador universal, U, y se supone que es conocido por cualquier otro nodo que desee comunicarse con el nodo en cuestión. Este identificador es totalmente independiente de cualquier característica referente a la red. El mismo podría ser una palabra, un valor numérico, o tal vez una dirección del estilo IP. El segundo identificador, la dirección virtual V, es la traducción de U dentro del

espacio virtual de direcciones, V. Esta traducción es realizada a través de una función de *hash* de tipo $f(U) = V = aU + (b \text{ mód } p)$. La dirección virtual V es utilizada para identificar el servidor de *rendezvous* correspondiente. El último identificador, la dirección relativa E, es la dirección dependiente de la topología actual del nodo. Observar que la dirección relativa del nodo va cambiando si el nodo se mueve, pero la dirección universal y la dirección virtual se mantienen sin cambios. La Figura 2.9 muestra el procedimiento de ruteo y el uso de los identificadores antes mencionados.

Figura 2.9: Fases de *Lookup*(flechas 1 y 2) y comunicación directa (flecha 3) en ruteo indirecto basado en DHT.



Cuando el nodo origen s necesita comunicarse con el destino d y no conoce la dirección relativa de d, primero debe contactar al *Rendezvous*, que es el nodo responsable de almacenar la dupla $\{U, E\}$, en nuestro caso $E = d$ (flecha 1). Llamemos a este nodo T_d. Entonces el mensaje enviado por s recorre la red hasta ser recibido por T_d, el nodo que maneja el sub-espacio de direcciones que contiene la dirección virtual V de d. Nótese que el nodo s no conoce la dirección relativa de d E_d, pero conoce la dirección V_d (obtenida a través de U_d). El nodo T_d conoce la dirección relativa E_d porque el nodo d previamente informó al nodo T_d su dirección actual. El nodo *rendezvous* T_d juega el rol de *rendezvous point* donde se almacena la localización de d.

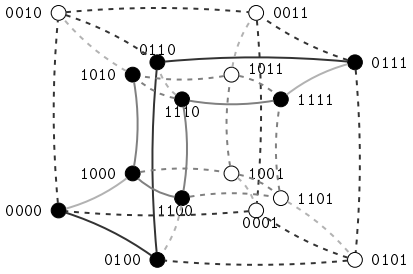
La particularidad de este enfoque es que el punto de *rendezvous* es identificado virtualmente y puede ser cualquier nodo físico en la red. Los nodos de *rendezvous* se encuentran distribuidos y dependen únicamente del identificador de cada nodo. Cuando interrogado por s a fin de resolver la localización de d, T_d responde con un mensaje que contiene la dirección relativa del nodo d, E_d (flecha 2). En este punto el nodo s puede comunicarse directamente con el nodo d (flecha 3).

La capa de red

Utilizando las coordenadas de los nodos en el hipercubo como direcciones relativas E, es posible mapear una red física en una red lógica. Para una red

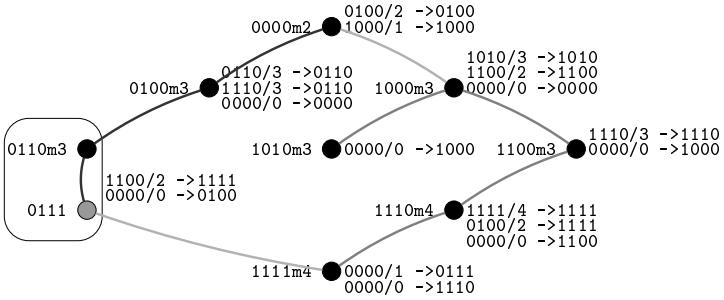
física cualquiera, el mapeo correspondiente produce un hipercubo incompleto, dado que el número de nodos presentes es menor a 2^d, y sus posibilidades de conexiones físicas no corresponden necesariamente a todas las aristas del hipercubo.

Figura 2.10: Hypercubo de dimensión = 4.



En la figura 2.11 mostramos una red arbitraria y su representación en el hipercubo en la figura 2.10, donde los nodos físicos son representados por círculos negros. La figura 2.11 también presenta una posible tabla de ruteo a la derecha de cada nodo.

Figura 2.11: Red aleatoria: Posición física de los nodos.



Hemos considerado que los nodos en la Figura 2.11 tienen un radio de cobertura circular. El hipercubo en la Figura 2.11 no representa todas las conexiones entre vecinos. Por ejemplo, el nodo 0100 tiene una conexión física con el nodo 1010, pero sus direcciones difieren en más de un bit, en consecuencia no están conectados en la estructura del hipercubo. Decimos entonces que el hipercubo esta incompleto. Sin embargo, aún perdiendo algunas conexiones, la red puede hacer uso de las adyacencias del hipercubo para ruteo.

La información contenida en cada nodo esta compuesta por la dirección principal, la dirección secundaria y su espacio de direcciones. La dirección principal corresponde a una dirección de red o dirección relativa E, la cual es asignada durante el proceso de conexión. Cuando un nodo se une a la red, la dirección

principal o primaria es seleccionada por el mismo a partir de las direcciones propuestas por sus vecinos (ya conectados a la red). Luego de obtener su dirección primaria, el nodo puede escoger una o más direcciones secundarias. Lo cual ocurre si estuviera físicamente conectado a otros vecinos que no son adyacentes en el hipercubo, por ejemplo sus direcciones de red E_i no son adyacentes a la dirección primaria del nodo. Por ejemplo el nodo 0110m3 en la figura 2.11, tiene su dirección y una dirección secundaria: 0111. Esta dirección secundaria es utilizada para conectar al nodo 0110 y 1111, dado que 0111 es adyacente a 1111, solamente difieren en un bit.

Cada nodo administra un sub-espacio del espacio de direcciones V. Este sub-espacio es utilizado para: (i) almacenar la base de datos para resolución de consultas de direcciones, 2 y (ii) asignar direcciones a nuevos nodos. La última función implica la delegación del espacio de direcciones correspondiente.

El espacio de direcciones de un nodo esta determinado por su dirección primaria y su máscara. La máscara esta representada por un determinado número de 'unos' desde la parte más izquierda, por ejemplo m3 es la máscara 1110 en un espacio de direcciones de dimensión 4. La dirección y su máscara (ejecutando un AND lógico con los bits) resultan en el espacio de direcciones administrado por el nodo. Este método es muy similar a las máscaras de subred IP, dado que la parte con ceros corresponde al espacio de direcciones administrado por el nodo. Por ejemplo, el nodo 0000m2 en la Figura 2.11 administra el espacio de direcciones 0000 (su dirección primaria), 0010, 0001 y 0011.

El primer parámetro a fijar es la dimensión d del hipercubo, la cual es conocida a priori por todos los participantes de la red. Por un lado este parámetro limita el número máximo de nodos posibles, pero por otro lado, otorga mayor flexibilidad para conectar nodos a través de direcciones secundarias. El problema es que cada nodo debería ser adyacente a un número máximo de nodos, idealmente a todos los nodos dentro de su radio de cobertura, a fin de estar fuertemente conectado. Intuitivamente, cuanto más grande sea el espacio de direcciones, más ricas las posibilidades de conexión de los nodos.

Ruteo indirecto en el hipercubo

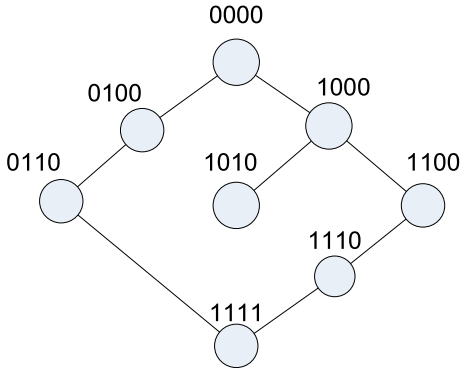
La técnica de ruteo indirecta consiste en la existencia de dos fases para el envío de información: (i) el origen consulta, al nodo de *rendezvous*, la dirección del nodo de destino utilizando su identificador universal, (ii) el origen envía el mensaje al destinatario. Este método presupone que hay un método para encontrar el nodo de *rendezvous*, dado que la única información disponible es la dirección de *rendezvous* del destinatario V la cuál es administrada por cierto nodo.

Como fue visto previamente, la dirección primaria y el espacio de direcciones de cada nodo es otorgado por otros nodos ya conectados. Cuando un nodo entrega una dirección, también delega una porción de su espacio de direcciones (generalmente la mitad superior del mismo) a un nuevo nodo. Por ejemplo en la figura 2.11, el nodo 0000m2 entregaría la dirección primaria y espacio de direc-

ciones 0010m3 a un nuevo nodo, produciendo un cambio en la máscara de 0000 de m2 a m3, y enviaría toda la información de resolución de direcciones almacenada para este espacio de direcciones. Esto significa que la dirección primaria de un nuevo nodo sería 0010, y administra las direcciones 0010 y 0011.

La utilización de este método para todos los nodos resulta en un árbol de distribución de las direcciones de red, al cual llamaremos árbol T en el resto de la tesis. La figura 2.12 muestra una posible topología, donde cortando el enlace entre los nodos 0111 y 1111, podemos observar un ejemplo del árbol T.

Figura 2.12: Arbol T.



Para una dirección de *rendezvous* V podrían darse 2 escenarios:

- Exista un nodo cuya dirección sea igual a V.
- La dirección V todavía no haya asignada a ningún nodo y esté contenida dentro del espacio de direcciones administrado de algún nodo.

En el caso de un hipercubo completo, solamente puede darse el primer escenario, y en un hipercubo incompleto puede suceder cualquiera de los 2. En el primer caso es cuestión de localizar al nodo correspondiente, siendo una tarea trivial en un hipercubo completo y una tarea un poco mas laboriosa en el caso del hipercubo incompleto. En el segundo caso es necesario localizar al nodo encargado de administrar el espacio de direcciones que contiene a la dirección V, lo cual es realizado a través del árbol T.

2.2. Problemática existente en ANTop-1

En esta sección se describen las diferentes problemáticas encontradas en el protocolo ANTop-1 , aquellas a las que el presente trabajo de tesis presenta soluciones. Comenzando por el esquema de direccionamiento utilizado y la manera de reciclar espacios de direcciones utilizados por nodos que abandonan la red, la recuperación y re-estructuración de las tablas de ruteo frente al abandono abrupto de nodos, y por último se describen las problemáticas relacionadas con la distribución de la información de localización, la selección de *Rendezvous* y

la respuesta frente a escenarios de falla en la red.

2.2.1. Direccionamiento en redes en hipercubo

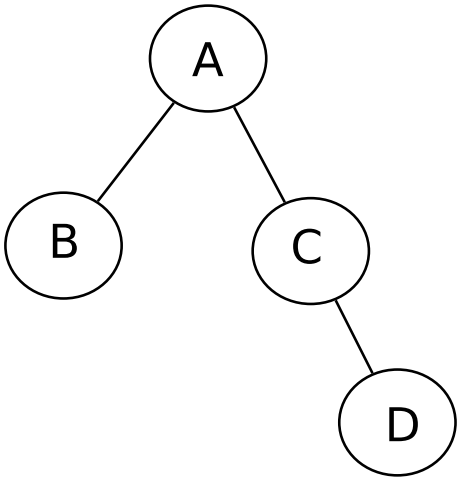
El hipercubo es una generalización de un cubo tridimensional a un número arbitrario d de dimensiones [34]. Cada nodo del hipercubo de dimensión d tiene coordenadas 0 o 1 para cada dimensión, cubriendo todas las combinaciones. Esto implica que el número total de nodos es 2^d. Cada nodo esta conectado a todos los nodos cuyas coordenadas difieren solamente en una dimensión. Por ejemplo, el cubo tiene un nodo en las coordenadas (0,0,0), o simplemente 000, el cual esta conectado con nodos en las coordenadas 001, 010, y 100, los cuales difieren solamente en una dimensión. Por lo cual, el grado, o el número de aristas de cada nodo es igual a la dimensión d.

La propiedad más importante del hipercubo es la adyacencia de nodos generada por su construcción, en la figura 2.10 se ve un hipercubo de dimensión $d = 4$. Podemos utilizar las coordenadas de un nodo como su dirección de red, luego la longitud de la dirección es d. Es simple ver que la distancia entre 2 nodos es medida mediante un XOR entre las 2 direcciones. Por ejemplo, la distancia entre los nodos 0100 y 0111 es 2 (solamente hay 2 bits de diferencia entre las direcciones), una ruta posible seria 0100 → 0110 → 0111.

Encontramos ejemplos interesantes de utilización de hipercubos en: parallel computing [35, 36] , peer-to-peer networks [37], genetic codes [38], fault-tolerant and redundant systems [39], message stability detection in distributed systems [40], parallel multiprocessor systems [41], data communication [34].

Terminología Antes de comenzar la descripción de los diferentes protocolos es necesario hacer algunas aclaraciones respecto de la terminología que utilizaremos a lo largo del documento, considérese topología presente en la figura 2.13:

Figura 2.13: Ejemplo de topología base



Tomando como referencia el nodo A, el nodo B y C serán sucesores de A, el nodo D será un sucesor de segundo orden de A. Tomando como referencia el nodo C, el nodo A será el predecesor de C, y el nodo D será un sucesor. Tomando como referencia el nodo D, el nodo C será el predecesor de D, y el nodo A será el predecesor de segundo orden.

Esquemas de ruteo Dependiendo de la disposición física de los nodos que interactúan en una red, el momento en que se conectan los mismos y su movilidad, podemos reconocer diferentes topologías, cada una de ellas presenta un comportamiento diferente que debe ser tomado en cuenta a la hora de implementar los diversos protocolos de control y transmisión de datos.

Una de las variables a considerar es la movilidad de los nodos dentro de la red, donde reconocemos 2 grandes tipos, aquellas donde los nodos pueden considerarse pseudo-estacionarios y las posiciones de los nodos pueden considerarse relativamente estables, y aquellas en las cuales los nodos participantes se mantienen en constante movimiento o donde la posición de un nodo tiene alta probabilidad de cambiar en el tiempo.

Cada una de las topologías antes mencionadas deberá contar con un algoritmo de ruteo diferente:

- Alta movilidad: en este caso es necesario tener un algoritmo altamente dinámico y poco predictivo, donde cada decisión de ruteo debe ser realizada sin consideración alguna de la posición del nodo en la red, donde lo más apropiado es realizar un descubrimiento de los nodos de la red.
- Baja movilidad: es el tipo de red en la que nos concentraremos en este trabajo, dado que presentan una topología relativamente estática podemos aplicar en el ruteo conceptos basados en la posición del nodo en la red, y distribuir en los nodos cercanos información referente al medio que rodea a cada nodo.

La variable de movilidad de los nodos lleva a otro aspecto a considerar en los protocolos de ruteo, la capacidad de resolver una petición de ruteo en forma autónoma o asistida:

- Resolución asistida: en este caso la resolución no es realizada en forma definitiva por cada nodo, sino que un nodo puede requerir la 'ayuda' de otros nodos para poder resolver el ruteo, este caso se presenta en aquellos esquemas de ruteo en que los nodos no tengan información completa sobre la red, que solamente sepan enrutar paquetes a destinos que fueron resueltos previamente y que no puedan tomar ningún tipo de decisión sobre un destino que no este presente en forma específica en su tabla de ruteo.
- Resolución autónoma: en este caso la resolución de ruteo es ejecutada en forma definitiva por cada nodo, cada nodo decide la mejor ruta para un paquete basándose completamente en la información que contienen sus tablas de ruteo, tiene la capacidad de tomar una decisión para cualquier tipo de destino y la decisión que el nodo tome es consistente en toda la red.

La combinación de las dos variables antes mencionadas lleva a un tercer aspecto, consecuencia de las anteriores, el tiempo que tarda en resolverse una ruta:

- Esquema asistido: la resolución asistida de peticiones de ruteo implica que los nodos no pueden mantener información predictiva para resolver el ruteo, el esquema asistido puede cometer 'errores' hasta que logra descubrir la ruta correcta para un paquete, por lo cual un paquete puede pasar varias veces por un nodo hasta que el nodo obtenga de algún nodo el ruteo 'correcto' para el destino en cuestión. En este esquema el tiempo que tarda en resolverse una ruta es elevado, consecuencia de la resolución asistida de la misma.

- Esquema autónomo: en este esquema dado que cada nodo tiene toda la información para realizar el ruteo en forma exitosa, el tiempo que toma resolver una consulta de ruteo es bajo, ya que consta solamente de procesar las tablas de ruteo correspondientes en el nodo local.

La ultima variable a considerar es el costo de mantenimiento de las tablas de ruteo utilizadas, puede verse que mantener una base de datos predictiva que pueda resolver las consultas de ruteo en forma definitiva en cada nodo presenta un costo de mantenimiento mucho mayor a un esquema que intenta resolver las consultas de ruteo solo cuando sea necesario.

El trabajo realizado por el ingeniero Alejandro Marcu, presta atención sobre las redes de alta movilidad, con ruteo asistido, en consecuencia con tiempos elevados de ruteo y bajo costo de mantenimiento de las tablas de ruteo. Implementa un esquema de ruteo reactivo, que presta servicios de resolución de rutas en reacción a solicitudes puntuales, necesitando en consecuencia un algoritmo de ruteo complejo que implica la interacción de diferentes nodos en la red para resolver una consulta.

El trabajo de tesis actual se centra en el desarrollo de los mecanismos necesarios para rutear paquetes en un esquema de baja movilidad, con ruteo autónomo en cada nodo, para lo cual se desarrolla un esquema de ruteo proactivo, en el cual se generan las tablas de ruteo sin que exista ninguna solicitud que dispare el proceso. En un esquema reactivo la información de ruteo se va recolectando *on demand* a medida que los paquetes necesitan ruteo hacia direcciones desconocidas, en cambio un protocolo proactivo debe mantener información de ruteo actualizada constantemente de manera tal que todas las resoluciones de rutas sean posibles de ejecutar en forma correcta y definitiva con la información contenida en la tabla de ruteo.

Un protocolo proactivo debe hacer uso de la información que puede obtener basándose en la ubicación del nodo en la red y la información que pueda obtener del conocimiento de ruteo que tengan sus vecinos, de esta manera para cualquier destino posible, debe poder resolverlo basándose en la información contenida en sus tablas de ruteo.

Donde la información contenida en sus tablas de ruteo tiene varias fuentes potenciales:

- Parientes: sean sucesores o predecesores, solo por asumir que el direccionamiento esta basado en una estructura de tipo árbol, mediante el análisis de prefijos debe poder reconocer que algún prefijo en particular se encuentra subiendo el árbol por su predecesor o descendiendo por alguna rama a través de un sucesor.

- Vecinos: el esquema de asignación de direcciones presenta la capacidad de establecer relación con vecinos físicos si los mismos tienen una dirección a distancia d de alguna de las direcciones locales del nodo, y permite a un nodo asumir N direcciones dentro de una misma rama a fin de poder establecer estas conexiones lógicas a distancia d. En el caso del hipercubo se utiliza d igual a 1. El protocolo debe poder hacer uso de estas relaciones con vecinos que no están relacionados a través de una relación sucesor-predecesor.

Recolectar la información de las fuentes antes mencionadas, su distribución hacia los nodos correspondientes en la red y el mantenimiento en respuesta a cambios en la topología de la red, deben ser tareas realizadas por el protocolo proactivo.

2.2.2. Recuperación ante fallas

El protocolo de ruteo reactivo implementado en ANTop-1 fue pensado y diseñado para ser aplicado en ambientes móviles, donde la posición física de los nodos no puede ser fijada en todo momento, por esta razón se entiende que las relaciones de los nodos (sucesor - predecesor - vecino físico - vecino lógico) están sujetas a cambios durante el tiempo, las razones pueden ser:

- Alejamiento: las relaciones entre nodos están sujetas al radio de cobertura de señal de cada nodo, si algún nodo se mueve fuera del radio de alcance de otro nodo, su relación se vera afectada temporal o definitivamente. El alejamiento puede producir que el nodo pierda contacto con alguno o todos sus vecinos, en el primer caso hablaremos de alejamiento parcial y en el segundo caso hablaremos de alejamiento total.

- Falla: de la misma manera que un nodo puede alejarse del radio de cobertura de otros, un nodo puede dejar de funcionar por completo, con lo cual este nodo pierde relación con todos los nodos de la red.

- Interferencia: sea el caso en que no existe alejamiento de un nodo, pero algún objeto se interpone entre 2 nodos impidiendo su comunicación, este caso es análogo al alejamiento parcial de un nodo.

A fines prácticos los efectos producidos por interferencia y falla son análogos al alejamiento parcial y total respectivamente.

Frente a una falla existe una serie de consecuencias que deben ser analizadas y tratadas:

- Aquellos nodos que utilizaban como destino para algún prefijo al nodo que abandona la red se verán obligados a utilizar alguna ruta alterna si es que la tuvieran.

- El predecesor del nodo que falla podría reutilizar el espacio de direcciones que en primera instancia fue cedido al nodo fallado.
- Los sucesores del nodo que falla podrían reestablecer su contacto con el árbol T a través de un predecesor sustituto.
- Las direcciones de *Rendezvous* que administraba el nodo deberán ser delegadas a otro nodo.

2.2.3. Distribución de información de localización

Como se fue discutido previamente al describir el ruteo indirecto el protocolo hace uso de 3 tipos de direcciones, una dirección universal U, una dirección virtual V y una dirección relativa E.

Donde la dirección virtual es obtenida en función de la dirección universal, la implementación del protocolo ANTop-1 utiliza esta dirección virtual V como la dirección relativa E del nodo que funcionará como servidor de *Rendezvous*. La traducción de dirección universal a dirección virtual es ejecutada mediante una función de *Hash*.

En resumen por cada dirección universal U es posible hallar V como resultado de una función de *Hash* aplicada a U, Hash(U), la cual es interpretada como la dirección relativa E del nodo que funcionará como *Rendezvous* para U.

Este mecanismo es utilizado para registrar a los nodos una vez que se conectan y para localizarlos cuando algún nodo quiere comunicarse con ellos.

Este esquema de registro presenta varios inconvenientes:

- Localización física de servidores
- Tolerancia ante fallas
- Distribución de información

Localización física de servidores El algoritmo de distribución no tiene manera de conocer la ubicación física de cada servidor de *Rendezvous*, con lo cual no se tiene control de la distancia que deben recorrer los paquetes para registrarse ni tampoco la distancia que debe recorrer una consulta de *Rendezvous*.

Por ejemplo supongamos que los nodos A y B se encuentran a 2 saltos de distancia uno de otro, y que el resultado de la función de *Hash* indica que ambos deben registrarse en el nodo C, el cual se encuentra a d nodos de distancia, siendo d el diámetro de la red. Ambos nodos al registrarse deberán esperar a que el paquete recorra d nodos para poder registrarse correctamente en la red, y en el momento en que A desee enviar un paquete a B, deberá esperar que un paquete recorra $2 * d$ nodos hasta obtener una respuesta indicando la localización del nodo B dando un tiempo elevado que depende del tamaño de la red, ver [2], claramente esta situación puede y necesita ser mejorada en alguna manera.

Tolerancia ante fallas Consideremos el caso en que un nodo se aleja completamente de todos los nodos de manera tal que su radio de cobertura no intersecta con el radio de cobertura de ningún nodo de la red, el nodo deja de ser parte de la red y podría considerarse que el mismo falló completamente.

En dicho caso las direcciones para las cuales el nodo funcionaba como servidor de *Rendezvous* se pierden y no podrán ser resultas por nadie, ya que era el único nodo en el cual las direcciones estaban registradas. En el caso en que un nodo abandona voluntariamente la red, ANTop-1 propone como primera aproximación a la solución, el delegar todos los registros de localización al predecesor del nodo que abandona la red. Este mecanismo no cubre los casos en que un nodo pierde su conectividad a la red sin previo aviso.

Distribución de información Una de las premisas de los esquemas de ruteo ad-hoc es que no existan nodos centrales que tengan una importancia superior sobre otros nodos de la red, en la cual todos los nodos sean de igual importancia donde la carga de almacenamiento y procesamiento se encuentra distribuida en todos los participantes de la red.

Al utilizar una función que distribuya la información de localización en la red, se irá cargando con información de *Rendezvous* a los diferentes nodos en la red, y debe promoverse un esquema de distribución que no cargue de información y procesamiento en forma despareja a los nodos, ya que una mayor cantidad de entradas de *Rendezvous* manejadas por un ruto se traduce en un mayor espacio utilizado para almacenar las rutas de *Rendezvous*, y en una mayor cantidad de procesamiento de solicitudes de resolución de localización, resultando en una mayor utilización de ciclos de procesamiento y en ultima instancia un mayor uso de energía. Además de convertir al nodo en punto de falla importante de la red.

En el próximo capítulo se discuten los mecanismos propuestos para solucionar las problemáticas antes mencionadas.

Capítulo 3

Mejoras introducidas en ANTop-2

En respuesta a las problemáticas introducidas en el capítulo anterior se propone como solución el desarrollo del protocolo ANTop-2, el cual toma como punto de partida el protocolo original y agrega las funcionalidades y mecanismos necesarios para superar las problemáticas descriptas.

El presente capítulo comienza con la descripción de una serie de cambios en la implementación del protocolo ANTop-1, los cuales fueron detectados al implementar y probar las características introducidas en ANTop-2. Luego se presenta un protocolo de ruteo proactivo, el esquema de ruteo proactivo si bien es parte de ANTop desde su concepción no fue incluido en la primer implementación del protocolo.

Luego se presentan los algoritmos desarrollados para resolver el reciclado de espacios de direcciones, seguido por el desarrollo de los algoritmos que resuelven los escenarios de falla en la red, y por último se presenta el desarrollo de un esquema de distribución de información de localización y los algoritmos necesarios para implementarlo.

La implementación del protoclo ANTop-2 puede encontrarse en el mismo sitio *web* donde se encuentra la versión original del simulador Quenas: *http://sourceforge.net/projects/quenas*.

3.1. Modificaciones en la nueva versión

Como consecuencia de las características introducidas en la nueva versión del protocolo y su implementación, se detectaron ciertos puntos de la implementación original que requerían ser modificados:

1. Secuencia en el mecánismo de conexión de nodos
2. Aprovechamiento de conectividad a través de direcciones secundarias

Secuencia en el mecánismo de conexión de nodos En la implementación actual la secuencia para conectar un nuevo nodo implica el envío y repeción de *Heart Beats* (HB) y en base a dicho intercambio infiere si el vecino está o no conectado físicamente, dicha interacción es la unión del algoritmo 1 y 3 de [2] junto con la maquina de estados descripta por la figura 2.11 de [2], una secuencia algoritmica que representa dicho comportamiento puede verse en el algoritmo 1.

Algorithm 1: Asignación de direcciones primarias

begin
  El nodo A detecta la presencia del nodo B mediante un paquete PAR;
  El nodo A propone una dirección primaria mediante un paquete PAP;
  El nodo B responde un paquete PANC;
  El nodo A agregado como vecino al nodo B en estado inactivo;
  El nodo B envía un HB luego de HB_TIMEOUT;
  El nodo A recibe el HB del nodo B y lo considera ahora como activo;
end

Considerando que el algoritmo de detección de ausencia de nodos verifica cada N milisegundos si su vecino esta conectado, podría darse que el mismo sea ejecutado en el momento en que se ha agregado el vecino como inactivo y todavía no se ha recibido un HB, razón por la cual podría erróneamente considerarse caído y comenzar a monitorear su presencia sin existir una necesidad real. En la implementación original del protocolo este comportamiento no produce ningún tipo de problema ya que la pérdida de un paquete de tipo HB de un vecino no inicia ningún mecanismo de monitoreo ni control. En la nueva implementación del protocolo este comportamiento no es desado y a fin de cambiarlo se implementa la solución descripta por el algoritmo 2.

Algorithm 2: Asignación de direcciones primarias

begin
  El nodo A detecta la presencia del nodo B mediante un paquete PAR;
  El nodo A propone una dirección primaria mediante un paquete PAP;
  El nodo B responde PANC y envía un paquete HB;
  El nodo A agregado como vecino al nodo B en estado inactivo;
  El nodo A recibe el HB del nodo B y lo considera ahora como activo;
end

De esta manera el nodo al momento de conectarse a la red envía inmediatamente un paquete de tipo HB, por lo cual el vecino automáticamente detecta su presencia activa en la red.

Aprovechamiento de conectividad a través de direcciones secundarias La conectividad mediante direcciones secundarias puede darse de 2 maneras:

1. Otorgamiento de direcciones secundarias

2. Establecimiento de adyacencia con un nodo cuya dirección se encuentra a 1 bit de distancia, sin contar la relación predecesor-sucesor

En el segundo caso, cuando un nodo recibe paquetes de tipo HB desde un vecino, analiza si puede establecer una relación a través de una dirección secundaria, si la dirección primaria del vecino se encuentra a un bit de diferencia de la dirección primaria del nodo que recibe el paquete de tipo HB se establece una relación a través del hipercubo, ver algoritmo 3.

Algorithm 3: Procesamiento de HB

```
1 begin
2 |  El nodo A recibe un HB del nodo B;
3 |  if El nodo B no esta registrado como vecino then
4 |  |  if Si la dirección primaria del nodo B dista en 1 bit de la
  |  |  dirección primaria del nodo A then
5 |  |  |  El nodo B es agregado como vecino activo;
6 |  |  end
7 |  end
8 end
```

Al ejecutar el algoritmo de detección de *gateway* alternativos, el cual será detallado más adelante, se detectaron numerosas conexiones entre nodos a distancia 1 que no podían ser utilizadas para ofrecerse como *gateway* alternativo. Este comportamiento se debe al hecho que existen nodos con dirección primaria a 1 bit de diferencia de algunas direcciones *secundarias* del nodo que recibe el HB, siendo que el algoritmo 3 solamente verifica la diferencia a 1 bit entre direcciones primarias, a fin de proveer una red de mayor conectividad se propone el algoritmo 4:

Algorithm 4: Procesamiento de HB

```
1 begin
2 |  El nodo A recibe un HB del nodo B;
3 |  if El nodo B no esta registrado como vecino then
4 |  |  if Si la dirección primaria del nodo B dista en 1 bit de alguna
  |  |  dirección del nodo A then
5 |  |  |  El nodo B es agregado como vecino activo;
6 |  |  end
7 |  end
8 end
```

De esta manera se logra una mayor aprovechamiento de las adyacencias y conexiones existentes, el algoritmo podría se extendido para verificar todas las direcciones secundarias de cada vecino contra todas las direcciones secundarias del nodo que recibe el HB, aunque este cambio implicaría que cada nodo HB enviado tenga un tamaño muy superior al actual, dado que debería contener las actuales direcciones secundarias del nodo, e implicaría un procesamiento mucho mayor en cada nodo, lo cual es contrario a las premisas de desarrollar un protocolo que sea efectivo tanto en gasto de energía como en poder de procesamiento.

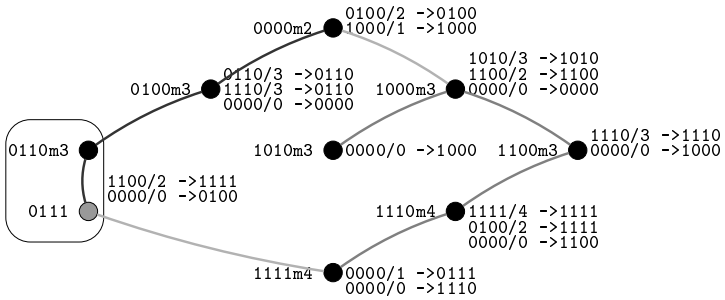
La implementación original al no soportar mecanismos de resistencia a fallas no requiere establecer la mayor cantidad posible de adyacencias.

3.2. Ruteo proactivo

En la sección 2.2.1 del presente trabajo de tesis, donde se introduce al lector de las limitaciones e información relevante al protocolo ANTop-1; se provee una explicación detallada de aquellas características que determinan y condicionan los diferentes esquemas de ruteo que un protocolo puede utilizar. En dicha sección se explica las razones de utilizar un protocolo de ruteo proactivo en redes cuya topología no cambia brusca y constantemente, dicho protocolo es adecuado a nuestro escenario ya que vamos a trabajar en este caso con redes cuya característica más notable es la baja movilidad física de los integrantes. En esta sección presentamos los fundamentos necesarios para el desarrollo de un protocolo de ruteo proactivo y la implementación desarrollada en ANTop-2.

A fin de comprender rápidamente el texto se recuerda al lector rever la terminología utilizada en el presente trabajo respecto de la nomenclatura de nodos según su posición en la red y relación con sus vecinos directos ver figura 2.13. Por otro lado en la figura 3.1 se muestra un ejemplo de una red junto con las respectivas tablas de ruteo en cada nodo, en la misma puede verse la estructura del árbol binario T y las rutas por defecto que se instalan hacia los predecesores de primer orden y sucesores de primer orden.

Figura 3.1: Árbol binario T y estructura de ruteo por defecto.



El protocolo de ruteo proactivo básicamente pretende hacer uso de la información ya existente en los nodos debido al mecanismo de asignación de direcciones tanto primarias como secundarias, y en base a determinados criterios que serán detallados más adelante permitir que diferentes nodos compartan información de ruteo y la distribuyan en un área de alcance acotado. Finalmente una vez que un nodo recibe información de ruteo de un vecino dependerá de los criterios de redistribución si la misma es propagada hacia otros vecinos cada vez más lejanos al nodo que inicialmente distribuye la información, luego será responsabilidad de cada nodo decidir si la información recibida le es de utilidad o

no.

El desarrollo del algoritmo proactivo de ruteo se basa en las siguientes consideraciones sobre el conocimiento de ruteo que se obtiene simplemente del mecanismo de asignación de direcciones:

- Todo nodo conoce el ruteo hacia sus sucesores, con lo cual si un nodo es parte de la subrama de un sucesor inmediatamente se sabe que se puede alcanzar al mismo a través del sucesor correspondiente.
- Todo nodo que no sea parte de una subrama de un sucesor y al que no se conozca otro medio especifico para alcanzarlo, se entiende que el mismo puede ser alcanzado a través del predecesor del nodo local. Ya que el paquete irá subiendo por el árbol T hasta llegar a algún nodo que sea predecesor del nodo destino. Nos referiremos a este comportamiento de ahora en más como Ruteo por *Default Gateway*.
- Las diversas conexiones secundarias que un nodo puede tener ofrecen visibilidad hacia otra subred fuera de la rama a la que pertenece el nodo, al momento de rutear paquetes se utilizaran estas conexiones alternas de la misma manera que se utilizan las subredes correspondientes a cada sucesor.
- Dado que las conexiones son estables, la información de ruteo de un nodo puede propagarse hacia otros nodos cercanos a fin de compartir información. Información referente a los predecesores y sucesores no es propagada ya que se infiere de la estructura del árbol. Pero las direcciones que se obtienen por enlaces secundarios pueden ser propagadas hacia nodos vecinos siempre y cuando dicha información les sea de utilidad.

