Algoritmos de Búsqueda y Actualización de Información para ruteadores IP

Para obtener el grado de
Maestro en Ciencias
(Ciencias y Tecnologías de la Información)

P R E S E N T A

Ing. Joel Yazbek Buendía Gómez

A S E S O R E S

Dr. Miguel Ángel Ruíz Sánchez
Dr. César Jalpa Villanueva

S I N O D A L E S

PRESIDENTE: Dr. Javier Gómez Castellanos
SECRETARIO: Dr. César Jalpa Villanueva
VOCAL: Dr.Ricardo Marcelín Jiménez
VOCAL: Dr. Miguel López Guerrero

30 de Julio de 2009

Dedicado a

M I F A M I L I A

Mi primera parte Angélica.

Mis Hijas Danaé, Sophia.

Mi madre EVA un gran nombre y un MAGNIFICO ser humano.

Mis hermanos Nayelly, Citlali, Jeyel.

Mis sobrinos Gadiel, y los que vengan.

La vida que he vivido.

Algoritmos de Búsqueda y Actualización de la Información para ruteadores IP

Ing. Joel Yazbek Buendía Gómez

Para obtener el grado de:

Maestro en Ciencias y Tecnologías de la Información
Junio 2009.

Resumen

En esta tesis se propone un nuevo algoritmo de búsqueda para encontrar el prefijo de red que coincida con los bits más significativos de la dirección IP de destino y además sea el más largo que exista en la tabla de ruteo. En el esquema del protocolo IP, después de quitar las clases e implementar CIDR, aparece el problema de no tener información acerca de la longitud del prefijo de red. El algoritmo UAM propone que el problema de buscar prefijos de red se cambie por una búsqueda exacta, es decir, convertimos la búsqueda de prefijos aplicando una función que cambia nuestro espacio de búsqueda. La idea principal es que los prefijos de red según su longitud pueden ser representados como ángulos en una circunferencia. Y ahora cuando se busca el prefijo más largo para una dirección IP de destino solamente se debe buscar el ángulo que le corresponde en un arreglo ordenado. La búsqueda de los prefijos y las estructuras de datos para organizar los prefijos son sencillas y de funcionamiento básico. Esta es la bondad de la primera versión del algoritmo UAM. Se utiliza el método de simulación para poder obtener el tiempo que se tarda el algoritmo UAM y compararlo con dos algoritmos clásicos de solución a este problema, pero la facilidad del manejo de las estructuras de datos es lo relevante de nuestra primera propuesta. Por lo tanto el algoritmo UAM es una solución al problema tratado en esta tesis.

Agradecimientos

A mi Familia, por ESTAR conmigo SIEMPRE.

A mis Asesores y Amigos Dr. Miguel Ángel y Dr. Cesar Gracias.

A mis profesores, de la Universidad Autónoma Metropolitana.

A mis coordinadores por su guía y apoyo, Dra. Elizabeth Pérez Cortés y Dr. Ricardo Marcelin Jiménez.

A todos los amigos y compañeros de la MCyTI.

Agradezco a las instituciones por los apoyos recibidos y bien gastados:

A la Universidad Autónoma Metropolitana

A la SEP - DGETI, COSDAC

Al CETis 39

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se presenta un panorama general de las redes de computadoras que permitirá ubicar el tema de esta tesis en el contexto de la Internet. En particular debemos conocer los sistemas de interconexión de redes, saber quién los norma, los protocolos involucrados y quienes los utilizan. Iniciamos con una descripción de lo que son las redes de computadoras, en seguida, presentamos el modelo OSI para después hablar del protocolo IP y finalmente trataremos lo correspondiente a los ruteadores.

1.1. Redes de computadoras

Las redes de computadoras nacen por la necesidad de compartir información y recursos. Una red de computadoras se define como: *"Conjunto de ordenadores o de equipos informáticos conectados entre sí que pueden intercambiar información"*[1].

A partir de los años '60 se comienza a investigar la forma de cómo establecer normas para la comunicación entre computadoras. Dentro de las redes de comunicación se propone utilizar la conmutación de paquetes, ya que Kleinrock demuestra que es la mejor opción para este tipo de comunicaciones. A partir de los años '70

[1]Definición de red décima acepción real academia de la lengua española, www.rae.es

en Estados Unidos se vio la necesidad de poder mantener un sistema de comunicaciones que tuviera la capacidad de poder seguir funcionando a pesar de posibles fallas en varias partes del sistema. A esta necesidad se encontró una solución, una interconexión de redes, es decir, una red de redes, este es el nacimiento de Internet.

Las reglas fundamentales para Internet fueron:

- Cada red distinta debería mantenerse por sí misma y no deberían requerirse cambios internos a ninguna de ellas para conectarse a Internet.
- Las comunicaciones deberían ser establecidas con base en la filosofía del mejor esfuerzo (best-effort). Si un paquete no llega a su destino debe ser retransmitido desde el emisor.
- Para interconectar redes se usarían cajas negras.
- No habría ningún control global a nivel de operaciones.

Internet se define como: *"Red informática mundial, descentralizada, formada por la conexión directa entre computadoras u ordenadores mediante un protocolo especial de comunicación"*[2].

Cada red puede utilizar diferentes tecnologías como canales de comunicación, desde cable de cobre hasta medios inalámbricos, y además puede tener diferentes topologías.

Los protocolos utilizados en las comunicaciones comienzan a normarse o estandarizarse para poder comunicar redes de diferentes tecnologías, el modelo de referencia, base de esta estandarización, se explica en la siguiente sección.

1.2. Modelo OSI

Las comunicaciones a través de las redes se organizan en una serie de capas con objeto de reducir la complejidad de su diseño, entendiéndose por "capa" una entidad que es capaz por sí misma de realizar una función específica. Cada una de

[2]Definición de Internet real academia de la lengua española, www.rae.es

estas capas se construye sobre la anterior. El número de capas, el nombre, contenido y función de cada una varían de una red a otra.

El propósito de una capa es ofrecer ciertos servicios a las capas superiores, liberándolas del conocimiento detallado sobre cómo se realizan dichos servicios.

La capa n en una computadora conversa con la capa n de otra computadora. Las reglas y normas utilizadas en esta conversación se conocen conjuntamente como protocolo de la capa n.

En 1984 la Organización Internacional de Normas (ISO) creó el modelo de referencia OSI (*Open Systems Interconnection*) para lograr una estandarización internacional de los protocolos. La ISO investigó varios modelos de redes para encontrar reglas que pudieran aplicarse a todas las redes a fin de lograr la estandarización. La ISO desarrolló un modelo que sirve a los fabricantes para crear redes compatibles.

Este modelo se ocupa de la Interconexión de Sistemas Abiertos y está divido en 7 capas que se muestran en la Figura 1.1.

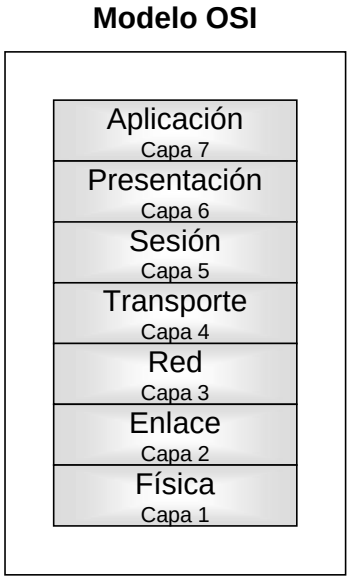


Figura 1.1: Capas del Modelo OSI

Los principios que se aplicaron para su división en capas son:

- Se debe crear una capa siempre que se necesite un nivel diferente de abstracción.

- Cada capa debe realizar una función bien definida.
- La función de cada capa se debe elegir pensando en la definición de protocolos estandarizados internacionalmente.
- Los límites de las capas deben elegirse a modo de minimizar el flujo de información a través de las interfaces.
- La cantidad de capas deben de ser suficientes para no tener que agrupar funciones distintas en la misma capa y lo bastante pequeña para que la arquitectura no se vuelva inmanejable.

A continuación se describe la función de cada una de las siete capas:

Capa Física (Capa 1) Se ocupa de la transmisión de bits a lo largo de un canal de comunicaciones. Se encarga de todas las conexiones físicas, terminales, características de transmisión, el medio de transmisión, la velocidad, etc.

Maneja por lo tanto diferentes tipos de señales (en esta capa no hablamos de datos sino de señales eléctricas) pudiendo ser variaciones eléctricas, lumínicas, campos electromagnéticos, determinando diferentes medios de transmisión y unidades de medición (frecuencia, amplitud, intensidad de luz, etc.)

Capa de enlace de datos (Capa 2) Esta capa se ocupa del direccionamiento físico dentro de una red local. Las señales empiezan a considerarse datos y su direccionamiento implica saber por dónde y en qué forma se van a distribuir los datos.

Esta capa se encarga de lo relacionado con la notificación de errores, el control del flujo y la correcta distribución de las tramas entre nodos vecinos.

Capa de red (Capa 3) Esta capa hace lo necesario para lograr el envío de datos desde el origen al destino a través de diferentes redes haciendo abstracción del tipo de conexión. Es en esta capa es donde los algoritmos de re-envío y ruteo tienen su funcionamiento. Por lo consiguiente para los fines de este proyecto esta capa es la más importante.

Capa de transporte (Capa 4) Su función es aceptar datos de la capa de sesión, dividirlos, si es necesario, en unidades más pequeñas, pasarlos a la capa de

red y asegurar que todos ellos lleguen correctamente al otro extremo. De tal forma que aísle la capa de sesión de los cambios de la tecnología de hardware.

Capa de sesión (Capa 5) Esta capa permite que los usuarios de diferentes computadoras puedan establecer sesiones entre ellos. Esta capa asegura que se puedan realizar todas las actividades previstas en una sesión determinada desde su comienzo hasta su finalización, y también en caso eventual de interrupción de la comunicación, proveer los medios para volver a conectarse.

Capa de presentación (Capa 6) Esta capa se encarga de manejar las estructuras de datos abstractos pero ya como contenido. Convierte datos para hacerlos compatibles. También está relacionada con la representación de la información por ejemplo cifrar datos y comprimirlos.

Capa de aplicación (Capa 7) Contiene varios protocolos que son utilizados con frecuencia. Cuando utilizamos un programa en particular, supongamos un cliente de correo cualquiera, estas aplicaciones actúan por medio de los protocolos que se manejan en esta capa. Cada día aumenta la cantidad de aplicaciones y con ellas los posibles protocolos a utilizar. Debemos entender que la utilización de este nivel no es directa hacia el usuario, sino por medio de utilidades que sirven de interfaz y nos hacen más fácil el uso de diferentes protocolos.

Cada capa tiene sus protocolos y la comunicación es realizada entre capas en distintas computadoras. En la Figura 1.2 se ilustra la forma en que se realiza esta comunicación.

Cuando la aplicación X de una computadora se comunica con la aplicación Y en una diferente computadora conectada a una red, la aplicación X sólo pasa los datos a la capa de aplicación, ésta capa empaqueta los datos y le pone una cabecera de información de control (CA), necesaria para poder desempaquetarlos en la capa de aplicación en la otra computadora.

A el paquete que genera cada capa se le denomina PDU (Protocol Data Unit) unidad de datos del protocolo, en el caso del protocolo de aplicación se le llama APDU (Aplication PDU).

La capa de aplicación pasa el APDU a la capa de sesión la cual también se encarga de generar su PDU llamado SPDU, el cual contiene el APDU más información de

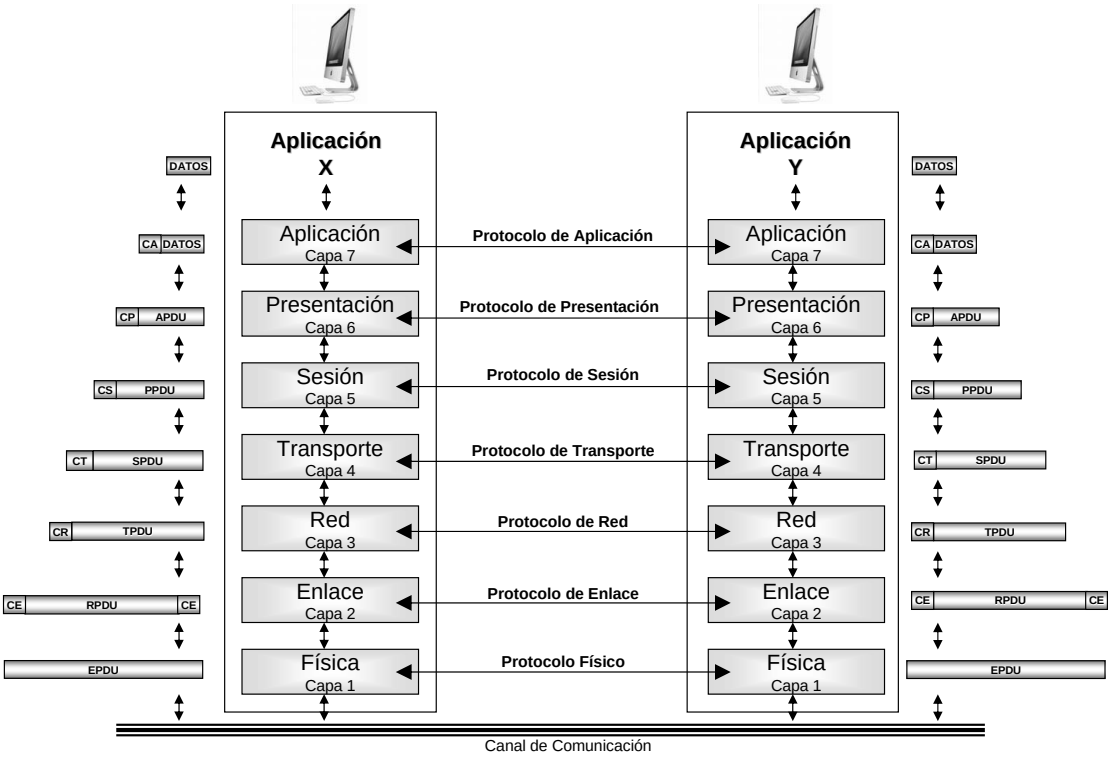


Figura 1.2: Estructura del Modelo OSI

control puesta en la cabecera de sesión (CS). Cada capa realiza la misma acción y en la Figura 1.2 se esquematiza la generación de paquetes por cada capa en forma descendente.