3.2.1. Componentes de la tabla de ruteo

La tabla de ruteo esta compuesta por diversos tipos de rutas, cada una de ellas proviniendo de un tipo de enlace diferente, en primera instancia consideraremos aquellas que se den por el establecimiento del árbol de direcciones y por la asignación de direcciones secundarias, más adelante consideraremos las direcciones provenientes de los algoritmos de tratamiento de fallas:

- *Default Gateway*: es una dirección del estilo 00000....0/0 que apunta al predecesor de primer orden del nodo, considerando la forma en que esta armado el árbol siempre que un paquete sea enviado al predecesor en algún momento llegara a un punto en la red donde encontrara una rama o sucesor por el cual se pueda enrutar correctamente el paquete, caso contrario el nodo destino simplemente no existe.
- Sucesores: son direcciones que apuntan al espacio de direcciones cedido a cada uno de sus sucesores, de esta manera al intentar rutear un paquete hacia un nodo dentro de la rama de algún sucesor, el paquete será enrutado directamente por el sucesor correspondiente.
- Secundarias: son aquellas direcciones que corresponden a los enlaces secundarios que el nodo posee, una vez que se detecta un enlace secundario se

agrega un prefijo relativo a la dirección secundaria del nodo, el mecanismo de cálculo del prefijo será detallado más adelante.

- Propagadas: estas rutas son agregadas por el protocolo de ruteo proactivo, y son aquellas que serán mantenidas por él. El protocolo de ruteo proactivo que será detallado en esta sección, plantea que una vez que un vecino agregue una ruta secundaria, la información de ruteo que la misma provee será propagada hacia otros vecinos. Una vez recibida esta información desde algún vecino cada nodo decide si la misma es o no de utilidad, de serlo será instalada en la tabla de ruteo como una ruta propagada.

3.2.2. Poblado de la tabla de ruteo

Una vez conocidos los tipos de rutas que pueden coexistir en la tabla de ruteo, veremos en que situación cada una es instalada y los mecanismos de importación y exportación de rutas.

Default Gateway El *Default Gateway* es instalado una vez que el nodo ingresa en la red y es parte del hipercubo, al momento en que su predecesor de primer orden le asigna una dirección primaria y esta es aceptada, la dirección hacia el predecesor se convierte en la ruta por defecto, que será utilizada siempre que no se encuentre otra ruta más específica.

Sucesores Al momento de ceder un espacio de direcciones a un nuevo sucesor se agregan en la tabla de ruteo rutas directas hacia cada uno de los sucesores, con una máscara igual al espacio cedido al sucesor, de esta manera cuando un paquete llega para ser ruteado hacia la subred contenida en el espacio de direcciones asignado al sucesor, el paquete encontrará una ruta a través del sucesor correspondiente.

Secundarias Una vez asignada e instalada una ruta secundaria (para mayor detalle del mecanismo de asignación de direcciones secundarias ver ANTop-1 [2], algoritmo 5 página 15), se propone utilizar el enlace secundario para enrutar paquetes destinados a la subred correspondiente al espacio de direcciones del vecino adyacente, el tamaño del espacio de direcciones al que se puede llegar a través del enlace secundario (la máscara del prefijo) será resultado de un algoritmo que gradualmente disminuye la máscara (haciendo más grande el espacio) hasta el punto en que ya no es conveniente utilizar el enlace secundario por sobre el ruteo a través del árbol T, en dicho momento se decide que máscara se utilizará. Una vez instalada la ruta es redistribuida a todos los nodos vecinos exceptuando sucesores y al nodo que dio origen a la nueva ruta.

Propagadas Aquellas rutas redistribuidas por nodos cercanos son procesadas para decidir si son o no instaladas en la tabla de ruteo, en primera instancia se verifica si la ruta recibida provee una mejor manera de llegar a dicha subred que utilizando el *Default Gateway* (a través del árbol T), para lo cual se calcula la cantidad de saltos que debería hacer en cada una de las 2 opciones, si la ruta es conveniente se instala. Para decidir la máscara del prefijo a instalar se utiliza un algoritmo que disminuye la máscara del prefijo anunciado hasta que ya no

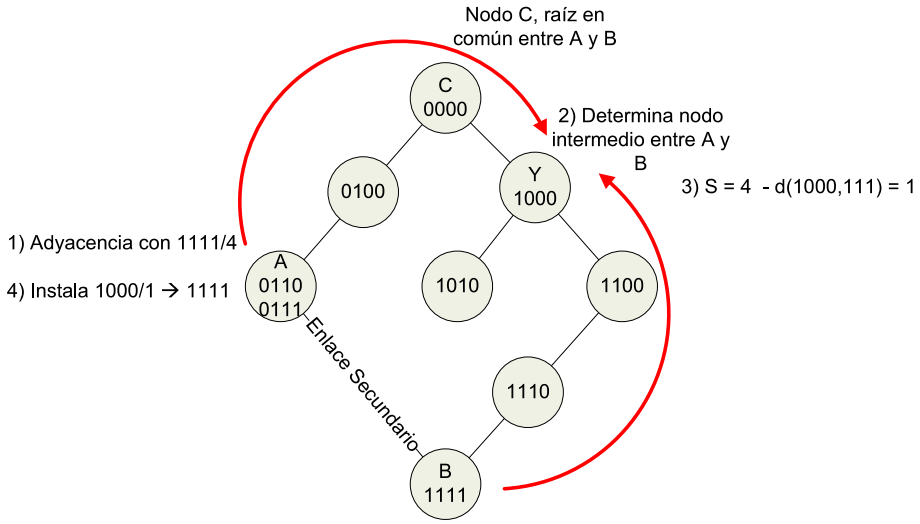
es conveniente utilizar la nueva ruta. Una vez instalada la ruta es redistribuida a todos los nodos vecinos exceptuando sucesores y al nodo que dió origen a la nueva ruta.

El algoritmo que decide la máscara a utilizar para los prefijos correspondientes a enlaces secundarios se basa en comparar en forma determinista la cantidad de saltos a realizar a través del árbol T versus la cantidad de saltos que requeriría utilizando el enlace secundario.

Considere un nodo A conectado a través de un enlace secundario al nodo B, ambos con un nodo raíz en común C, la distancia a través del árbol T desde A a C es 2 saltos, y la distancia entre B y C son 4 saltos.

El algoritmo comienza por encontrar el punto intermedio entre los nodos A y B, entendiéndose como nodo intermedio a aquél que se encuentre a mitad de camino entre los 2 recorriendo a través del árbol T; en este caso se encontrará a 3 saltos de distancia de A, llámese Y al nodo intermedio, el cual para este ejemplo se encontrará localizado dentro de la misma rama de B, ver figura 3.2.

Figura 3.2: Instalación de rutas secundarias.



Luego la cantidad de bits en común entre el nodo Y y el destino será la máscara S del prefijo a agregar. Ver algoritmo 5.

En el nodo T la dirección Y/S se instala directamente en la tabla de ruteo, luego la misma es propagada al predecesor de A. Al llegar al nodo predecesor de A nuevamente se calcula el valor de la máscara que se utilizará junto con el prefijo, obteniéndose la dirección $Y/S1$. Una vez obtenido el prefijo y su máscara, se calcula la distancia hasta el prefijo utilizando el árbol T y utilizando el hipercubo, si la distancia a través del árbol T es mayor entonces la ruta es aceptada y propagada, de no ser así se prefiere utilizar la ruta a través el árbol T. Ver pseudo codigo en el algoritmo 6.

Algorithm 5: Procesamiento de rutas secundarias

```
1  begin
2  |  Encontrar el nodo intermedio Y entre el destino y el nodo en cuestión;
3  |  Calcular la máscara S como la distancia entre el destino y el nodo Y;
4  |  Instalar la nueva ruta;
5  |  Propagar la nueva ruta;
6  end
```

Algorithm 6: Procesamiento de rutas propagadas

```
1  begin
2  |  Encontrar el nodo intermedio Y entre el destino y el nodo en cuestión;
3  |  Calcular la máscara S como la distancia entre el destino y el nodo Y;
4  |  Calcular la distancia DH a través del hipercube desde Y/S hasta el
   |  nodo en cuestión;
5  |  Calcular la distancia DT a través del árbol T desde Y/S hasta el
   |  nodo en cuestión;
6  |  if DT ≥ DH then
7  |  |  Instalar la nueva ruta;
8  |  |  Propagar la nueva ruta;
9  |  else
10 |  |  Ignorar la ruta;
11 |  end
12 end
```

De esta manera se distribuye información hacia los nodos que se encuentran en un área cercana al nodo que agrega el enlace secundario, la información dejará de propagarse cuando deje de ser conveniente. Si bien la metodología de calculo para decidir si la ruta es o no conveniente permite elegir la utilización del árbol T en algunos casos en que podría preferirse ir por el camino alterno, se prefiere esta manera de calculo a fin de que el comportamiento sea determinista y que todos los nodos en la red tomen decisiones consistentes.

Bajo el esquema propuesto, los nodos realizan ruteo buscando la mejor correspondencia entre prefijos dentro de su tabla de ruteo, donde podrán enrutar un paquete, preferentemente, a través de la información propagada, y en caso de no tener información propagada que sea de utilidad siempre podrá rutear a través de las direcciones correspondientes al árbol T.

En caso de encontrarse otro enlace secundario que provea una mejor manera de alcanzar algún prefijo, la mejor ruta será instalada y nuevamente propagada, al ser propagada todos los nodos cercanos nuevamente tomarán una decisión consistente respecto de la instalación del prefijo. El caso de retracción de rutas propagadas de la tabla de ruteo debido a abandono o partida de un nodo será visto en la próxima sección.

3.3. Resistencia a fallas

En esta sección veremos las soluciones propuestas al escenario donde algunos nodos abandonan repentinamente de la red, cabe recordar que ANTop-1 ofrece mecanismos para desconexión de nodos en los cuales un nodo avisa que se va a desconectar y el protocolo tiene tiempo de redistribuir toda la información referente al mismo. En el presente trabajo veremos los mecanismos necesarios para responder a las desconexiones abruptas de nodos, en la cual la red no tiene un previo aviso de la condición problemática.

De las problemáticas mencionadas en el capítulo 2 del presente trabajo de tesis, en esta sección nos centraremos en las siguientes:

- Detección de caídas: Los mecanismos de detección de fallas tienen como punto de partida la detección de la falla, en este caso nos interesa ver como se detectan las caídas o alejamientos de los diferentes tipos de vecinos que un nodo puede tener y que tipo de mecanismos son iniciados en cada caso.

- Descubrimiento de *Gateway* Alternos: el protocolo proactivo rutea principalmente mediante sucesores y rutas por defecto, en este punto se trata el caso en que sea necesario obtener un nuevo *Default Gateway* debido a que un predecesor de orden N abandona la red. En este caso enfrentaremos una reestructuración de la tabla de ruteo basada en la existencia de direcciones secundarias.

- Recuperación de direcciones puntuales: Un nodo que se aleja de la red dispara una serie de eventos que pueden llevar a una reestructuración de la misma, dicho nodo luego puede volver a ser parte de la red sin causar un impacto mayor, aunque existen casos en que nodos sustitutos pueden tomar el lugar de un nodo que previamente se alejó de la red.

- Reciclado de espacio de direcciones: Cuando existen desconexiones de nodos quedan ciertos espacios que pueden ser reutilizados o reasignados, se plantea la solución a dicho escenario y todos los posibles casos.

3.3.1. Detección de caídas

En primera instancia es necesario que los nodos involucrados y que siguen siendo parte de la red reconozcan que un vecino, sea sucesor o predecesor ha abandonado la red, las acciones a seguir serán diferentes dependiente sea un sucesor de primer orden o predecesor de primer orden del nodo que abandona la red. La manera en que se detecta que los nodos siguen o no en contacto con sus vecinos es efectuada mediante el mecanismo de *Heart beats* utilizado en ANTop-1, las modificaciones introducidas por ANTop-2 se encuentran en el tratamiento y los mecanismos que son activados cuando no se reciben los paquetes de tipo *Heart beat* desde algún vecino.

Cuando un nodo detecta que un sucesor de primer orden no esta presente, la primera acción que debe realizar es marcar la entrada asociada indicando que esta temporalmente perdida, de esta manera en una etapa posterior, podría volver a utilizarse dicho espacio. Luego de un tiempo determinado, sino se

tienen noticias respecto del sucesor ausente, se puede considerar que el mismo efectivamente se ha desconectado de la red, y dispone de la dirección que usaba el mismo para futuro uso.

Cuando un nodo detecta que su predecesor de primer orden esta ausente, no tiene el problema o necesidad de reciclar direcciones, sino que debe encontrar otro nodo que cumpla las funciones que le proveía el *Default Gateway*. El mecanismo y los criterios que serán utilizados para encontrar dicho reemplazo serán descriptos en la sección 3.3.2. Una vez que logra descubrir el nuevo nodo que funcionará como *Default Gateway*, es necesario ponerse en contacto con su predecesor de orden N, de esta manera el predecesor tendrá mayor información cuando realice tareas de reciclado en sus espacios asignados. En caso de haberse producido una caída de enlace el nodo en cuestión se pondrá en contacto con su predecesor de primer orden, en este caso el predecesor entiende que solo se ha caído el enlace y que el espacio no puede reciclarse de ninguna manera. En caso de haberse producido una desconexión de un nodo, la comunicación es establecida con su predecesor de segundo orden, el cual será el encargado de reciclar la dirección del nodo que abandonó la red. Una vez que el predecesor correspondiente fuere notificado, el predecesor confirmará la recepción de la notificación dando final al proceso de detección de caídas.

Posibles estados de un espacio de direcciones Los mecanismos descriptos en esta sección establecen un estado asociado a cada espacio de direcciones que posee un nodo, el estado es utilizado para identificar que tipo de eventos han sucedido en el sucesor al cual se delegó el espacio, y en última instancia como se puede disponer de dichos espacios de direcciones, los diferentes estados en los cuales puede encontrarse un espacio de direcciones son:

- ASSIGNED: Cuando el espacio fue asignado a un sucesor, y el sucesor se mantiene en permanente contacto con su predecesor a través de HB.
- LOST: Cuando el sucesor esta ausente o el predecesor esta ausente, pero el espacio no puede considerarse como libre.
- AVAILABLE: Corresponde a los sucesores que han abandonado la red y se considera que no van a volver, el espacio es reutilizable.
- UNAVAILABLE: Espacio de direcciones que fue asignado a un sucesor al cual no se tiene conexión, si bien el sucesor correspondiente no mantiene contacto con el predecesor a través de HB, su espacio de direcciones no puede ser utilizado para ningún propósito. Este sucesor virtual podría volverse real si comenzara a mantener contacto permanente con el predecesor a través de HB momento en el cuál su estado pasa a ser ASSIGNED.

A fin de detectar la ausencia de nodos o caídas de enlaces se monitorea la presencia de *Heart Beats* (HB), al detectarse la perdida de HBs de algún nodo se comienza un conteo para determinar la acción a seguir, el tiempo a esperar varia según el tipo de nodo que haya dejado de enviar los HB, la diferencia la dictamina el algoritmo que se ejecuta como consecuencia y los requerimientos del mismo, ver algoritmo 7

Algorithm 7: Vecinos que abandonan la red

```
   ;   /* Existe un tratamiento diferente por cada tipo de vecino
   que abandona la red, a fin de ejecutar cada uno de ellos se
   espera un tiempo prudencia diferenciado por cada tipo de
   vecino  */
 1 begin
 2   switch Tipo de vecino do
 3     case El vecino era un sucesor
 4       Esperar tiempo CHILD_LOST_WAIT_TIME;
 5     end
 6     case El vecino era el predecesor
 7       Esperar tiempo PARENT_LOST_WAIT_TIME;
 8     end
       ;  /* Nodo adyacente es aquel cuya dirección difere en 1
       bit sin ser sucesor o predecesor */
 9     case El vecino era un nodo adyacente
10       esperar ADYACENT_LOST_WAIT_TIME;
11     end
12   end
13 end
```

Una vez que se perdió contacto con los diferentes tipos de nodos, y los *timers* correspondientes a cada uno de ellos comienzan a expirar, deberán tomarse acciones correctivas que varían según el caso.

Una vez que expira el tiempo de espera asociado con la ausencia de un sucesor, debe verificarse si la subrama que se desprende del sucesor todavía esta conectada al árbol T a través de algún enlace secundario, siendo los paquetes de tipo *Child Alive Notification*(CAN) aquellos que se espera recibir de los sucesores de orden N en esta situación. De confirmarse la existencia de sucesores de orden N, con $N \geq 2$ que se encuentran conectados al árbol T a través de un enlace secundario, se indica que la dirección esta perdida, pero que no puede ser reasignada. Si no se recibió ningún paquete de tipo CAN, el espacio de direcciones es marcado como AVAILABLE y podrá ser reutilizado la próxima vez que un nodo solicite un espacio de direcciones mediante un paquete *Primary Address Request*(PAR).

En cualquier de los dos casos se interpreta que las rutas que provenían de dicho nodo y todos aquellos prefijos a los cuales se accedía a través del sucesor ya no son válidos, por lo cual son retirados de la tabla de ruteo, y se envía un mensaje de tipo *Route Retract*(RTR) hacia el predecesor del nodo notificando que retraiga todas aquellas rutas que hayan sido propagadas y que utilizaban este enlace como *gateway*. Ver algoritmo 8

Una vez que se determina que un predecesor esta fuera del radio de cobertura, y dado que el algoritmo necesita contar con un *Default Gateway*, el nodo debe comenzar la búsqueda para encontrar ruteo a través de un camino alterno, por lo cual comienza el algoritmo de descubrimiento de *gateway* alternos. Una

Algorithm 8: Sucesor que abandona la red

```
;       /* El algoritmo es ejecutado una vez que transcurre el
tiempo CHILD_LOST_WAIT_TIME y afecta a los espacios de
direcciones correspondientes a los sucesores que no enviaron
un paquete HB durante dicho lapso */
1  begin
2      if No se recibieron paquetes CAN de algún sucesor del nodo se
       ausenta ( sucesor de segundo orden de este nodo ) then
3          Marcar la entrada como AVAILABLE;
4          Eliminarlo de la tabla de vecinos;
5      else
           ;       /* Recibió algún paquete CAN desde un sucesor de
           segundo orden */
6          Marcar la dirección como LOST;
7      end
8      Armar un paquete RTR con todas las direcciones que fueron
       propagadas a través del sucesor;
9      Enviar paquete RTR al predecesor;
10     Notificar a otras aplicaciones de la ausencia ( ver que la aplicación de
       ruteo detecte el evento y marque la entrada como inactiva );
11 end
```

vez completado el algoritmo se analiza su resultado, de haber conseguido un nuevo *gateway*, se espera un tiempo determinado para que la red se estabilice, luego se notifica la identidad del *Gateway* alterno a todos los sucesores que se encuentren entre el nodo que detecta la caída de su predecesor y el nodo que funcionará como *Gateway* alterno.

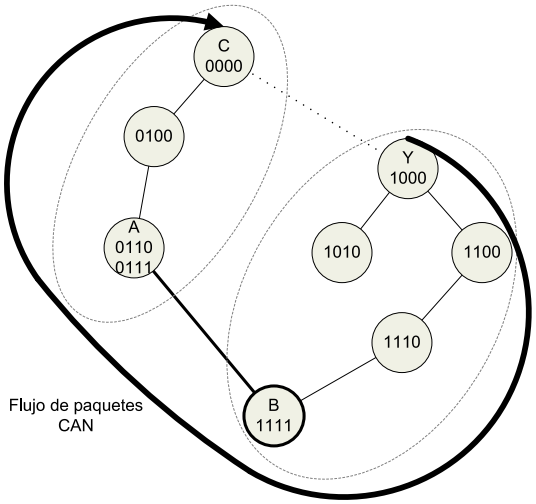
Al reestablecer contacto con el árbol T a través del *Gateway* alterno, el nodo debe ponerse en contacto con su sucesor de primer o segundo orden según haya ocurrido una caída de enlace o de nodo. Esta comunicación es necesaria a fin de que el nodo sucesor de orden N no interprete que la rama completa ha perdido contacto con el árbol T y disponga del espacio de direcciones para futuro reciclado. La comunicación necesaria es establecida a través de paquetes *Child Alive Notificaction*(CAN), los cuales dada la criticidad de la operación serán enviados hasta que sean confirmados mediante un paquete de tipo *Acknowledgment* ACK por parte del nodo sucesor de orden N, o que ya no sea necesario enviarlos debido a que la red se recupera o se pierde completo contacto con el árbol T. Ver algoritmo 9. En la figura 3.3 puede verse el recorrido que tuvo que haber logrado realizar un paquete de tipo CAN a fin de confirmar la existencia de sucesores conectados a la red.

Si no se logra determinar un *gateway* alterno, se entiende que la red esta particionada, es en este punto donde debería estudiarse el uso de algoritmos de *MERGE/SPLIT* de redes. El tratamiento de *MERGE* y *SPLIT* de redes excede el alcance del presente trabajo de tesis y no será discutido.

Algorithm 9: Predecesor que abandona la red

```
   ;       /* El algoritmo es ejecutado una vez que transcurre el
   tiempo PARENT_LOST_WAIT_TIME y el predecesor no envía ningún
   HB */
 1 begin
 2     Inicializar el algoritmo de descubrimiento de Gateway alternos, ver
       algoritmo 13;
 3     Esperar a la completitud del algoritmo de solicitud de Gateway
       alterno;
 4     if Se logra determinar un Gateway then
 5         Esperar un tiempo STABILIZE_GATEWAY_TIMEOUT a que
           todos los nodos instalen el gateway alterno;
 6         Enviar un paquete CAN al predecesor y al predecesor de segundo
           orden del nodo predecesor indicando que esta activo y dentro de
           la red;
 7         Esperar paquete ACK del paquete CAN enviado o tiempo
           CAN_ACK_TIMEOUT;
 8     else
           ;       /* No se logra determinar un Gateway alterno */
 9         La red esta particionada ;
           ;     /* podría darse un reset ordenado o el nuevo nodo
           raíz podría actuar como Rendezvous por defecto, todos
           los paquetes RVRegister le llegarían por defecto a el,
           y las consultas también  */
10         ;
11     end
       ;       /* La aplicación de ruteo debe detectar el evento y
       marcar la entrada como inactiva  */
12     Notificar a otras aplicaciones de la ausencia;
13 end
```

Figura 3.3: Recorrido de un paquete CAN para notificar a un predecesor sobre la existencia de sucesores todavía conectados al árbol binario T a través de algún enlace secundario que oficia como *gateway* alterno.



Dada la importancia de los paquetes de tipo CAN, es necesario confirmar la recepción de los mismos por parte de los predecesores de orden N, si el paquete no es confirmado mediante un ACK, el mismo es reenviado, ver algoritmo 10.

Algorithm 10: No confirmación de recepción de paquetes CAN

```
;       /* El algoritmo se ejecuta cuando transcurre un tiempo
CAN_ACK_TIMEOUT */
1 begin
2   if Hay paquetes que no hayan recibido paquete ACK then
3     Enviar nuevamente todos los paquetes sin ACK;
4     Esperar CAN_ACK_TIMEOUT por las respuestas;
5   end
6 end
```

Si es un nodo adyacente el cual se considera ausente, deben retraerse todas aquellas rutas que lo usen como *gateway*, sean propagadas o simplemente el prefijo correspondiente al enlace secundario, además de retraer las rutas debe notificarse a todos los nodos a los cuales previamente se propago estas rutas.

Adicionalmente, si dicho vecino estaba siendo utilizado como *default Gateway* debido a un evento de perdida de *gateway* anterior, debe se recomenzar el algoritmo de búsqueda de *gateway* Alternos. Ver algoritmo 11.

A fin de proveer un correcto funcionamiento a los algoritmos de detección de caída, es necesario mediante los paquetes de HB reconocer si algún nodo vecino, sea sucesor o predecesor esta nuevamente presente en el radio de alcance del nodo actual. Al recibir un paquete HB desde un nodo que sea considerado un sucesor podrían darse 2 situaciones:

Algorithm 11: Vecino adyacente que abandona la red

```
;       /* El algoritmo es ejecutado una vez que transcurre el
  tiempo ADYACENT_LOST_WAIT_TIME y afecta a aquellos vecinos que
  no hayan enviado un paquete HB durante dicho lapso */
1 begin
2   Marcar al vecino como ausente, estado LOST;
3   Desactivar cualquier ruta secundaria cuyo prefijo sea predecesor de la
    dirección del nodo que abandona la red;
4   Desactivar todas las rutas que hayan sido aprendidas a través de este
    nodo;
5   Armar un paquete RTR con todas las rutas que se desactivaron en los
    pasos anteriores;
6   Enviar paquete de RTR hacia el predecesor;
7   if El vecino estaba siendo utilizado como default gateway alterno
    then
8     Recomenzar algoritmo de detección de gateway alternos;
9   end
10 end
```

- Que provenga de un sucesor que se consideraba perdido, su espacio de direcciones marcado como LOST, por lo cual el estado del mismo pasa a ASSIGNED.
- Que provenga de un nodo que se sabe esta presente en la red, pero sin estar dentro del radio de cobertura del nodo actual, en cuyo caso se encontraba en estado UNAVAILABLE, en este caso el sucesor pasa a estado ASSIGNED. Un sucesor puede estar en estado UNAVAILABLE por 2 razones, en primera instancia por notificación a través de paquetes CAN una vez que se encuentra un *gateway* alterno, y por otro lado en aquellos momentos donde un nodo toma el lugar de otro adoptando a su predecesor y a aquellos sucesores que tenga en su radio de cobertura, ver sección 3.3.3.

Si el predecesor del nodo abandonó momentáneamente de la red, o el enlace entre ambos se vio afectado, el predecesor podría volver a entrar en contacto con el nodo, en dicho caso se recibirá un paquete HB proveniente del predecesor, en consecuencia deberán deshacerse aquellas modificaciones temporales inducidas por la perdida del predecesor inmediato, mediante paquetes de tipo *Gateway Reset*(GWRS) se inicia el procedimiento para revertir los efectos del algoritmo de detección de *gateway* alternos, ver algoritmo 12, el funcionamiento de estos últimos paquetes será descripto en la sección 3.3.2.

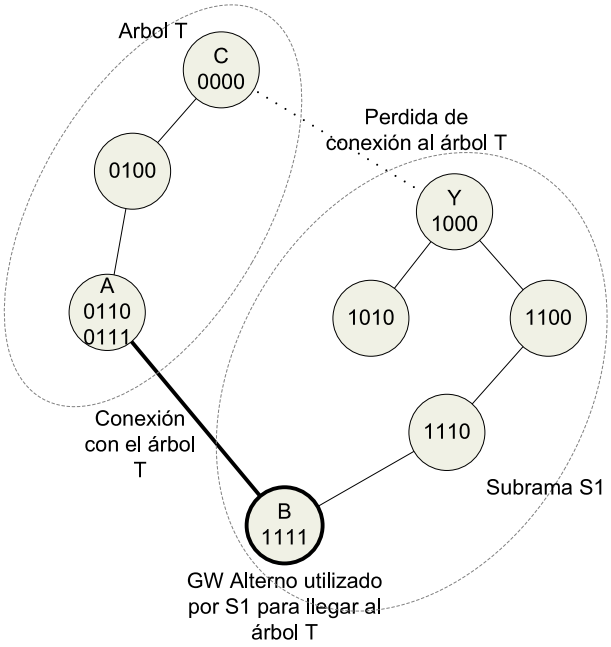
Algorithm 12: Procesamiento de *Heart Beats*

```
1  begin
2      Esperar paquetes HB durante un tiempo T_LHB;
3      Marcar como activos a los vecinos que enviaron paquetes HB;
4      for Cada vecino que envía un HB do
5          switch Tipo de vecino do
6              case Es el predecesor
7                  if La dirección del predecesor esta marcada como LOST
                   then
8                      Instala nuevamente el default gateway hacia el
                       predecesor;
9                      Vuelve al estado Estable;
10                     Desinstala las rutas temporales instaladas;
11                     Envía un paquete GWRS a todos los sucesores
                       indicando que vuelvan a usar el default gateway;
12                 end
13             end
14             case Es un sucesor
15                 if La direccion estaba marcada como UNAVAILABLE
                   then
16                     Marcar la dirección como ASSIGNED;
17                     switch Relación previa con el sucesor do
18                         case Previamente fue su sucesor
19                             Activarlo en la tabla de ruteo como sucesor;
20                         end
                           ;   /* En los casos donde existe sustitución
                           de nodos */
21                         case Previamente nunca tuvo contacto con el nodo
22                             Agregarlo en la tabla de ruteo;
23                         end
24                     end
25                 end
26             end
27         end
28     end
29 end
```

3.3.2. Descubrimiento de *Gateway* Alternos

La situación existe en el momento en que se ausenta un predecesor de un nodo, ver algoritmo 9, cuando esto ocurre, dicho nodo pierde a su *Default Gateway*, por lo cual ya no tiene forma de enrutar paquetes hacia los cuales no tiene una ruta específica, de existir dentro de la subrama que acaba de perder la conectividad por defecto al árbol T, una conexión con el árbol T mediante una ruta secundaria, podría utilizarse dicho enlace como nuevo *Default Gateway*. En la figura 3.4 se ve un ejemplo la mecánica propuesta para descubrir y hacer uso de los enlaces secundarios para volver a rutear paquetes hacia todos los destinos posibles.

Figura 3.4: Conectividad lograda con el árbol T a través de un enlace secundario que oficiará como *gateway* alterno.



Una vez detectada la perdida de un predecesor, comienza el algoritmo de búsqueda de *Default Gateway*, para lo cual se envía a todos los sucesores un paquete indicando la dirección del predecesor ausente, dicha dirección será utilizada para reconocer si algún nodo tiene o no una salida mediante un enlace secundario.

De existir algún nodo que tenga una dirección conectada a una subred externa a la rama cuya raíz es el nodo que se desconecta, el mismo indicará que es un posible *Gateway* alterno enviando un paquete de tipo *gateway proposal* al nodo que inicio el proceso.

De todos los posibles *gateway* recibidos se debe elegir uno, en este caso se propone elegir el primero que llega, el cuál debe ser propagado enviándolo a los sucesores que sea necesario. Se propone elegir la primer propuesta que llega al

suponer que llegó primera por recorrer menor cantidad de saltos. Todos aquellos nodos que estén entre el nodo que perdió el enlace y aquel que provee el *gateway*, deben actualizar el *gateway* por defecto que utilizan, y reemplazarlo por el recién descubierto *gateway* alterno.

Los nodos que se encuentran en el árbol T, y que se conectan a la rama que ejecuto el algoritmo de descubrimiento de *gateway*, deben actualizar su manera de llegar a esa rama, antes por el ruteo por defecto a través de las direcciones de los espacios cedidos, ahora mediante el enlace secundario, dentro del árbol T esta modificación debe ser instalada en todos los nodos que existan entre el nodo que oficia de *gateway* alterno y el nodo que perdió el enlace.

Cabe destacar que algunas rutas pueden ser redundantes, ya que mediante los mecanismos utilizados por el ruteo proactivo existen rutas en el árbol T agregadas fruto de la aparición de enlaces secundarios, las mismas deben ser sobrescritas por estas nuevas rutas temporales debido al *default gateway* alterno. Las nuevas rutas temporales no pueden competir con las previamente instaladas por el algoritmo de selección de rutas ya que poseen un máscara fija, cuando las anteriores tenían una máscara variable dependiendo de la distancia al enlace secundario, por dicha razón simplemente reemplazan a las rutas antiguas.

El nodo que detecta la ausencia de un predecesor, o la caída del enlace con el mismo, es el encargado de administrar el algoritmo de obtención de un *gateway* alterno, una vez detectada la caída del nodo/enlace, el nodo actual verifica si posee un enlace secundario que pueda ser utilizado como nuevo *gateway*, si lo tiene directamente lo utiliza y el algoritmo finaliza, de no encontrar una ruta válida envía un paquete *Gateway Request* (GWR) a sus sucesores, ellos son ahora los que deben intentar localizar una ruta alterna y notificarlo a este nodo.

Dentro del paquete de solicitud del paquete GWR debe indicar que dirección representa el espacio de direcciones que necesita un *gateway* alterno, para ello se toma la dirección del nodo ausente (el ancestro del nodo que detecta la caída) y se le coloca la máscara que originalmente tenía el nodo, dado que al ir asignando hijos su máscara fue incrementándose para representar todos aquellos espacios de direcciones que fueron cedidos. Ver algoritmo 13.

El algoritmo de obtención de *gateway* alternos no se ejecuta por siempre, el mismo tiene asociado un *timer* que indica cuando ha fallado el algoritmo, una vez que dicho *timer* expira, se reintenta 5 veces, si no se logra encontrar un *gateway* alterno se considera que la red esta particionada, ver algoritmo 14.

Si el nodo/enlace predecesor vuelve a ser detectado, se deben deshacer los cambios introducidos por el algoritmo de detección de *gateway* alternos, y notificar a los nodos sucesores mediante un paquete *Gateway Reset* (GWRS) que deben volver a su estado estable anterior, ver algoritmo 12.

Una vez comenzado el algoritmo de solicitud de *gateway* alternos, un paquete GWR es enviado a todos los sucesores del nodo que inicia el proceso, en cada nodo se realiza la misma verificación que hizo el nodo anterior, esto es, verificar si tiene acceso a través de un enlace secundario hacia el árbol T, no cualquier

Algorithm 13: Inicialización de algoritmo de descubrimiento de *gateway* alternos

```
   ;    /* Luego que detecta que el predecesor abandona la red */
1  begin
2  |  if Existe alguna ruta secundaria cuyo gateway no este dentro de la
   |  rama que se esta separando del árbol then
3  |  |  Instala una ruta default alternativa a través del enlace secundario;
4  |  else
5  |  |  Enviar un paquete GWR a todos los sucesores indicando la subred
   |  |  que necesita ser enrutada;
6  |  |  Esperar paquete GWU ó DEFAULT_GATEWAY_TIMEOUT;
7  |  end
8  end
```

Algorithm 14: Vencimiento del tiempo de espera para el descubrimiento de *gateway* alterno on Timeout

```
   ;                        /* Es ejecutado una vez que el tiempo
   DEFAULT_GATEWAY_TIMEOUT transcurre sin haber logrado
   encontrar un gateway alterno */
1  begin
2  |  if Intento encontrar un gateway alterno menos de 5 veces then
3  |  |  Duerme por un tiempo T_RETRY_GW_TIMER;
4  |  |  Recomienza el algoritmo de descubrimiento de gateway alternos e
   |  |  incrementa un contador de intentos, ver algoritmo 13;
5  |  else
6  |  |  Vuelve el contador de intentos a 0;
7  |  |  RED PARTICIONADA;
8  |  end
9  end
```

dirección secundaria es adecuada, sino alguna que esté relacionada con un nodo afuera de la rama que desciende del nodo que solicita la localización de un *gateway* alterno. Si no logra encontrar un *gateway* apropiado y tiene sucesores reenvía el paquete GWR hacia todos los sucesores. En cambio si logra encontrar un *gateway* apropiado envía un paquete *Gateway Proposal* (GWP) al nodo que solicitó el inicio del algoritmo de descubrimiento de *gateway* alternos. Este comportamiento esta representado en el algoritmo 15.

Algorithm 15: Solicitud de *gateway* alterno

```
;        /* El algoritmo es ejecutado una vez que se recibe un
paquete de tipo GWR, el cual indica la solicitud de localizar
un gateway alterno para un determinado prefijo */
1 begin
2   if Tiene ruteo hacia una rama fuera de la dirección a la cual se
    perdió conexión then
3     Enviar un paquete GWP al gateway al que detectó la caída de un
      enlace indicando su dirección y la dirección secundaria por la cual
      llega al prefijo solicitado ;
4   else
5     if Tiene sucesores then
6       Enviar un paquete GWR a todos los sucesores;
7     end
8   end
9 end
```

Una vez que algún sucesor encuentra un *gateway* apropiado el mismo envía un paquete *Gateway Proposal* (GWP), al ser recibido por el nodo destino se instala como *gateway* alterno. En la implementación actual se toma la primer respuesta que llega y no espera más paquetes GWP, otra opción posible es esperar múltiples respuestas y bajo un determinado criterio elegir una de ellas. Luego pasa a un estado temporal donde utiliza un *gateway* alterno que no es el principal, y propaga esta decisión hacia todos sus sucesores mediante un paquete de tipo GWU (Gateway Update). Ver algoritmo 16.

Una vez que se encontró y se decidió instalar un nuevo *gateway*, el mismo es propagado mediante el paquete GWU, el cual se procesa en diferentes secciones del árbol y de diferentes maneras, en los nodos que son sucesores del nodo que detecta la caída del enlace/nodo se debe instalar el nuevo *gateway* y reenviar a todos aquellos sucesores que corresponda. Si el nodo es el *gateway* instala su propia conexión secundaria como *gateway* y reenvía el paquete a través del enlace secundario para que en el árbol T estén notificados de la situación, finalmente crea un paquete RTR indicando todas las direcciones que deben ser retraídas y también lo envía hacia el árbol T.

Una vez que el paquete llega al árbol T también debe ser procesado hasta llegar al nodo predecesor del nodo/enlace que se detecto como caído. Todos los nodos en el camino entre el enlace secundario entre el árbol T y el predecesor del nodo/enlace caído reinstalan un nuevo *gateway* para el prefijo del nodo/enlace

Algorithm 16: Actualización de *gateway* alterno

```
;       /* El algoritmo es ejecutado en el nodo que detectó la
pérdida del sucesor e inició el algoritmo de descubrimiento
de gateway alterno */
1 begin
2   Vuelve el contador de intentos a 0;
3   Instalar el gatewayalterno;
4   Pasa a un estado de Estabilidad Temporal hasta que el enlace se
    reestablezca;
5   Armar paquete GWU con el nuevo gateway a instalar y todas las
    rutas que fueron aprendidas a través del predecesor;
6   Enviar paquete GWU a todos los sucesores indicando el nuevo
    gateway a instalar;
7 end
```

caído y propagan la información a su predecesor, hasta que llegue a algún nodo predecesor de la rama en la que se encuentra el nodo/enlace caído, desde ese punto el mensaje es reenviado a través del sucesor que corresponda a dicha rama, hasta llegar a su destino final. Ver algoritmo 17.

En el caso en que sea necesario retraer rutas se hará mediante un paquete de tipo *Route Retract* (RTR), en el mismo se indica el prefijo que se pierde y que esta provocando la retracción de rutas y las rutas que dicho nodo había propagado en primera instancia, tanto el prefijo como las rutas deben ser retraídas de la tabla de ruteo. Una vez procesado el paquete es reenviado al predecesor que corresponda. Ver algoritmo 18.

Algorithm 17: Propagado de *gateway* alterno

```
;        /* El algoritmo es ejecutado una vez que se recibe un
   paquete GWU, el cual es enviado por el nodo que inicia el
   algoritmo de descubrimiento de gateway alterno */
1  begin
2      switch Relación con el nodo que dió origen al paquete GWU do
3          case Es un predecesor de orden n
4              if El gateway es él mismo then
5                  Instalar como default gateway la conexión secundaria
                   correspondiente;
6                  Pasa a un estado de Estabilidad Temporal para dicha ruta;
7                  Reenviar a través del enlace secundario correspondiente;
8                  Armar un paquete RTR con todas las direcciones que
                   venían adjuntas al paquete GWP;
9                  Enviar paquete RTR a través de la conexión secundaria;
10             else
11                 Instalar nueva ruta;
12                 Pasa a un estado de Estabilidad Temporal para dicha ruta;
13                 Reenviar a todos los sucesores;
14             end
15         end
16         case Es un sucesor
17             Instalar la nueva ruta por defecto;
18             if Es un sucesor de orden mayor o igual a 2 then
19                 Reenviar al sucesor que esta en la misma rama que el
                   requesting Node;
20             end
21         end
22         case Es un vecino adyacente sin relación directa
23             Instalar la nueva para el prefijo que solicito el gateway a través
               del enlace que envío el paquete, marcarla como temporal;
24             Reenviar al predecesor;
25         end
26     end
27 end
```

Algorithm 18: Procesamiento de retracción de rutas

```
  ; /* Es ejecutado una vez que se recibe un paquete de tipo RTR
  */
1 begin
2 |  Desactivar aquellas rutas PROPAGADAS de la tabla de ruteo cuyo
  |  prefijo contenga a la dirección de la ruta perdida;
3 |  Desactivar todas las rutas contenidas en el paquete RTR;
4 |  if El paquete viene desde un sucesor del nodo y fue originado por un
  |  predecesor de la ruta retraída, ó si el evento ocurrió en una rama a la
  |  que no pertence el nodo que procesa then
5 |  |  Reenviar el paquete RTR al predecesor;
6 |  end
7 end
```

3.3.3. Reciclado de direcciones

El abandono de nodos en la red implica que el espacio de direcciones que tenia asignado queda totalmente en desuso y susceptible de ser reutilizado para otro nodo. Existen 2 escenarios en los cuales puede darse el caso de reutilización de direcciones:

- Reutilización de un espacio de direcciones: Se da en el caso que un nodo abandona completamente de la red, y el mismo no tiene sucesores o sus sucesores no mantienen contacto con el árbol T mediante un enlace secundario. En este caso el espacio de direcciones es completamente reutilizable.
- Reutilización puntual de direcciones: Cuando un nodo abandona de la red y sus sucesores siguen en contacto con el árbol T mediante enlaces secundarios, puede suceder que un nodo externo a la red ocupe físicamente el lugar que ocupaba el nodo ausente, en dicho caso el nuevo participante podría sustituir al anterior y asumir su dirección puntual.

Cuando un nodo pierde conectividad con su predecesor, y sus sucesores siguen conectados al árbol T, el predecesor del nodo no tiene posibilidad alguna de reutilizar el espacio de direcciones cedido a dicho nodo, el espacio de direcciones se considera inutilizable (UNAVAILABLE), el sucesor se conserva en forma virtual (el espacio de direcciones sigue estando asignado pero no hay conexión física). Estos espacios podrían ser reutilizados si los sucesores de orden N pierden su conexión con el árbol T a través de sus enlaces secundarios. Entonces es importante determinar la existencia de sucesores de orden N de aquellos nodos con los cuales se pierde conectividad, aquellos sucesores que sigan en contacto con el árbol T deberán comunicarlo utilizando paquetes destinados a tal fin.

Los paquetes *Child Alive Notification* (CAN) indican la existencia de un sucesor que todavía esta conectado a la red, el mismo puede ser enviado por un nodo a su predecesor inmediato o a su predecesor de segundo orden, según sea el caso de un abandono de un nodo o simplemente la caída de un enlace, si el paquete CAN llega desde un sucesor entonces se entiende que se cayó el enlace entre los nodos, si el paquete llega desde un sucesor de segundo orden se entiende que el nodo sucesor se ausentó, y que su sucesor (de segundo orden para este nodo) detecto la caída del mismo, ver algoritmo 19.

Una vez que un nodo se aleja del radio de cobertura de sus vecinos, puede ocurrir que el mismo nodo vuelva al radio de cobertura ocupado anteriormente o puede darse el caso que un nuevo nodo intente unirse a la red, y dicho nodo podría físicamente estar localizado cerca de la posición que ocupaba el nodo que anteriormente abandonó la red. A fin de representar dicha situación reconocemos 3 tipos de nodos:

- Predecesor del nodo ausente: Este nodo debe reconocer que su sucesor no esta presente, y debe tener la habilidad de reutilizar el espacio de direcciones que el mismo manejaba.
- Nodo sucesor del nodo ausente: Este nodo debe reconocer que su predecesor no esta presente en la red y debe comenzar a localizar un nuevo *Default Gateway* mediante el cual rutear paquetes.

Algorithm 19: Procesamiento de notificaciones de sucesores sin conexión directa

```
   ;  /* Ejecutado en respuesta a la llegada de paquetes de tipo
   CAN, luego del establecimiento de un gateway alterno */
 1 begin
 2     if El sucesor es de orden 1 entonces se perdió el enlace then
 3         Marcar el espacio como UNAVAILABLE;
 4         Enviar paquete ACK al sucesor de segundo orden;
 5     else
 6         Localizar el sucesor que sea ancestro del nodo que envía el
           paquete CAN;
 7         switch Estado del sucesor encontrado do
               ; /* Representa una pérdida de enlace del sucesor de
               segundo orden con el sucesor de primer orden */
 8             case Sigue conectado
 9                 Descartar, el paquete también llegó al sucesor;
10             end
               ;                    /* El sucesor se fue de la red */
11             case Esta en estado LOST
12                 Almacenar la dirección del sucesor de segundo orden
                   asociada a la entrada del sucesor perdido;
13                 Enviar paquete ACK al sucesor de segundo orden;
14             end
15         end
16     end
17 end
```

- Nodo Reemplazante: Este nodo es el que toma el lugar del nodo que se ausento de la red, el mismo solicita un espacio de direcciones al predecesor del nodo original, el cual puede cederle un espacio de direcciones nuevo o reutilizar uno correspondiente al nodo ausente, en caso de reutilizar el espacio de un nodo ausente deberá resolver como comunicarse con los sucesores del nodo ausente que todavía sean parte de la red.

Al momento de unirse a la red un nodo solicita a los nodos en su rango de alcance que le asignen una dirección primaria, aquellos nodos que detectaron la desconexión de un sucesor, además de proponer alguna dirección indicarán la dirección de su sucesor ausente, aquellos nodos que han perdido su predecesor, además de proponer una dirección primaria indican también la dirección del predecesor ausente.

Un nodo podrá aceptar reemplazar a otro solamente si recibe paquetes PAP desde al menos un sucesor del nodo ausente y del predecesor del nodo ausente. Cuando un nodo acepta utilizar una dirección que pertenecía a un nodo que se desconecto de la red, necesita que saber cuantos sucesores tenía el nodo anterior, ya que necesita conocer si existen espacios de direcciones que no estén disponibles, esto aplica solo para conocer a aquellos sucesores que no están dentro del radio de cobertura del nodo, por definición al menos un sucesor previo esta dentro de su radio de cobertura. Para averiguar quienes son sus sucesores virtuales (sucesores en el espacio de direcciones pero sin conexión física real) envía un paquete de consulta a su predecesor, el cual le informa quienes son todos los sucesores que el nodo anterior tenía, el predecesor conoce a todos los sucesores que todavía están conectados a la red ya que cada uno de ellos se puso en contacto mediante paquetes CAN luego de encontrar un *Gateway* alterno. Aquellos sucesores que no se pusieron en contacto con el predecesor de segundo orden no son de interés en el presente trabajo ya que corresponden al tratamiento de *Split and Merge* de redes, el cual no será tratado aquí. Una vez recibidas las direcciones de todos los sucesores del nodo anterior, las almacena en su vector de direcciones asignadas, indicando que no las tiene ni las puede asignar. Luego mediante los paquetes HB de los nodos dentro del radio de alcance, puede detectar si algún sucesor marcado como no disponible esta a su alcance, de ser así lo agrega a la tabla de ruteo y lo marca como sucesor.

El primer algoritmo que sufre modificaciones es el relacionado al ofrecimiento de direcciones primarias, ya que será el encargado de verificar la posibilidad de reutilizar direcciones y ofrecerlas como tal. Al momento de ofrecer direcciones primarias un nodo gradualmente reduce el espacio de direcciones que administra, aunque al detectar ausencias de sucesores puede recuperar en parte su espacio de direcciones original, un sucesor ausente puede todavía contener sucesores de orden N conectados al árbol T mediante enlaces secundarios, pero podría suceder que luego ese enlace secundario deje de existir, permitiendo al nodo recuperar otro espacio de direcciones previamente cedido. Consultar constantemente para verificar si algún nodo que previamente comunico su presencia mediante un paquete CAN sigue estando conectado a la red implicaría un gran número de paquetes extra cursando la red, por lo cual dicha verificación se realiza solamente en el caso que el nodo se este quedando sin espacios de direcciones a asignar.

Algorithm 20: Inicialización de algoritmo de recuperación de espacios de direcciones

```
   ;   /* El algoritmo se inicia en forma asincrónica una vez que
   se detecta que no quedan muchos espacios de direcciones para
   delegar */
 1 begin
 2 |   Enviar un paquete CAQ a todos los sucesores de segundo orden
   |   marcados como LOST;
 3 |   Esperar paquete CAR o tiempo T_CAR;
 4 |   switch Evento sucedido do
 5 |   |   case Llega un paquete T_CAR
 6 |   |   |   Dejar de esperar paquete CAR para ese espacio;
 7 |   |   end
 8 |   |   case Pasa tiempo T_CAR
 9 |   |   |   Marcar como AVAILABLE a todos los espacios que sigan
   |   |   |   esperando el paquete CAR;
10 |   |   end
11 |   end
12 end
```

Durante el proceso de ofrecimiento de direcciones primarias la primera acción es verificar si comienza o no el algoritmo de recupero de direcciones, lo cual será solo necesario cuando la cantidad de direcciones disponibles para asignar sea menor a un limite predefinido, ver algoritmo 20. En caso de ser necesario recuperar espacios de direcciones envía paquetes *Child Alive Query* (CAQ) a todos los sucesores virtuales, ver algoritmo 21, aquellos que no contesten mediante un paquete *Child Alive Response* (CAR) verán su estado cambiar a AVAILABLE.

Aquellos sucesores que sigan conectados con el árbol T mediante algún enlace secundario recibirán un paquete de tipo *Child Alive Query* CAQ luego que se inicie el mecanismo de recuperación de direcciones, la respuesta a dicho paquete es mediante paquetes de tipo *Child Alive Response*(CQR), los cuales al ser de gran importancia necesitan ser confirmados, por lo cual manejan un mecanismo de confirmación basado en ACK. Ver algoritmo 21.

Algorithm 21: Respuesta consulta sober estado del sucesor

```
1 begin
2 |   if LLega paquete de tipo CAQ then
3 |   |   Responder con un paquete CAR;
4 |   end
5 end
```

Luego comienza la recolección de direcciones para formar el paquete PAP. En este punto si el nodo ha detectado que su predecesor ha abandonado la red lo informa dentro del paquete PAP mediante un campo agregado para tal fin.

Por otro lado si el nodo ha detectado que existen sucesores de primer orden que se han ido pero su espacio de direcciones se encuentra en estado LOST, agrega dichos sucesores al paquete PAP mediante un campo agregado para dicho fin

En última instancia se realiza el armado del paquete PAP, la dirección primaria a proponer es alguna que pueda ser reutilizada (espacio de direcciones en estado AVAILABLE) o un nuevo espacio de direcciones. Ver algoritmo 22

Algorithm 22: Ofrecimiento de direcciones primarias

```
   ; /* El algoritmo se ejecuta una vez que se recibe un paquete
   PAR, solicitando asignación de una dirección primaria */
 1 begin
 2   if Hay menos de ASIGNABLE_LOWER_LIMIT direcciones
     disponibles para asignar then
 3     Iniciar el algoritmo de recuperación de espacios de direcciones, ver
       algoritmo 20;
 4   end
     ;                    /* Busca direcciones para paquete PAP  */
 5   if Tiene sucesores perdidos then
 6     Agregar al paquete PAP todos los rangos marcados como LOST;
 7   end
 8   if Su predecesor esta perdido then
 9     Agregar al paquete PAP la dirección del predecesor;
10   end
11   if Tiene alguna dirección marcada como AVAILABLE then
12     Utilizar la dirección para el paquete PAP;
13   else
14     if Quedan direcciones para asignar then
15       Asignar dirección al paquete PAP;
16     else
17       No poner dirección primaria en el paquete PAP;
18     end
19   end
20   Enviar paquete PAP;
21 end
```

Una vez enviado la propuesta de direcciones solamente resta esperar la aceptación de las mismas o el *timeout* de la propuesta, en caso de recibir una aceptación de una propuesta pueden darse varias situaciones, en el caso más sencillo simplemente se acepta una dirección nueva y el nodo cede el espacio correspondiente de direcciones. En el caso de recibir un paquete aceptando una dirección que corresponde a un sucesor del nodo, el mismo será reincorporado a la tabla de sucesores, el estado del espacio de direcciones correspondiente cambiará a ASSIGNED y el nodo será tratado como un sucesor más. En el caso de recibir una notificación indicando que se acepta la dirección de su predecesor, el mismo debe ser agregado y considerado como el nuevo predecesor, actualizando la tabla de ruteo como sea correspondiente, en este caso el nodo contará con un nuevo *Default Gateway*. Ver algoritmo 23.

Algorithm 23: Procesamiento de aceptación de direcciones propuestas

```
   ;  /* Se ejecuta al momento de recibir un paquete PAN el cual
   indica que un nodo aceptó una de las direcciones propuestas
   por este nodo */
 1 begin
 2     switch Según cual de las direcciones propuestas acepta el nuevo
       vecino do
 3         case Es una de las direcciones propuestas del espacio de
           direcciones local
 4             Ceder el espacio;
 5             Almacenar la dirección y máscara cedida;
 6             Marcarla como ASSIGNED;
 7         end
 8         case Corresponde a uno de los sucesores perdidos
 9             Almacenar la dirección y máscara cedida;
10             Agregarlo como sucesor;
11             Notifica a aplicaciones interesadas de la incorporación del
               nodo;
12             Marcarla como ASSIGNED;
13         end
14         case Corresponde con la antigua dirección del predecesor del nodo
15             Notifica a aplicaciones interesadas de la incorporación del
               nodo;
16             Instalarlo como nuevo predecesor;
               ;                    /* en ruteo proactivo solamente */
17             Reestablecer el Default Gateway;
18         end
19     end
20 end
```

Dado que se agregan nuevos campos en el paquete PAP, los mismos deben ser procesados en el nodo que esta solicitando la dirección primaria, las direcciones que se agregan al paquete PAP son analizadas por el nodo que envío el paquete PAR, solamente en el caso en que una misma dirección figure tanto en los sucesores perdidos de un PAP como en el predecesor perdido de otro paquete PAP será considerada para ser utilizada, en ningún otro caso dichas direcciones agregadas al protocolo tendrán efecto.

Una vez elegida la dirección primaria a utilizar se debe confirmar la aceptación de la misma, por lo cual es enviado el paquete PAN. Ver algoritmo 24

Una vez que es enviado el paquete PAN hacia el nodo que propuso la dirección primaria, es necesario esperar confirmación del mismo, la cual indica que el espacio de direcciones ya fue correctamente cedido.

Algorithm 24: Selección de direcciones primarias

```
;  /* Nodo reemplazante: El nodo reemplazante debe asumir una
   dirección a reutilizar solo si tiene en su radio de alcance
   algún sucesor del nodo que se perdió previamente. Una vez que
   toma la dirección del nodo perdido debe adoptar a los
   sucesores que estén dentro de su alcance. */
1  begin
2      if Se recibió al menos un paquete PAP con dirección disponible
       entonces then
3          if Recibió en un paquete PAP una dirección de un sucesor perdido
           adjunta, y al menos recibió otro paquete PAP con la misma
           dirección como predecesor perdido then
4              Elegir dicha dirección como primaria;
5              Asumir el rol del nodo perdido;
6              Adoptar a los sucesores;
7          else
8              Elegir entre las direcciones propuestas una con la menor
               máscara;
9          end
10         Enviar un paquete PAN en broadcast notificando la dirección
           elegida;
11     else
12         Es el nodo raíz de la red, adoptar la dirección 0000...000
13     end
14 end
```

Si la dirección que acepto el nodo proviene del reciclado de una dirección, es necesario consultar a su predecesor sobre los espacios de direcciones previamente asignados por el nodo anterior que ocupaba el espacio de direcciones que se esta aceptando. Para lo cual envía paquetes *Child Query* (CQY) a su predecesor de primer orden, el cual responde con paquetes de tipo *Child Query Response*(CQR) indicando todos los espacios de direcciones previamente asignados por el nodo previo. Cabe destacar que el predecesor de primer orden solo informará los espacios de direcciones correspondientes a los nodos que se hayan puesto en contacto con él mediante paquetes de tipo *Child Alive Notification*(CAN), se entiende que aquellos que hayan podido comunicar paquetes CAN, no están en contacto con el árbol T, por lo cual se puede considerar que no son parte de la red.

Una vez recibidos todos los espacios de direcciones previamente asignados, serán agregados en estado UNAVAILABLE por el nodo que se esta conectando a la red. El estado del espacio asignado cambiará a AVAILABLE una vez que se establezca contacto físico con los sucesores mediante paquetes HB.

En este punto concluye el proceso de conexión a la red, y el nodo ya tiene la capacidad de enviar paquetes en la red. Ver algoritmo 25

Algorithm 25: Establecimiento de dirección primaria

```
;     /* El algoritmo se ejecuta al recibir un paquete de tipo
   PANC, el cual confirma que la dirección primaria puede ser
   utilizada */
1  begin
2      if La dirección a utilizar es reutilización de una dirección previa then
3          Enviar un paquete CQY a su predecesor consultando por
           sucesores existentes del nodo que esta siendo reemplazado;
4          Esperar paquetes CQR o que transcurra un tiempo T_CQR;
           switch Evento sucedido do
5              case Transcurre el tiempo T_CQR
6                  Volver a solicitar los sucesores;
7              end
8              case Se reciben paquetes CQR
9                  Marcar como UNAVAILABLE el espacio de direcciones
                   correspondiente a cada nodo que envió un paquete CQR;
10                 Envía paquete ACK por cada paquete CQR recibido;
11             end
12         end
13     end
14     La dirección primaria elegida ya puede ser utilizada;
15 end
```

Una vez que un nodo reemplaza a otro, asumiendo sus sucesores y predecesores, necesita conocer todos los sucesores que el nodo anterior tenia, a fin de no reasignar el mismo espacio de direcciones 2 veces, dado que cada nodo envía

un CAN a sus predecesores de primer y segundo orden, cada nodo que detecta la caída de un sucesor, recibe un CAN por cada sucesor de segundo orden que todavía tiene acceso al árbol T, con lo cual cuando un nodo asume reemplaza físicamente a otro envía un paquete *Child Query* (CQY) a su nuevo sucesor a fin de conocer todos los sucesores existentes, el sucesor responde con paquetes de tipo *Child Query Response* (CQR) indicando todos los espacios de direcciones que corresponden a los sucesores del nodo reemplazado. Ver algoritmo 26.

Algorithm 26: Proceso de paquetes CQY - Solamente cuando tiene una dirección estable

```
1 begin
2 |  Esperar paquetes CQY;
3 |  Notificar al nuevo sucesor de todos los sucesores de segundo orden
  |  existentes mediante paqutes CQR;
4 |  Esperar paquete ACK, de no recibirlo reenviar los datos;
5 end
```

3.4. Distribución de información de localización

La red esta diseñada para almacenar la información referente al ruteo en forma distribuida a través de toda la red, sin dar preferencia a ningún nodo en particular.

A fin de mantener el concepto y a su vez proveer mayor robustez a la red, se propone una mecánica para distribuir en forma equitativa información redundante sobre la red, a fin de:

- Soportar caídas de nodos sin impactar en el servicio de localización más que en su desempeño.
- Registar en forma simultánea en N Randes Vouz servers la información de localización de los nodos.
- Recuperar la información que un nodo contenía una vez que el mismo abandona la red.
- Dado que la información estará distribuida en forma redundante, poder obtener información desde el *Rendezvous* más cercano a la posición del nodo que esta ejecutando la consulta.

3.4.1. Recuperación ante caídas de nodos

Dado que en primera instancia la información particular de localización correspondiente a un nodo determinado, se encuentra almacenada en un solo nodo que funciona como *Rendezvous Server*, la simple caída de dicho *Rendezvous Server* implica la pérdida de la información de localización. El mismo efecto se produce si se decidiera replicar la información en N *Rendezvous Servers*, ya que la caída de uno de ellos dejaría solamente $N-1$ *Rendezvous Servers* conteniendo la información.

Por otro lado debe considerarse que en el caso que un nodo abandone la red, el espacio de direcciones que el mismo administraba será recuperado por su predecesor, por lo cual la próxima vez que un nodo intente localizar dicho espacio de direcciones será su predecesor quien este encargado de administrar dicho espacio de direcciones y responda a la consulta.

Por lo cual a fin de recuperar las direcciones almacenadas por un nodo Randes Vouz Server, se propone agregar al algoritmo de registro del *Rendezvous* un *timer* para enviar el registro al server en forma periódica , conceptualmente un *Heartbeat*, para que el *Rendezvous* mantenga información actualizada del nodo, por 2 razones:

- Mantener actualizada la información en el *Rendezvous Server* de la dirección del nodo.
- De caerse algún *Rendezvous Server*, el próximo mensaje de registro llegará al predecesor del *Rendezvous Server* caído y la dirección universal del nodo seguirá registrada en la red.

3.4.2. Algoritmo de distribución

La distribución de información de localización se basa en determinar unívocamente un servidor *Rendezvous* por cada dirección, la cual será utilizada tanto por el nodo que desea registrarse como por el nodo que esta consultando dicha dirección. Dado que los hipercubos no suelen ser completos, existen espacios de direcciones que no han sido cedidos, dando origen a direcciones que no están asignadas a ningún nodo. Cuando un paquete de registro o de consulta a *Rendezvous* es destinado a una dirección que todavía no ha sido asignada, la misma es tomada por aquel nodo que administra el espacio de direcciones dentro del cual esta contenida la dirección solicitada.

Al momento de distribuir la información de localización en forma equitativa, tomamos como objetivo balancear la cantidad de entradas de *Rendezvous* según la cantidad de espacios de direcciones administrados por un nodo, en el esquema actual un nodo que maneja un espacio de 16 direcciones probablemente administre un mayor número de entradas de *Rendezvous* que un nodo que administre un espacio de 2 direcciones.

En resumen es nuestra intención darle carga a los nodos que son hoja. Un nodo se considera más hoja cuanto más lejos este de la raíz, lo cuál será cuantos más alejados estén los 1 unos del lado izquierdo de su dirección, así un nodo en 1101100100 seguramente está más cerca de ser una hoja que 100100000. Este criterio es aceptable si consideramos que hay una distribución pareja en la profundidad de las ramas, cualquier otro tipo de criterio requiere tener información específica sobre algún árbol en particular. La premisa de ofrecer mayor prioridad a los nodos hoja, se traduce entonces en darle prioridad a aquellos nodos con ciertas características en su dirección.

Por otro lado, además de la necesidad de ofrecer una carga acorde tanto a nodos raíz como a nodos hoja, también es necesario considerar la distribución dentro del árbol de manera tal de ofrecer una carga apropiada a diferentes ramas, a dicho fin se presenta la necesidad de determinar 2 parámetros diferentes con las funciones de distribución:

- La profundidad dentro de la rama del Randes vouz server.
- La rama en la cuál residirá el Randes vouz server.

3.4.3. Profundidad del *Rendezvous Server*

En primera instancia es necesario determinar la profundidad dentro del árbol, cuan hoja va a ser el nodo, manteniendo preferencia por los nodos hoja sobre los nodos raíz. Siendo necesario calcular la posición del último 1 ya que determina que tan alejado de la raíz se encuentra un nodo, luego del último 1 el resto de los bits serán 0. El resto de los bits de la dirección a la izquierda del último 1, deben ser elegidos de la misma manera sin importar en que nodo se realice el cálculo.

Considerando que la dirección esta formada por n dígitos, siendo el dígito 0 el que se encuentra más a la izquierda y el dígito n el que se encuentra más a

la derecha, es necesario elaborar una función para calcular el último 1 que de prioridad a números cercanos a n. Una función que ofrece ese crecimiento es la función logarítmica, luego a fin de calcular la posición del último bit, la función logarítmica debería devolver valores que se encuentren entre 1 y n. Con n igual a la cantidad de dígitos se tiene que:

$$1 \leq LastBit \leq n \tag{3.1}$$

Utilizando una función logarítmica en base e, se tiene que:

$$LastBit = log_e(arg) \leq n \tag{3.2}$$

Entonces el valor del argumento queda limitado a:

$$arg \leq e^n \tag{3.3}$$

Por último, para calcular el valor del último bit, debe utilizarse de alguna manera la dirección universal del nodo y debe introducirse una variable de distribución controlada a fin de dispersar la información en forma pareja dentro del árbol. Una función de *hash* provee dicho comportamiento, ofreciendo un valor determinístico por cada argumento de entrada y de estar bien construida distribuye en forma equitativa dentro del espacio de resultados posibles, entonces el cálculo del último bit sigue la forma de:

$$LastBit = log_e(hash) \tag{3.4}$$

Luego de las consideraciones pertinentes, se propone para calcular el último bit:

$$LastBit = int(log_e(hash1 * m + 1)) \tag{3.5}$$

Donde log_e es el logaritmo natural, $m = (e^{n-1} + 1)$ con n el número de bits de la dirección, y $hash1$ una función de *hash* que mapea cualquier dirección universal U entre los números reales 0 y 1. Donde U es la dirección del nodo que quiere distribuirse.

3.4.4. Rama donde reside el *Rendezvous Server*

Por otro lado a fin de distribuir en forma correcta dentro del árbol de direcciones, es necesario determinar en que rama del mismo residirán los *Rendezvous Servers*, en este punto de cálculo entra en juego la cantidad K de *Rendezvous Servers* sobre los cuales se desea distribuir la información.

Para lo cual utilizaremos el siguiente criterio. Considerando que el árbol de direcciones en su mayor distribución sigue la forma descripta en la figura 3.5.

Al dividir el árbol en ramas iguales, podemos ver en la figura 3.6 una tendencia que en las figuras 3.7 y 3.8 se terminará de delinear al dividir el árbol en 4 y 8 ramas respectivamente, la tendencia se va marcando según los primeros n bits del árbol de direcciones.

Al tomar los primeros 2 bits del árbol de direcciones podemos generar las combinaciones 00, 01, 10 y 11, donde las ramas que representan dichas combinaciones pueden verse en la figura 3.7.

Figura 3.5: Distribución del árbol binario T en un espacio con direcciones de dimensión 4.

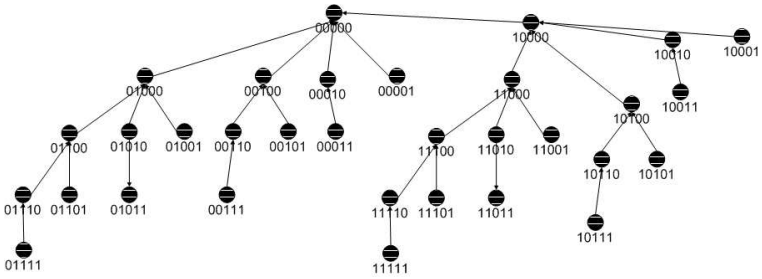


Figura 3.6: Distribución de espacio de direcciones en 2 ramas de igual cantidad de nodos en el árbol T.

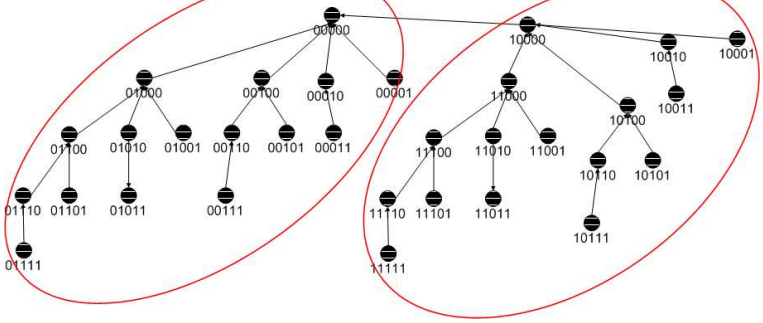
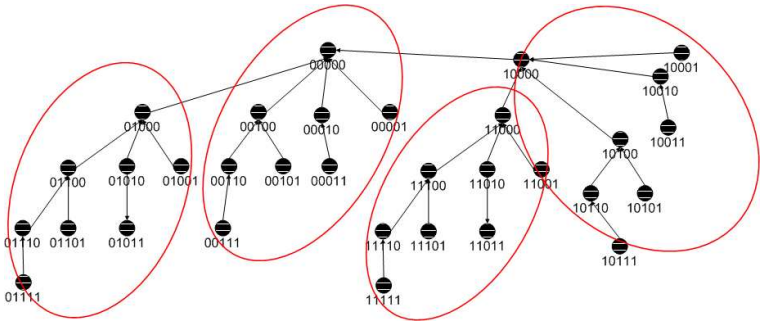
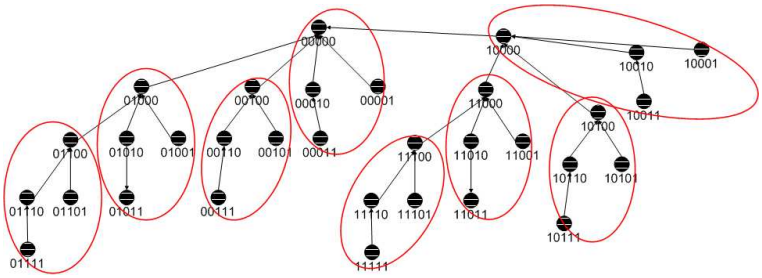


Figura 3.7: Distribución de espacio de direcciones en 4 ramas de igual cantidad de nodos en el árbol T.



Finalmente y para terminar de delinear la tendencia vemos en la figura 3.8 el resultado de tomar los primeros 3 bits del direccionamiento del árbol T.