En la computadora con la aplicación Y cada capa desempaqueta los PDU qué le corresponden. Con la información de la cabecera se sabe que protocolo se utiliza y la estructura de los PDU empaquetados. Así cada capa se encarga de verificar que los datos sean manejados como fueron generados y así pasarlos a la siguiente capa superior, hasta llegar a la capa de aplicación que se encarga de entregar los datos que envío la aplicación X a la aplicación Y.

La capa de red es la capa con la que tiene que ver el trabajo de esta tesis y el protocolo que nos interesa explicar es el Protocolo de Internet (IP), en la siguiente sección se expone este protocolo.

1.3. Protocolo IP

Los protocolos TCP/IP son los estándares en torno a los cuales se desarrolló Internet dando como resultado el modelo de protocolos TCP/IP donde la capa de internet es equivalente con la capa de red del modelo de referencia OSI. Podemos decir que el protocolo base en la capa 3 es el protocolo IP.

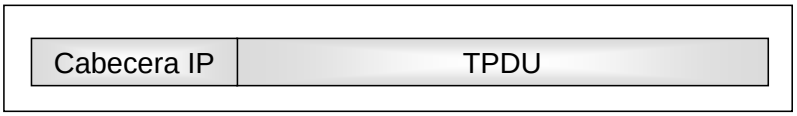


Figura 1.3: PDU de Red OSI

El Protocolo Internet [1] proporciona las funciones necesarias para enviar un paquete de bits llamado datagrama desde un origen a un destino a través de un sistema de redes interconectadas. El protocolo IP es utilizado por aplicaciones Host a Host en el entorno de Internet. También controla la fragmentación y reensamblado si el paquete es enviado por una red de trama pequeña.

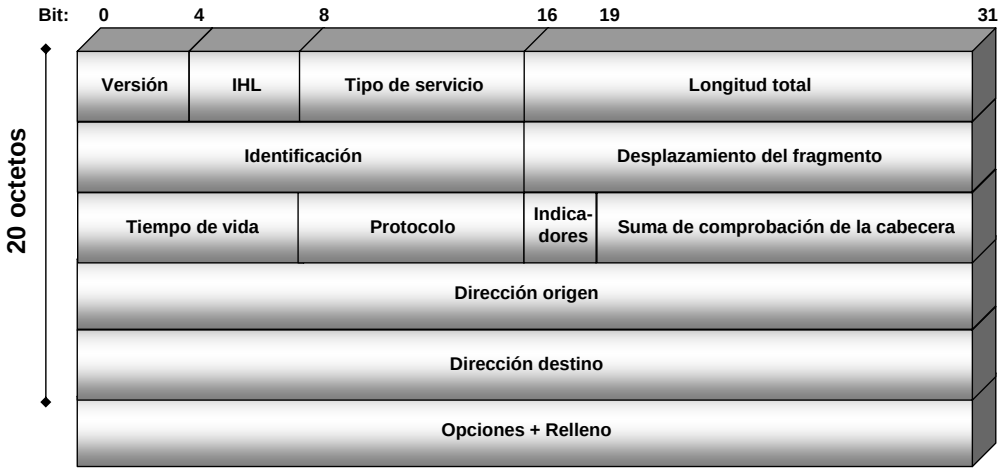


Figura 1.4: Cabecera de Red

El datagrama mostrado en la Figura 1.3 está formado por una cabecera más el TPDU. La cabecera del protocolo IP se observa en la Figura 1.4. En la cabecera IP se tiene la dirección IP de origen y la dirección IP de destino que son etiquetas de

identificación que tiene cada Nodo de Internet, son de una longitud de bits fija, de 4 octetos, es decir, 32 bits.

Se define como un Nodo de Internet a un dispositivo electrónico que es capaz de manejar los protocolos que se utilizan en la capa de red, es decir, tienen las primeras tres capas o más del modelo OSI. Un Nodo de Internet por lo tanto puede ser una computadora o un ruteador.

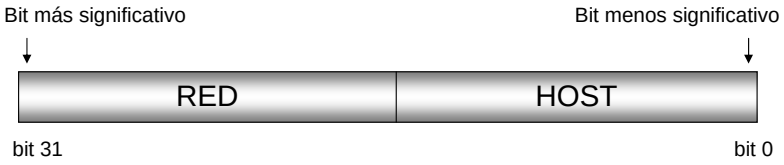


Figura 1.5: Dirección IP

Las direcciones IP son únicas para cada Nodo de Internet. Para ser precisos, cada dirección es única para cada una de las interfaces de red IP de cada Nodo de Internet. Si un Nodo de Internet dispone de más de una interfaz de red, necesitará una dirección IP para cada interfaz.

La dirección IP esta formada por dos partes una que identifica a la red y la otra que identifica al host, los bits más significativos corresponden al prefijo de red y los bits menos significativos a la etiqueta de host en la red. La Figura 1.5 muestra la estructura de la dirección IP.

El protocolo de Internet en su propuesta oficial especifica tres clases de red básicas mostradas en la Figura 1.6.

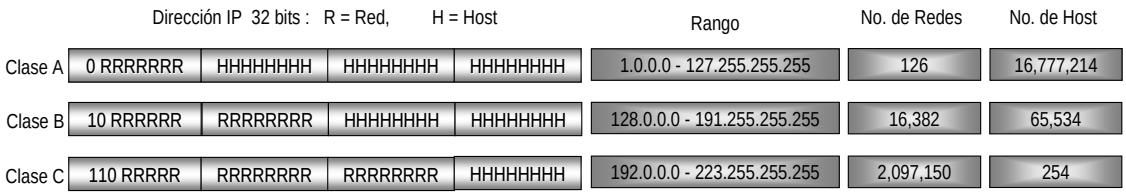


Figura 1.6: Clases definidas en IP

Clase A: El bit más significativo es cero y los siguientes 7 bits corresponden a la red y los 24 restantes son direcciones de host.

Clase B: Los primeros dos bits más significativos son “10”, los 14 bits siguientes identifican la red y los 16 bits restantes son direcciones de host.

Clase C: Los tres bits más significativos son “110”, los 21 bits siguientes son de red y los 8 restantes son la dirección de host.

Para mayo de 1993 se describe la regionalización de los prefijos de red en el documento de definición de protocolos RFC-1466 (Request for Comments), llamado “Guidelines for Management of IP Address Space”, donde se asigna a cada continente un espacio de direcciones IP a manejar mediante una empresa dedicada a la asignación de direcciones IP por región.

En la actualidad las clases de red no se utilizan, dejando para el siguiente capítulo la explicación de la problemática que tiene utilizar este esquema.

Para que un paquete recorra la red es necesario solamente tres capas: la capa física, la capa de enlace de datos y la capa de red. En la Figura 1.7 se muestra la interacción de estas tres capas.

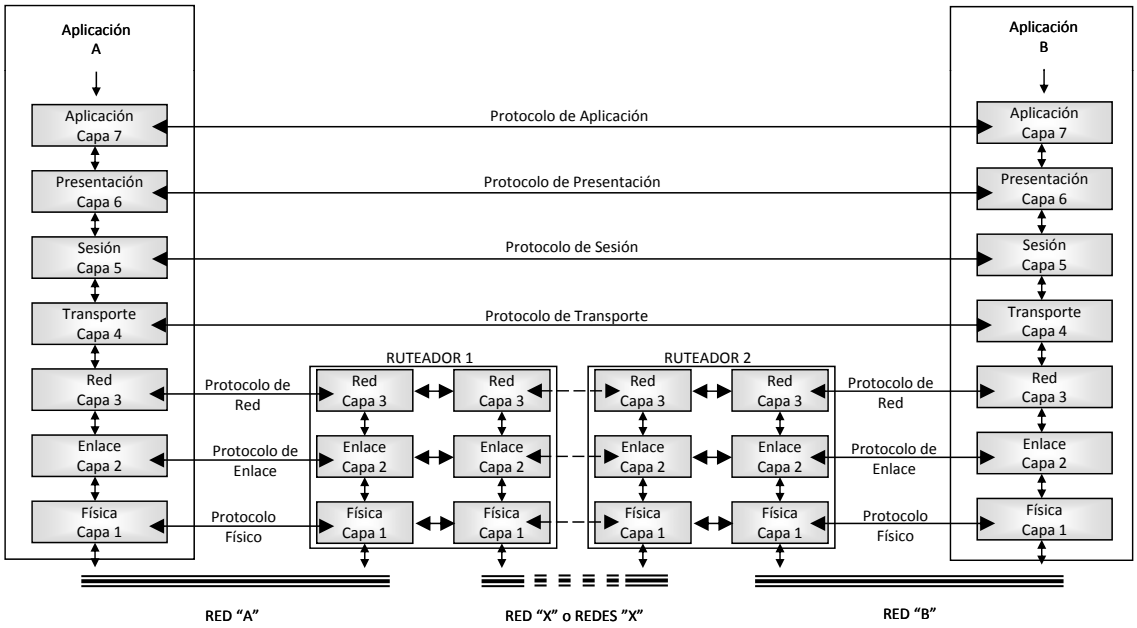


Figura 1.7: Modelo OSI Completo

Los datagramas son enrutados desde un Nodo de Internet a otro a través de redes individuales basándose en la interpretación de la dirección IP, es por esto que es un importante mecanismo el protocolo Internet.

Los ruteadores son Nodos de Internet, que son importantes para esta tesis, en donde tiene efecto el protocolo IP, a continuación se definen sus funciones.

1.4. Ruteadores

Un ruteador es un dispositivo con varias interfaces de red y está dedicado a realizar dos tareas el ruteo y reenvío.

1.4.1. Ruteo

En el ruteo, se lleva a cabo un proceso distribuido que se encarga de recolectar la información de las redes a las que se puede alcanzar.

Los prefijos de red son los bits que identifican a una red, y la longitud del prefijo es el número de bits que contiene el prefijo.

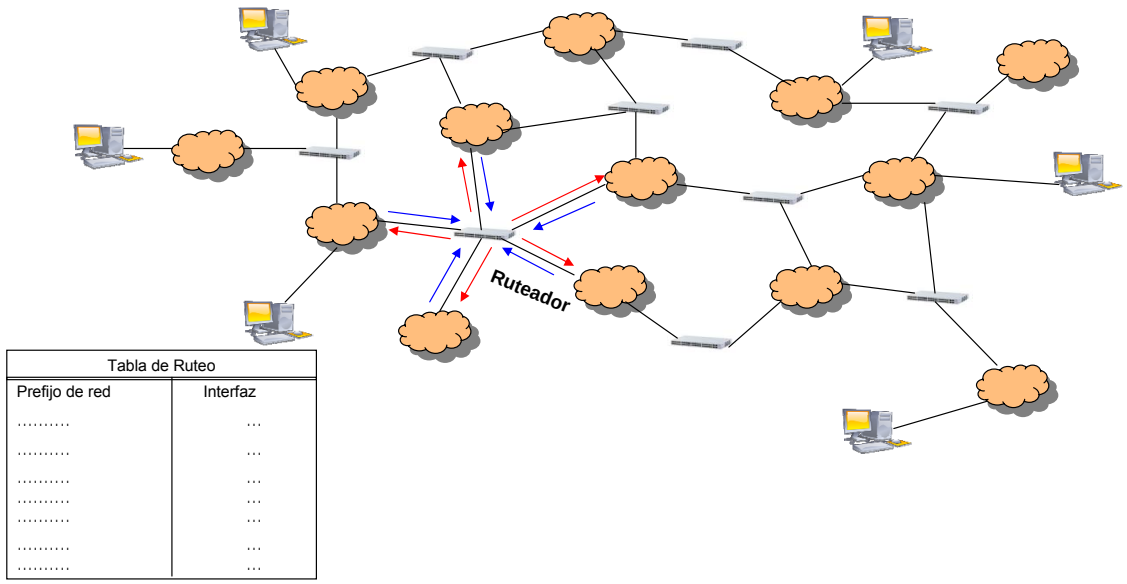


Figura 1.8: Generación de la Tabla de Ruteo

La información que recogen los ruteadores es precisamente los prefijos de red de las redes que están cerca de los ruteadores, este prefijo siempre debe ser asociado con la interfaz del ruteador por la que esta red puede ser comunicada.