Figura 3.8: Distribución de espacio de direcciones en 8 ramas de igual cantidad de nodos en el árbol T.



Extrapolando la idea, si queremos distribuir K servers, donde K es una potencia de 2, podemos correctamente colocar 1 *Rendezvous Server* en cada rama del árbol, una vez determinadas las K ramas queda colocar el *Rendezvous Server* en algún lugar dentro de cada rama. A fin de definir como se arman las ramas deben separarse los primeros bits de la dirección, y para calcular cuantos bits de la dirección serán separados se utiliza la ecuación 3.4.4 .

$$bits = log\_2(K) \tag{3.6}$$

3.4.5. Ejemplo de funcionamiento

Se desean distribuir K Servers, donde K es una potencia de 2, se toman los primeros $log\_2(K)$ bits del espacio de direcciones y se forman las K combinaciones necesarias, por ejemplo para distribuir 4 servidores, se toman los primeros $log\_2(4) == 2$ bits de la dirección y se hacen las combinaciones necesarias:

$00 - 01 - 10 - 11$

Luego mediante la ecuación 3.4.3 se determina la 'profundidad' del nodo, que tan hoja será el *Rendezvous Server*. Una vez determinada la rama dentro de la cuál reside el nodo, los primeros k bits de la dirección quedarán fijos, por lo cual para calcular la posición del último nodo no se puede utilizar n como la cantidad de bits que forman la dirección, sino que hay que tener en cuenta cuantos nodos fueron ya prefijados, utilizando entonces $n' = n - log_2(K)$.

Para el ejemplo se utilizará $n = 12$ con distribución en 4 *Rendezvous Servers*, se obtienen 4 posibles ramas dentro de las cuales distribuir $00 - 01 - 10 - 11$, siendo estas 4 ramas obtenidas con los 2 primeros bits de la dirección, luego para el calculo de la profundidad del nodo se descuentan los 2 bits establecidos

por la rama de residencia del *Rendezvous Server*, obteniendo $n' = n - 2$:

$$n' = 10; \tag{3.7}$$

Siendo el valor del argumento que utilizaremos en el logaritmo neperiano:

$$m = e^{(n'-1)} + 1 = 8104,08 \tag{3.8}$$

Hash1 = real entre 0 y 1. Luego $log\_e(hash1 * m + 1)$ oscila entre

$$Log_e(1) = 0 \tag{3.9}$$

$$log_e(8104,08 + 1) = 10, xxxxxx. \tag{3.10}$$

Finalmente la posición oscila entre:

$$LastBit = 1 + (0 < x < 10) \tag{3.11}$$

Asumiendo que $LastBit$ resulto con valor 6, quedan determinadas 4 direcciones
$00xxxxx10000$
$01xxxxx10000$
$10xxxxx10000$
$11xxxxx10000$

Con estas 4 direcciones esta asegurado que los 4 *Rendezvous Servers* se distribuirán en forma pareja en las diferentes ramas de del árbol. Los 5 bits que quedan sin determinar pueden ser cualquier combinación de 0 o 1. Para este caso usaremos los primeros bits resultantes del *hash* original de la dirección Universal que utilizaba ANTop-1 para localizar el *Rendezvous Server*.

3.5. Resumen de paquetes utilizados en ANTop-2

Esta sección tiene como objeto indicar todos los paquetes utilizados por ANTop-2, tanto aquellos heredados y modificados de ANTop-1 como aquellos introducidos específicamente para implementar ANTop-2.

3.5.1. Paquetes utilizados en ANTop-1

Esta sección presenta los distintos paquetes utilizados en la primer implementación de ANTop, comenzando por los componentes que serán utilizados por los diferentes paquetes de control, y finalizando con una reseña de cada paquete de control utilizado por el protocolo, para un mayor detalle de cada uno de los paquetes utilizados por ANTop-1 ver la sección 2.2 de [2]. Aquellos tipos de paquetes que fueron modificados respecto de la implementación original presentan una descripción de los campos agregados.

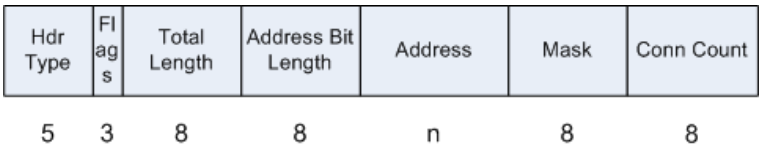
Paquete de control El plano de control del protocolo es administrado a través de paquetes de control con una estructura básica que permite identificar el tipo de paquete y el nodo que originador del paquete, adicionalmente algunos tipos de paquetes agregan algunos campos a la estructura básica para propósitos particulares. La estructura de un paquete de control puede verse en la figura 3.9, el cual está compuesto por (fuente ver sección 2.2.4 de [2]):

Figura 3.9: Paquete de tipo *Control Packet*, debajo de cada componente se indica la cantidad de bits que ocupa.

- Packet type: el tipo de paquete en 5 bits, permitiendo hasta 32 tipos distintos.
- Flags: 3 flags de uso general, que cada tipo de paquete puede asignar para lo que necesite.
- Total length: longitud total del paquete, incluyendo encabezados opcionales.
- MAC address: dirección física del nodo que envía el paquete. Se utiliza para identificar unívocamente a los nodos cuando aún no poseen una dirección de hipercubo.
- Primary Address Length: longitud en bits de la dirección primaria (dimensión del hipercubo).
- Primary Address: dirección primaria, almacenada en n bits, siendo n el mínimo múltiplo de 8 mayor que Primary Address Length.
- Optional Headers: cada paquete puede transportar encabezados opcionales.

Additional Address Los paquetes de control describen las direcciónes utilizadas por los nodos mediante elementos de tipo *Additional Address*, ver figura 3.10.

Figura 3.10: Elemento de tipo *Additional Address*, debajo de cada componente se indica la cantidad de bits que ocupa.



- Header Type: 1 para este encabezado opcional
- Flags

  - flag 0: indica si el encabezado tiene el campo Mask
  - flag 1: indica si el encabezado tiene el campo Connection Count
- Total Length: longitud total del encabezado
- Address Bit Length: longitud en bits de la dirección primaria (dimensión del hipercubo)
- Address: dirección primaria, almacenada en n bits, siendo n en mínimo múltiplo de 8 mayor que Address Bit Length
- Mask: si el flag 0 está seteado, se incluye este campo que indica la máscara asociada a la dirección.
- Connection Count: si el flag 1 está seteado, se incluye este campo que indica la cantidad de conexiones que tiene el nodo con esa dirección.

HypercubMaskAddress

- Address Bit Length: longitud en bits de la dirección (dimensión del hipercubo)
- Address: Dirección, almacenada en n bits, siendo n en mínimo múltiplo de 8 mayor que Address Bit Length
- Mask: Indica la máscara asociada a la dirección.

Primary Address Request -PAR Se utilizan para hacer un pedido de dirección primaria a los vecinos. Se envían siempre en modalidad de *broadcast*. Se espera como respuesta un paquete PAP (Primary Address Proposal), tanto para ofrecer dirección como para informar que no se dispone del espacio necesario. Tipo de paquete de control:

- Hdr Type = 1.

Primary Address Proposal - PAP Se envía en respuesta a un paquete PAR para proponer direcciones de conexión. En este paquete se agregan los espacios para proponer direcciones susceptibles de ser reutilizadas:

- Lost Prefixs count: de tipo entero, indica la cantidad de direcciones que serán incluidas en el vector de direcciones perdidas.
- Lost Prefixs: de tipo vector de AdditionalAddress, contiene todos aquellas direcciones con las que se ha perdido contacto, sean correspondientes a un predecesor o algún sucesor, cada una posee un tipo que identifica el tipo de relación que se posee con la dirección indicada.

Tipo de paquete de control:

- Hdr Type = 2.

Primary Address Notification - PAN Una vez que un nodo eligió la dirección primaria con la que se va a conectar, envía este paquete en modalidad de *broadcast* para indicarle a sus vecinos su nueva dirección, y que su padre le ceda el espacio.
Tipo de paquete de control:

- Hdr Type = 3.

Primary Address Notification Confirmation Lo envía como respuesta a un paquete PAN el nodo que cede el espacio de direcciones, confirmando que efectivamente recibió el paquete PAN.
Tipo de paquete de control:

- Hdr Type = 4.

Disconnection - DISC Se envía para notificar que el nodo se está desconectando, en modalidad de *broadcast*.
Tipo de paquete de control:

- Hdr Type = 5.

Heart Beat - HB Se debe enviar a intervalos regulares para notificar que el nodo sigue conectado, en modalidad de *broadcast*.
Tipo de paquete de control:

- Hdr Type = 6.

Secondary Address Proposal - PAN Se envía para proponer una dirección secundaria.
Tipo de paquete de control:

- Hdr Type = 7.

Secondary Address Notification - SAN Se envía como respuesta a un paquete SAP para indicar si la dirección secundaria fue aceptada o rechazada.
Tipo de paquete de control:

- Hdr Type = 8.

3.5.2. Paquetes agregados en ANTop-2

En esta sección se presentan los paquetes introducidos en ANTop-2 y aquellos agregados para implementar el protcolo de ruteo proactivo de ANTop.

Route Update - RouteUpdatePacket Este paquete tiene como objeto propagar actualizaciones de rutas obtenidas a través de enlaces secundarios entre nodos vecinos, y sus campos son los siguientes, ver figura 3.11:

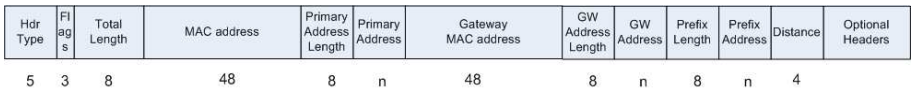
- Gateway Physical Address: de tipo MACAddress, e indica la dirección física del *gateway* a utilizar para el set de prefijos de rutas, dado que de un nodo pueden tenerse múltiples direcciones lógicas.

- Gateway Primary Address: de tipo HypercubeMaskAddress, e indica la dirección lógica del nodo a utilizar como *gateway* para el set de rutas
- Destination Prefix: de tipo HypercubeMaskAddress, e indica el prefijo de redes que deben enrutarse a través del *gateway* propuesto.
- Distance: de tipo entero, distancia lógica existente entre el *gateway* y el nodo que propuso el prefijo originalmente.

Tipo de paquete de control:

- Hdr Type = 9.

Figura 3.11: Paquete de tipo *Route Update*, debajo de cada componente se indica la cantidad de bits que ocupa.



Child Alive Query - CAQPacket Este paquete tiene como objeto solicitar a un descendiente de orden N que indique si todavía esta presente en la red, y sus campos son los siguientes:

- Basicos de paquete de Control.

Tipo de paquete de control:

- Hdr Type = 10.

Child Alive Response - CARPacket Este paquete tiene como objeto responder a la solicitud enviada por un predecesor de orden N que todavía esta presente en la red, y sus campos son los siguientes:

- Basicos de paquete de Control.

Tipo de paquete de control:

- Hdr Type = 11.

Child Alive Notification - CANPacket Este paquete tiene como objeto notificar a un predecesor de orden N que todavía esta presente en la red en caso de sufrir una desconexión con su predecesor de primer orden, y es enviado sin que ningún nodo lo solicite, y sus campos son los siguientes:

- Secuence ID: de tipo entero, indicador de secuencia de paquete utilizado para determinar si el paquete recibió un ACK o no.

Tipo de paquete de control:

- Hdr Type = 12.

Child Query - CQYPacket Este paquete tiene como objeto solicitar a un predecesor de primer orden que notifique todos los espacios de direcciones que este sucesor previamente otorgo a otros sucesores, esto es realizado en caso de substitución de nodos, y sus campos son los siguientes:

- Basicos de paquete de Control.

Tipo de paquete de control:

- Hdr Type = 13.

Child Query Response - CQRPacket Este paquete tiene como objeto informar a un sucesor que esta substituyendo a otro nodo de la cantidad y descripción de espacios de direcciones cedidos por el nodo que anteriormente ocupaba su posición en la red, y sus campos son los siguientes:

- Addresses: de tipo HypercubeMaskAddresses vector, contiene todos los prefijos de red ya cedidos por un nodo.

Tipo de paquete de control:

- Hdr Type = 14.

Gateway Request - GWRPacket Este paquete tiene como objeto solicitar que comience el algoritmo de descubrimiento de *gateway* alternos, y sus campos son los siguientes:

- Requesting Node: de tipo HypercubeMaskAddresses, e indica el prefijo representado por el nodo origina la solicitud y descubrimiento de *gateway* alternos.

Tipo de paquete de control:

- Hdr Type = 15.

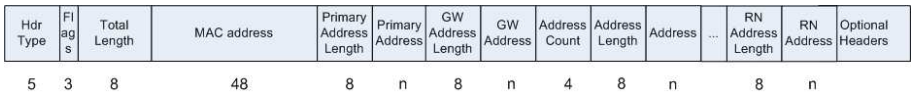
Gateway Update - GWUPacket Este paquete tiene como objeto propagar la elección de un *gateway* alterno para el segmento, y sus campos son los siguientes, ver figura 3.12:

- Gateway Address: de tipo HypercubeMaskAddresses, e indica la dirección del *gateway* seleccionado para oficiar de *gateway* alterno para el segmento.
- Address count: de tipo entero, indica la cantidad de rutas que son anunciadas para ser retraídas.
- Addresses: de tipo HypercubeMaskAddresses vector, e indica todas las rutas que fueron propagadas por el nodo original y que deben ser retraídas.
- Requesting Node: de tipo HypercubeMaskAddresses, e indica el prefijo representado por el nodo origina la solicitud y descubrimiento de *gateway*.

Tipo de paquete de control:

- Hdr Type = 16.

Figura 3.12: Paquete de tipo *Gateway Update*, debajo de cada componente se indica la cantidad de bits que ocupa.



Gateway Proposal - GWPPacket Este paquete tiene como objeto proponer un nodo como posible *gateway* alterno, y sus campos son los siguientes:

- Gateway Address: de tipo HypercubeMaskAddresses, e indica la dirección del *gateway* propuesto para oficiar de *gateway* alterno para el segmento.

Tipo de paquete de control:

- Hdr Type = 17.

Acknowledgment - ACKPacket Este paquete tiene como objeto confirmar la recepción numerosos paquetes del protocolo, y sus campos son los siguientes:

- ACK Type: de tipo entero, e indica a que tipo de paquete del protocolo esta sirviendo de confirmación, actualemente se utilizan paquetes de tipo *Acknowledgment* para confirmar la recepción de paquetes de tipo *Child Alive Notification*. Este campo toma un valor igual a 0, futuras características del protocolo podrían requerir confirmación para otro tipo de paquetes los cuales serán confirmados utilizando paquetes de tipo *Acknowledgment* pero utilizando valores diferentes en el campo *ACK Type*.
- Secuence ID: de tipo entero, y es utilizado para poder correlacionar una solicitud de algún tipo con un paquete de tipo ACK en particular.

Tipo de paquete de control:

- Hdr Type = 18.

Gateway Reset - GWRSPacket Este paquete tiene como objeto solicitar que se deshagan las acciones realizadas por la elección de un *gateway* alterno en el segmento, y sus campos son los siguientes:

- Requesting Node: de tipo HypercubeMaskAddresses, e indica el prefijo representado por el nodo origina la solicitud y descubrimiento de *gateway*.

Tipo de paquete de control:

- Hdr Type = 19.

Route Retract - RTRPacket Este paquete tiene como objeto retraer un set de rutas previamente propagadas, y sus campos son los siguientes, ver la figura 3.13:

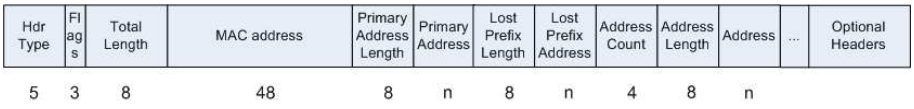
- Lost Prefix: de tipo HypercubeMaskAddress, prefijo del nodo que abandona la red y cuyas rutas deben ser retraídas

- Address count: de tipo entero, indica la cantidad de direcciones que son anunciadas apra ser retraídas.
- Addresses: de tipo HypercubeMaskAddress vector, grupo de rutas que fueron propagadas originalmente por el nodo que abandona la red.

Tipo de paquete de control:

- Hdr Type = 20.

Figura 3.13: Paquete de tipo *Route Retract*, debajo de cada componente se indica la cantidad de bits que ocupa.



Capítulo 4

Simulaciones

En el presente capítulo se desarrollan las simulaciones correspondientes a probar el funcionamiento de los mecanismos descriptos en las secciones anteriores referentes al desarrollo del protocolo proactivo, desarrollo de mecanismos de recuperación ante fallas y por ultimo distribución de información de localización. Cada tipo de prueba será ejecutado tanto en topologías cuya estructura y secuencia de conexión es completamente aleatoria y en topologías cuya estructura y secuencia de conexión simula el comportamiento de Internet.

En el caso del análisis del diseño del protocolo proactivo se pretende demostrar su funcionalidad sobre redes con una dinámica pseudo-estacionaria, la forma en que administra las tablas de ruteo, la manera en que cursa paquetes sobre la red y el desempeño que presenta en su funcionamiento.

En el caso de los mecanismos de recuperación ante fallas se pretende analizar en que condiciones logran resolver la problemática correctamente, sus límites y desempeño final de la red una vez que los mecanismos correspondientes reestructuraron la red posterior al escenario de fallas.

Al analizar los mecanismos de distribución de información de localización se pretende demostrar que el algoritmo propuesto distribuye correcta y equitativamente la información en la red, de manera tal que provee una carga controlada en los servidores de *Rendezvous* y provee resistencia ante posibles y eventuales fallas en la misma.

A fin de verificar el correcto funcionamiento de los diferentes algoritmos propuestos, se presenta una serie de simulaciones diseñadas específicamente para cada situación.

4.1. Mecánica de las simulaciones

Las simulaciones se separan en 3 secciones, orientadas a:

- Analizar el comportamiento del protocolo proactivo, a fin de verificar su

correcto funcionamiento en las topologías pseudo-estacionarias y su comparación frente al desempeño del protocolo de ruteo reactivo.

- Analizar la efectividad de los protocolos de tratamiento de fallas, frente a la caída de un determinado porcentaje equipos en la red, como queda conformada la nueva topología y como es distribuida la información de ruteo en la misma.

- Analizar la distribución realizada por el mecanismo de registro de información de localización en múltiples nodos *Rendezvous*.

En cada caso se utiliza una red de nodos cuya composición y conexiones se generan en forma aleatoria, respetando los criterios particulares a cada simulación, diferenciando la secuencia de conexión entre nodos y su distribución para redes que siguen una topología aleatoria o para las redes cuya topología responde a una ley de potencias. Por cada caso específico se generaron 20 redes que comparten todas las características excepto la secuencia específica de conexión o desconexión de nodos y de envío de paquetes en la red.

Las simulaciones fueron ejecutadas utilizando extensiones desarrolladas al simulador QUENAS diseñado por Alejandro Marcu (ver Capítulo 3 de [2]). El mismo utiliza archivos de entrada escritos en un formato propietario, generando como resultado de la simulación archivos en formato XML. La estructura básica y contenido de los archivos de entrada y salida se encuentra descriptos en [2]. Una vez generados los archivos de salida en formato XML, los mismos son ingresados en un procesador XSLT para lo cual fue necesario codificar archivos de instrucciones específicos para el procesador XSLT. El procesador XSLT genera archivos de salida en formato CSV, los cuales son procesados directamente por la herramienta SCILAB, la cual es un software de procesamiento matemático utilizado para ejecutar los cálculos y generar las gráficas presentadas en este trabajo.

La ejecución de las simulaciones incluye los siguientes procesos:

- Generación de archivos de entrada: para la generación de archivos de entrada se desarrolló una versión modificada del generador utilizado por el Ingeniero Alejandro Marcu, en la cual el requerimiento principal fue la posibilidad de poder eliminar nodos en forma aleatorio y en un determinado momento en la red, generando múltiples simulaciones idénticas pero variando la secuencia y la identidad de los nodos que abandonan la red.

- Ejecución de las simulaciones: en este punto fue necesario desarrollar todos los algoritmos y mecanismos descriptos en el capítulo anterior, a fin de brindar soporte a múltiples protocolos de ruteo, a los mecanismos de tratamiento de fallas y al mecanismo de distribución de información de localización.

- Pre procesamiento de simulaciones: Implica el procesamiento de los archivos XML resultantes del simulador QUENAS en un formato entendible por aplicaciones como SCILAB o MATLAB.

- Procesamiento de simulaciones: Fue necesario desarrollar los scripts necesarios para procesar los archivos generados con la aplicación de procesamiento de datos SCILAB.

Anteriormente mencionamos que existen diferentes tipos de topologías que fueron consideradas a la hora de generar los archivos de entrada para la simulación, estas topologías presentan propiedades y una composición completamente diferente:

Redes aleatorias Son aquellas donde la disposición de los nodos en el plano sigue un patrón aleatorio, en la cual aquellos nodos que estén dentro del radio de cobertura de otro nodo pueden establecer un enlace físico entre ellos, luego según las reglas de asignación de direcciones podrán convertirse en enlaces lógicos y ser utilizados para enviar o recibir paquetes. En esta topología la secuencia en la que se conectan los nodos y se convierten en parte de la red es completamente aleatoria, excepto que la red debe generarse a partir de un nodo raíz. El primer nodo de la red se considera como la raíz de la red, luego se crea otro nodo cuya posición es aleatoria, si dicho nodo no está dentro del radio de cobertura de algún nodo que ya es parte de la red es descartado. El procedimiento se repite por cada nodo que debe ser creado hasta formar completamente la red.

Redes con topología sin escala Utilizamos este tipo de redes para modelar el comportamiento del ruteo en Internet, en particular consideramos la topología de sistemas autónomos (A.S.), siendo un sistema autónomo una red (ruteadores, LANs, etc.) que se encuentran bajo un mismo dominio administrativo. Este tipo de red esta formada por sistemas autónomos conectados entre sí, donde cada sistema autónomo es considerado un nodo de la red. En cada nodo la cantidad de conexiones hacia otros sistemas autónomos/nodos es considerado el grado de conexión del nodo.
Esta topología es llamada "sin escala", ya que la distribución del grado de conexión de los nodos sigue una ley de potencias. Luego los nodos se consideran pertenecientes a diferentes capas, donde un nodo i tiene número de capa c, si dicho nodo pertenece al $c - core$ pero no al $(c+1) - core$. Tomamos como notación c_i el número de capa del nodo i. Para definir si un nodo pertenece a un $k - core$ debemos en primera instancia definir que es un $k - core$, un subgrafo $H = (C, E|C)$ inducido por el conjunto $C \subseteq V$ es un $k - core$ o un núcleo de orden k sii $\forall v \in C : grado_H(v) \geq k$, y H es el máximo subgrafo con esta propiedad, para un mayor detalle ver [46].
En este tipo de topología existirán nodos con gran cantidad de vecinos, y recordando que el mecanismo de asignación de direcciones de ANTop-1 constantemente intenta establecer direcciones secundarias entre vecinos, se entiende que aquellos nodos con un gran número de vecinos no deberían agotar su espacio de direcciones con adyacencias a través de direcciones secundarias, sino que deberían reservarlo a fín de poder delegarlo frente a peticiones de dirección primaria de sus vecinos. Razón por la cual la red comienza conectando aquellos nodos de mayor grado de conexión(mayor cantidad de vecinos), y capa por capa comienza a efectuar la conectividad de los nodos de menor grado de conexión. Una vez finalizada la asignación de direcciones primarias recién permite la asignación de direcciones secundarias para proveer un mayor mallado de conectivi-

dad.

Tanto para las simulaciones en topologías aleatorias y en aquellas con topologías sin escala se envían 500 paquetes entre orígenes y destinos aleatorios, cada paquete es enviado 3 veces; de origen a destino, de destino a origen y finalmente de origen a destino nuevamente.

4.2. Extensiones al simulador QUENAS

El trabajo de tesis actual introduce una gran cantidad de cambios en el comportamiento y diseño del simulador QUENAS, las cuales fueron descriptas en el capítulo 3.

En adición a las modificaciones mencionadas anteriormente, se introdujeron una serie de mejoras que afectan al comportamiento global del simulador, y que fueron utilizadas a fin de realizar las simulaciones presentes:

- *Rendezvous* global: Se provee de un *Rendezvous* Global, externo a la red, que es parte del simulador, el cual provee servicios de registro y localización a cualquier nodo que lo solicite, siendo inmune a la caída o el abandono de nodos en la red. De esta manera de producirse una partición tal que para alguna dirección no tenga ningún servidor de *Rendezvous* dentro de la partición, el servicio de localización sigue prestando servicio. Este *feature* es utilizado solo para fines de estudio.

- Solo Medición de topología: Característica agregado para realizar pruebas utilizando archivos de simulación existente pero ignorando las peticiones de envío de datos, procesa la topología, genera los reportes correspondientes y abandona la ejecución. Este *feature* es agregado para poder probar los mecanismos de registro en el servicio de localización en forma independiente del envío de datos en la red.

4.3. Comportamiento protocolo proactivo

El diseño del mecanismo de ruteo en el esquema proactivo toma como premisa que los nodos no poseen un alto grado de movilidad, razón por la cual podemos asumir que si colocamos información en la tabla de ruteo, la misma tendrá validez por un tiempo no despreciable, en consecuencia podría contener mayor información dentro de la tabla de ruteo en comparación a la información mantenida por el protocolo reactivo.

Teniendo en consideración el punto anterior es de vital importancia analizar la evolución del poblado de la tabla de ruteo, tanto en tamaño como en composición, dado que el protocolo esta pensado para ser utilizado en dispositivos móviles, y se debe considerar cuestiones tales como ahorro de energía consumida ,almacenamiento utilizado y ciclos de procesamiento.

Según el esquema de funcionamiento del protocolo de ruteo utilizado, los paquetes realizarán algún recorrido para ir de un nodo a otro, aunque dicho recorrido no necesariamente sea el mejor posible. Considerando la red como un grafo y utilizando los enlaces lógicos entre los nodos como aristas del grafo, pueden utilizarse diferentes tipos de algoritmos para encontrar el camino mínimo entre cualquier par de nodos, en esta tesis se utiliza un algoritmo similar al propuesto por *Dijkstra* para obtener el camino mínimo entre 2 nodos. A fin de evaluar si el recorrido elegido por el algoritmo de ruteo fue o no bueno se compara contra el camino mínimo obtenido mediante un algoritmo determinista.

En las siguientes secciones se presentan los resultados obtenidos luego de simular el comportamiento del protocolo en redes aleatorias y redes sin escala.

4.3.1. Redes aleatorias

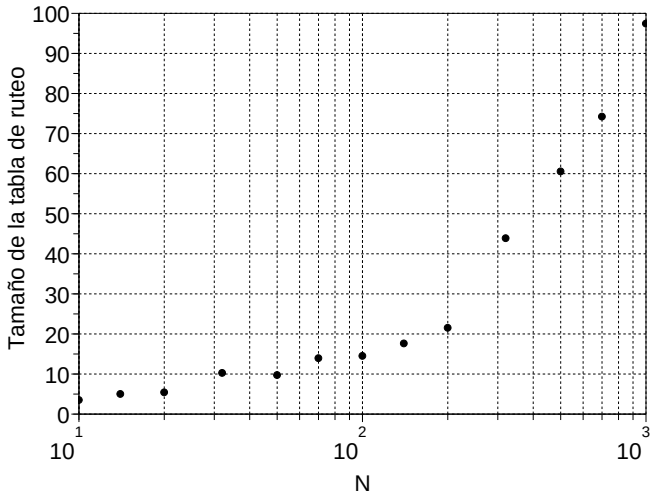
Comenzamos por analizar el desempeño y comportamiento del protocolo proactivo en redes cuya conexión y distribución sigue un patrón aleatorio, y con diferentes tamaños de red.

El primer parámetro que mediremos es la cantidad de entradas presentes en las tablas de ruteo de los nodos de la red, cabe recordar que la composición de la misma consta de una ruta hacia el predecesor, una ruta por cada sucesor del nodo, una cantidad variable de rutas secundarias y una cantidad variable de rutas que han sido propagadas por otros nodos con enlaces secundarios.

En figura 4.1 se presenta el tamaño promedio de las tablas de ruteo para diferentes dimensiones de redes, con:

$N \in \{10, 14, 20, 32, 50, 70, 100, 140, 200, 320, 500, 700, 1000\}$.

Figura 4.1: Tamaño promedio de las tablas de ruteo para diferentes tamaños de redes.



Analizando el tamaño promedio de las tablas de ruteo en el caso reactivo, capítulo 4.1.11, figura 4.27 del trabajo previo a esta tesis (ver [2], puede verse que el promedio del tamaño de las tablas de ruteo en el caso proactivo es aproximadamente la mitad que en el caso reactivo.

En comparación contra el ruteo reactivo era esperado encontrar un promedio significativamente más alto en la cantidad de entradas de la tabla de ruteo, esto es cierto para el caso en que no se envíen paquetes en la red, ya que el tamaño de las tablas de ruteo en el caso proactivo se encuentra acotado a la información que sea distribuida entre los nodos, y el mismo es independiente a la cantidad de paquetes enviados en la red permaneciendo estable mientras que la red no sufra cambios en su topología. En cambio en el protocolo reactivo se parte de un número mínimo de entradas en la tabla de ruteo y debido a que el envío de cada paquete agrega entradas en todos los nodos que recorre entre el origen y destino, vemos que si bien inicialmente la cantidad de entradas en sus tablas de ruteo es menor, el tamaño de las tablas de ruteo en el caso reactivo sigue creciendo a medida que se envían paquetes mientras que en el caso proactivo este tamaño es acotado y fijado a cambios topológicos.

Como muestra representativa del comportamiento del algoritmo se toman redes de 500, 1000 y 5000 nodos.

La presentación de resultados se organiza en 2 partes:

- Tamaño y distribución de componentes en la tabla de ruteo.
- Distancia recorrida por los paquetes ruteados

En primera instancia comenzaremos analizando la distribución de componentes en la tabla de ruteo, considerando todos los tipos de rutas existentes en la tabla de ruteo la distribución de número de entradas en la tabla de ruteo 4.2.

La figura 4.2 muestra la totalidad de las rutas almacenadas por las tablas de ruteo, aunque es necesario recordar que la tabla de ruteo esta compuesta por distintos tipos de rutas (secundarias, propagadas, etc), las cuales son instaladas en diferentes etapas de funcionamiento del protocolo proactivo. Por lo cual es necesario verificar como esta conformada la tabla, siendo las rutas propagadas y secundarias las de mayor interés ya que son resultado de los mecanismos de actualización y mantenimiento de rutas del protocolo proactivo.

En la figura 4.3 se presenta la distribución de rutas secundarias, rutas regulares y propagadas, siendo las rutas regulares las que representan la estructura básica del árbol T, las rutas secundarias son consecuencia del mecanismo de asignación de direcciones secundarias y finalmente las rutas propagadas muestran el resultado de propagar las rutas secundarias.

Claramente se puede apreciar que la mayoría de las rutas que componen la tabla de ruteo son aquellas propagadas por el algoritmo de ruteo proactivo, si consideramos la posibilidad de utilizar un ruteo mínimo, solo compuesto por la estructura del árbol T estaremos frente a la composición mínima posible para las tablas de ruteo, y utilizando como punto de comparación las figuras anteriores

Figura 4.2: Distribución de cantidad de entradas en la tabla de ruteo para redes de 500, 1000 y 5000 nodos.

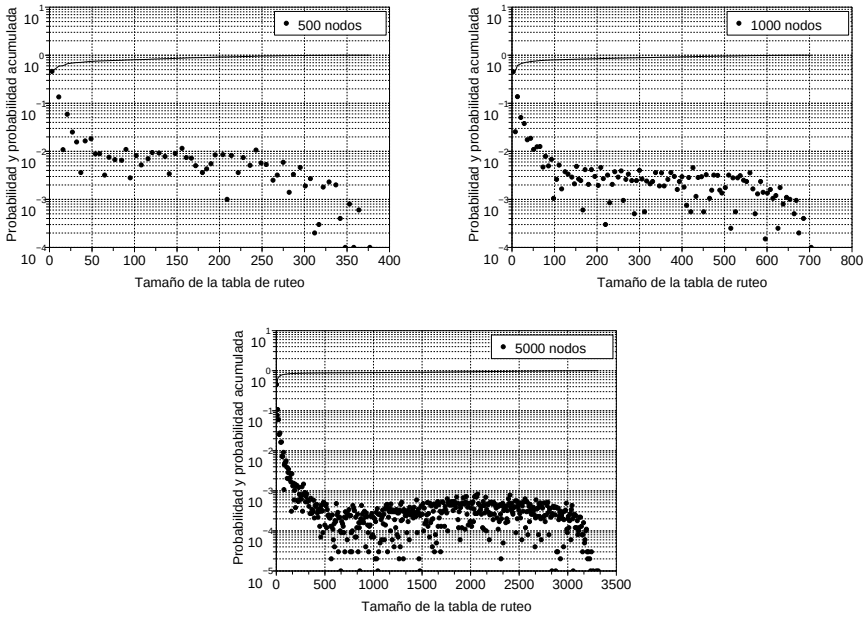
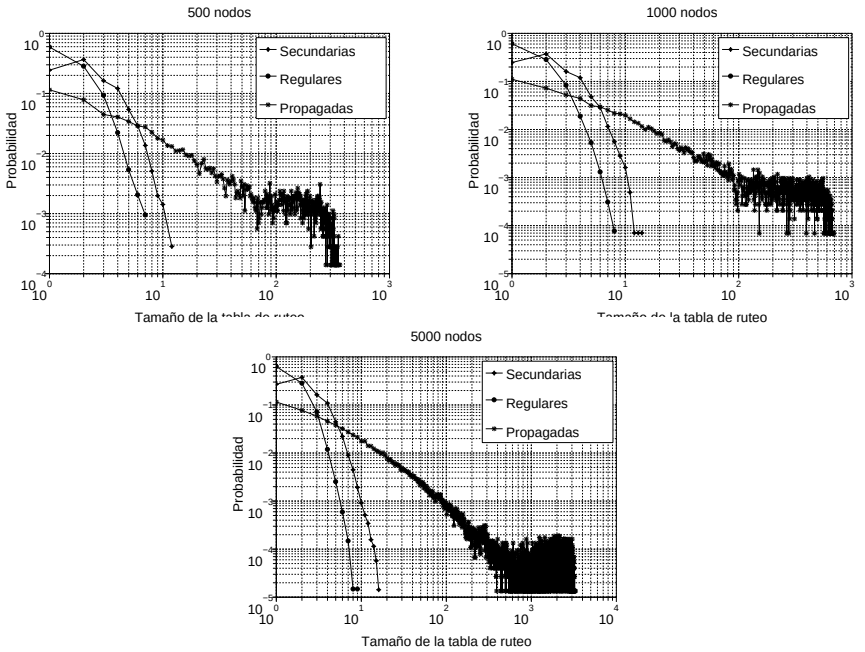


Figura 4.3: Composición de la tabla de ruteo por rutas secundarias, regulares y propagadas para redes de 500, 1000 y 5000 nodos.



puede apreciarse que el espacio que ocuparían las tablas de ruteo es insignificante, en consecuencia menor utilización de recursos por el nodo y de energía en última instancia.