Existen protocolos de ruteo para que el prefijo se asocie a la interfaz que le genere el menor costo en el atributo que se quiera minimizar. El prefijo de red y su interfaz de salida se guardan en la llamada tabla de ruteo, que además del prefijo y la interfaz tiene también algún tipo de ponderación debido al algoritmo de ruteo utilizado.

1.4.2. Reenvío

El reenvío comienza cuando llega un paquete al ruteador, con la información que tiene el paquete de su destino se busca la mejor ruta y es reexpedido para que continúe su recorrido.

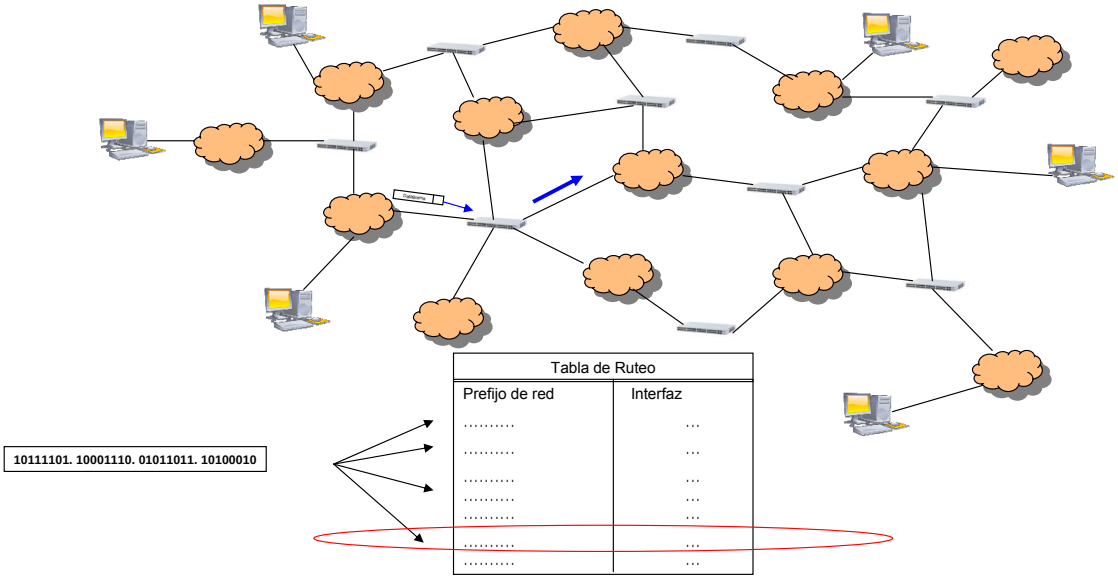


Figura 1.9: Reexpedición del paquete

De la cabecera IP se obtiene la dirección IP destino y con las entradas en la tabla de ruteo se realiza una búsqueda para encontrar el prefijo de red que le corresponde a la dirección IP destino del paquete. El prefijo está asociado a una interfaz y es por allí donde se reexpide el paquete.

Así, el paquete es enviado por la ruta de menor costo para la dirección destino del paquete. Y continúa su viaje por las redes.

En cada ruteador se realiza esta búsqueda hasta que finalmente llega a un ruteador que conoce el host destino y entrega el paquete.

Ahora ya, con los conocimientos adquiridos en el siguiente capítulo se podrá definir cual es el problema tratado en esta tesis.

Capítulo 2

Búsqueda y actualización de la información en ruteadores IP

2.1. Búsqueda con Clases

El IP propone que las clases de red se utilicen para saber cuál es la estructura de la dirección IP, con esto se sabe cuantos bits son utilizados para representar la red y cuantos bits para representar el host. La idea es buena, el problema ahora es el crecimiento de las redes a nivel mundial.

La tabla de ruteo está agrupada en clases, solamente tenemos tres tipos de posibles prefijos de red, las longitudes son 8, 16 y 24 bits, y los primeros bits de la dirección destino nos indica cual es la longitud.

La búsqueda de la información de ruteo para que un paquete sea reenviado por la interfaz que le corresponde, se realiza como sigue, se busca la dirección IP destino del paquete y se identifica qué clase de red le corresponde, la búsqueda se realiza en los prefijos de red de la clase de red a la que pertenece, es decir, se sabe de antemano cual es la longitud del prefijo de red. En la Figura 2.1 se ejemplifica la búsqueda con clases de red.

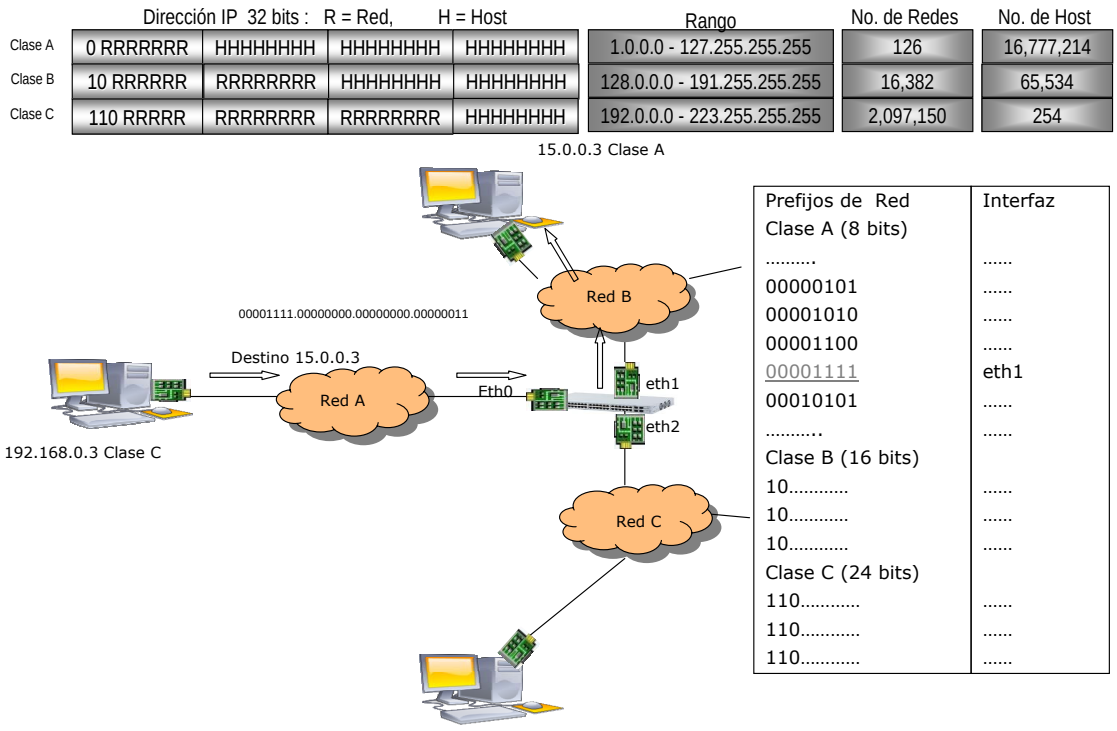


Figura 2.1: Búsqueda del prefijo dependiendo de la Clase

Se observa que tener tres clases de red tiene los siguientes problemas:

- En la clase A el número de redes es pequeño y tiene excesiva cantidad de hosts, si una empresa obtenía una dirección de red de clase A desperdiciaría gran cantidad de direcciones de host.

- En la clase C se tienen suficientes prefijos de red pero solamente 254 identificadores de hosts para cada red, si la empresa tenía más de 254 host era necesario obtener otra dirección de red clase C y el problema es que se debía hacer trabajo de ruteo en la red interna de la empresa.

- El crecimiento de las redes y de los usuarios de Internet hace que la asignación de direcciones IP tenga el problema de acabarse las direcciones de red de Clase B que son las más manejables.

En el siguiente esquema se minimizan los problemas que tienen las clases, pero se agregan nuevos problemas.

2.2. Búsqueda con CIDR

Los problemas descritos por los que se define el nuevo esquema de direccionamiento son:

- Agotamiento del espacio de direcciones de la clase B. Una causa fundamental de este problema es la falta de una clase de red de un tamaño apropiado para una organización de tamaño mediano. La clase C, tiene como máximo 254 direcciones de host que es demasiado pequeño, mientras que la clase B, permite un máximo de 65,534 direcciones y es demasiado grande. El resultado de este problema es el uso ineficiente de números de red de clase B.
- Sobrecarga de información de ruteo. El tamaño y la tasa de crecimiento de las tablas de ruteo en los ruteadores de Internet está más allá de la capacidad actual de software (y personas) para administrarlo eficazmente.
- Eventualmente el número de redes IP se acabará.

La solución que se plantea es CIDR (Classless Inter Domain Routing), quitar las clases y proponer que la dirección de red pueda tomar diferentes longitudes de prefijos de red. Así las empresas obtienen una dirección de red acorde con sus necesidades y el mal uso de direcciones de host se minimiza ya que no se desperdician.

CIDR propone una máscara de red que hace la separación entre la dirección de red y el host. La máscara son 32 bits donde los bits más significativos hasta el número de longitud del prefijo de la red están en uno y los bits menos significativos a partir del número de la longitud del prefijo están en cero, esto para realizar la operación lógica “and” con la dirección destino y así tener la dirección de red. El prefijo de red puede ser de cualquier longitud entre 1 y 32, con esto se quitan las clases y se hace un uso efectivo de todo el espacio de direcciones IP.

En el entorno de la red se utilizan las máscaras de red, pero el problema es que en las tablas de ruteo se almacena la información como prefijo de red. Con CIDR las tablas de ruteo de los ruteadores guardan el prefijo de red de diferentes longitudes, la interfaz por donde debe ser alcanzada esta red y algún tipo de costo asociado a esa interfaz.

2.3. El problema

¿Cuál es el precio que se debe pagar con el cambio de clases a CIDR?.

En la cabecera IP solamente tenemos campos que indican la dirección IP destino, pero ninguno indica la longitud del prefijo de red.

Ahora el problema es que no sabemos cuál es la longitud del prefijo de red que le corresponde a la dirección IP destino, no sabemos hasta donde se identifica la red ni donde comienza la identificación del host. Y además nadie tiene información referente a la longitud del prefijo.

La búsqueda en la tabla de ruteo con las clases era sencilla ya que los primeros bits indicaban la estructura de la dirección, la longitud de la dirección de red, y solamente buscábamos en tres clases de red. Ahora con prefijos de diferentes longitudes, se busca en la tabla de ruteo el prefijo de red que coincida perfectamente con los bits más significativos de la dirección IP de destino y que sea el más largo que exista en la tabla de ruteo, a éste prefijo se le llama **BMP**(Best Matching Prefix), en algunos textos también se le conoce como LMP (Longest Matching Prefix).

La búsqueda no es exacta en la mayoría de los casos, solamente se puede dar la solución más próxima, ya que al momento de asignarse las direcciones de red quedaron agrupadas por localidad como ya se había explicado en el capítulo anterior, en la Figura 2.2 observamos un ejemplo de la búsqueda en CIDR.

En una empresa que tiene un número reducido de máquinas el ruteo no es costoso en tiempo, la búsqueda no representa un retardo significativo ya que se encuentra la información de ruteo relativamente rápido.

El problema es en los ruteadores de alto desempeño que existen en el backbone de Internet, ya que ellos conocen miles de prefijos de red y hacer la búsqueda del prefijo de red más largo representa un costo en tiempo de procesamiento en cada ruteador. Y se debe tomar en cuenta que se busca el prefijo de red más largo que coincide con los primeros bits de la dirección destino que esta asociado a la información de la interfaz por la que debe ser enviado el paquete.

La transmisión en el backbone tiene velocidades muy grandes del orden de los Gbps o Tbps, y por tanto, el cuello de botella está en el tiempo de procesamiento para reexpedir un paquete. ¿Cómo resolver este problema? La idea importante es que la

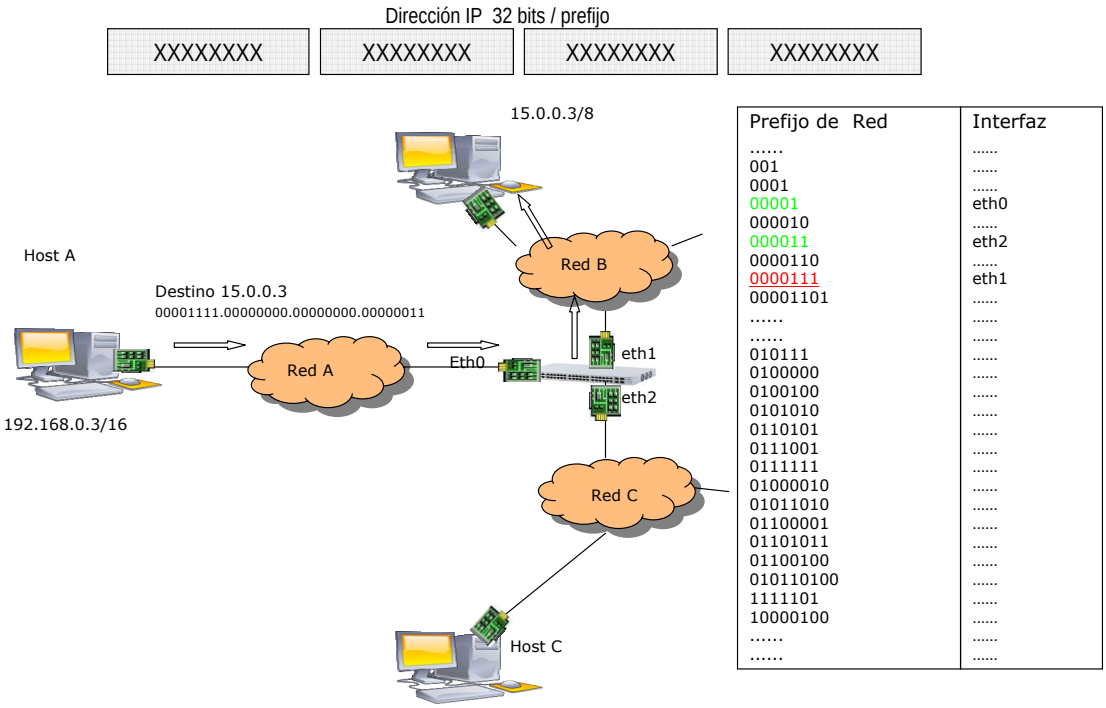


Figura 2.2: Búsqueda del prefijo en CIDR

tabla de ruteo debe estar organizada en una estructura de datos que permita realizar búsquedas sobre el espació de prefijos, con la dirección IP destino se pueda encontrar el BMP y regrese la información de la interfaz de salida. Definimos información de ruteo como la interfaz de salida que está asociada con un prefijo de red.

Cuando la tabla se convierte en la estructura de datos hay que tener en cuenta que la tabla está inmersa en un sistema distribuido en tiempo real, y que siempre es cambiante, así que, tenemos que considerar la actualización de información de ruteo; la adición de nuevos prefijos de red y cuando se cae un nodo o se satura y es necesario quitar este prefijo de red de la tabla de ruteo. Por lo tanto, el tiempo de búsqueda es directamente proporcional a la complejidad de la estructura de datos para organizar la tabla de ruteo, es decir, si el tiempo de búsqueda se minimiza, la actualización de los prefijos de red se complica y si la estructura de datos es muy simple el tiempo de búsqueda para encontrar la información de ruteo crece. Las soluciones que hay para resolver este problema son variadas, en el siguiente capítulo daremos un breve recorrido por las más representativas.

Capítulo 3

Algoritmos Representativos

El crecimiento de las redes y de los requerimientos de aplicaciones como video a la demanda y cómputo en tiempo real, necesitan menor retardo al atravesar Internet.

Para realizar la búsqueda de la información de ruteo se han creado diferentes algoritmos. A continuación se describen las ideas principales de algunos de los algoritmos presentados como solución.

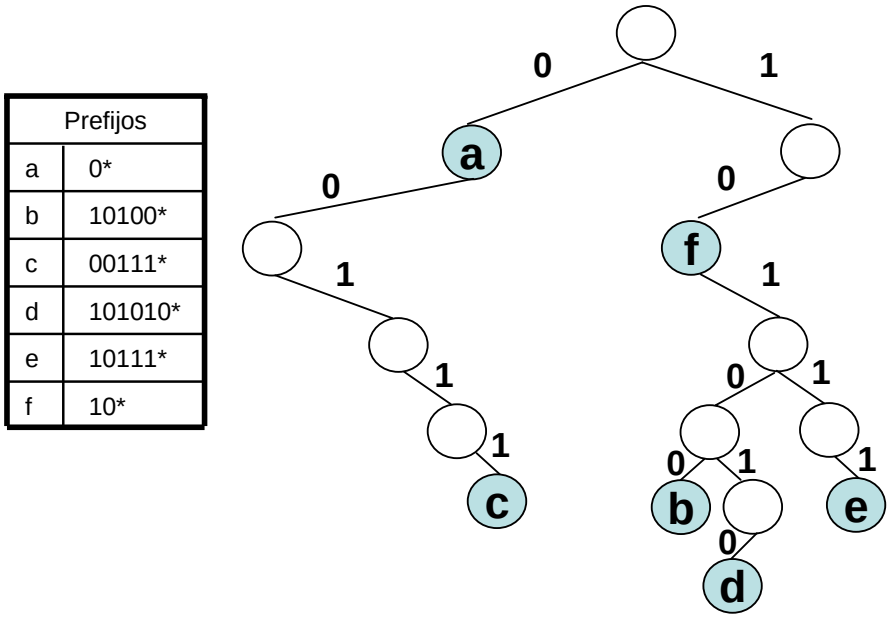
3.1. "Trie" Binario

La solución más básica es poner la información de la tabla de ruteo en un árbol binario de recuperación de datos [4]. La información de ruteo es guardada en los nodos del árbol y la posición de los hijos está dada por el valor del bit que se compara.

3.1.1. Construcción del "Trie" Binario

Para construir el árbol los prefijos de la tabla de ruteo son insertados uno a uno comenzando por el bit más significativo. Se compara el valor del bit, si es uno se inserta un nodo a la derecha si es cero se inserta un nodo a la izquierda. Si un nodo ya existe únicamente se salta hacia este nodo y se continúa con la inserción de

los siguientes bits. Cuando se llega al final de los bits del prefijo de red se inserta en el último nodo la información de ruteo asociada a este prefijo. Así, para poder representar los siguientes prefijos se tiene el árbol de la Figura 3.1.



Prefijos	
a	0*
b	10100*
c	00111*
d	101010*
e	10111*
f	10*

Figura 3.1: "Trie" Binario

3.1.2. Búsqueda en el "Trie" Binario

La búsqueda del prefijo de red más largo, se obtiene mediante un recorrido del árbol. Se toma la dirección de destino de la cabecera IP, se comienza con el bit más significativo comparando si es uno o cero y recorriendo los nodos del árbol hasta tener alguno de los siguientes criterios de paro:

- Cuando se encuentre una dirección de 32 bits de longitud.
- Cuando para la comparación que se realiza, no existe el nodo que se debe recorrer.

Se utiliza una variable para guardar la información de ruteo del prefijo más largo encontrado. Cada vez que se visita un nodo se revisa si existe información de ruteo

de la interfaz asociada al nodo. Si existe información, está se actualiza en la variable de la interfaz encontrada.

La dirección IP destino llega a un nodo de profundidad máxima, el cual es el prefijo con la mayor cantidad de bits significativos iguales.

El tiempo de búsqueda de cualquier dirección está relacionado con la longitud de los prefijos en la estructura de datos, por lo que la búsqueda es O(W) donde W es la longitud máxima de un prefijo.

3.2. Árboles de Ruta Reducida

La idea de este algoritmo es encontrar ramas en el árbol binario que puedan contraerse [4]. Se analiza el árbol para contraer ramas donde la bifurcación se pueda tomar en base a un bit de la dirección IP destino. Se contraen las ramas para las que las comparaciones no tienen significado en la búsqueda y que solamente ocupan espacio en memoria.

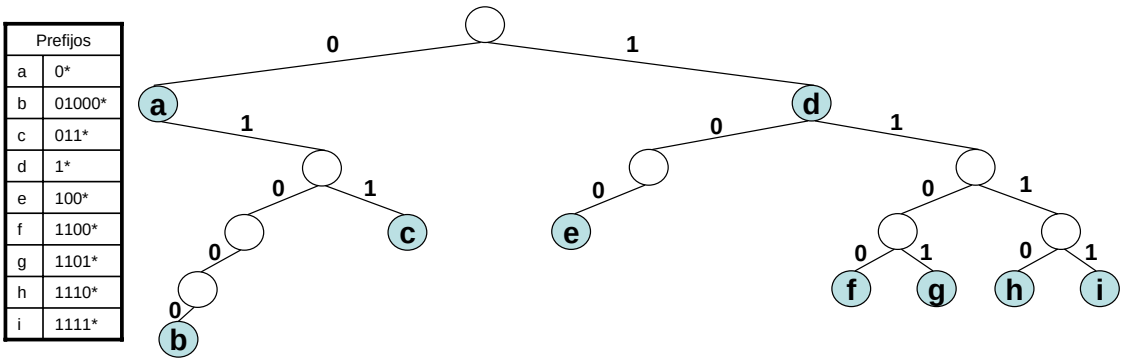


Figura 3.2: Árbol binario

Se puede observar un ejemplo en la Figura 3.2. Para este caso se comprime la ruta hacia el nodo “b” ya que en este camino no hay ninguna bifurcación, también después del nodo “a” solamente el camino posible es 1, por lo que esta rama se comprime.

Para el nodo “e” ocurre lo mismo, pero no para los nodos que tienen decisiones y el nuevo árbol se muestra en la Figura 3.3. La información en los nodos indica la posición del bit que se debe comparar y el prefijo que representa.

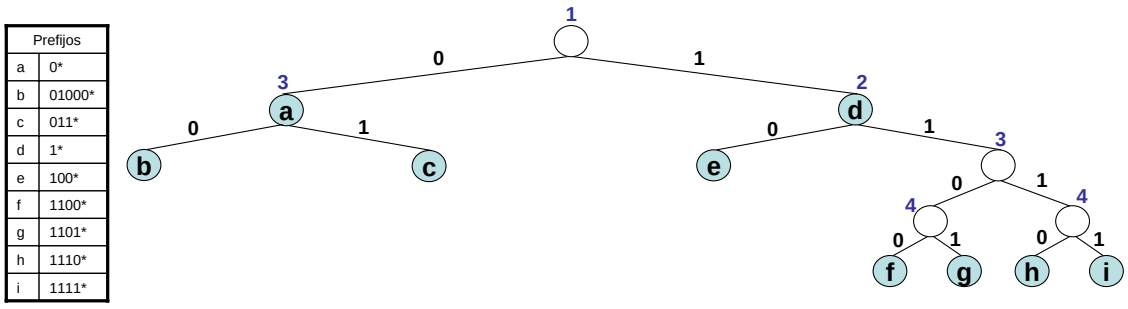


Figura 3.3: Árbol de ruta reducida

En el árbol se contrae la rama derecha del nodo “a” y la rama izquierda del nodo “d”. En los nodos donde se realizó la reducción de ruta se agrega información con respecto la posición del bit que se debe comparar de la dirección IP destino.

El nodo también tiene guardado el prefijo para que se compare con la dirección IP destino. Si esta comparación es positiva, se toma la información de ruteo y se continúa la búsqueda; en caso contrario, termina la búsqueda.

Depende de la distribución de prefijos de la tabla de ruteo, se puede tener el caso de que el árbol no pueda contraerse teniendo como resultado el algoritmo “trie” binario. En el peor de los casos su complejidad es O(W) donde W es la longitud máxima de un prefijo.

3.3. Expansión controlada del prefijo

La idea de este algoritmo es que los prefijos se expandan [6] hasta una longitud que permita hacer comparaciones con secciones de la dirección IP destino, es decir que se toma un grupo de bits y se comparan en una iteración. Cuando se tiene la expansión controlada, se construye un árbol de menor profundidad ya que los nodos son arreglos que tienen todas las posibles combinaciones para la longitud de bits

que se compara, esto representa menos iteraciones del algoritmo. En la Figura 3.4 se presenta un ejemplo.

Nombre	Original	Longitud
P5	0*	1
P1	10*	2
P2	111*	3
P3	11001*	5
P4	1*	1
P6	1000*	4
P7	100000*	6
P8	1000000*	7

Nombre	Expansión	Longitud
P5	00*	2
P5	01*	
P1	10*	
P4	11*	
P2	11100*	5
P2	11101*	
P2	11110*	
P2	11111*	
P3	11001*	
P6	10000*	
P6	10001*	
P7	1000001*	7
P8	1000000*	

Figura 3.4: Expansión del prefijo

Observamos que los prefijos en la tabla se expanden a longitudes establecidas de 2, 5 y 7 bits.

Para el prefijo P5 de longitud uno (Figura 3.4 izquierda) con valor 0 completando con la combinación 00 y 01 se expande a un prefijo de longitud dos (Figura 3.4 Derecha). Del prefijo P5 ahora se tienen dos prefijos. Esta longitud a la que se expanden los prefijos es llamada *stride*, y podemos tener *strides* variables, para la Figura 3.4 del lado izquierdo tenemos *strides* de 2 y 3 bits. Con esta información podemos generar un nuevo árbol de prefijos expandidos con los arreglos para cada grupo de bits que se debe comparar, se muestra en la Figura 3.5.

Para pasar de un nivel a otro se debe comparar cierta cantidad de bits. En el ejemplo, en la raíz se comparan dos bits, para llegar al primer nivel se toman los siguientes 3 bits, es decir, la longitud que se compara en el primer nivel es de 5 bits y en el último nivel se toman otros 2 bits para llegar a la longitud de 7 bits.