El efecto negativo de utilizar un esquema de ruteo mínimo es una mayor distancia recorrida por los paquetes ya que recorrerán siempre el árbol T, y la distancia que los mismos recorrerán estará siempre acotada, siendo posible calcularla previamente a cada envío, siendo el resultado de sumar la distancia desde el origen hasta un nodo predecesor en común con el destino, y luego sumado la distancia de ese nodo en común con el nodo de destino.

Por otro lado es necesario analizar el comportamiento y desempeño de la red a la hora de cursar tráfico, más allá de utilizar poco o mucho espacio de almacenamiento o producir un consumo inferior o superior de energía, la red debe ser capaz de cursar tráfico en forma eficiente.

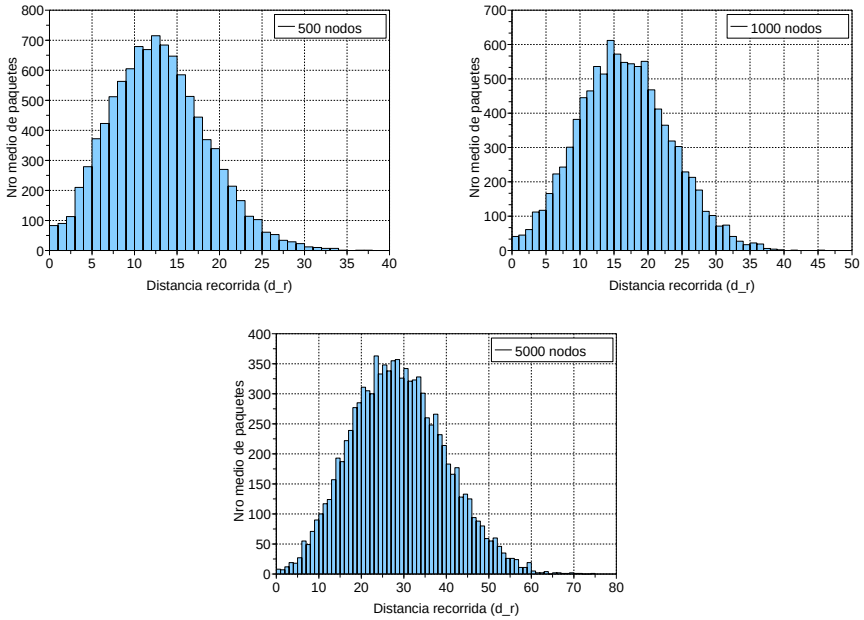
Uno de los objetivos del desarrollo del protocolo de ruteo proactivo es lograr probar que a diferentes esquemas de movilidad de nodos en una red es posible encontrar un esquema de ruteo que se ajuste al tipo de movilidad que presentan los nodos y lograr así obtener el mayor desempeño posible para cada tipo de red.

Se pretende que en un esquema de baja movilidad sea posible mejorar notablemente los resultados del desempeño de la red en comparación a los obtenidos por el protocolo reactivo en un esquema de alta movilidad, en este caso se puede apreciar el la figura 4.4 el histograma de distancias recorridas por los paquetes en redes de 500, 1000 y 5000 nodos, de la cual se observa que los mismos tienen una media de distancia recorrida de 12, 15 y 30 saltos aproximadamente, con un desvío estándar de unos 10 saltos aproximadamente en los 3 casos, lo cual es notablemente superior al desempeño logrado por el protocolo reactivo (ver sección 4.1.9 Envío de datos de [2]), esto es posible solamente considerando la premisa de baja movilidad de los nodos, este resultado es consistente con los resultados esperados al momento de diseñar el protocolo proactivo.

Una vez analizados los resultados generales de desempeño es necesario verificar como se comporta el protocolo frente a un ruteo de tipo *link-state* o *distance-vector*, como ser OSPF(Open Shortest Path First) definido en [44] o ISIS(Intermediate System to Intermediate System) definido en [43], y para en el primer caso y RIP(Routing Information Protocol) definido en [45] en el segundo, dichos protocolos son de tipo *Interior Gateway Protocol*(IGP), son utilizados dentro de un mismo sistema autónomo, razón por la cual utilizan la premisa de contar con una red relativamente pequeña, con cambios topológicos conocidos y esquemas de movilidad nula, un nodo no puede desaparecer de un lugar y reaparecer en otro como caso normal de operación, este tipo de movimientos son previstos y realizados en forma manual por operadores de red.

En los protocolos de ruteo de tipo IGP como OSPF e ISIS, se mantiene información detallada de la composición de la red, donde cada nodo contiene una base de datos con todo el detalle de la red necesario para realizar un ruteo óptimo. Ambos protocolos toman toda la información posible de todos los nodos y enlaces de la red, con lo cual elaboran un mapa detallado de la topología

Figura 4.4: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, 1000 y 5000 nodos luego de enviar 500 paquetes.



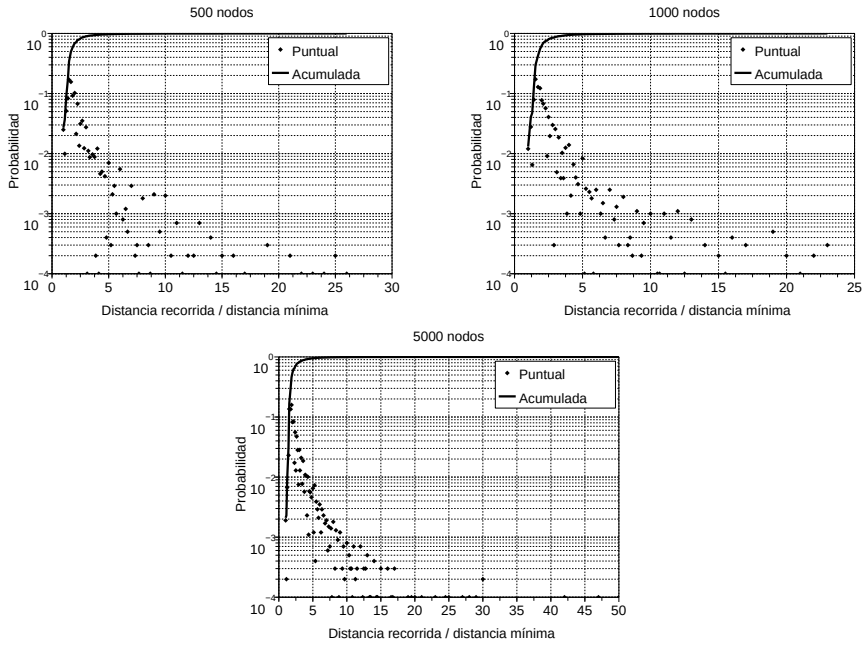
de la red completa, luego utilizando algoritmos de camino mínimo (Dijkstra) calculan el mejor camino para cada destino posible, y en base a dichos cálculos conforman las tablas de ruteo.

Si bien el protocolo ANTop-1 esta pensado para trabajar como *External Gateway Protocol*(EGP), brindando ruteo entre diferentes sistemas autónomos, es interesante ver la relación de su desempeño frente a un protocolo de ruteo óptimo.

En la figura 4.5 se presenta la distribución de probabilidad para la relación existente entre la distancia recorrida con un paquete versus la distancia mínima ofrecida por un protocolo de ruteo óptimo, lo cual fue calculada para cada paquete enviado por las 20 simulaciones de cada tipo de red, también se presenta la gráfica de probabilidad acumulada para cada caso:

Analizando la figura 4.5 puede verse que aproximadamente el 95 % de los paquetes enviados recorren una distancia entre 2 y 3 veces mayor a la que recorrerían utilizando un protocolo de ruteo óptimo, por otro lado vemos que hay algunos paquetes que recorren distancias mucho mayores pero con una probabilidad mucho más baja, si se quisiera lograr una relación 1 a 1 con un protocolo de ruteo óptimo, ANTop-2 debería mantener información detallada de la composición de la red, lo cual claramente no es el objetivo, debe recordarse que el esquema de distribución de rutas planteado por el protocolo proactivo hace uso del principio de localidad, donde la información se considera útil y es distribuida solo en las cercanías del nodo que comienza la distribución o mientras sea útil, mediante los criterios desarrollados al proponer la solución llega un punto en

Figura 4.5: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos, presentado para redes de 500, 1000 y 5000 nodos.

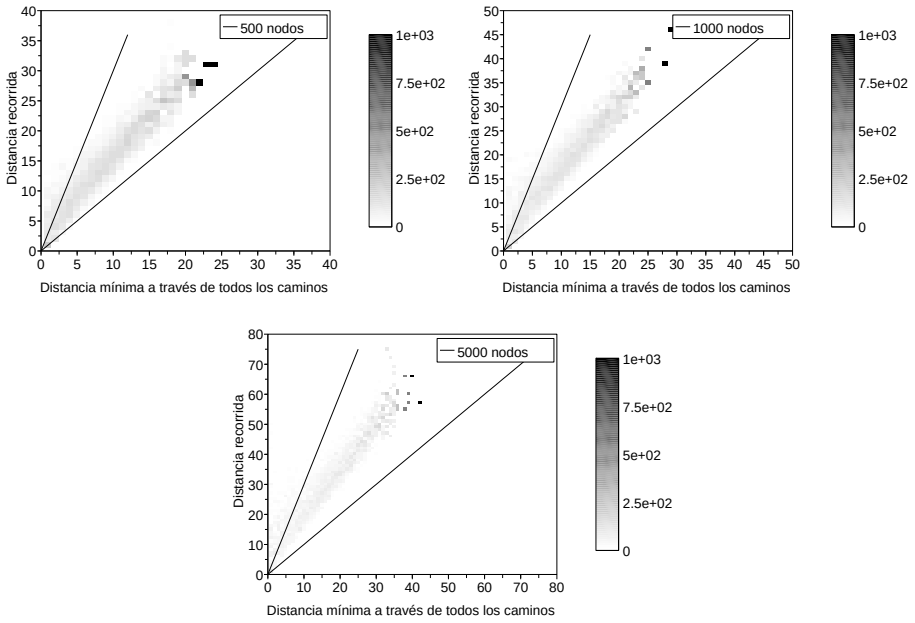


que las rutas dejan de ser propagadas y se decide proseguir con el ruteo a través del árbol T.

Una vez analizada la relación entre la distancia recorrida y la distancia mínima para todos los paquetes es interesante analizar para cada distancia mínima cual es la probabilidad de ocurrencia de las diferentes distancias, esto nos informa cuales son las distancias que presentan mayor o menor desempeño. Para lo cual se elabora una gráfica que presenta para cada distancia mínima, la probabilidad de ocurrencia para cada distancia recorrida, representado la probabilidad 0 y 1, con el color blanco y negro respectivamente, con todas sus variaciones internas como escala de grises, además se agregan 2 líneas continuas una con pendiente 1 y otra con pendiente 3, donde la recta de pendiente 1 representa el resultado que se obtendría utilizando un protocolo de ruteo óptimo y la recta de pendiente 3 representa el *stretch* teórico máximo para un protocolo de ruteo, estudios y resultados que llevan a la utilización de un stretch de 3 como techo pueden encontrarse en [47] y [48].

En las gráficas de dispersión 4.6 puede observarse que si bien en la gráfica de probabilidad se observaban algunos casos en que la relación entre distancia recorrida y distancia mínima era bastante elevado, en la gráfica de dispersión puede verse que en general los paquetes recorren distancias similares a las propuestas por un algoritmo de ruteo óptimo y que no se salen de la relación 1:3 planteada por la línea de pendiente 3.

Figura 4.6: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.



Experimentalmente se observó que ciertos paquetes recorren una distancia notablemente superior a la que recorrerían utilizando un algoritmo de camino mínimo. Dichas relaciones entre distancia recorrida y distancia mínima se encontraron en nodos que si bien están cerca uno de otro, no son adyacentes en el hipercubo. A fin de comprender las razones del resultado es necesario recordar que la estructura de hipercubo lógico desplegada sobre la red ofrece el mismo grado de conectividad que un grafo normal, ya que no todas las adyacencias físicas se convierten en adyacencias lógicas. El grado de conectividad logrado por el hipercubo es controlado por la distancia en bits entre las direcciones de los vecinos físicos, siendo que en la implementación actual se consideran adyacentes cuando las direcciones de dos vecinos físicos distan 1 bit. Una posible forma de mejorar la relación entre el camino recorrido y el camino mínimo es haciendo más flexible el criterio para que dos vecinos físicos se vuelvan adyacentes, por ejemplo llevando la diferencia a 2 o 3 bits. El precio que debe pagar dicho cambio es un gran incremento en el tamaño de las tablas de ruteo, debido especialmente a las rutas que son propagadas por el algoritmo de ruteo.

La opción de disminuir la restricción sobre la distancia en el hipercubo para formar adyacencia debería plantearse solamente si existiese la necesidad de reducir aún más la relación entre distancia recorrida versus distancia óptima.

4.3.2. Redes con topologías sin escala

El protocolo ANTop-1 fue pensado para poder funcionar en un ambiente como Internet, cumpliendo funciones de EGP, proporcionando ruteo entre diferentes sistemas autónomos, razón por la cual es preciso analizar su comportamiento en dicho ambiente. En el capítulo anterior se analizó el comportamiento del protocolo al utilizar redes aleatorios, en este capítulo nos centraremos en analizar los mismos resultados pero obtenidos sobre redes que presentan una topología similar a la de Internet.

La organización de la presentación de resultados es igual a la utilizada en el capítulo anterior por lo cual se seguirá el mismo formato haciendo las aclaraciones necesarias en cada caso, aunque determinadas secciones que contengan justificaciones de la organización no serán desarrolladas nuevamente, simplemente se hará mención a la sección correspondiente donde encontrar la información necesaria.

El generador de simulaciones utilizado genera las redes de entrada hace uso de un mapa de nodos obtenido a partir los datos publicados por CAIDA (Cooperative Association for Internet Data Analysis) del mes de abril de 2005. En el cual se generan redes de 8500 nodos aproximadamente, nuevamente se generan 20 series de redes en las cuales se envían paquetes entre orígenes y destinos aleatorios.

Para un mayor detalle del funcionamiento del generador de simulaciones para redes sin escala referirse (ver [2] Desarrollo y simulación de un protocolo para redes ad-hoc, Sección 4.2).

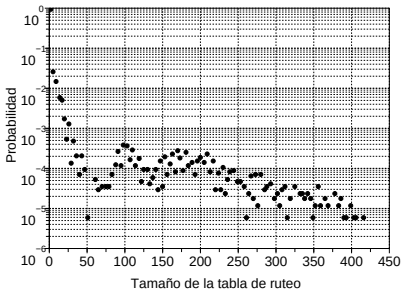
El análisis de los datos se plantea de la misma manera que fue realizado para el caso de redes aleatorias, con una presentación dividida en 2 etapas:

- Tamaño y distribución de componentes en la tabla de ruteo.
- Distancia recorrida por los paquetes ruteados.

Nuevamente comenzamos analizando el poblado de la tabla de ruteo así como su composición, en primera instancia se presenta la distribución de probabilidad para las diferentes cantidades de entradas en la tabla de ruteo.

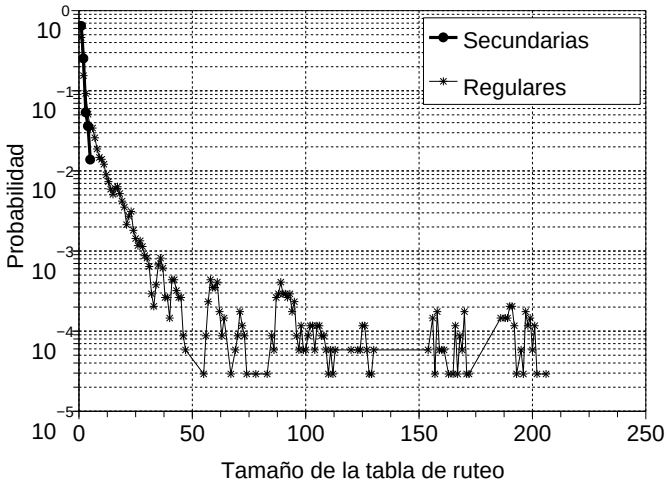
En la figura 4.7 vemos una distribución de probabilidades con aproximadamente 450 entradas como máximo, este número es sensiblemente inferior al caso del ruteo proactivo en redes aleatorias. La topología de las redes sin escala se basa principalmente en la alta conectividad que poseen los nodos, y hay que recordar que con el objetivo de no agotar el espacio de direcciones debido a la asignación de rutas secundarias, se inhibe la delegación de las mismas hasta que todos los nodos hayan obtenido una dirección primaria, esto se puede implementar en la realidad porque existe un algoritmo distribuido para calcular el número de capa de un nodo. En el caso de las redes aleatorias vimos que la mayor cantidad de entradas en la tabla de ruteo provenía de las rutas propagadas por el algoritmo de distribución de rutas, en este caso dado que se prioriza la asignación de direcciones antes de la múltiple conectividad por direcciones secundarias esperamos ver ese comportamiento invertido o mitigado en gran parte.

Figura 4.7: Distribución de probabilidad para la cantidad de entradas de la tabla de ruteo



A fin de corroborar que se asignan mayor cantidad de rutas a sucesores en este caso, la figura 4.8 presenta una gráfica comparativa con los diferentes tipos de rutas que conforman la tabla de ruteo.

Figura 4.8: Distribución de probabilidad para rutas secundarias y rutas regulares en redes sin escala.

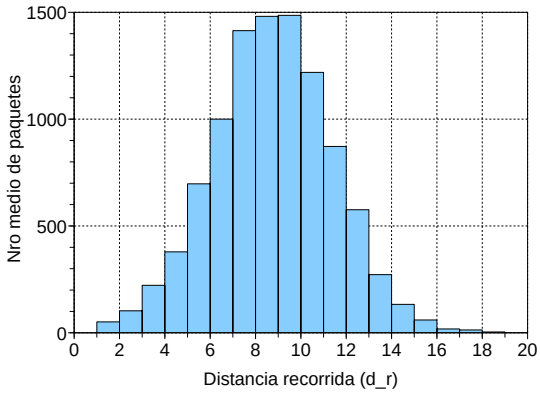


En este caso, la cantidad de rutas propagadas es menor, producto de la existencia de menos rutas secundarias.

Al igual que en el caso de redes aleatorias, vemos que las tablas de ruteo tienen un tamaño acotado a las reglas de distribución y propagación de rutas. Al comparar contra los resultados obtenidos por el protocolo de ruteo reactivo implementado en ANTop-1, vemos que finalmente se obtiene un tamaño inferior en las tablas de ruteo, cuando la suposición original indica que las tablas deberían ser comparativamente mayores, ver pagina 139, sección 4.2.7 de [2].

Una vez analizada la composición de la tabla de ruteo es necesario comenzar a verificar el desempeño de la red a la hora de cursar paquetes, por lo cual comenzamos presentando un histograma de distancias recorridas por los paquetes al ser enviados en forma aleatoria en la figura 4.9.

Figura 4.9: Histograma de ocurrencia de las distancias recorridas por los paquetes.

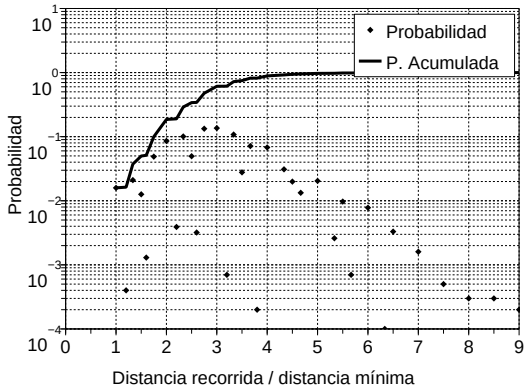


Nuevamente para obtener los resultados se enviaron 3 paquetes por cada uno de los 500 pares de nodos aleatorios, sumando un total de 1500 envíos, vemos una media de 9 saltos con una desviación estándar de 3 saltos. Se puede ver la mejora en cuanto a desempeño al considerar redes pseudo estacionarias, donde con un menor poblado de las tablas de ruteo se logra cursar paquetes recorriendo en promedio menos saltos que en el caso reactivo.

Si bien en redes similares a Internet los protocolos de ruteo utilizados son de tipo EGP, volveremos a comparar su desempeño frente al logrado por un protocolo de ruteo óptimo como ser ISIS u OSPF, ambos de tipo IGP. Para lo cual la figura 4.10 presenta una gráfica con la distribución de probabilidad de la relación entre la distancia recorrida y la distancia mínima entre todos los nodos, y la probabilidad acumulada de dicho valor. Si bien un protocolo de ruteo óptimo claramente presenta un desempeño superior al lograda por ANTop cabe recordar que dicho tipo de protocolos mantiene en sus tablas de ruteo una entrada por cada nodo de la red, además de otras entradas relativas a las direcciones internas de la red. El mantenimiento de dicha estructura de ruteo en los protocolos de tipo IGP mencionados, impone tiempos elevados de convergencia y demanda mucho más espacio para almacenar información en cada nodo.

En este caso vemos que el 95 % de los envíos tienen una relación máxima de 3.5 veces el camino mínimo posible, este número es un poco mayor al obtenido en las redes con topologías aleatorias, esto se debe a que es una topología diferente lo cuál como se explicó previamente requiere un esquema de asignación de direcciones primarias y secundarias diferente, resultando en una menor conectividad a través de direcciones secundarias, lo cual se traduce en una menor eficiencia a la hora de enrutar paquetes. Aunque comparando su desempeño comparado contra el ofrecido por el protocolo reactivo presenta resultados superiores, cabe

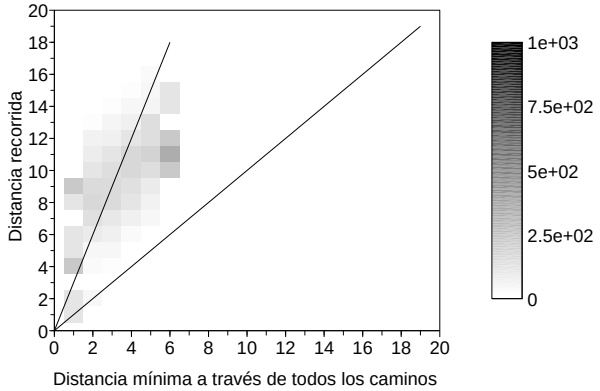
Figura 4.10: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos.



recordar la premisa de baja movilidad de nodos que se asume.

A fin de obtener una idea sobre cuales son las distancias que presentan una relación entre distancia recorrida versus distancia mínima más pobre, se elabora una gráfica que presenta para cada distancia mínima, la probabilidad de ocurrencia para cada distancia recorrida, representado la probabilidad 0 y 1, con el color blanco y negro respectivamente, con todas sus variaciones internas como escala de grises, además se agregan 2 líneas continuas una con pendiente 1 y otra con pendiente 3, a fin de facilitar la comparación visual de los resultados:

Figura 4.11: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.



El resultado de la comparación presentado en la figura 4.11 muestra que el esquema de ruteo proactivo ofrece mayores beneficios para redes que poseen una topología sin escala, ya que en primera instancia ofrece una media de saltos

efectuados inferior a que se observó en redes aleatorias, y la relación con la distancia mínima contiene al 95 % de los envíos por debajo de 3.5, ambos hechos en conjunto ofrecen un desempeño apto para utilizarlo como protocolo de ruteo en este tipo de redes.

4.3.3. Conclusiones del protocolo de ruteo proactivo

La expectativa al diseñar el protocolo era recorrer distancias menores, y que la relación entre distancia recorrida versus distancia mínima sea de 2 ó 3 como máximo, aunque experimentalmente se encuentran valores algo mayores.
Se realizaron pruebas para verificar si al agregar a la tabla de ruteo todos los vecinos físicos aunque no tengan distancia igual a 1, se lograba mejorar la relación. Los resultados no muestran una diferencia significativa y luego de analizar la composición de las redes analizadas y los paquetes enviados se encuentra la razón en las siguientes consideraciones:

- El ruteo proactivo se basa en el principio de localidad, donde la información es propagada a los nodos cercanos y no más allá de cierto radio de alcance definido por el prefijo a propagar.
- Las mediciones son realizadas entre orígenes y destinos aleatorios, pudiendo estar los mismos en posiciones opuestas de la red.

Entendiendo que el ruteo proactivo tiene alcance local y no se distribuye a toda la red, un nodo lejano no puede hacer uso de la información de localidad propagada por el ruteo proactivo, de esta manera cuando decide enrutar un paquete no tiene información adicional a la proporcionada por el árbol T y sus rutas por defecto, por lo cual el paquete es montado sobre el árbol T directamente, solo cuando llegue a la cercanía del nodo de destino y que llegue a un nodo cercano a través de un enlace secundario al nodo destino, que podrá hacer uso de la información de localidad propagada por el ruteo proactivo.

Por esta razón se observa que el ruteo proactivo presenta sus mayores beneficios en los paquetes que inicialmente se encuentran cercanos a través de direcciones secundarias del destino.

Cuando se trata de orígenes y destinos lejanos el ruteo proactivo actúa por su mecanismo por defecto. Este comportamiento es correcto ya que no es el objetivo del protocolo comportarse como un protocolo de tipo *link-state* como OSPF o ISIS, en los cuales todos los nodos tienen información detallada del estado y composición de todos los enlaces de la red, la cual usan para calcular el camino mínimo entre los orígenes y destinos conocidos. Es necesario recordar que protocolos como OSPF o ISIS requieren una gran cantidad de paquetes para mantener sus tablas de ruteo y requieren mucho espacio de almacenamiento para las mismas. El protocolo ANTop-2 con esquema de ruteo proactivo hace uso de información extra de ruteo solo en zonas localizadas, no en la red completa.

4.4. Recuperación ante fallas

En la sección anterior se analiza el comportamiento del protocolo y su desempeño en escenarios donde todos los nodos una vez que se conectan a la red se quedan en la misma, con lo cual tanto los nodos de *Rendezvous* y la estructura del hipercubo se mantienen estables y no necesitan modificación alguna, en el diseño original del protocolo se específica el comportamiento del protocolo en los momentos en que un nodo decide voluntariamente abandonar la red, en el presente trabajo se desarrollaron los mecanismos necesarios para soportar la situación en la que los equipos abandonan la red sin previo aviso y en forma abrupta, en este sección se presentan los resultados de simular dichas situaciones.

A fin de simular el abandono abrupto de un nodo, se efectuaron las modificaciones necesarias tanto al generador de archivos de simulación como al simulador en sí para soportar dicha funcionalidad, los archivos de entrada que utiliza el simulador contienen un listado de los comandos que debe ejecutar el simulador, la sintaxis general de los comandos indica el instante de tiempo y la acción particular que debe ser ejecutada, por ejemplo:

$$[1500ms]node(Y).leave \tag{4.1}$$

La sentencia anterior indica al simulador, que en el milisegundo 1500, el nodo Y debe ejecutar la acción *leave*, la acción *leave* indica al nodo que debe salir de la red automáticamente, para lo cual se simula una pérdida total de señal de transmisión, el radio de cobertura del nodo se reduce completamente y no puede recibir ni enviar información.

Una vez implementada la funcionalidad de desconexión abrupta de nodos, se generan los archivos de simulación correspondientes, para lo cual se decide probar el protocolo simulando la desaparición del 0.1 %, 0.2 %, 0.5 %, 1 % y 2 % del total de los nodos que conforman la red. Para cada porcentaje de abandono se generan nuevamente 20 archivos de simulación con 500 envíos entre pares aleatorios, tomando en cuenta de no enviar datos desde ni hasta un nodo que haya abandonado la red.

Una vez que un nodo abandona la red, sus sucesores pierden conectividad con sus predecesores de orden N, a los cuales llegaban a través del *Default Gateway* que proveía el predecesor de primer orden, al no poder acceder directamente a sus predecesores de orden N, pierde su conectividad directa al árbol T.

Los mecanismos de recuperación ante fallas descriptos en este trabajo se basan en el concepto de la conectividad a través de direcciones secundarias, ya que el concepto es utilizar los enlaces fuera del árbol T provistos por las rutas secundarias para ofrecer una forma alterna de conectarse al árbol T. Cuando no existe posibilidad de conectarse al árbol T nuevamente a través de las direcciones secundarias se produce una partición en la red aislada del árbol T.

A medida que mayor número de nodos abandonan la red, es más probable que se produzcan particiones y que la red no pueda recuperarse, en el presente

trabajo es necesario analizar hasta que punto los mecanismos de recuperación ante fallas son capaces de rearmar las tablas de ruteo y proveer conectividad y alcanzabilidad entre los nodos. Para lo cual en las simulaciones se produce el abandono abrupto de los nodos, se deja la red estacionaria por algunos minutos para que los mecanismos de recuperación ante fallas rearmen las tablas de ruteo, y luego se realiza el envío de los 1500 paquetes en la red. Lo cual ofrece los resultados que serán utilizados para el análisis.

Con los datos ofrecidos por la simulación y con el objetivo de analizar el desempeño de la red frente a escenarios de falla se define una métrica cuyo propósito es considerar todos los factores que hacen a la funcionalidad de la red y el estado de la misma de manera tal de poder evaluar en que punto una red ya no es útil debido al número de nodos que abandonan la misma.

Se propone medir el grado de desconexión de una red o grado de separación, tomando en cuenta los siguientes conceptos para la construcción de la métrica:

Grado de particionamiento El grado de separación debe ser directamente proporcional al grado de particionamiento de la red, dado que un mayor número de particiones presentes resulta en una red menos conexa.

Centro de masas comparativo Otra medida que debe ser tomada en cuenta es la relación entre el tamaño de las particiones que se generan, o la "comparabilidad" de los tamaños (que sean comparables entre sí), ya que el mayor tamaño en las particiones existentes resulta una red menos conexa. Una vez particionada una red, comparando los tamaños de las particiones se puede diferenciar a una gran red con pequeñas particiones, de un grupo de redes con tamaños similares, haciendo un análisis de la comparabilidad de los tamaños de las particiones:

- La Medida de comparabilidad debe ser baja para redes con muchas particiones pequeñas
- La Medida de comparabilidad debe ser alta cuando existen muchas particiones grandes

Porcentaje de paquetes enviados exitosamente La cantidad de paquetes exitosamente cursados debe impactar en forma inversa en el cálculo del grado de separación, ya que claramente cuanto mayor es la cantidad de paquetes que no logran encontrar su destino, menos útil será la red.

Cada uno de los factores antes mencionados dará como resultado una parte de la ecuación para obtener el grado de separación, a continuación se presentan los criterios utilizados para representar cada uno de los factores y finalmente formar la ecuación del grado de separación. Cada uno de los factores que serán utilizados debe variar en forma consistente a las variaciones de la topología, variando entre un valor mínimo en el momento en que la red funciona exitosamente sin ningún tipo de falla y tomando su valor máximo cuando la red esta completamente colapsada. Se toma como concepto que en una red sin particionar, donde todo funciona correctamente, el grado de separación debe tender a 1, y en una red completamente colapsada el grado de separación debe tender a infinito.

Grado de particionamiento Debe ser una medida que indique que tan particionada esta una red, tomando en cuenta el tamaño de la misma y la cantidad de particiones generadas, donde si la red no tiene particiones este valor debe tender a 1, y en caso contrario aumentar a medida que la cantidad de particiones aumenta.
La cantidad de particiones que se genera no considera a los nodos que abandonan la red, no se los considera una partición aislada de tamaño 1, simplemente no son parte de la ecuación, y el árbol T tampoco se considera una partición, por lo cual el número de particiones es igual a 0 en el caso de no existir fallas. En consecuencia la cantidad de particiones varia entre 0 y la cantidad total de nodos, descontando los nodos que han abandonado la red y descontando el árbol T.

$$0 \leq CP \leq CN - DN - 1 \tag{4.2}$$

Donde:

- CN: Cantidad de nodos
- DN: Cantidad de nodos que abandonan la red
- CP: Cantidad de particiones existentes en la red

Como antes mencionamos debe tender a 1 en una red sin alteraciones y aumentar a medida que aumentan las particiones, siendo representativo según el tamaño de la red, ya que no es lo mismo que se generen 5 particiones en una red de 500 nodos que 6 particiones en una red de 5000 nodos.

$$GCP = \frac{1}{1 - \frac{CP}{CN}}; \tag{4.3}$$

Donde:

- GCP: Grado de particionamiento de la Red.

A fin de verificar la ecuación, vemos que en una red sin particiones el valor es siempre igual a 1, y a medida que las particiones aumentan el denominador tiende a 0 por la relación CP/CN, en consecuencia el grado de particionamiento es cada vez mayor. A su vez una red con 5 particiones en una red de 500 ofrece un denominador menor a una red con 5 particiones en una red de 5000 nodos, debido a que 5/500 es mayor a 5/5000.

Centro de masas comparativo Para representar el grado de comparabilidad comenzamos analizando el momento de inercia de primer orden, o centro de masas, donde se considera a cada partición como un objeto de masa igual a la cantidad de nodos presentes en la partición localizado en una posición en X igual al valor de masa del cuerpo, de esta manera la presencia de grandes objetos de mayor masa inclina el centro de masas hacia valores mayores, y una cantidad elevada de objetos de menor masa inclina el centro de masas hacia valores inferiores.

$$CM = \frac{\sum_{i=1}^{CP} CNP_i * CNP_i}{CN} \tag{4.4}$$

Donde:

- CNP_i: Es la cantidad de nodos presentes en la partición i.
- CM: Centro de masas.
- CN: Cantidad total de nodos.

De esta manera se obtiene el centro de masas de la red, aunque es necesario obtener un valor relativo al tamaño de la red, de manera tal de obtener un valor que sea comparable tanto en una red de 10 nodos como en una red de 5000 nodos. Por lo cual buscamos relacionarlo con el centro de masas de una red sin particionar, obteniendo como resultado un indicador que nos muestra que tan corrido se encuentra el centro de masas con respecto del caso ideal.

Siendo el centro de masas de una red ideal.

$$CM_0 = \frac{CN * CN}{CN} = CN \tag{4.5}$$

Donde:

- CM_0: Es el centro de masas de una red en la que ningún nodo abandona la red.

Donde el centro de masas comparativo se define como la relación entre el centro de masas en el caso ideal y el centro de masas de la red particionada.

$$CMC = \frac{CM_0}{CM} \tag{4.6}$$

Donde:

- CMC: Centro de masas comparativo de la red.