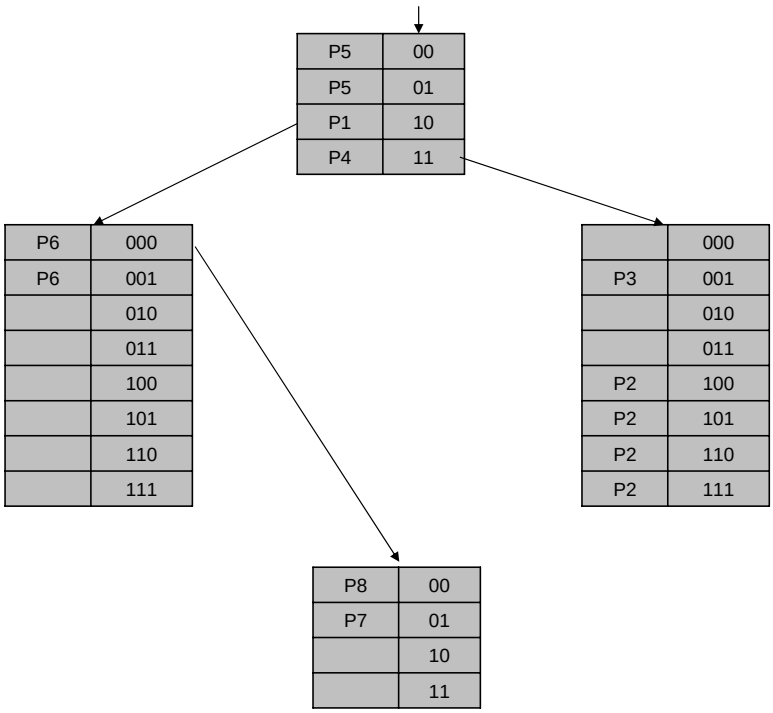


Figura 3.5: Árbol de tablas de Expansión del prefijo

En cada nodo se tiene un arreglo de tamaño 2^n donde ahora queda claro que n son los bits que se comparan por nivel, para el ejemplo el nivel cero tiene 2 bits a comparar y se crea un arreglo de 4 entradas, en el segundo nivel se tiene 3 bits y se genera un arreglo de 8 entradas, es así que se genera este nuevo árbol.

Cada entrada tiene la información de ruteo asociada al prefijo y un apuntador hacia otro arreglo para continuar la comparación. El algoritmo tiene un tiempo O(k) donde k es la longitud de la ruta más larga del nuevo árbol.

3.4. Búsqueda Multi-dirección y Multi-columna

La búsqueda no exacta se modela como una búsqueda exacta, añadiendo información en forma de rango de prefijos [7]. Usando un ejemplo con direcciones de 6 bits y con los siguientes prefijos de red: 1*, 101* y 10101*. Se toman los límites inferior y superior de los prefijos de red mostrados en la Figura 3.6.

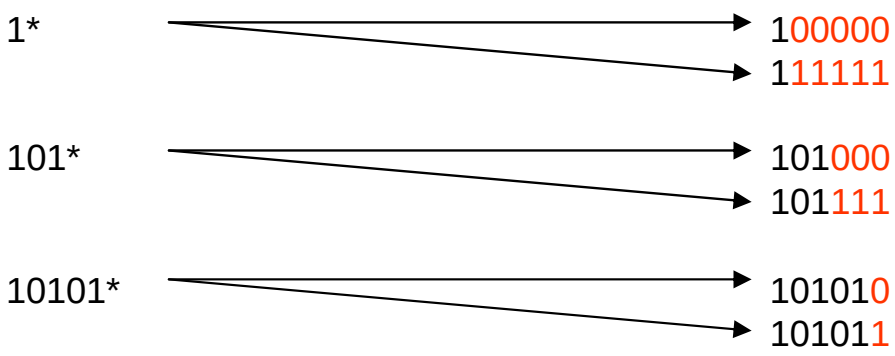
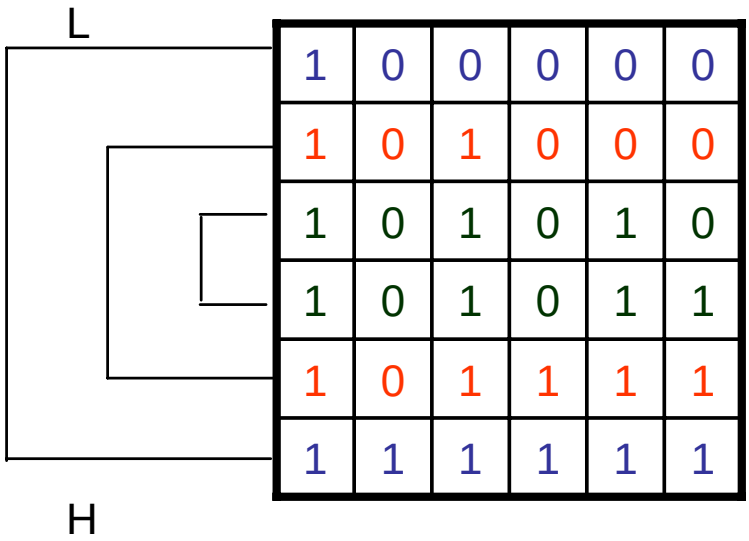


Figura 3.6: Límites inferior y superior de los prefijos

Con esta información se genera la Tabla 3.1, donde se ordenan las direcciones de menor a mayor.



1	0	0	0	0	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	0	1	1
1	0	1	1	1	1
1	1	1	1	1	1

Tabla 3.1: Límites inferior y superior de los prefijos

Los prefijos ahora representan rangos donde el prefijo 10101* cubre los valores 101010 y 101011. Si una dirección cae dentro de este rango es considerada como prefijo 10101*.

Si llegan las siguientes direcciones 101011, 101110 y 111110 su búsqueda nos daría el resultado mostrado en la Tabla 3.2.

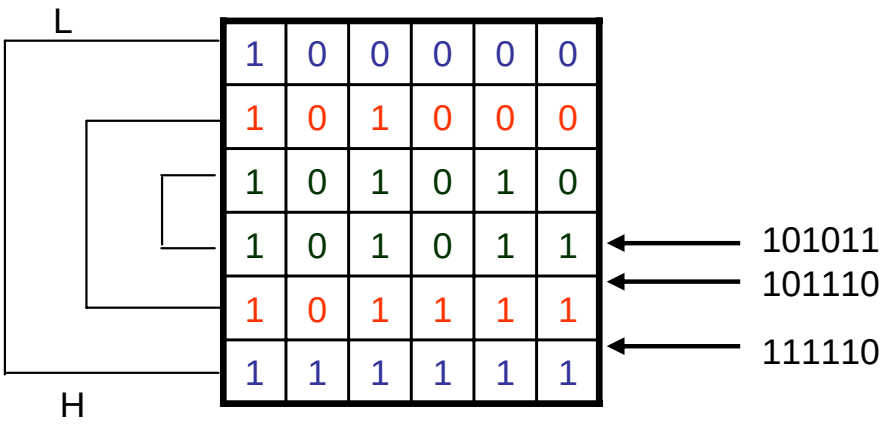


Tabla 3.2: Búsqueda de direcciones en la tabla expandida

La codificación de los rangos se puede visualizar en la Figura 3.7

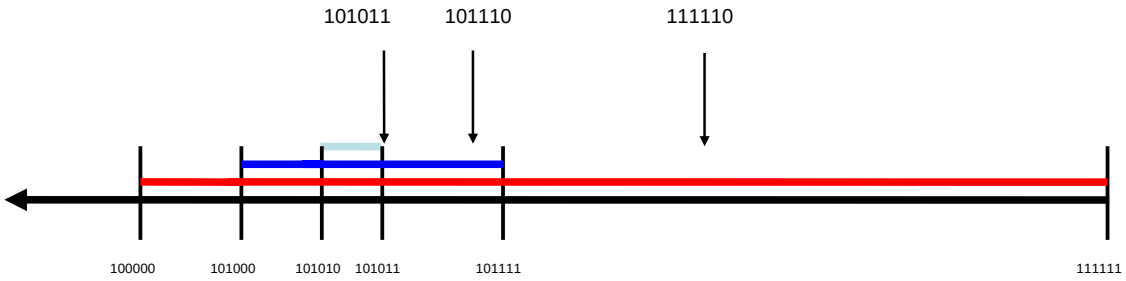


Figura 3.7: Rangos de los prefijos

Cuando utilizamos los beneficios de la memoria cache y la segmentación de la dirección en columnas, la búsqueda es más rápida, los tiempos de acceso a memoria están dados por $log_2(N + W/M)$, donde N es el número de entradas en la tabla, W es la longitud de los prefijos y M es el tamaño de palabra que maneja el bus del dispositivo de procesamiento.

3.5. Búsqueda en Tablas *Hash*

La idea de las tablas *Hash* [8] es ordenar la información de la tabla de ruteo con respecto a la longitud del prefijo, así la búsqueda está guiada por la longitud del prefijo.

3.5.1. Búsqueda Lineal

Esta se realiza mediante un arreglo que contiene las longitudes de los prefijos y un apuntador a la tabla hash que los contiene, así dependiendo de la longitud se realiza la búsqueda en una sola tabla. Un ejemplo con prefijos de longitudes 5,7 y 12 bits con los prefijos 01010, 0101011, 0110110 y 011011010101 se muestra la Figura 3.8.

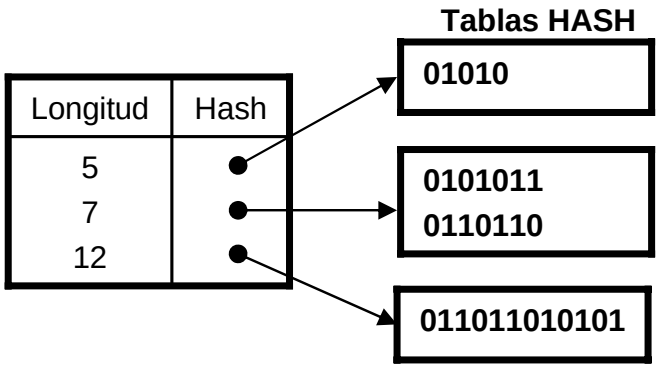


Figura 3.8: Búsqueda *Hash* ineal

3.5.2. Búsqueda Binaria

Para el arreglo que contiene las longitudes de los prefijos se realiza una búsqueda binaria. El inconveniente es que los prefijos de longitudes más grandes deben informar que existen a los prefijos de longitudes menores, es por esto que se utiliza una marca en los arreglos de menor longitud para solucionar este problema. En la Figura 3.9 observamos que 11 no existe como prefijo, éste es una marca de 111 para que cuando se busque, tenga una representación para referencia hacia este prefijo.

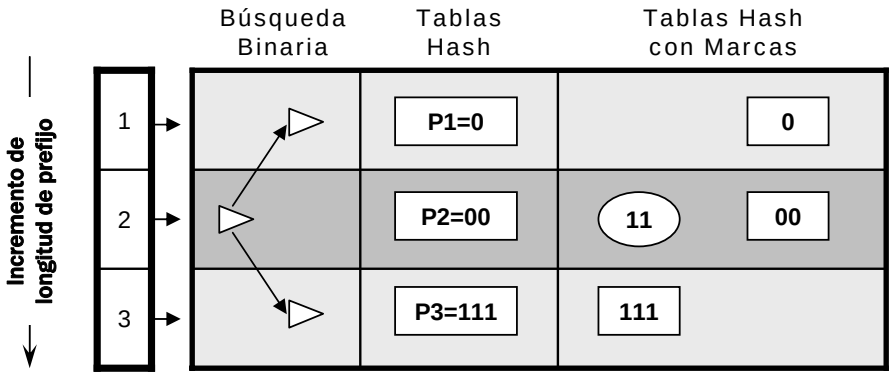


Figura 3.9: *Hash* binario con marcas

En la Figura 3.10 la búsqueda binaria se realiza en la parte izquierda del gráfico y en la parte derecha se muestra una visualización de los prefijos.

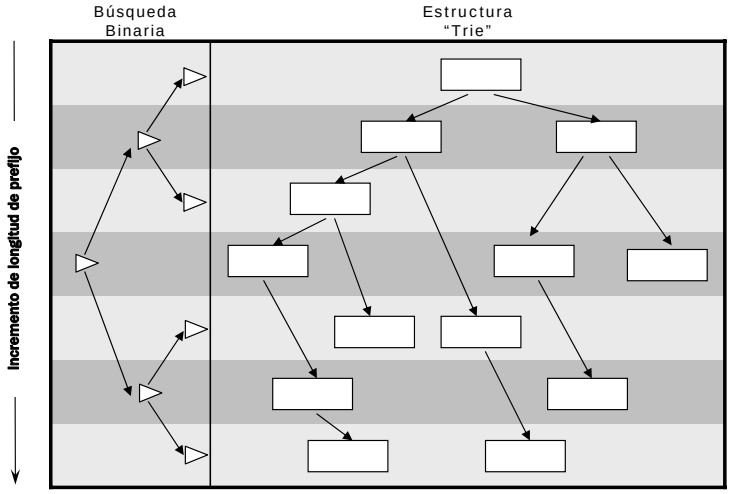


Figura 3.10: Búsqueda en *Hash* binario

3.5.3. Búsqueda Binaria Asimétrica

Para este tipo de búsqueda se utiliza información estadística con la que se toman decisiones para generar nuevos árboles. En la Figura 3.11 se muestra un histograma de frecuencias de longitud de prefijos de red, esta información es utilizada en la búsqueda asimétrica.