Combinando las ecuaciones 4.4, 4.4 y 4.4, obtenemos:

$$CMC = \frac{\frac{CN*CN}{CN}}{\frac{\sum_{i=1}^{CP} CNP_i*CNP_i}{CN}} \tag{4.7}$$

Y luego de simplificar términos, resulta el centro de masas comparativo:

$$CMC = \frac{CN}{CM} \tag{4.8}$$

Nuevamente verificamos que en el caso ideal, el centro de masas de la red es igual al centro de masas en el caso ideal, con lo cual resulta igual a 1. A medida que el centro de masas se aleja del caso ideal, tomando su valor mínimo en una red poblada de particiones mínimas, resultando en un valor de centro comparativo de masas máximo.

Porcentaje de paquetes enviados exitosamente En este caso el cálculo es directo, simplemente se toma la cantidad de paquetes enviados y recibidos correctamente y se divide por la cantidad de paquetes enviados, en este caso igual a 500.

El porcentaje de paquetes enviados exitosamente es:

$$PE = \frac{PEE}{PET} \tag{4.9}$$

Donde:

- PEE: Siendo igual a los paquetes enviados exitosamente.
- PET: Siendo igual a la cantidad total de paquetes enviados, 500.
- PE: Porcentaje de paquetes enviados exitosamente de origen a destino y viceversa.

Para el grado de separación el porcentaje de paquetes enviados es utilizado como denominador, con lo cual en el caso ideal no existe perdida de paquetes, siendo $PE = 1$, y a medida que los paquetes se van perdiendo en el camino el porcentaje tiende a 0, y el grado de separación aumenta en consecuencia.

Grado de separación Considerando la manera en que deben interactuar los diferentes factores que determinan el grado de separación de una red, se propone la ecuación 4.4 para su cálculo:

$$GD = \frac{(KGCP * GCP) * (KCMC * CMC)}{KPE * PE} \tag{4.10}$$

Siendo los factores que forman parte de la ecuación:

- GD: Grado de separación.
- GCP: Grado de particionamiento de la Red.
- $KGCP$: Factor de peso del grado de particionamiento de la red.
- CMC: Centro de masas comparativo de la red.
- $KCMC$: Factor de peso del centro de masas de la red.
- PE: Porcentaje de paquetes enviados exitosamente de origen a destino y viceversa.
- KPE: Factor de peso del porcentaje de paquetes exitosamente enviados.

4.4.1. Redes aleatorias

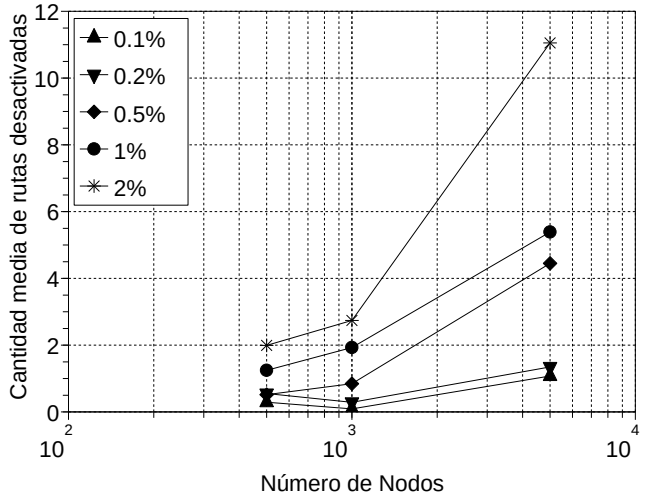
En esta sección se presentan los resultados del desempeño del protocolo en redes aleatorias, nuevamente se toma como muestra redes de 500, 1000 y 5000 nodos.

Comencemos verificando como quedaron compuestas las tablas de ruteo en cada caso. Según lo indica el mecanismo de recuperación ante fallas se agregan 2 tipos de rutas, aquellas que corresponden a los *gateway* alternos y aquellas correspondientes a las rutas temporales, donde las rutas temporales son el resultado de propagar el descubrimiento de *gateway* alternos.

Otro resultado del mecanismo de recuperación ante fallas es la desactivación de rutas del protocolo proactivo, dado a que dejan de ser viables debido a la caída de enlaces, en este caso nos interesa comenzar con las rutas que deben ser desactivadas, para compararlas con el total de las rutas referirse al capítulo 4.3.1 donde se presentan las gráficas de tamaño de tablas de ruteo en caso ideal.

En la figura 4.12 presenta para los diferentes tamaños de redes cual es el promedio de rutas que fueron desactivadas, donde para cada porcentaje de falla el promedio graficado es la media calculada en base a un conjunto formado por todas las rutas que fueron desactivadas por cada nodo en todas las simulaciones. Donde el promedio graficado es la media calculada en base a un conjunto formado por todas las rutas que fueron desactivadas por cada nodo en todas las simulaciones.

Figura 4.12: Número medio de rutas que son desactivadas.

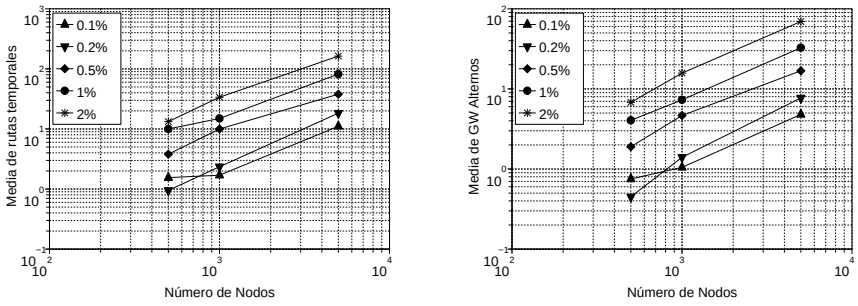


Cabe recordar que no todos los nodos de la red sufren modificaciones ni son afectados por los nodos que abandonan la red, razón por la cual vemos un promedio bajo de rutas inactivas, lo cual se hace aun más evidente cuando analizamos la cantidad de rutas que son agregadas por los mecanismos de recu-

peración ante fallas, ya que por cada *gateway* alterno que se agrega son muchas más las rutas que se inactivan.

A fin de analizar la cantidad de rutas que se agregan en caso fallas en la red, la figura 4.13 presenta la cantidad promedio de nodos que instala un *default gateway* alterno, junto con una gráfica que muestra la cantidad promedio de rutas temporales que hay instaladas en la red luego de procesar el algoritmo de recuperación ante fallas:

Figura 4.13: Número medio de nodos que asumen un *gateway* alterno.



Vemos que el número de rutas que agregan los mecanismos es realmente bajo en comparación al número de rutas que maneja el protocolo proactivo, cabe recordar que la mayoría de las rutas en el protocolo son debidas a propagación de rutas, lo cual aquí se ve reflejado, existe un alto número de rutas desactivadas y un muy bajo número de rutas nuevas, siendo que la mayoría de las rutas que se desactivan provienen de los mecanismos de propagación de rutas.

Luego de analizar el contenido de las tablas de ruteo, continuaremos por analizar el desempeño logrado por la red una vez que se recupera del abandono repentino de nodos, para lo cual la figura 4.14 presenta nuevamente la comparación entre camino recorrido por los paquetes versus el camino mínimo que hubieran seguido si hubieran utilizado un protocolo de ruteo óptimo.

Si bien las simulaciones presentan datos para redes con para 0.1, 0.2, 0.5, 1 % de nodos que abandonan la red, analizamos solamente el caso mínimo y máximo por ser los más representativos, en la figura las 2 gráficas superiores corresponden al 0.1 y 2 % de fallas respectivamente para una red de 1000 nodos, y las gráficas inferiores muestran el mismo caso para una red de 5000 nodos, en ambas vemos que lo que varia en gran medida es la cantidad de paquetes que logran ser cursados exitosamente, la menor cantidad de paquetes cursados se debe a la inexistencia de conexiones secundarias susceptibles de ser utilizadas como *gateway* alterno, dada la alta cantidad de paquetes que no logran ser cursados se toma a 2 % como el umbral máximo para las pruebas a realizar.

Analizando las gráficas no se detecta un incremento notable en la distancia recorrida por los paquetes, esta situación es comprensible cuando se analiza este

Figura 4.14: Distancia recorrida versus distancia mínima, para 0.1 y 2 % de fallas respectivamente, en una red de 1000 y 5000 nodos respectivamente. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.

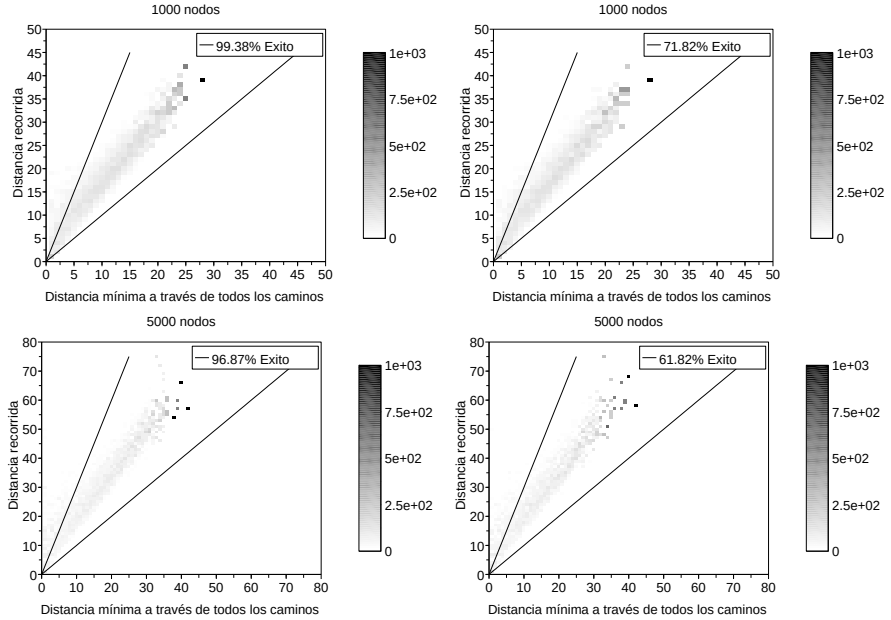


gráfico en conjunto con la figura que muestra la cantidad promedio de nodos que instalan un *gateway* alterno. El mecanismo de recuperación ante fallas indica que una vez que un nodo pierde su *gateway* debe comenzar a solicitar a sus sucesores que localicen un nuevo *gateway* alterno, con lo cual si un nodo encuentra un *gateway* alterno en un sucesor de orden N, entonces como mínimo N nodos tendrán que instalar un *gateway* alterno.

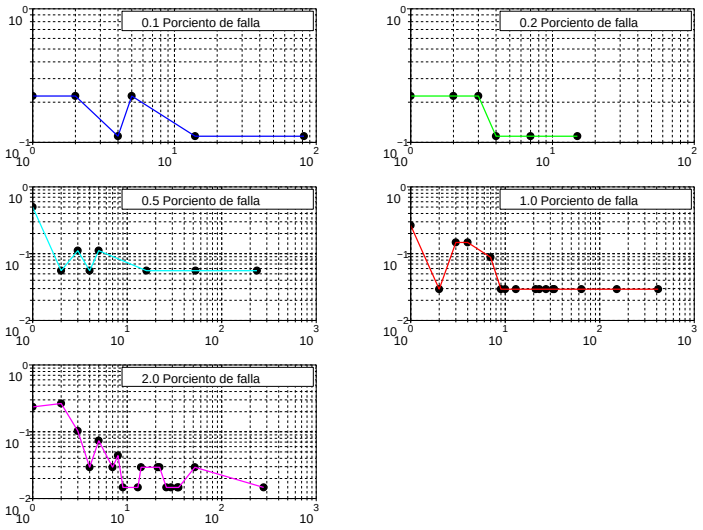
Las gráficas muestran una relación casi directa entre la cantidad de nodos que abandonan la red y la cantidad de nodos que instalan un *default gateway* alterno, esto significa que la mayoría encontró un *gateway* alterno sin solicitárselo a un sucesor. Un enlace secundario que el mismo nodo posee provee la conectividad, con lo cual una caída implica solo un nodo que instala un *gateway* alterno, lo cual es exactamente lo que se ve en la figura 4.13. En consecuencia la distancia recorrida por los nodos varia como mucho en 1 salto.

Cabe recordar que en las redes aleatorias existe una gran conectividad a través de enlaces secundarios, esto se ve reflejado en los resultados en el momento en que los nodos encuentran *gateway* alternos sin la necesidad de atravesar un gran número de sucesores.

Luego de analizar el desempeño de la red es hora de analizar la métrica propuesta, el grado de separación, para lo cual todavía nos falta analizar la composición de las particiones de la red y como están distribuidas, para lo cual se presenta la figura 4.15 y 4.16 que presentan la distribución de probabilidades

para los tamaños de las particiones presentes en redes de 500 y 5000 nodos para todos los porcentajes de falla probados.

Figura 4.15: Distribución de probabilidad de tamaño de particiones para una red de 500 nodos



La figura de distribución de probabilidad para los tamaños de particiones en la red para 500 y 5000 nodos muestra como a medida que existen más nodos que abandonan repentinamente la red mayor será la cantidad de islas o particiones que se producen en la red, el resultado es consistente con el obtenido al enviar los paquetes y presentado anteriormente, donde veíamos que el porcentaje de paquetes exitosamente enviados se vuelve notable cuando el 2 % de los nodos abandonan la red, en este caso lo vemos reflejado en la distribución de las particiones donde al 2 % la cantidad de particiones presentes tiene mayor influencia en los calculos.

Como resultado de los análisis anteriores llegamos a la figura 4.17 en la cual se puede apreciar el grado de separación presente en la red.

El grado de separación claramente respeta las condiciones previstas, se ve afectado por la mayor cantidad de particiones existentes en la red, y por la cantidad de paquetes exitosamente cursados, siendo cercano al valor unitario en los casos mínimos de falla. Analizando el gráfico se ve que el grado de separación se ve acotado para valores bajos de falla pero extrapolando la función más allá del 2 % de falla claramente se ve que el grado de separación se eleva considerablemente a medida que más nodos abandonan la red.

En base a los resultados presentados en las gráficas anteriores y el grado de separación visto vemos en 2 % de fallas un umbral más allá del cual los mecanismos de recuperación ante fallas y la topología misma de la red requieren la presencia de algoritmos que ofrezcan la posibilidad de rearmar el ruteo completo

Figura 4.16: Distribución de probabilidad de tamaño de particiones para una red de 5000 nodos

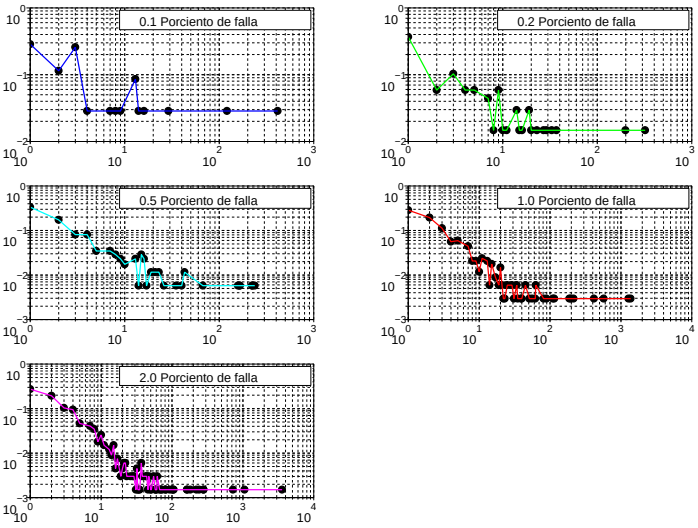
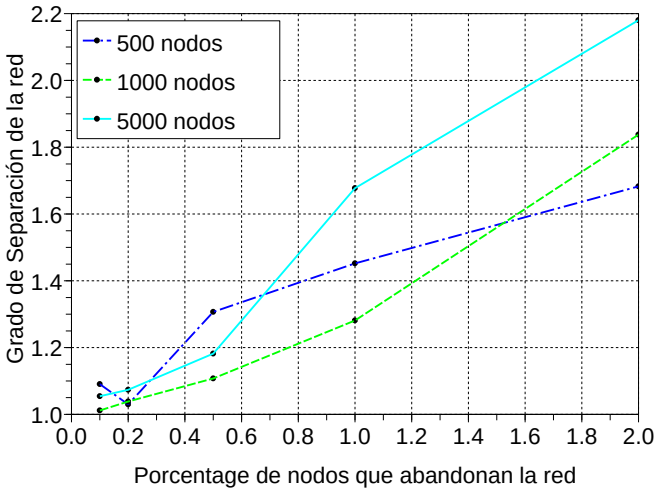


Figura 4.17: Grado de separación presente a medida que aumenta el porcentaje de nodos que abandonan la red sin aviso.



de la red considerando cada partición como una red autónoma. Lo cual debe ser analizado mediante mecanismos de *Merge and Split*.

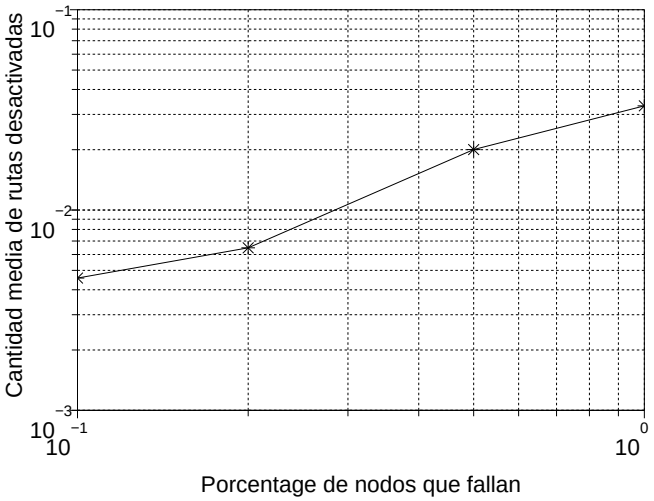
4.4.2. Redes sin escala

En esta sección veremos el resultado de ejecutar las simulaciones en redes que presentan una distribución de nodos que sigue una ley de potencias, las redes cuentan aproximadamente con 8500 nodos, y se utilizo el mismo porcentaje de fallas que en las redes aleatorias 0.1, 0.2, 0.5, 1 %, siendo la cantidad de nodos que abandonan la red 8, 16, 45 y 85 respectivamente, no se utilizo un porcentaje de fallas de 2 % ya que representa 170 nodos que abandonan la red, siendo este número demasiado alto para las intenciones del protocolo de tratamiento de fallas.

De la misma manera que se realizó el análisis de los resultados para el caso de redes con topologías aleatorias, se realizará el análisis para redes con topología sin escala, comenzaremos analizando la composición de las tablas y el efecto que los mecanismos de recuperación ante fallas tienen sobre las mismas.

La figura 4.8 muestra la relación entre rutas secundarias y aquellas asignadas en forma directa de predecesor a sucesor en forma de dirección primaria, la misma muestra que existen pocas direcciones secundarias asignadas, y cabe recordar que la mayor cantidad de rutas en la tabla de ruteo del protocolo proactivo son aquellas provenientes de la propagación de rutas secundarias.

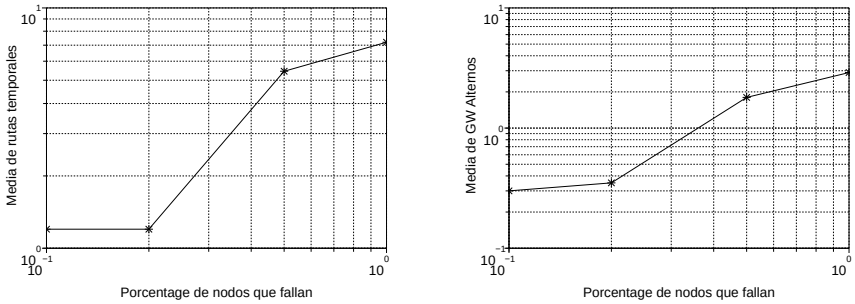
Figura 4.18: Número medio de rutas que son desactivadas.



En la figura 4.18 se pueden ver las rutas que son desactivadas una vez que los mecanismos de recuperación ante fallas actualizan la información en las tablas de ruteo, donde el promedio graficado es la media calculada en base a un conjunto formado por todas las rutas que fueron desactivadas por cada nodo en todas las simulaciones. La cantidad de rutas que son desactivadas es notablemente inferior al caso de las redes aleatorias, la razón es la menor existencia de rutas propagadas en la tabla de ruteo, siendo las rutas propagadas aquellas que principalmente son desactivadas por estos mecanismos.

La segunda modificación importante en las tablas de ruteo es la adición de rutas temporales, sea para instalar un *gateway* alterno o para informar la alcanzabilidad de ciertos prefijos debido a la instalación de un *gateway* alterno, en la figura 4.8 se puede ver la cantidad de rutas que son agregadas en forma temporal y la cantidad promedio de nodos que instalan un *gateway* alterno producto de los mecanismos de recuperación ante fallas.

Figura 4.19: Número medio de nodos que asumen un *gateway* alterno.



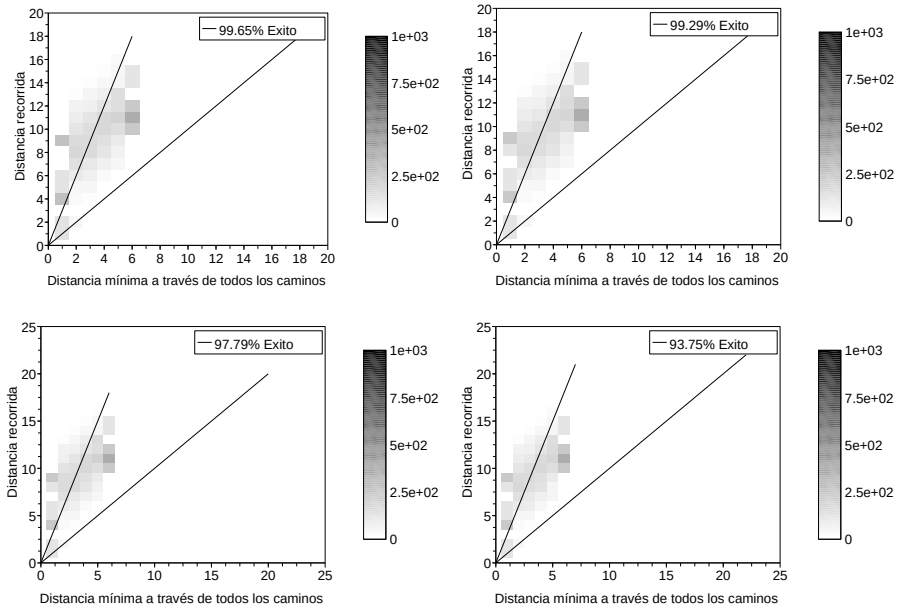
Nuevamente como consecuencia de la baja conectividad a través de enlaces secundarios que existe en las redes con topologías sin escala, vemos un promedio mucho menor de rutas temporales y un número menor de nodos que instalan un *gateway* alterno, este resultado no es inesperado ya que los mecanismos de recuperación se basan exclusivamente en la existencia de rutas secundarias, y las mismas son escasas en este tipo de redes.

Luego de analizar la composición de las tablas de ruteo es tiempo de analizar el desempeño de la red a la hora de cursar tráfico, para lo cual se utilizó el mismo esquema de simulaciones que para las redes aleatorias, generando envíos de paquetes entre orígenes y destinos aleatorios, teniendo en cuenta que tanto el destino como el origen no sean los nodos que voluntariamente se alejan de la red, y realizando 500 envíos entre ellos.

Como valor comparativo nuevamente se toma la distancia que los paquetes hubieran recorrido si hubiesen utilizado un protocolo de ruteo óptimo como ser un IGP (OSPF o ISIS), de esta distancia se obtiene la relación con la distancia realmente recorrida por los paquetes y se presenta en una gráfica junto con 2 líneas de pendiente 1 y 3, a fin de poder visualizar fácilmente el momento en que los paquetes recorren más de 3 veces la distancia mínima.

En la figura 4.20 puede verse tanto la relación entre distancia recorrida contra distancia mínima y la cantidad de paquetes que son correctamente enviados en la red, de los resultados obtenidos puede verse que la distancia recorrida por los paquetes no se ve afectada, esto es debido a la baja cantidad de rutas que fueron agregadas en este caso y la baja cantidad de nodos que obtienen un *gateway* alterno.

Figura 4.20: Distancia recorrida versus distancia mínima, para 0.1, 0.2, 0.5, 1 % de fallas. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.



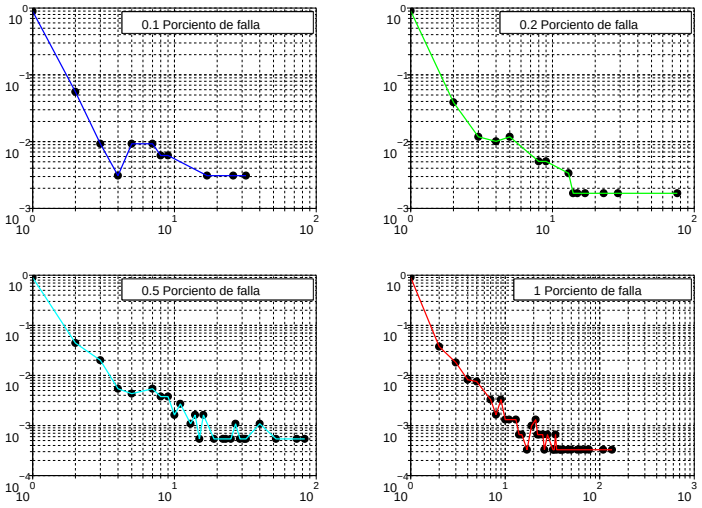
En cuanto a la cantidad de paquetes correctamente cursados, se puede ver que son pocos los paquetes que no logran cursar correctamente de origen a destino, claramente la cantidad de paquetes cursados correctamente es notablemente superior a los resultados obtenidos en una red aleatoria de 5000 nodos. La razón de esta diferencia se encuentra en la composición misma de la red, ya que en una red sin escala al existir nodos que abandonan la red en forma aleatoria es altamente probable que se elija un nodo hoja en vez de un nodo raíz, dado que en las redes sin escala la mayor cantidad de nodos son hojas.

De esta manera la baja cantidad de nodos que instalan un *gateway* alterno no tiene un impacto directo en la cantidad de nodos cursados correctamente. En una red sin escala el problema reside cuando un nodo con un alto grado de conectividad abandona la red, no cuando los nodos que abandonan la red son nodos hoja.

A fin de comprender correctamente la situación analizaremos como se componen las particiones generadas en la red luego de que los nodos seleccionados abandonen la red.

En la figura 4.21 puede verse que el número de particiones generadas tiene un tamaño menor al que existe en las redes aleatorias, en este caso las particiones provienen de la partida de nodos con alto grado de conectividad.

Figura 4.21: Distribución de probabilidad de tamaño de particiones.

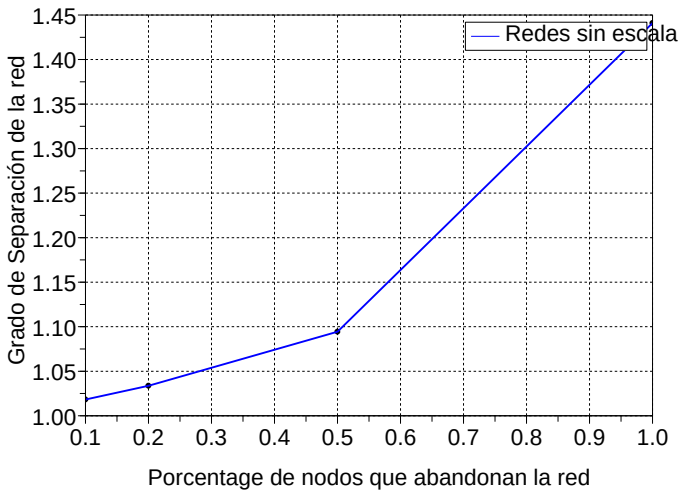


El grado de separación que presenta la red en la figura 4.22 es similar al obtenido en redes aleatorias, aunque los resultados esperados hubiesen sido de esperar un grado de separación mayor, la razón es que experimentalmente es muy probable desconectar nodos que son hojas en el caso de redes sin escala, y en el caso de redes aleatorias es más probable desconectar nodos de importancia.

Considerando las características de ambas topologías, en las redes que siguen una topología sin escala la asignación de rutas secundarias es mucho menor, lo cual como fue visto anteriormente afecta directamente en la capacidad de los mecanismos de recuperación ante fallas de encontrar *gateway* alternos, cuando en redes aleatorias esta capacidad es mucho mayor, siendo más propensas a recuperarse ante eventos de fallas. El resultado esperado difiere debido a la baja probabilidad de desconectar nodos importantes en redes sin escala.

Analizando los resultados presentados en redes aleatorias y los presentados en la presente sección vemos en el grado de separación una representación correcta de la situación y comportamiento de las diferentes redes frente a escenarios donde nodos abandonan la red sin previo aviso, la curva crece considerablemente al 2 % respecto del valor obtenido al 1 % y valores inferiores, lo cuál es consistente con la perdida de paquetes observada en dichos porcentajes de falla.

Figura 4.22: Grado de separación presente a medida que aumenta el porcentaje de nodos que abandonan la red sin aviso.



4.5. Distribución de servidores de *Rendezvous*

En esta sección nos concentramos en analizar el resultado del algoritmo de distribución de información, el cual hace uso de la estructura binaria que tiene la red en su direccionamiento, y de la manera en que las direcciones primarias son asignadas, como resultado permite distribuir la información de localización en una cantidad aleatoria *nrv* de nodos, donde *nrv* toma valores en potencias de 2, mediante la división en subramas a partir del nodo raíz de la red, 00000.../0.

La mecánica que se sigue para analizar los resultados es la misma que fue utilizada anteriormente, utilizando un set de 20 simulaciones en cada configuración de escenarios, siendo en este caso la cantidad de nodos en los cuales se distribuye la información la característica que sufre variación. La cantidad de servidores que se utilizaron depende de la cantidad de nodos presentes en la red, donde tomamos como criterio que el número máximo de servidores de *Rendezvous* es el logaritmo en base 2 de la cantidad de nodos presentes en la red.

$$RVCount = \lceil log_2 CN \rceil \tag{4.11}$$

Siendo:
RVCOunt = Cantidad de nodos que oficiaran como servidores de *rendezvous*.
CN = Cantidad de nodos presentes en la red.

Aplicando la ecuación 4.5 y tomando en cuenta que la división de ramas de igual nivel del árbol binario sigue potencias de 2, para redes de 500 nodos se realizaron pruebas utilizando 1, 2, 4 y 8 servidores de *Rendezvous*, mientras que en redes de 1000 y 5000 nodos la ecuación 4.5 indica que también se debe simular utilizando 16 servidores de *Rendezvous*.

A fin de verificar si el mecanismo de distribución de direcciones funciona correctamente se pretende medir la cantidad de entradas resultantes en cada nodo y verificar que la distribución es equiprobable.

Una vez más al analizar la solución planteada se organizan las simulaciones según las características de la topología de red sobre la cual son ejecutadas, redes con topologías en las cuales los nodos se conectan en forma aleatoria y redes en las cuales la conexión y topología presentan una distribución de grados de conexión que sigue una ley de potencias.

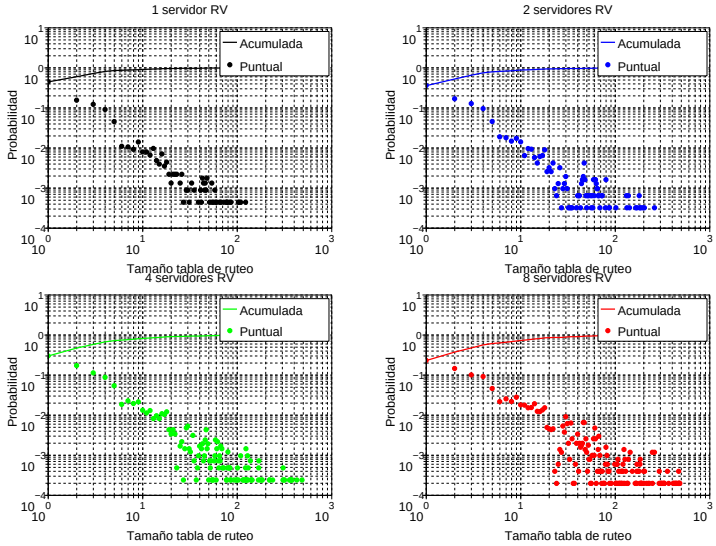
4.5.1. Redes Aleatorias

En la presente sección se presenta el resultado de implementar el algoritmo de distribución de información en redes cuya topología se arma en forma aleatoria, siendo la medida de interés la distribución de entradas en las tablas de *Rendezvous* de los nodos. Se tomaron 20 topologías de red base para 500, 1000 y 5000 nodos, luego se realizaron las simulaciones correspondientes utilizando la cantidad de servidores de *Rendezvous* como la ecuación 4.5 indica, luego se

toma de cada nodo la cantidad de entradas que posee su tabla de *Rendezvous* y mediante estos valores se obtiene una distribución estadística del tamaño de las tablas de *Rendezvous* en los nodos de la red.

En la figura 4.23 se puede apreciar la distribución de tamaños de las tablas de *Rendezvous* en una red compuesta por 500 nodos.

Figura 4.23: Distribución tamaños de tablas de *Rendezvous* para una red de 500 nodos con 1, 2, 4 y 8 servidores de *Rendezvous*



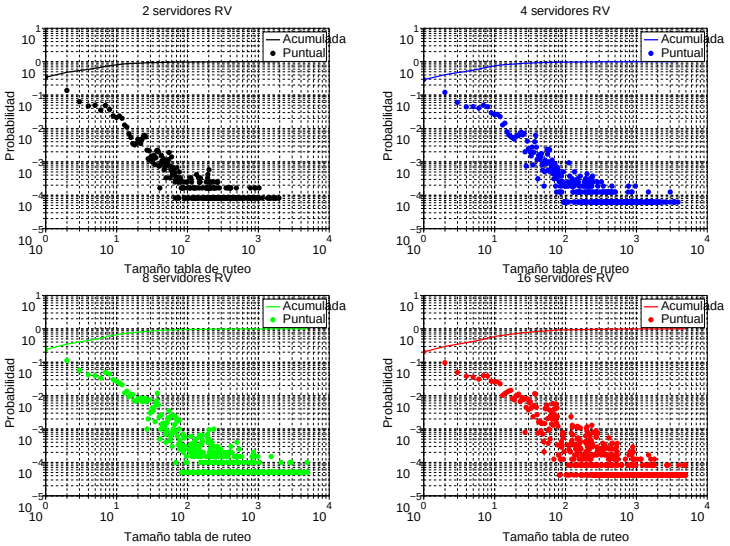
A modo de comparación en la figura 4.24 podemos observar la distribución de tamaños de la tabla de *Rendezvous* para una de 5000 nodos. Comparativamente podemos ver que a medida que aumenta la cantidad de servidores en los cuales se distribuye la información, la cantidad de entradas en las tablas de ruteo aumenta, pero no aumenta en forma lineal.