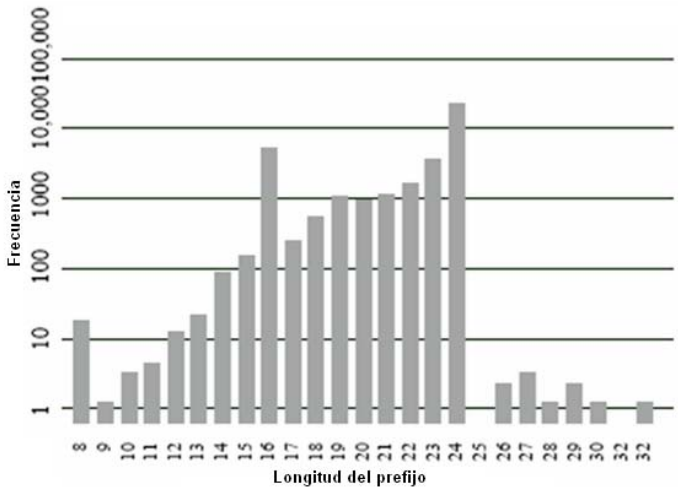


Figura 3.11: Histograma de frecuencias de longitud de prefijos

Una búsqueda binaria normal genera el árbol mostrado en la Figura 3.12, el lado izquierdo representa direcciones de 32 bits y en el lado derecho se utiliza el histograma de frecuencias. Como éste no contiene las primeras longitudes, además comienza con la longitud de 8 bits y termina con 32 bits, se toma la mitad que es 19 y así se construye el árbol únicamente con las longitudes que estadísticamente ocurren.

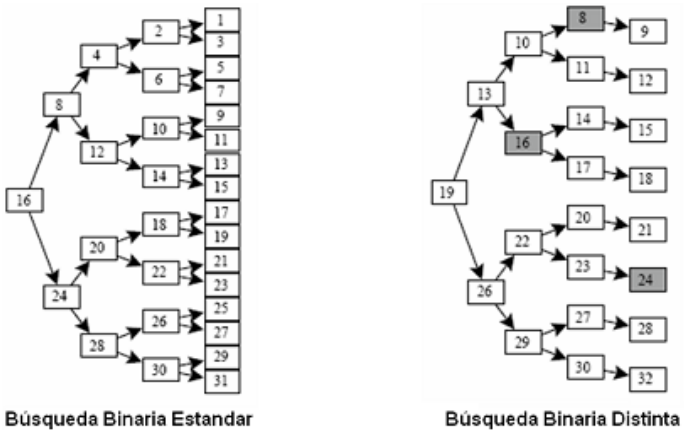


Figura 3.12: Comparación entre el Árbol binario y un Árbol con información estadística

3.6. Algoritmo de la Universidad de Lulea

La idea principal de este algoritmo es que la información de ruteo será almacenada en un arreglo, para encontrar el prefijo que le corresponde a una dirección IP se obtiene el índice del arreglo mediante otros tres arreglos [5]. El índice se puede obtener con segmentos de la dirección IP destino y los tres arreglos que contienen información de una cuenta de unos.

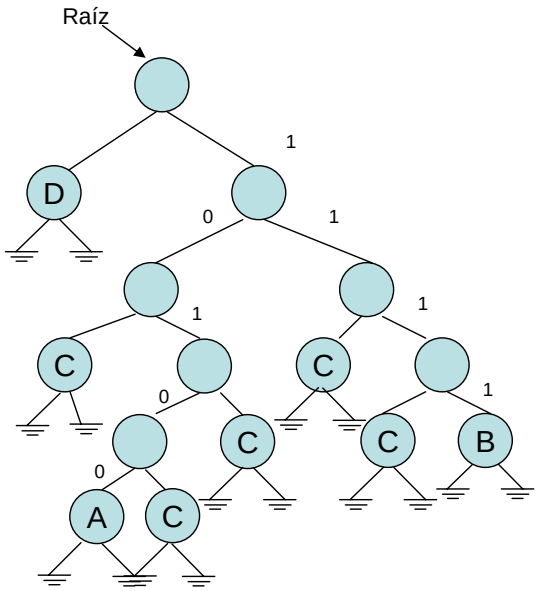


Figura 3.13: Árbol binario completo

Este algoritmo utiliza como base el árbol binario completo y lo divide en tres secciones de búsqueda. La primera sección es a una profundidad de 1 a 16, la segunda es de 17 a 24 y la tercera es de 25 a 32, esto hace que los arreglos tengan un número reducido de entradas.

La búsqueda se realiza por sección, encontrando el índice del arreglo donde está la información de ruteo o el apuntador a la rama del árbol donde se continuará con la búsqueda.

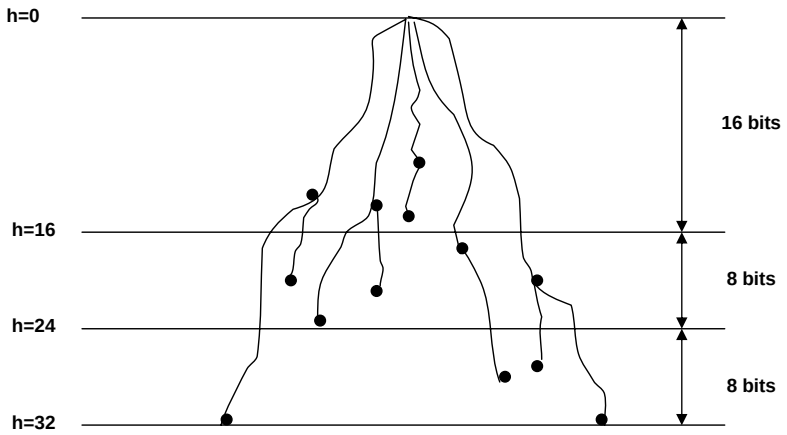


Figura 3.14: División del árbol en secciones

Dependiendo de la densidad de prefijos de la rama de la siguiente sección, se aplica el mismo algoritmo o se utiliza un algoritmo menos complicado.

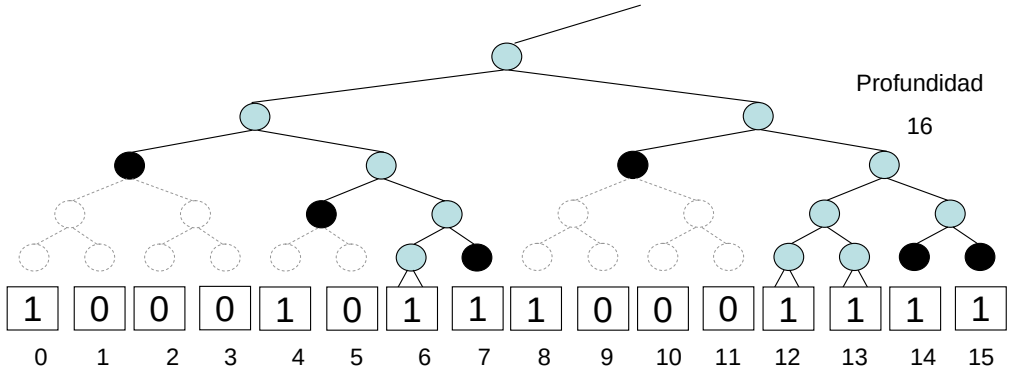


Figura 3.15: Arreglo en el corte a profundidad 16

3.7. Análisis de complejidad de los algoritmos

La variable que representa a la búsqueda es el tiempo, y esta relacionada con la complejidad computacional, esta complejidad se debe a una o más de las siguientes características: la cantidad de prefijos de red de la tabla de ruteo, la longitud de la dirección IP o con la división de la información de los prefijos. En la Tabla 3.3 se

compara la complejidad de los algoritmos vistos en este capítulo.

Tabla 3.3: Complejidad computacional de los algoritmos estudiados

Nombre del Algoritmo	Peor caso tiempo de búsqueda	Memoria	Construcción
Trie Binario	O(W)	O(NW)	O(NW)
Árbol de ruta reducida	O(W)	O(N)	O(W)
Expansión controlada del prefijo	O(W/k)	O(2^kNW/k)	O(W/k + 2k)
Multi-dirección y multi-columna	$O(log_2(N))$	O(N)	O(N)
Búsqueda en tablas Hash	O(log_2 (W))	O(N log_2 (W))	O(N log_2 (W))
Universidad de Lulea	O(W/k)	O(2^kNW/k)	-

Donde W es la longitud de la dirección, N es el número de prefijos en la tabla de ruteo y k es el tamaño de stride es decir el número de bits que se comparan.

En el siguiente capítulo se describe la implementación y análisis del desempeño de dos de los algoritmos descritos, esto con el propósito de tener un marco de comparación para el algoritmo que proponemos.

Capítulo 4

Marco de Comparación

En este capítulo se presenta la implementación y evaluación de desempeño de dos algoritmos de búsqueda de información en tablas de ruteo que se utilizarán como referencia con propósitos comparativos. Los algoritmos son:

1. "Trie" binario [4].

2. Algoritmo de la Universidad de Lulea [5].

Para realizar evaluaciones del desempeño de los algoritmos, se propone que se implementen en el lenguaje de programación C y que se realice la simulación de la búsqueda de prefijos de red en la tabla de ruteo tomando mediciones de tiempo de búsqueda.

Para probar las implementaciones se utiliza una tabla de ruteo de un ruteador de alto desempeño [9]; estos datos son reales y son proporcionados para fines de investigación. La tabla de ruteo tiene 213086 prefijos de red al mes de diciembre de 2006.

Los prefijos de red están guardados en un archivo del tipo texto, en el que por cada línea se tiene un prefijo, este prefijo está en binario, teniendo en cuenta que solamente están los bits que corresponden a la longitud del prefijo.

Para probar las implementaciones de los algoritmos, se efectuaron búsquedas para encontrar el prefijo más largo que le corresponde a una dirección IP, esto para medir el tiempo que se tarda en encontrar la información de ruteo. Para probar el correcto funcionamiento de las implementaciones, se generan direcciones IP con información de los prefijos de red de la tabla de ruteo. Con estas direcciones el algoritmo debe encontrar el prefijo de red que utilizamos para formar la dirección IP.

Para medir el tiempo de búsqueda, se guarda el registro del contador de tiempo con la instrucción RDTSC (read time stamp counter), primero al inicio de la búsqueda y una segunda vez al regresar la información de ruteo encontrada, con estos valores obtenemos los ciclos de reloj que tarda en realizarse la búsqueda.

Las búsquedas que se realizan son eventos independientes e idénticamente distribuidos, y se obtienen promedios de tiempo de búsqueda.

Es necesario que todas las simulaciones se realicen en el mismo equipo de computo, es decir, que las características del sistema sean las mismas para las implementaciones de los algoritmos. El equipo en donde se realizaron las simulaciones es en un servidor DELL Power Edge 1800 con procesador Intel Xeon TM 3.00 GHz con 2M en cache 2 y memoria RAM de 2GB, con sistema operativo (fedora 4) Linux.

En la siguiente sección se presenta el trabajo realizado con el algoritmo «Trie» Binario. En la sección 4.2 se presenta el trabajo realizado con el algoritmo de la Uni-

versidad de Lulea y por último en la sección 4.3 se presenta el análisis de resultados de la simulación.

4.1. “Trie” Binario

La estructura de datos que más se utiliza para realizar búsquedas es un árbol binario de recuperación de datos (Binary Trie). En el árbol binario se insertan los prefijos de red codificados en binario de la tabla de ruteo. La búsqueda se realiza recorriendo el árbol tomando como ruta los bits de la dirección IP de destino.

Se presenta a continuación el funcionamiento básico del “trie” binario.

4.1.1. Funcionamiento Básico “Trie” Binario

Construcción del “Trie” Binario

El “Trie” se construye a partir de la información contenida en la tabla de ruteo. Para construir el árbol primero se define la estructura de un nodo. Este nodo tiene una variable que guarda información de la longitud del prefijo que representa la información de ruteo para esta implementación. Esta variable es de tipo entero que se inicializa con el valor cero, lo que significa que no hay información de ruteo en el nodo. El nodo tiene también dos apuntadores, un apuntador al hijo izquierdo y otro al hijo derecho.

La construcción del “Trie” se inicia insertando en un nodo llamado raíz nuevos nodos para cada bit de cada prefijo de red contenido en la tabla de ruteo. Comenzando con el bit más significativo se realiza una comparación, si es uno se crea un nuevo nodo y se enlaza como hijo derecho de la raíz, en caso contrario cuando el

bit es cero se crea el hijo izquierdo de la raíz. Este procedimiento es iterativo, si el nodo ya existe no se crea un nodo solamente se visita para continuar la creación del árbol.