Siendo esta una característica interesante ya que demuestra la no existencia de nodos concentradores, e indica una distribución equitativa, dado que si el algoritmo siempre eligiera al mismo tipo de nodos en la red, al ir aumentando la cantidad de servidores de *Rendezvous* serían siempre los mismos nodos los que recibirían toda la información resultando en un crecimiento lineal de la cantidad de entradas en las tablas de *Rendezvous*.

En las figuras se detecta la existencia de nodos con un elevado número de entradas con respecto a la mayoría, siendo que para una red de 500 nodos el 95 % de los nodos tienen menos de 10 entradas y en el caso de una red de 5000 nodos el valor no es muy diferente. La existencia de dichos nodos con un elevado tamaño de la tabla de *Rendezvous* no puede ser explicado con la información hasta aquí presentada, para lo cual se procede a introducir una nueva medición.

A fin de obtener información sobre la razón de la existencia de nodos con grandes tamaños de la tabla de *Rendezvous*, se decide analizar de alguna manera

Figura 4.24: Distribución tamaños de tablas de *Rendezvous* para una red de 5000 nodos con 2, 4, 8 y 16 servidores de *Rendezvous*



que nodos son aquellos que presentan dicha característica y encontrar alguna relación entre ellos que explique el resultado. Para lo cual medimos la relación que existe entre el tamaño promedio de la tabla de *Rendezvous* y el tamaño de la máscara de cada nodo, de esta manera se intenta localizar que posición dentro del árbol base poseen los nodos con grandes tablas de *Rendezvous*.

En la figura 4.25 puede verse para una red de 500 nodos la relación entre la cantidad promedio de entradas de *Rendezvous* en un nodo versus el tamaño de la máscara de red que posee cada nodo. A continuación de la misma también se presentan en la figura 4.26 los resultados obtenidos en redes de 5000 nodos, siendo los resultados obtenidos en redes de 1000 nodos similares, no serán mostrados en esta sección, para ver mayor detalle de los resultados obtenidos referirse al Anexo de figuras 6.

Los resultados muestran que los mayores tamaños de tablas de *Rendezvous* se encuentran en nodos con máscaras chicas, siendo estos nodos aquellos que han cedido pocos espacios de direcciones y por lo tanto aglomeran o representan a una cantidad superior de nodos, considerando este dato se entiende que dichos nodos presenten una mayor cantidad de entradas en la tabla de *Rendezvous*. En el momento en que dichos nodos deleguen su espacio de direcciones también delegarán las entradas de *Rendezvous* correspondientes a dichos espacios, y en ese momento el tamaño de sus tablas se verá reducido.

Queda demostrado que el algoritmo planteado para la distribución de información de localización funciona correctamente si consideramos que la red es un hipercubo completo, aunque es raro el caso en el que a partir de una red aleatoria se logre formar un hipercubo completo. En los casos en que la red no es un hipercubo completo se encontrarán necesariamente nodos con tablas de

Figura 4.25: Tamaño promedio de la tabla de *Rendezvous* versus tamaño de la máscara de los nodos para 500 nodos

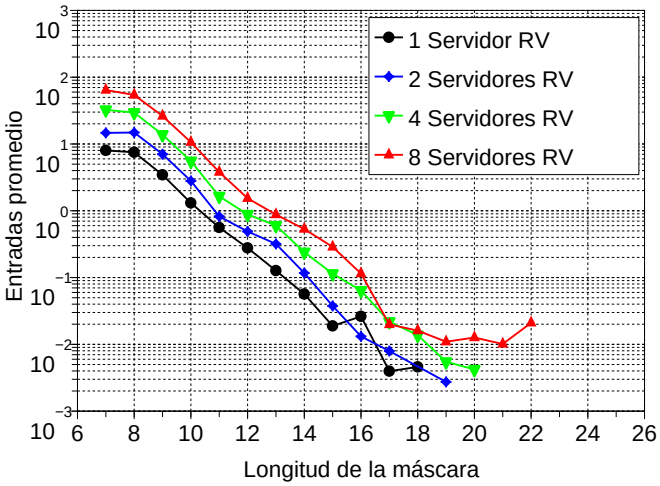
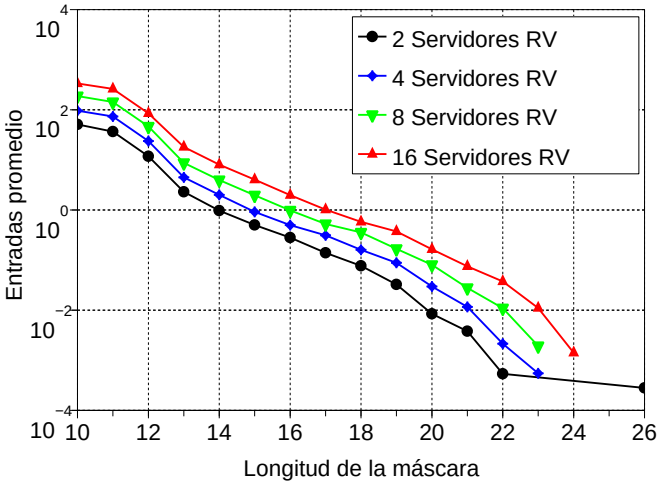


Figura 4.26: Tamaño promedio de la tabla de *Rendezvous* versus tamaño de la máscara de los nodos para una red con 5000 nodos



Rendezvous de mayor tamaño.

Devido al esquema de generación de las redes aleatorias no es posible realizar algún tipo de análisis que permita plantear un esquema de distribución de servidores *Rendezvous* basado en experiencias previas y que ofrezca una distribución más equitativa de la información de localización. Este concepto no es compatible con el tipo de red que se está analizando, ya que la característica de la red es la no predictivilidad de su topología.

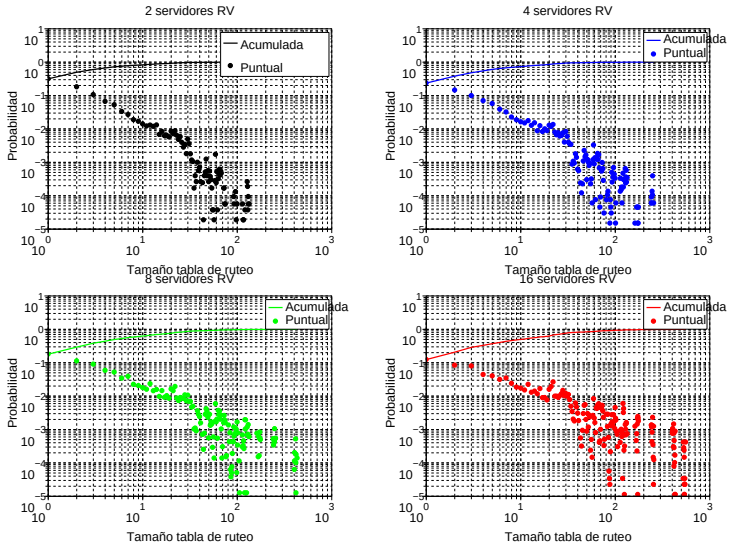
4.5.2. Redes sin escala

En forma análoga al análisis realizado para redes con topologías aleatorias, es de interés analizar como se comporta el protocolo en una red cuya topología presenta características similares a las de Internet, recordemos algunas características que pueden ser de importancia para este escenario en particular, las redes con topologías sin escala tienen un alto número de nodos con muy pocas conexiones, es decir nodos hoja, donde el núcleo de la red posee un alto grado de conectividad.

Una elevada existencia de nodos hoja y el echo que los nodos de núcleo más elevado tienen alta conectividad, indica que si bien el hipercubo no es completo, existe potencialmente una mejor distribución del espacio de direcciones, deberían quedar pocos nodos que almacenen un gran espacio de direcciones sin delegar, aunque cabe recordar que sigue existiendo un componente aleatorio al crear este tipo de redes, el cual puede ocasionar la existencia de nodos en el núcleo más elevado que no tengan nodos hoja conectados.

En la figura 4.27 podemos ver la distribución de tamaño de las tablas de *Rendezvous*, obtenido de la ejecución de 20 simulaciones con el número de servidores de *Rendezvous* indicado para cada caso.

Figura 4.27: Distribución tamaños de tablas de *Rendezvous* con 2, 4, 8 y 16 servidores de *Rendezvous*

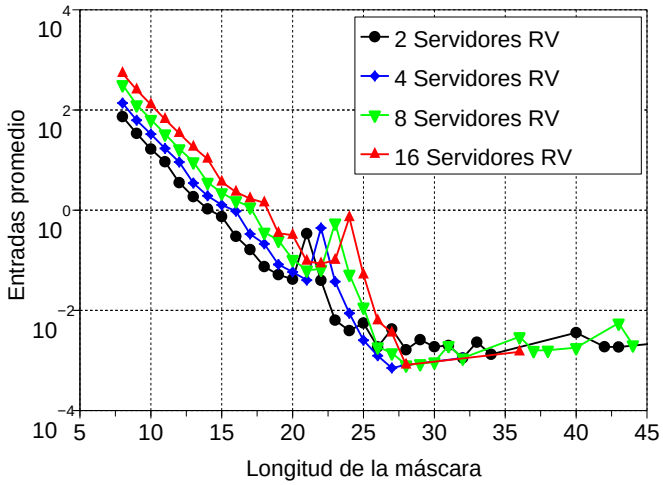


Por la cantidad de nodos podría directamente compararse contra el caso de redes aleatorias de 5000 nodos, recordemos que en las topologías sin escala utilizamos aproximadamente 8500 nodos, claramente el tamaño de las tablas de *Rendezvous* es mucho menor en el caso de redes sin escala que el obtenido en las redes aleatorias, ver figura 4.24, siendo que posee una mayor cantidad de nodos vemos que el máximo en 16 servidores de *Rendezvous* no llega a 1000 entradas, cuando en el caso de las redes aleatorias ese valor tiene un orden de magnitud

más, llegando casi a las 6.000 entradas.

Es fácil observar que existe un desempeño superior en el caso de redes sin escala, esto se debe a la estructura antes mencionada, existe un menor número de nodos que concentran grandes espacios de direcciones sin delegar, por diseño en este tipo de redes los nodos en el núcleo más elevado, que son los que en primera instancia tenían los grandes espacios de direcciones, necesariamente los delegan. Nuevamente hay que recordar que sigue existiendo un componente aleatorio al formar la red por lo cual pueden existir algunos nodos en el núcleo más elevado con menor conectividad.

Figura 4.28: Tamaño promedio de la tabla de *Rendezvous* versus tamaño de la máscara.



En la figura 4.28 podemos observar la relación existente entre el tamaño de la máscara de los nodos con el número de entradas de *Rendezvous* en promedio que existe para ese tamaño de máscara, nuevamente se confirma que los nodos que concentran un gran número de entradas de *Rendezvous* son aquellos con una máscara pequeña, en una red con una topología sin escala estos nodos tienden a no existir, en este caso por la componente aleatoria de generación existen.

Si bien las figuras correspondientes a redes aleatorias y las correspondientes a las redes sin escala presentan una distribución de probabilidades similar, encontramos en el caso de redes sin escala un limite cercano a 10^3 entradas en las tablas de *Rendezvous* y un limite cercano a 10^4 entradas para redes aleatorias de 5000 nodos, recordando que las redes sin escala contienen aproximadamente 8500 nodos vemos en este aspecto un mejor resultado para redes con topologías sin escala.

Capítulo 5

Conclusiones

El objetivo del presente trabajo fué analizar el comportamiento del protocolo ANTop-2, tomando como parámetros el resultado de la implementación de un esquema de ruteo proactivo, sus ventajas y desventajas, y la implementación de los *features* agregados en ANTop-2.

En el presente capítulo presentaremos las conclusiones sobre los resultados obtenidos en el capítulo anterior y el trabajo futuro que se desprende luego de analizar los resultados obtenidos.

5.1. Análisis de Resultados

En el capítulo anterior se desarrollaron las simulaciones ejecutadas sobre la implementación del protocolo proactivo, los mecanismos de recuperación ante fallas y la distribución de información de localización en la red, en la presente sección se toma cada uno de dichos puntos por separado y se presenta el análisis de los resultados obtenidos.

5.1.1. Ruteo Proactivo

La mecánica de trabajo del ruteo proactivo en una primera impresión indicaba que el tamaño de las tablas de ruteo luego de propagar toda la información de ruteo correspondiente, seria de gran tamaño. Aunque experimentalmente se demuestra que en comparación al protocolo reactivo presenta tablas de ruteo de menor tamaño, encontramos la razón justamente en su diseño. El protocolo reactivo agrega una entrada a la tabla de ruteo por cada resolución que debe hacer, mientras que el protocolo proactivo agrega aquellas que son fruto de los algoritmos de distribución y propagación.

El protocolo proactivo mantiene un tamaño de tablas de ruteo acotado a los criterios de distribución y propagación de rutas. El ruteo reactivo implementado en ANTop-1 esta pensado para funcionar en ambientes móviles donde es premisa pensar en un bajo consumo energético y capacidad limitada de memoria, el esquema de ruteo proactivo demuestra que puede funcionar correctamente

con una estructura de ruteo mínima compuesta exclusivamente por la ruta por defecto y las rutas hacia los espacios de direcciones delegados. La limitante de utilizar un esquema de ruteo mínimo es la cantidad de saltos promedio que debe dar cada salto, la cual es conocida de antemano como la distancia en el árbol T entre origen y destino.

La mayor cantidad de rutas que componen la tabla de ruteo pertenece a aquellas que son propagadas por el protocolo de ruteo proactivo, las cuales nacen en la conectividad a través de direcciones secundarias. En el caso de las redes sin escala donde la premisa es lograr la mayor delegación de espacios de direcciones posibles y luego comenzar a obtener conectividad a través de direcciones secundarias, vemos en promedio un número menor de entradas en la tabla de ruteo en comparación con el caso de redes aleatorias.

El mecanismo de distribución de rutas se puede ir aplicando gradualmente para mejorar la eficiencia en el ruteo de paquetes mientras que se controla el crecimiento de las tablas de ruteo. Recordando el algoritmo 6 línea 5, se puede ver que variando la desigualdad pueden regularse la cantidad de rutas que son propagadas y distribuidas.

Con respecto al desempeño logrado en el momento de enrutar paquetes pudimos observar que la distancia recorrida por los paquetes al atravesar la red no supera por mucho la cantidad de saltos ofrecida por un protocolo de ruteo óptimo. Cabe recordar que para poder funcionar de la misma manera que un protocolo de ruteo óptimo deberían mantenerse bases de datos contengan la información completa y detallada de toda la red y sus enlaces.

A fin de graduar el desempeño del protocolo vimos que variando el algoritmo 6 se puede lograr que el mismo disminuya la cantidad de rutas que son redistribuidas y propagadas por la red, en consecuencia disminuyendo su desempeño. Una forma de aumentar el desempeño del protocolo es variar el criterio de adyacencia en el hipercubo, en este caso se plantea que 2 nodos son adyacentes en el hipercubo si sus direcciones distan solamente en un bit. Cambiando la cantidad de bits que se requieren para ser adyacentes en el hipercubo se logra establecer relaciones mediante direcciones secundarias en una medida superior, aumentando en consecuencia el tamaño de las tablas de ruteo.

5.1.2. Recuperación ante fallas

En el caso de los mecanismos de recuperación ante fallas es necesario analizar la medida en la que crecen las tablas de ruteo, el desempeño en el envío de paquetes una vez que la red se recupera y la medida en que la red soporta recuperarse frente a la caída de nodos.

Analizando los resultados se ve que en promedio el crecimiento de las tablas de ruteo es despreciable, y que la cantidad de rutas que se desactivan debido a los protocolos de recuperación ante fallas no es despreciable. Las rutas que componen las tablas de ruteo como vimos anteriormente son aquellas propagadas debido a la existencia de conexiones secundarias entre los nodos, por lo cual la

caída de un nodo implica que el mismo ya no podrá ser utilizado por otros como un *gateway* para los prefijos que antes era utilizado, entonces las rutas para las que servía como *gateway* deben ser retraídas, esta es la razón de la cantidad de rutas desactivadas luego del accionar de los mecanismos de recuperación.

Los mecanismos de recuperación se basan en la existencia de rutas secundarias, por lo cual aquellos nodos que detectaron la caída de su predecesor y tenían una dirección secundaria hacia una rama fuera del sucesor caído, directamente encuentran en si un nuevo *Default Gateway*, luego no existe necesidad de notificar a sus sucesores sobre la caída del predecesor. Esta es la razón por la cual no existe un gran número de rutas agregadas consecuencia de los mecanismos de recuperación ante fallas. Por otro lado las redes que siguen una topología sin escalas se ven afectadas por el hecho de basar el funcionamiento del mecanismo de recuperación ante fallas en la existencia de rutas secundarias, ya que se comprueba experimentalmente que la cantidad de enlaces secundarios que presenta es notablemente menor que los presentes en redes aleatorias.

Como consecuencia de la baja cantidad de rutas agregadas notamos que no existe una variación notable en la cantidad promedio de saltos que dan los paquetes hasta llegar a destino, dado que en promedio aquellos paquetes que logran ser cursados exitosamente darán solo 1 salto hasta llegar a la nueva rama que los transportará hasta el destino, siendo diferente solamente la distancia que recorren en la nueva rama.

A fin de verificar la manera en la que los mecanismos de recuperación ante fallas responden a la caída de nodos se introduce al cálculo del grado de separación, siendo esta una métrica concebida específicamente para este trabajo de tesis. El grado de separación representa correctamente el desempeño de la red, tendiendo al valor 1 en aquellos escenarios que la red funciona sin problemas, y tendiendo a infinito cuando una red colapsa, considerando la cantidad y el tamaño de las particiones resultantes y el porcentaje de paquetes que exitosamente son cursados en la red. La medición del grado de separación nos indica que la red no puede recuperarse en forma satisfactoria más allá del 2 % de fallas de nodos.

5.1.3. Distribución de información de localización

En cuanto a la distribución de información el objetivo del presente trabajo fue proveer de robustez ante el abandono repentino de nodos, manteniendo el concepto original de no generar nodos concentradores de información. Se toma como premisa la equidad entre nodos, siendo todos de igual importancia en el funcionamiento del protocolo.

Experimentalmente podemos ver que a medida que aumenta la cantidad de nodos *Rendezvous* en el sistema, la cantidad promedio de entradas en los nodos no aumenta en forma lineal, con lo cual se demuestra claramente la no existencia de nodos concentradores. Sin embargo se ve la existencia de nodos con una cantidad notablemente mayor de entradas en sus tablas de *Rendezvous* lo cual se explica analizando la máscara de dichos nodos, aquellos nodos que no han

delegado por completo su espacio de direcciones presenta un gran número de entradas en sus tablas. La mecánica de distribución prueba ser correcta cuando hablamos de un hipercubo completo, aunque en los casos en que el hipercubo es incompleto puede darse que existan nodos con gran cantidad de entradas. El hecho que un nodo tenga una gran cantidad de entradas lo convierte en un punto de fallas importante en la red, aunque dado el mecanismo de replicación de información utilizado, el único impacto real es en la carga de procesamiento y almacenamiento que se impone en dichos nodos.

La premisa de construcción de las redes cuya topología sigue un patrón aleatorio impide pensar en la creación de un mecanismo que pueda predecir la forma en que la red será constituida y que distribuya la información de localización basado en dichas asunciones. En cambio en el caso de las redes que siguen una topología sin escala se demuestra que es posible distribuir correctamente la información, aunque debido a la componente aleatoria en el proceso de creación de las redes sin escala se encuentran nodos que no delegan por completo su espacio de direcciones, y en consecuencia contienen un gran número de entradas en sus tablas de *Rendezvous.*

5.2. Trabajo futuro

Los algoritmos propuestos aportan robustez y resistencia ante fallas al protocolo ANTop, aunque es necesario destacar que queda trabajo a ser realizado dentro del mismo, sea en características no desarrolladas o limitaciones del algoritmo actual, las mismas a saber al momento de escribir el trabajo presente son:

- Tratamiento y reordenamiento de direcciones en caso de *split/merge* de 2 o más redes.
- Alcance de algoritmo de detección de *gateway* alternos en caso de múltiples particiones simultáneas sobre la misma subrama, el algoritmo actual soporta una sola partición y no guarda registro de todas las rutas propagadas por la primera detección de fallas.
- Definir un algoritmo de ruteo dinámico, donde la red es capaz de utilizar protocolo reactivo o proactivo dependiendo de la topología y movilidad actual en los nodos.
- Permitir la asociación de enlaces secundarios mediante un criterio dinámico, de existir poca conectividad mediante enlaces secundarios, la red debería poder elegir utilizar asociaciones mediante 2, 3 ,4 o n bits de similitud entre las direcciones.
- Definir un algoritmo de distribución de información de localización que tenga en consideración los resultados de la presente tesis.
- Una vez que un nodo adyacente pierde contacto, las rutas que el mismo propagó son desactivadas, una vez que el nodo vuelve a estar en contacto dichas rutas no vuelven a activarse dado que no se sabe cual es el alcance

de la misma, podría suceder que un predecesor del nodo adyacente también haya tenido problemas, y volver a analizar todos los prefijos posibles implicaría un recorrido de gran parte de la red y un análisis del alcance de cada prefijo.

- Respecto de la utilización de un hipercubo para el ruteo, una manera de mejorar este mapeo y lograr que más cantidad de conexiones físicas se conviertan en aristas del hipercubo, es asignando múltiples direcciones a algunos nodos. Dado que dos nodos pueden no ser vecinos en la estructura del hipercubo aunque estén físicamente conectados, esto nos ofrece mejor representación de las adyacencias físicas.

Capítulo 6

Anexo 1

En este anexo se presentan todas las gráficas correspondientes a las simulaciones propuestas en esta tesis, en los capítulos previos la mayoría no se agrego dado que se decidió poner gráficas que resuman la situación completa, en este anexo se presenta toda la información a fin de que el lector tenga la visión completa de toda la solución.

6.1. Redes Aleatorias

En la presente sección se presentan los resultados obtenidos en redes aleatorias en cuanto al tamaño de las tablas de ruteo en el protocolo proactivo luego de aplicar los mecanismos de fallas, la distancia recorrida por los paquetes, la relación entre la distancia mínima y distancia recorrida.

6.1.1. Tamaño de tablas de ruteo

Una vez que sucede el abandono repentino de nodos en la red se disparan los mecanismos de recuperación ante fallas, luego del cual las tablas de ruteo sufren cambios quedando algunas rutas inactivas y otras rutas activas, las siguientes graficas presentan la distribución de probabilidad para las rutas activas e inactivas para redes de 500, 1000 y 5000 nodos en los casos de falla de 0.1, 0.2, 0.5, 1 y 2 % de los nodos.

Figura 6.1: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 500 nodos con 0.1 % de fallas .

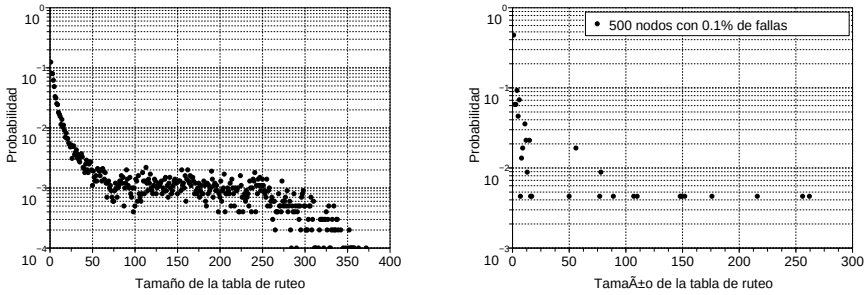


Figura 6.2: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 500 nodos con 0.2 % de fallas .

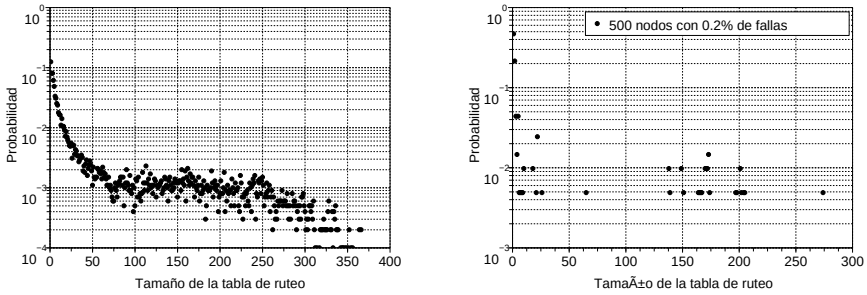


Figura 6.3: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 500 nodos con 0.5 % de fallas .

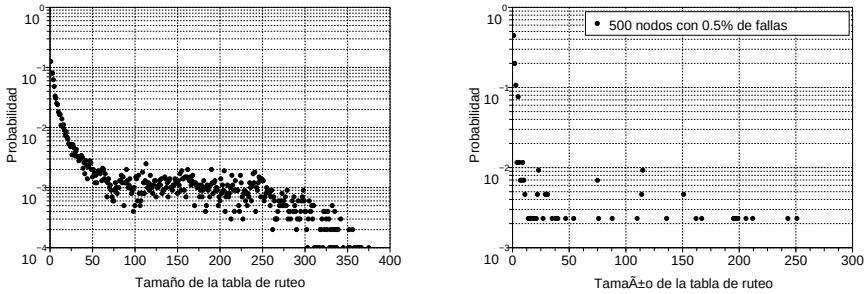


Figura 6.4: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 500 nodos con 1 % de fallas .

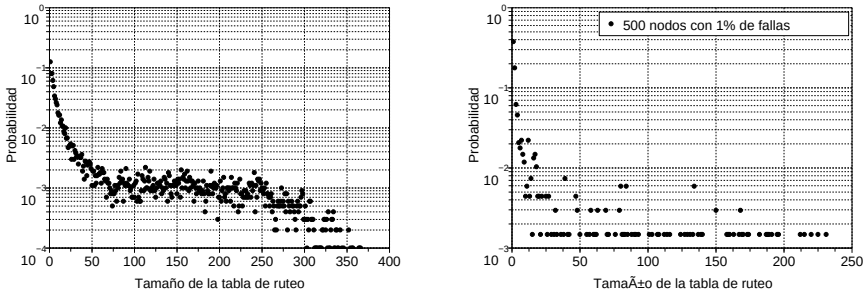


Figura 6.5: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 500 nodos con 2 % de fallas .

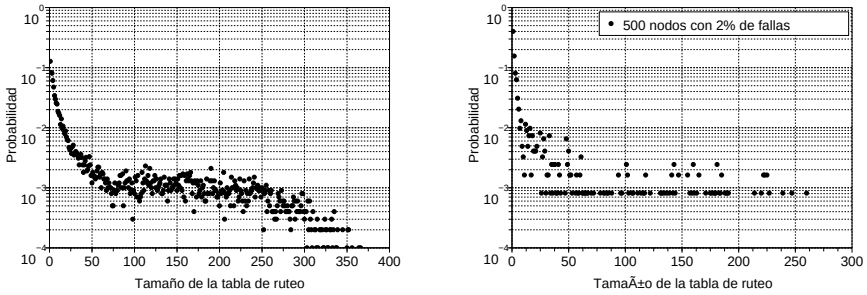


Figura 6.6: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 1000 nodos con 0.1 % de fallas .

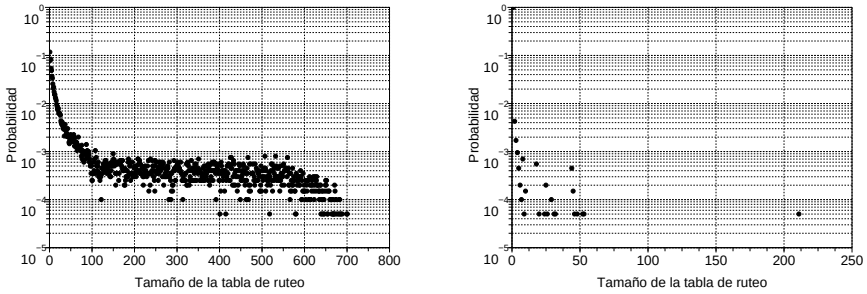


Figura 6.7: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 1000 nodos con 0.2 % de fallas .

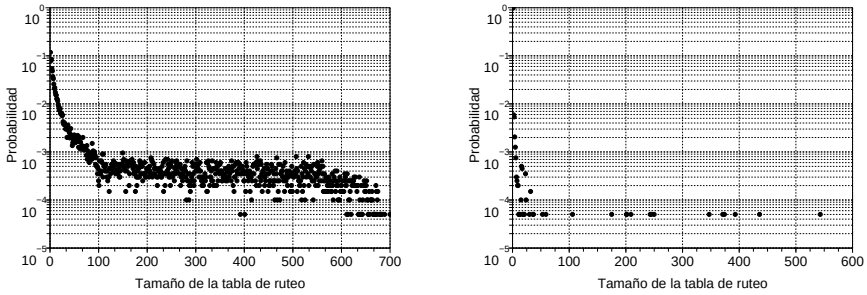


Figura 6.8: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 1000 nodos con 0.5 % de fallas .

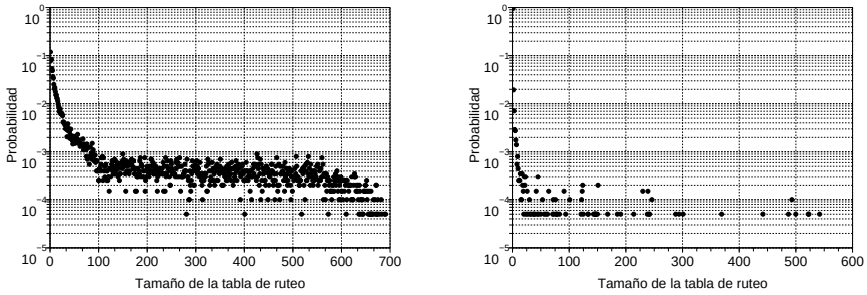


Figura 6.9: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 1000 nodos con 1 % de fallas .

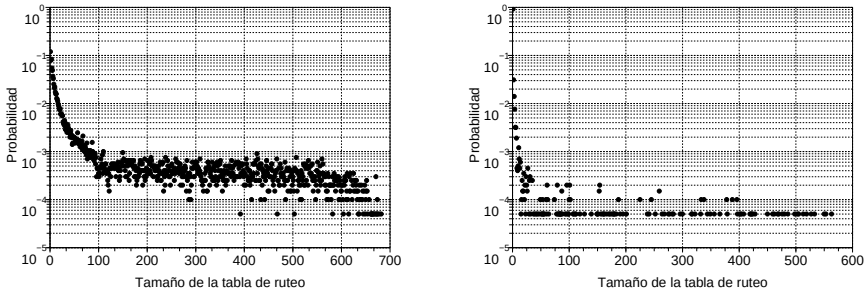


Figura 6.10: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 1000 nodos con 2 % de fallas .

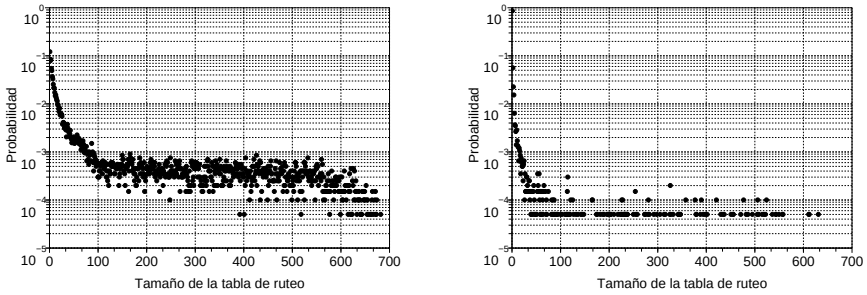


Figura 6.11: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 5000 nodos con 0.1 % de fallas .

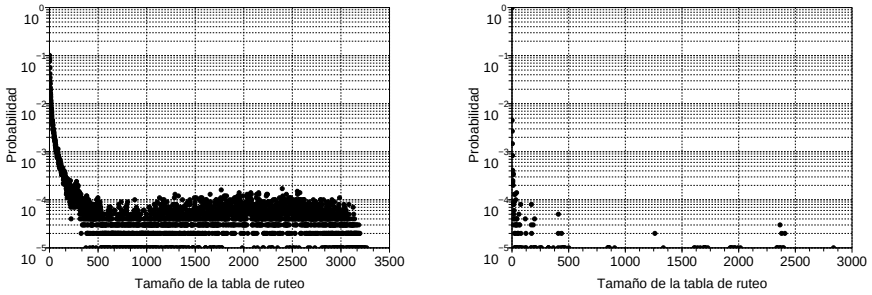


Figura 6.12: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 5000 nodos con 0.2 % de fallas .

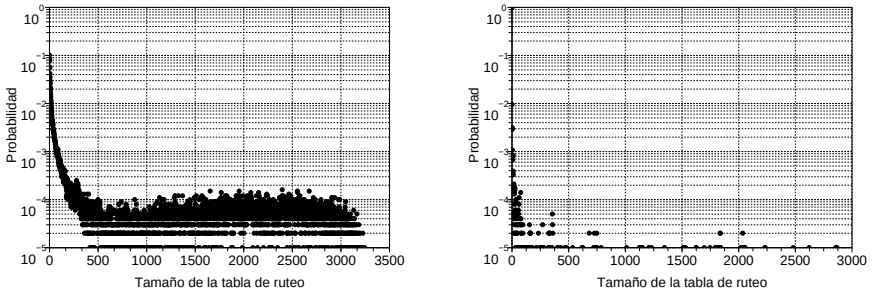


Figura 6.13: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 5000 nodos con 0.5 % de fallas .

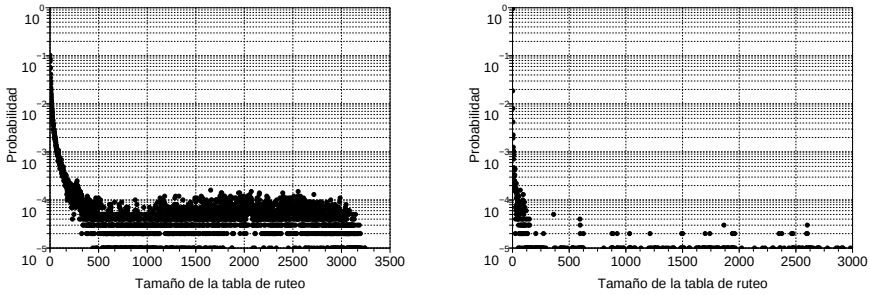


Figura 6.14: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 5000 nodos con 1 % de fallas .