Por ejemplo si se quiere insertar el prefijo de red:

Nombre	Prefijo de Red
A	10100*

Se procede como sigue. Se inicia con el nodo raíz, se toma el bit más significativo, como este bit es uno se inserta un nodo del lado derecho, el siguiente bit es cero por lo que se inserta un nodo hacia el lado izquierdo, así hasta insertar cinco nodos y en el último nodo se guarda la información de ruteo del prefijo "A". Gráficamente el procedimiento se observa en la Figura 4.1.

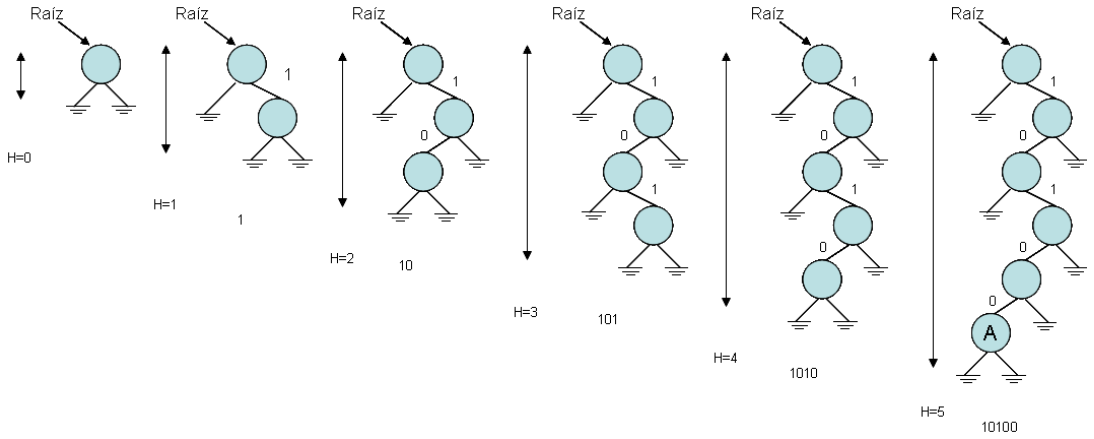


Figura 4.1: Inserta Prefijo de Red A : 10100*

Si ahora queremos insertar en el árbol anterior el siguiente prefijo de red:

Nombre	Prefijo de Red
B	1111*

Tomamos el bit más significativo y como el valor es uno, debemos insertar un nodo a la derecha pero ya existe, así que solamente se inserta si no existe el nodo, si existe se continua con el recorrido y en el último nodo se guarda la información de ruteo del prefijo “B”, ya sea que el nodo se inserte o se visite . Esto se observa en la Figura 4.2.

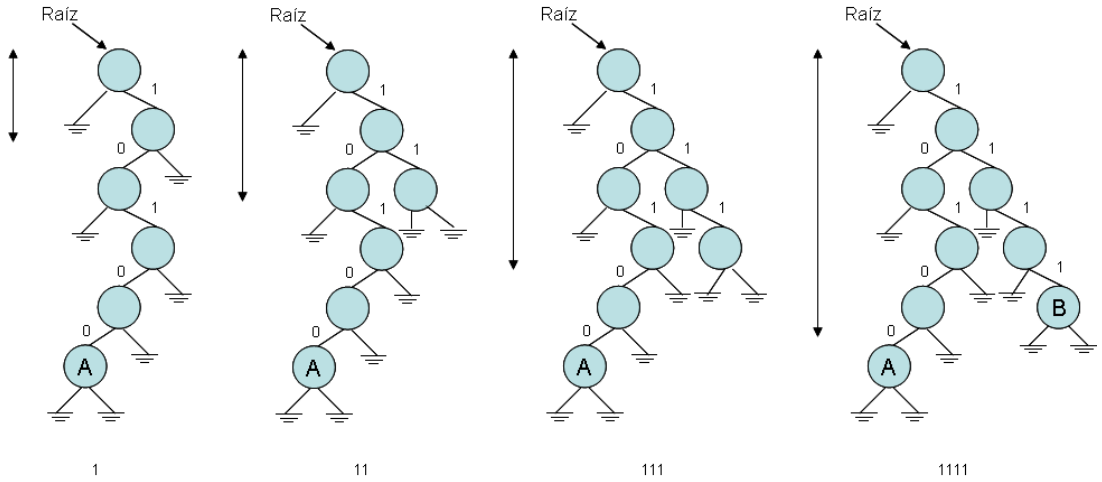


Figura 4.2: Inserta Prefijo de Red B : 1111*

Con esta manera de insertar prefijos de red se forma un árbol que contiene tanto en nodos intermedios como en nodos hoja la información de ruteo por la que deben ser re-enviados los paquetes.

Búsqueda de Prefijos del “Trie” Binario

Para la búsqueda de direcciones se recorre el árbol en base a una dirección IP de destino de 32 bits. Se compara desde el bit más significativo y se recorre el árbol bit por bit hasta que suceda cualquier de las siguientes condiciones:

1. Que se encuentre la dirección de 32 bits exactamente igual.

2. Que en una comparación el nodo que se debe visitar no existe.

El prefijo más largo se determina guardando la información de ruteo del nodo visitado solamente si es diferente de cero, ya que los nodos se inicializaron con cero y si tienen información de ruteo el valor es diferente de cero. Esto garantiza que el último nodo que se visita y que tiene información de ruteo, es el prefijo más largo que coincide con los bits más significativos de la dirección IP destino del paquete, es el BMP.

A continuación se muestra la implementación en pseudocódigo de las funciones principales de este algoritmo.

4.1.2. Implementación del Algoritmo “Trie” Binario

Inserción.

Con el archivo que almacena los prefijos de la tabla de ruteo se alimenta el generador de árbol, la función que se utilizó para generar el árbol es iterativa. A la función de inserción se le pasan como argumentos el prefijo de red en un arreglo de tamaño 32, el valor de la longitud del prefijo de red y el apuntador al nodo raíz del árbol.

```
función ins_pref(RAIZ, long_prefijo, arr_prefijo)
INICIO
    PARA i=0 HASTA i<long_prefijo HAZ
        SI arr_prefijo[i]=0 ENTONCES
            SI (RAIZ->der)==NULO ENTONCES
                RAIZ->der=inicializa_hoja()
            FIN SI
        RAIZ=RAIZ->der
        SINO
            SI (RAIZ->izq)==NULO ENTONCES
                RAIZ->izq=inicializa_hoja()
            RAIZ=RAIZ->izq
            FIN SI
        FIN SI
    FIN PARA
    RAIZ->info_ruteo=long_prefijo
FIN
```

Esta función inserta nodos y actualiza el valor de la longitud del prefijo en cada nodo en la variable prefijo, si el valor de la variable prefijo del nodo es cero indica que para ese prefijo no se asocia ninguna información de ruteo.

Si el valor de la variable prefijo del nodo es diferente de cero indica que el nodo representa un prefijo, de longitud igual al valor guardado en el nodo. Este valor es una representación de la información de ruteo que es guardada en estos nodos. Teniendo la estructura base se carga la tabla de ruteo y se generó un árbol de más de 900,000 nodos para esta tabla de ruteo que tiene 213,086 prefijos.

Búsqueda.

La búsqueda es iterativa. A la función que busca el prefijo de red se le pasan como parámetros el arreglo que contiene la dirección IP de 32 bits y el apuntador al nodo raíz. Esta función realiza un recorrido en el árbol. Con el bit más significativo se realiza la comparación en el primer nodo, si es '1' se recorre hacia el nodo derecho, si es '0' se recorre hacia el nodo izquierdo, el siguiente bit ahora es utilizado para tomar la siguiente decisión del recorrido, por lo tanto la profundidad está relacionada con la posición del bit que se compara en la dirección IP.

```
función busca_pref(RAIZ, direccion)
INICIO
ENCONTRADO=FALSO
    MIENTRAS (ENCONTRATO=FALSO)
        SI RAIZ->info_ruteo != 0 HAZ
            Informacion_de_ruteo=RAIZ->info_ruteo
        FIN SI
        SI direccion[i]=0 ENTONCES
            SI (RAIZ->der)==NULO ENTONCES
                ENCONTRADO=VERDADERO
            SI NO
                RAIZ=RAIZ->der
            FIN SI
        SINO
            SI (RAIZ->izq)==NULO ENTONCES
                ENCONTRADO=VERDADERO
            SI NO
                RAIZ=RAIZ->izq
            FIN SI
        FIN SI
    FIN PARA
FIN
```

La variable que se decidió estudiar es el tiempo que tarda una iteración, es decir un salto en el árbol. Para encontrar un prefijo es necesario realizar tantos saltos como la longitud del prefijo más uno.

Los resultados de esta simulación se muestran en la siguiente sección.

4.1.3. Resultados de la simulación del algoritmo “Trie” binario

Tomando en cuenta el análisis de la desviación estándar y la varianza mostrado en el apéndice A, se realizan búsquedas para mil direcciones de cada longitud de prefijo de 9 a 32. El algoritmo busca para cada dirección IP la información del prefijo más largo que coincide con los bits más significativos de la dirección IP destino, los promedios para la búsqueda de prefijos con respecto a su longitud se muestran en la Gráfica 4.3.

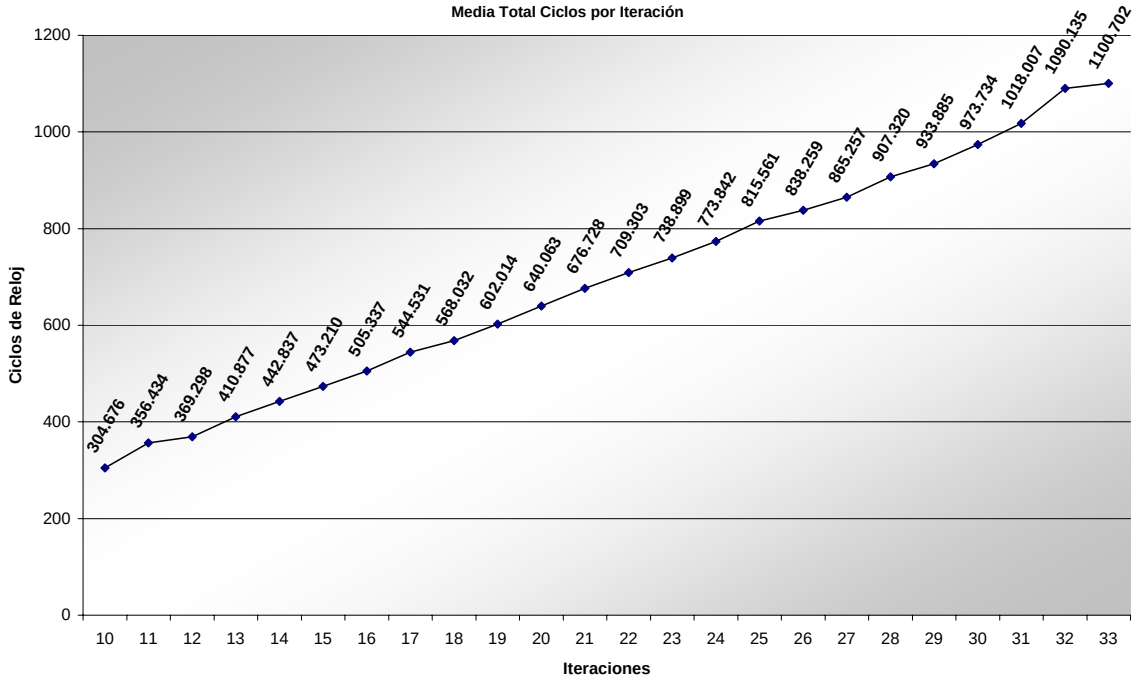


Figura 4.3: Gráfica de resultados en ciclos por segundo: cada iteración consume ciclos de reloj por lo tanto para 9 bits longitud el algoritmo realiza 10 iteraciones que son 304.676 ciclos de reloj, y para 32 bits de longitud se realizan 33 iteraciones que representa 1100.702 ciclos de reloj.

Estas mediciones son con respecto a ciclos de reloj, ya que cada iteración en promedio tarda los mismos ciclos de reloj, una iteración es un salto en el árbol, para reportar estos datos en segundos es necesario convertirlos con respecto al equipo de cómputo utilizado. La Gráfica 4.4 muestra los datos correspondientes al equipo utilizado.