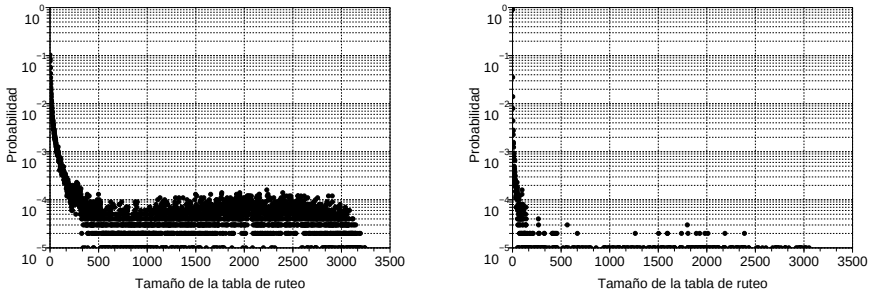
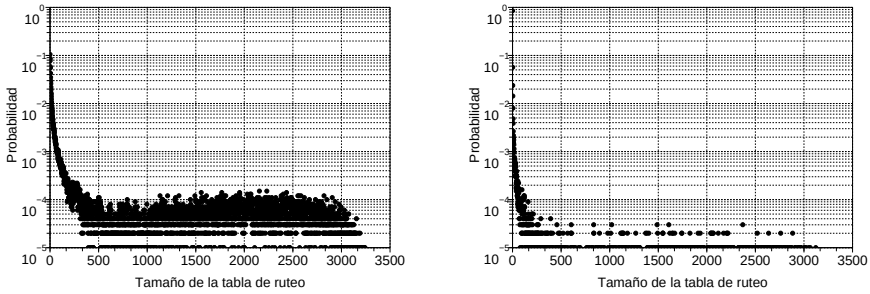


Figura 6.15: Distribución de cantidad de entradas activas(izquierda) e inactivas(derecha) en la tabla de ruteo para redes de 5000 nodos con 2 % de fallas .



6.1.2. Distancia recorrida

En esta sección se presentan los histogramas de las distancias recorridas por aquellos paquetes que logran enrutarse correctamente en todo su camino, para redes de 500 y 1000 nodos en los casos de falla de 0.1, 0.2, 0.5, 1 y 2 % de los nodos.

Figura 6.16: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, y 1000 nodos luego de enviar 500 paquetes, habiendo fallado el 0.1 % de los nodos.

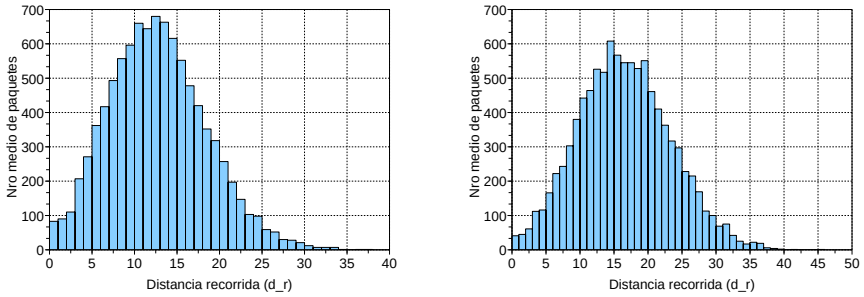


Figura 6.17: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, y 1000 nodos luego de enviar 500 paquetes, habiendo fallado el 0.2 % de los nodos.

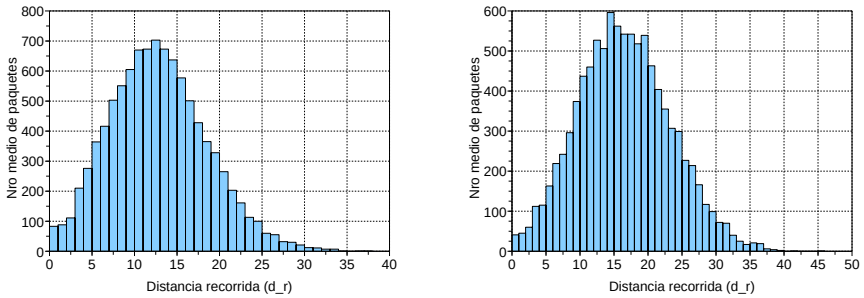


Figura 6.18: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, y 1000 nodos luego de enviar 500 paquetes, habiendo fallado el 0.5 % de los nodos.

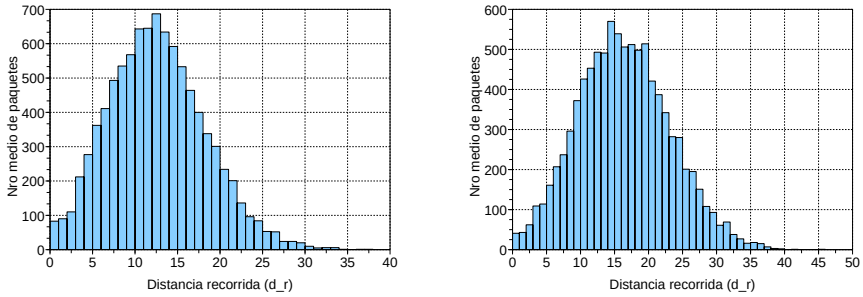


Figura 6.19: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, y 1000 nodos luego de enviar 500 paquetes, habiendo fallado el 1 % de los nodos.

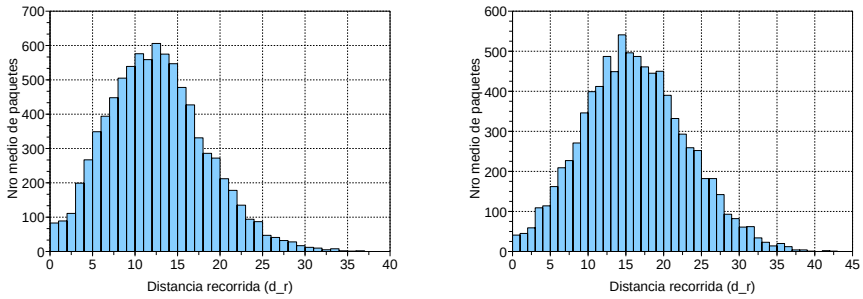
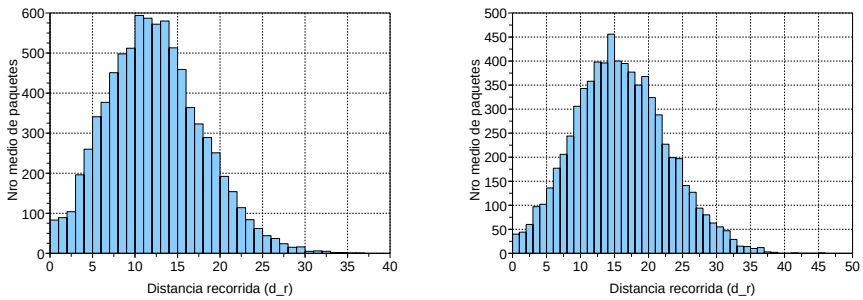


Figura 6.20: Histograma de ocurrencia de las distancias recorridas por los paquetes en redes de 500, y 1000 nodos luego de enviar 500 paquetes, habiendo fallado el 2 % de los nodos.



6.1.3. Relación distancia mínima y distancia recorrida

Las gráficas de la presente sección presentan la relación entre la distancia mínima y la distancia recorrida para redes de 500, 1000 y 5000 nodos.

Figura 6.21: Distancia recorrida versus distancia mínima, para 0.2, 0.5 y 1 % de fallas respectivamente, en una red de 500 nodos respectivamente. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.

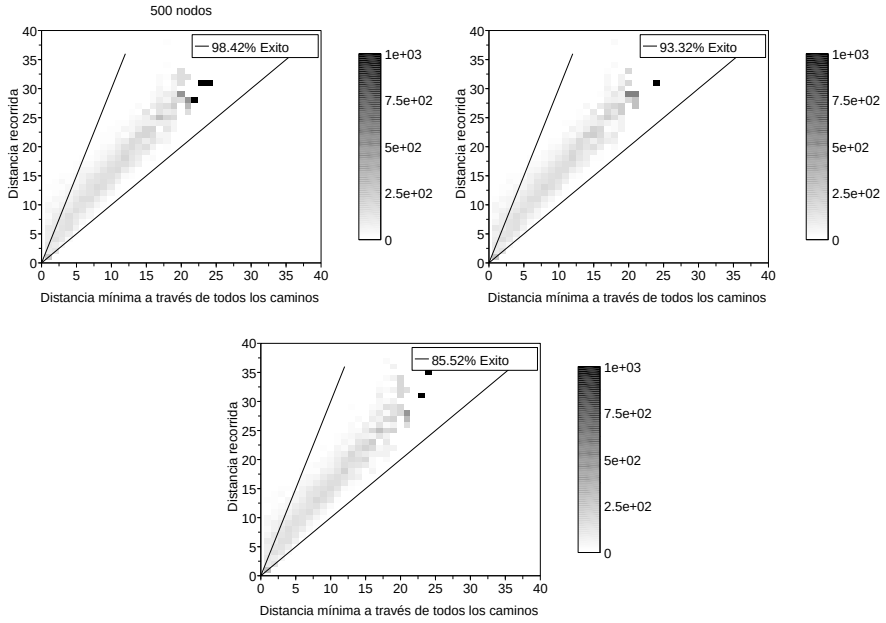


Figura 6.22: Distancia recorrida versus distancia mínima, para 0.2, 0.5 y 1 % de fallas respectivamente, en una red de 1000 nodos respectivamente. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.

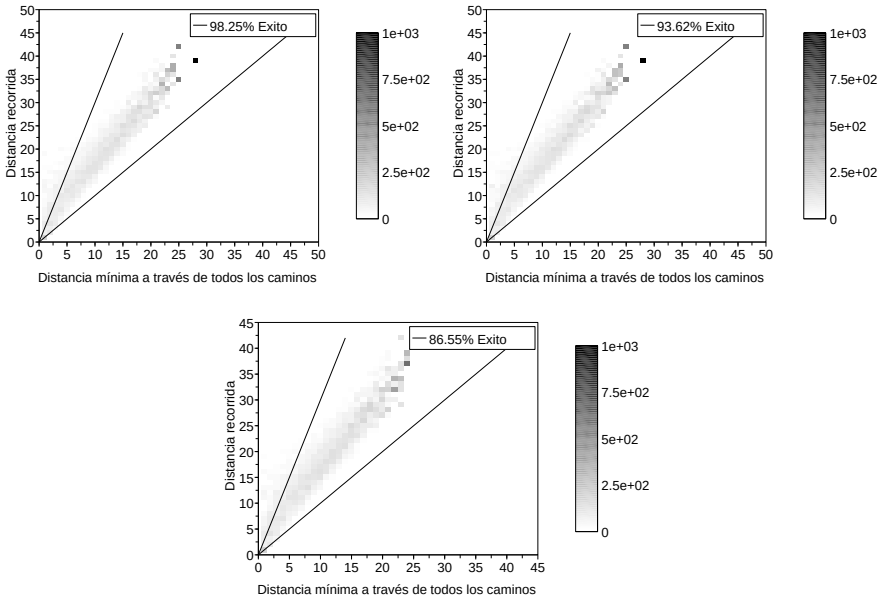
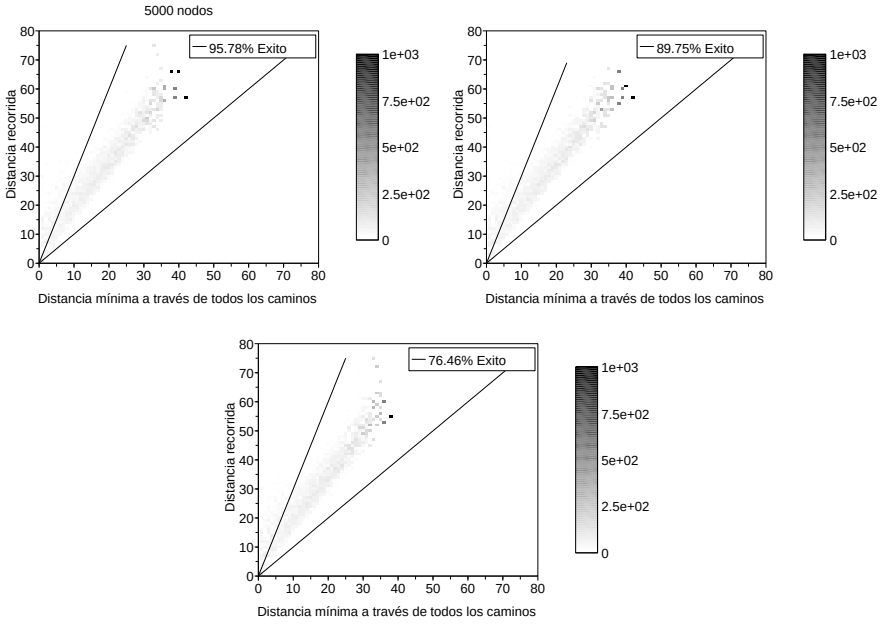


Figura 6.23: Distancia recorrida versus distancia mínima, para 0.2, 0.5 y 1 % de fallas respectivamente, en una red de 5000 nodos respectivamente. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo.



6.2. Redes sin escala

En la presente sección se presentan los resultados obtenidos en redes con topologías sin escala en cuanto a la distancia recorrida por los paquetes, la relación entre la distancia mínima y distancia recorrida.

6.2.1. Distancia recorrida

En esta sección se presentan los histogramas de las distancias recorridas por aquellos paquetes que logran enrutarse correctamente en todo su camino, en los casos de falla de 0.1, 0.2, 0.5 y 1 % de los nodos.

Figura 6.24: Histograma de ocurrencia de las distancias recorridas por los paquetes en una red donde falla el 0.1 % de los nodos.

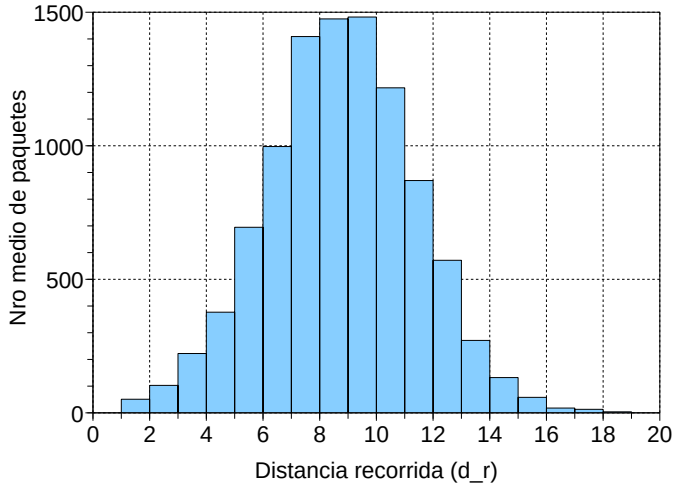


Figura 6.25: Histograma de ocurrencia de las distancias recorridas por los paquetes en una red donde falla el 0.2 % de los nodos.

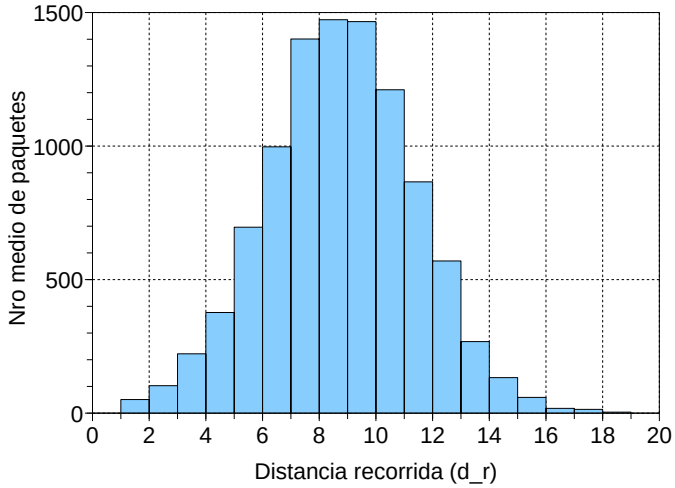


Figura 6.26: Histograma de ocurrencia de las distancias recorridas por los paquetes en una red donde falla el 0.5 % de los nodos.

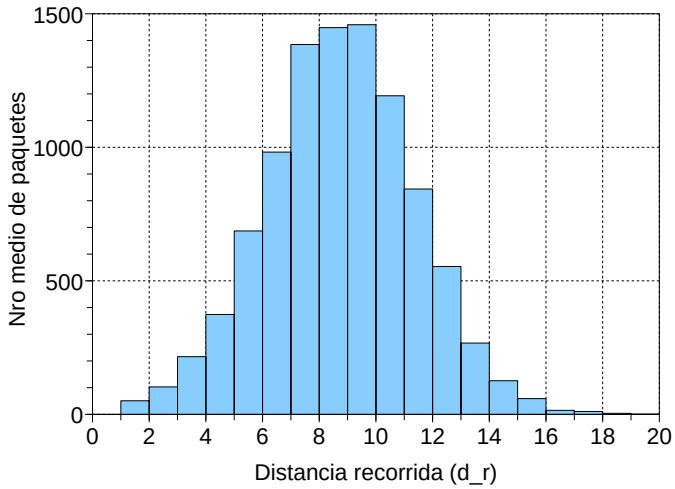
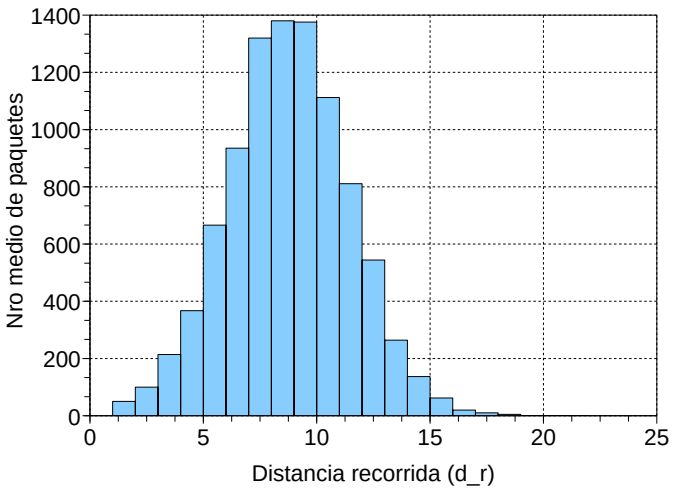


Figura 6.27: Histograma de ocurrencia de las distancias recorridas por los paquetes en una red donde falla el 1 % de los nodos.



6.2.2. Relación distancia mínima y distancia recorrida

Las gráficas de la presente sección presentan la relación entre la distancia mínima y la distancia recorrida.

Figura 6.28: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos en una red donde falla el 0.1 % de los nodos.

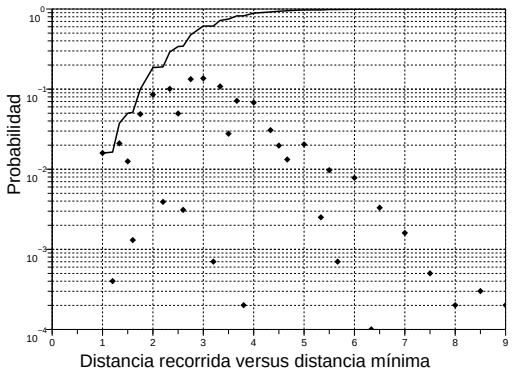


Figura 6.29: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos en una red donde falla el 0.2 % de los nodos.

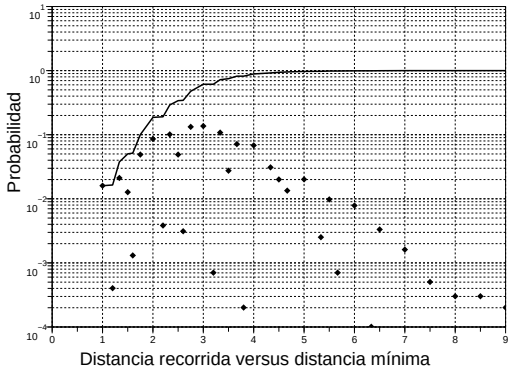


Figura 6.30: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos en una red donde falla el 0.5 % de los nodos.

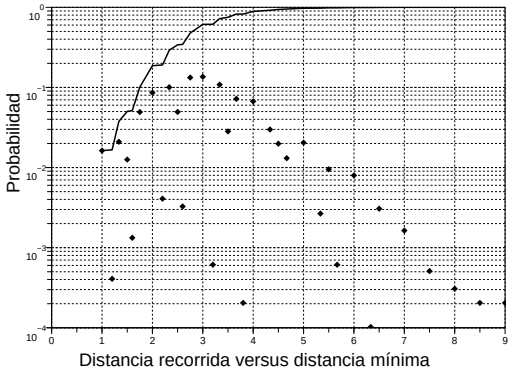


Figura 6.31: Distribución de probabilidad y probabilidad acumulada de la relación entre distancia recorrida y la distancia mínima a través de todos los nodos en una red donde falla el 1 % de los nodos.

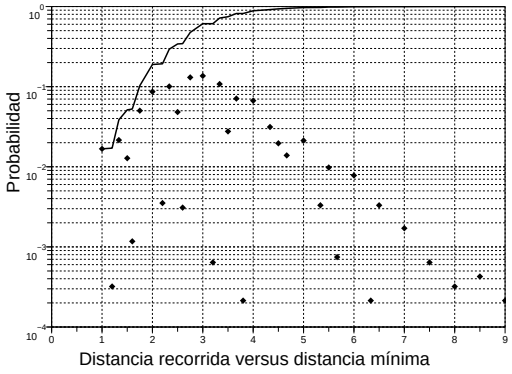


Figura 6.32: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo en una red donde falla el 0.1 % de los nodos.

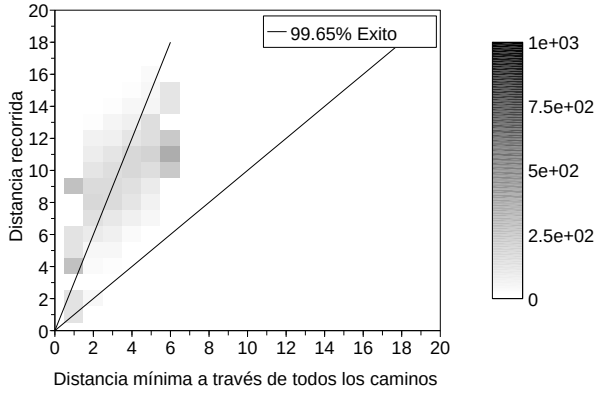


Figura 6.33: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo en una red donde falla el 0.2 % de los nodos.

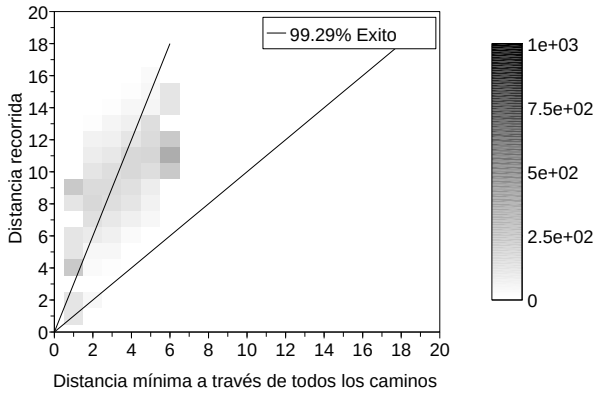


Figura 6.34: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo en una red donde falla el 0.5 % de los nodos.

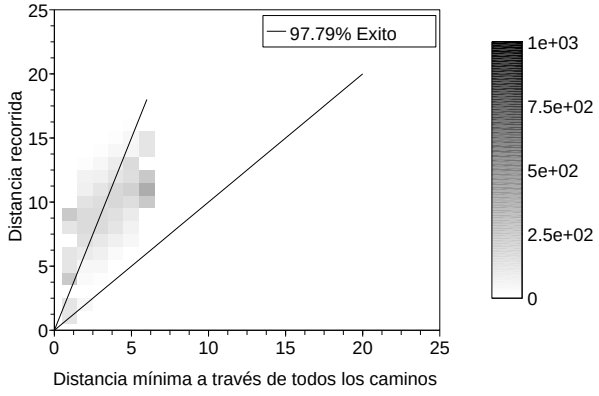
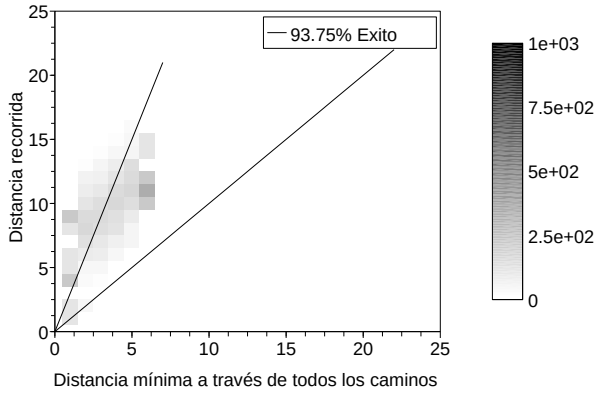


Figura 6.35: Grafica de dispersión en la relación distancia recorrida versus distancia mínima a través de todos los enlaces. Se presentan 2 rectas comparativas una de pendiente 1 representando un algoritmo de ruteo ideal y otra de pendiente 3 representando el *strech* teórico máximo de un algoritmo de ruteo en una red donde falla el 1 % de los nodos.



Bibliografía

[1] Jose I. Alvarez-Hamelin , Aline C. Viana , and Marcelo D. de Amorim "DHT-based Functionalities Using Hypercubes",*IFIP International Federation for Information Processing* Publisher: Springer Boston, ISSN: 1571-5736 (Print) 1861-2288 (Online) Volume: Volume 212/2006, Book:Ad-Hoc Networking, Pages 157-176

[2] Alejandro Marcu, "Desarrollo y simulacion de un protocolo para redes ad-hoc", Julio 2007.

[3] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, "Peernet: Pushing peer-to-peer down the stack," *Proceedings of International Workshop on Peer-To-Peer Systems (IPTPS'03)*, Feb. 2003.

[4] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 17–32, Feb. 2003.

[5] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proceedings of ACM MOBICOM'00*, Aug. 2000.

[6] Y. Xue, B. Li, and K.Ñahrstedt, "A scalable location management scheme in mobile ad-hoc networks," in *Proceedings of IEEE Conference on Local Computer Networks (LCN)*, (Tampa, FL, USA), Nov. 2001.

[7] B. Chen and R. Morris, "L+: Scalable landmark routing and address lookup for multi-hop wireless networks," tech. rep., Massachusetts Institute of Technology, Cambridge, Massachusetts - MIT LCS Technical Report 837 (MIT-LCS-TR-837), Mar. 2002.

[8] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, "Scalable ad hoc routing: The case for dynamic addressing," in *Proceedings of IEEE INFOCOM'04*, (Hong Kong), Mar. 2004.

[9] A. C. Viana, M. D. Amorim, S. Fdida, and J. F. Rezende, "Indirect routing using distributed location information," *ACM Wireless Networks*, vol. 10, no. 6, pp. 747–758, Dec. 2004.

[10] Aline Carneiro Viana - Marcelo Dias de Amorim - Serge Fdida - Jose Ferreira de Rezende "Self-organization in spontaneous networks: the approach of DHT-based routing protocols", *Networks Journal, 2005*

[11] Ye Xia, Shigang Chen and Vivekanand Korgaonkar, Computer and Information Science and Engineering Department, "Load Balancing with Multiple Hash Functions in Peer-to-Peer Networks" *ICPADS archive, Proceedings of the 12th International Conference on Parallel and Distributed Systems - Volume 1 table of contents* , Pages: 411 - 420 , Year of Publication: 2006 , ISBN ISSN:1521-9097 , 0-7695-2612-8

[12] Mema Roussopoulos - Mary Baker, "Practical load balancing for content requests in peer-to-peer networks", *eprint arXiv:cs/0209023* Publication Date: 09/2002 Origin: ARXIV

[13] Christos Gkantsidis, John Miller, Pablo Rodriguez, "Anatomy of a P2P Content Distribution system with Network Coding", *In IPTPS, 2006*

[14] David R. Karger - MIT, Matthias Ruhl - IBM Almaden "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems",*ACM Symposium on Parallel Algorithms and Architectures archive Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures table of contents* Barcelona, Spain , SESSION: Peer-To-peer systems table of contents Pages: 36 - 43 , Year of Publication: 2004, ISBN:1-58113-840-7

[15] sonesh surana - brighten godfrey - karthik lakshminarayanan - Richar karp - Ion Stoica "Load balancing in dynamic structured Peer-To-Peer systems", *Performance Evaluation Volume 63 , Issue 3 (March 2006) P2P computing systems, Pages: 217 - 240 Year of Publication: 2006 ,ISSN:0166-5316*

[16] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, 'Chord: a scalable peer-to-peer lookup protocol for internet applications,' IEEE/ACM Transactions on Networking, vol. 11, no. 1, pp. 17?32, Feb. 2003.

[17] L. Blazevic, L. Buttyan, S. G. S. Capkun, J. P. Hubaux, and J. Y. L. Boudec, 'Self-organization in mobile ad-hoc networks: the approach of terminodes,' IEEE Computer Communications Magazine, June 2001.

[18] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, 'A scalable location service for geographic ad hoc routing,' in Proceedings of ACM MOBI- COM'00, Aug. 2000.

[19] Y. Xue, B. Li, and K. Nahrstedt, 'A scalable location management scheme in mobile ad-hoc networks,' in Proceedings of IEEE Conference on Local Computer Networks (LCN), (Tampa, FL, USA), Nov. 2001.

[20] B. Chen and R. Morris, 'L+: Scalable landmark routing and address lookup for multi-hop wireless networks,' tech. rep., Massachusetts Institute of Technology, Cambridge, Massachusetts - MIT LCS Technical Report 837 (MIT-LCS-TR-837), Mar. 2002.

[21] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, 'Scalable ad hoc routing: The case for dynamic addressing,' in Proceedings of IEEE INFOCOM'04, (Hong Kong), Mar. 2004.

[22] A. C. Viana, M. D. Amorim, S. Fdida, and J. F. Rezende, 'Indirect routing using distributed location information,' ACM Wireless Networks, vol. 10, no. 6, pp. 747?758, Dec. 2004.

[23] A. C. Viana, M. D. Amorim, S. Fdida, and J. F. Rezende, 'Self-organization in spontaneous networks: the approach of dht-based routing protocols.' to appear in Ad Hoc Networks Journal, 2005.

[24] . J. P. Hubaux, T. Gross, J. Y. L. Boudec, and M. Vetterli, 'Towards self-organized mobile ad hoc networks: the terminodes project,' IEEE Communications Maga- zine, vol. 39, no. 1, pp. 118?124, Jan. 2001.

[25] Terminodes Project. http://www.terminodes.com/.

[26] Grid Project. http://www.pdos.lcs.mit.edu/grid/.

[27] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, 'Peernet: Pushing peer-to-peer down the stack,' Proceedings of International Workshop on Peer-To-Peer Systems (IPTPS'03), Feb. 2003.

[28] P. F. Tsuchiya, 'The landmark hierarchy: a new hierarchy for routing in very large networks,' in Proceedings of ACM SIGCOMM'88, Aug. 1988.

[29] P. F. Tsuchiya, 'Landmark routing: Architecture, algorithms and issues,' tech. rep., MTR-87W00174, MITRE Corporation, Sept. 1987.

[30] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva, 'A performance comparison of multi-hop wireless ad hoc network routing protocols,' in Proceed- ings of ACM MOBICOM'98, Oct. 1998.

[31] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, 'The broadcast storm problem in a mobile ad hoc network,' in Proceedings of ACM MOBICOM'99, pp. 152?162, Aug. 1999.

[32] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, 'Internet indirection infrastructure,' in Proceedings of ACM SIGCOMM'02, Aug. 2002.

[33] C. E. Perkins and P. Bhagwat, 'Highly dynamic destination sequenced distance- vector routing (dsdv) for mobile computers,' in Proceedings of ACM SIG- COMM'94, Oct. 1994.

[34] Y. Saad, 'Data communication in hypercubes,' tech. rep., Research Report 428, Department of Computer Science, Yale University, New Haven, CT, 1985.

[35] F. T. Leighton, Introduction to parallel algorithms and architectures: array, trees, hypercubes. Morgan Kaufmann Publishers Inc. San Francisco, CA, US, 1991.

[36] E. Oh and J. Chen, 'Parallel routing in hypercube networks with faulty nodes,' in IEEE International Conference on Parallel and Distributed Systems (ICPADS '01), pp. 338?345, July 2001.

[37] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl, 'Hypercup - hypercubes, ontolo- gies, and efficient search on peer-to-peer networks,' in Agents and Peer-to-Peer Computing: A Promising Combination of Paradigms, LNCS 2530, pp. 112?124, July 2003.

[38] M. A. Jimenez-Montano, C. R. de la Mora-Basanez, and T. Poeschel, 'On the hypercube structure of the genetic code,' in Proceedings of Bioinformatics and Genome Research, pp. 445?459, Oct. 1994.

[39] D. Wang, 'A low-cost fault-tolerant structure for the hypercube,' Journal of Supercomputing, vol. 20, no. 3, Nov. 2001.

[40] R. Friedman, S. Manor, and K. Guo, 'Scalable stability detection using logical hypercube,' tech. rep., Technion, Department of Computer Science Technical Re- port 0960, May 1999.

[41] J. Slack, 'Visualization of embedded binary trees in the hypercube,' tech. rep., Final Report of the Project for Information Visualization, Department of Com- puter Science, University of British Columbia, Apr. 2003.

[42] Matthew Caesar,Miguel Castro,Edmund B. Nightingale,Greg O'Shea,Antony Rowstron, 'Virtual Ring Routing: Network Routing Inspired by DHTs', Microsoft Research Cambridge, UK. University of California Berkeley, USA. University of Michigan Ann Arbor, USA.

[43] ISO, "Intermediate System to Intermediate System Intra-Domain Routeing Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-mode Network Service (ISO 8473)", International Standard 10589:2002, Second Edition.

[44] RFC 1247, J. Moy, Proteon, Inc., July 1991.

[45] Hedrick, C., "Routing Information Protocol", STD 34, RFC 1058, Rutgers University, June 1988.

[46] V. Batagelj and M. Zaversnik , "Generalized Cores",CoRR, cs.DS/0202039, 2002, http://arxiv.org/abs/cs.DS/0202039,DBLP, http://dblp.uni-trier.de.

[47] M. Thorup and U. Zwick, "Compact routing schemes", In Proc. of SPAA, Jul. ,2001.

[48] L. Cowen, "Compact routing with minimum stretch", J. of Algorithms, 2001.