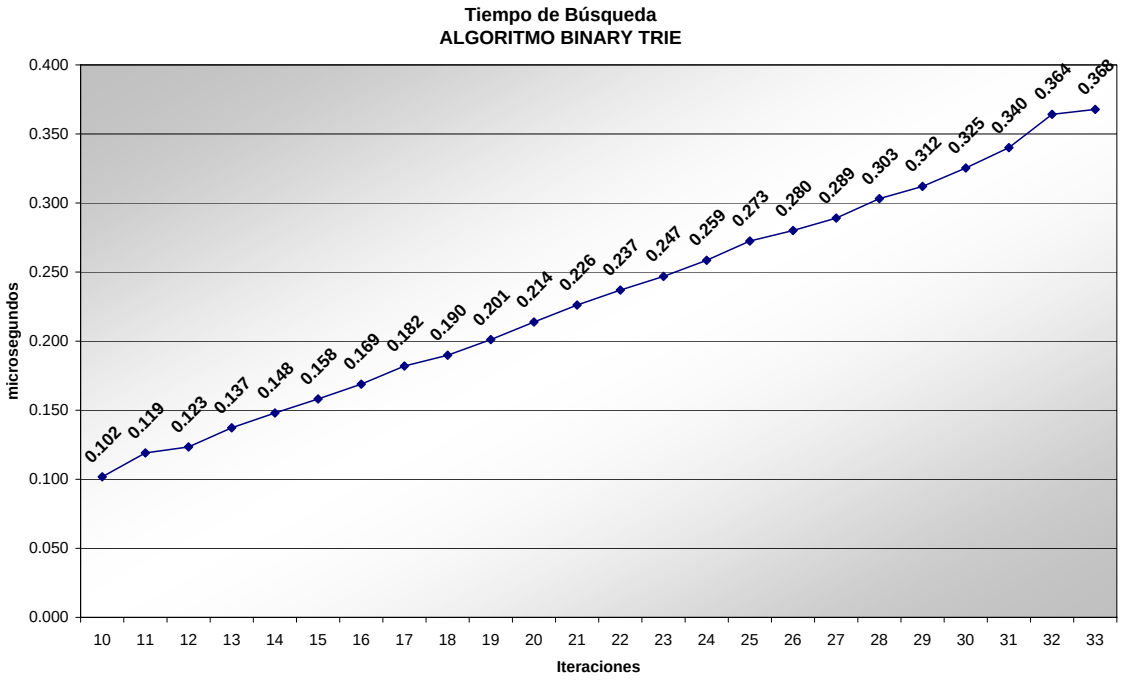


Figura 4.4: Gráfica de Resultados en Microsegundos

De aquí se puede obtener el promedio del tiempo que se tarda en realizar una iteración en el árbol y concluimos que para este equipo es de 12 nanosegundos.

4.1.4. Análisis de resultados del algoritmo "Trie" binario

Los resultados de las simulaciones, nos permiten entender cuáles son los factores que afectan la medición en el tiempo de búsqueda.

La variación en tiempo ocurre cuando se recorre el árbol, cada vez que se actualiza

el valor del prefijo más largo que se encuentra. Si en un nodo que se visita, tiene información de la interfaz, ésta es actualizada utilizando tiempo de procesamiento. Por lo que el tiempo de búsqueda está también en función de cuantas veces se actualiza este dato.

El tiempo de búsqueda está determinado por cuantos nodos recorrió y cuantas veces actualizó la información de ruteo. Para obtener los promedios mostrados en las gráficas se tomaron el tiempo completo que tardó en encontrar el BMP que incluye el tiempo de actualización y el tiempo de visitar los nodos.

En la siguiente sección se presenta el Algoritmo de la Universidad de Lulea.

4.2. Algoritmo de la Universidad de Lulea

Este algoritmo guarda la información de la tabla de ruteo en un arreglo. A la información de ruteo de cada prefijo le corresponde una entrada en el arreglo. Con la dirección IP destino se puede encontrar el índice que le corresponde dentro del arreglo y así obtener la información de ruteo.

El algoritmo propone hacer la búsqueda por secciones, cada sección trabajando con la longitud de los prefijos como se indica a continuación:

- 1^{era} Sección De 0 a 16 bits.
- 2^{da} Sección De 17 a 24 bits.
- 3^{era} Sección De 25 a 32 bits.

En la siguiente sección se explica el funcionamiento del algoritmo.

4.2.1. Funcionamiento Básico

Construcción de Tablas y Arreglos

Para encontrar el índice del arreglo en donde está la información de ruteo se construyen otros tres arreglos y una tabla donde se manejará la información de control para la búsqueda, esto se realiza por cada sección de búsqueda.

El algoritmo propone crear un arreglo con respecto a un corte en el árbol binario a diferentes profundidades. Trabajando igual para cada una de las tres secciones de búsqueda.

El árbol binario tiene que ser "completo", es decir cada nodo en el árbol sólo tiene dos posibilidades:

1. Debe tener siempre dos hijos.

2. Ningún hijo.

Es necesario empujar la información de ruteo de los prefijos internos y llevarla a las hojas del árbol, a está técnica se le conoce como “leaf pushing”.

Así el árbol es completo, con información de ruteo de los prefijos de red en las hojas como se muestra en la Figura 4.5.

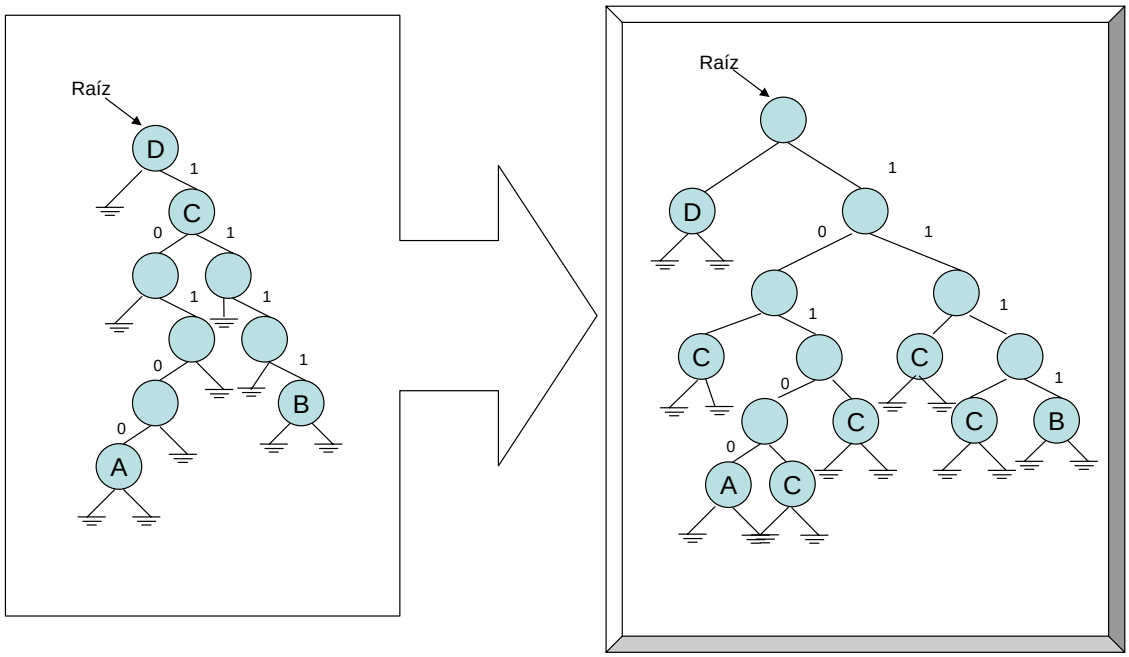


Figura 4.5: Completar árbol

Se realiza un corte del árbol en profundidad 16 y se genera el primer arreglo de tamaño $2^{16} = 65,536$ entradas. Este arreglo es binario y se llena con la información de los nodos, en la Figura 4.6 se muestra el corte.

Cada bit dentro del arreglo puede ser “cero” o “uno” con las siguientes condiciones:

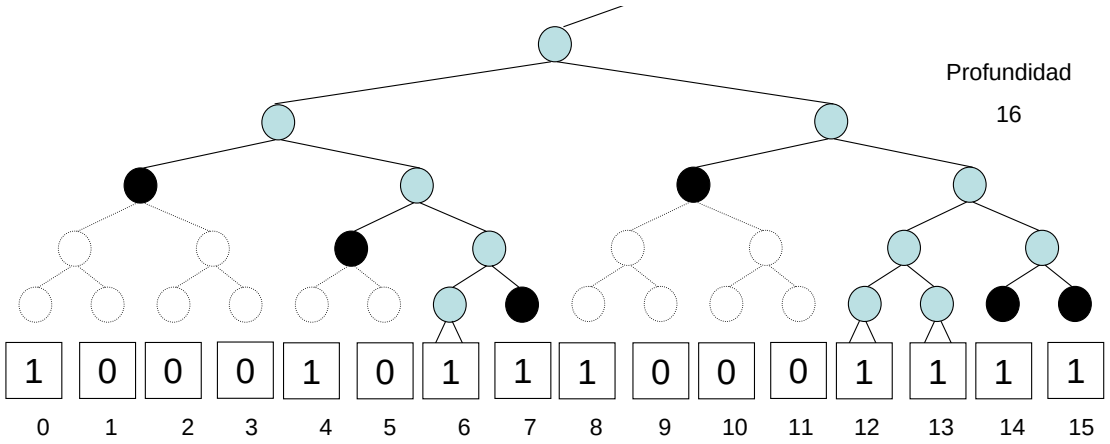


Figura 4.6: Arreglo de Bits

- Es '1' cuando el árbol continua, es decir, cuando el corte indica que esta rama continúa, a este tipo de bit se le denomina "root head".

- También es '1'no cuando en profundidad igual a 16 o menor tenemos un nodo hoja, a este tipo de bit se le llama "genuine head".

- Es '0' cuando está contenido en un rango que está cubierto por una hoja a profundidades menores a 16.

Aquí se genera el arreglo que contiene información referente a las cabeceras que se identificaron con las condiciones anteriores.

Para "root heads" se necesita guardar un apuntador a la siguiente sección del árbol donde se realizará una segunda iteración de búsqueda, es decir un apuntador a la segunda sección de búsqueda.

Para "genuine heads" se guarda el índice en donde se tiene que buscar la información de ruteo en el arreglo que contiene la tabla de ruteo.

Esta información de las cabeceras se guarda en otro arreglo donde 16 bits son

asignados para cada prefijo de red, 2 bits para identificar qué tipo de información contiene y 14 bits para apuntar al siguiente árbol de búsqueda, o como un índice que apunta a la posición de la información del siguiente salto en la tabla de ruteo.

Para saber cuál es la posición dentro del arreglo donde está la información de la tabla, se necesita generar los otros dos arreglos.

Por las características de cómo se completó el árbol, se tiene una combinación de posibilidades en el arreglo de bits. Esto dependiendo de la cantidad de bits que sean tomados para llenar los nuevos arreglos. Se observa que siempre los rangos son dados por $2^{(16-profundidad)}$. Las combinaciones posibles tomando n bits del arreglo de bits están dados por la siguiente formula:

$$a(0) = 1, a(n) = 1 + a(n-1)^2 \tag{4.1}$$

Todas las combinaciones posibles se guardan en una tabla llamada "Map Table" que tiene 678 renglones y en cada renglón se guardan 16 bits, que representan la combinación.

Un ejemplo es la Tabla 4.1, donde se observan las combinaciones posibles para cuatro bits.

00000
1000
1010
1011
1110
1111

Tabla 4.1: Map Table para 4 bits

Así, tomando 16 bits como máscara, se puede dividir el tamaño del primer arreglo

de bits que es de 35,536 entre 16, para tener el tamaño del nuevo arreglo.

Este segundo arreglo es de tamaño 4,096 y se llama arreglo “code word”, aquí se estructura información en palabras de 16 bits, 10 para tener el índice hacía la “Map Table” y 6 bits para representar un offset. Los primeros 10 bits direccionan hasta 1,024 localidades, y sólo necesitamos 678, por lo que es suficiente con este número de bits. Los 6 bits restantes van a tener información referente al número de unos que tiene la suma de máscaras de bits acumulados hasta antes de esta palabra.

Utilizando 6 bits solamente se pueden contar hasta 64 unos, por lo tanto, cada cuatro palabras se debe guardar el conteo que se lleva y comenzar de cero para contar otras 4 palabras. Este número se debe guardar en el tercer arreglo llamado “Base Index” donde cada entrada tiene información de cuatro palabras dentro del code word, este arreglo es de tamaño 1024. Dicho procedimiento se muestra en la Figura 4.7.

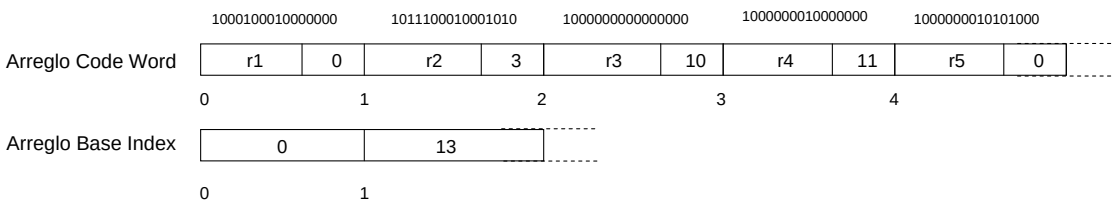


Figura 4.7: Generar Arreglos

Ya que se generan los 4 arreglos se procede a realizar la búsqueda.

Búsqueda de prefijos de red

La búsqueda en la primer sección se realiza tomando los primeros 16 bits de la dirección IP destino, para encontrar el índice del code word se utilizan 12 bits, para

el índice del arreglo base index se utilizan 10 bits y para el número de columna que se toma de la map table se utilizan 4 bits.

En la Figura 4.8 se explica como se forma el índice del arreglo de la tabla de ruteo.

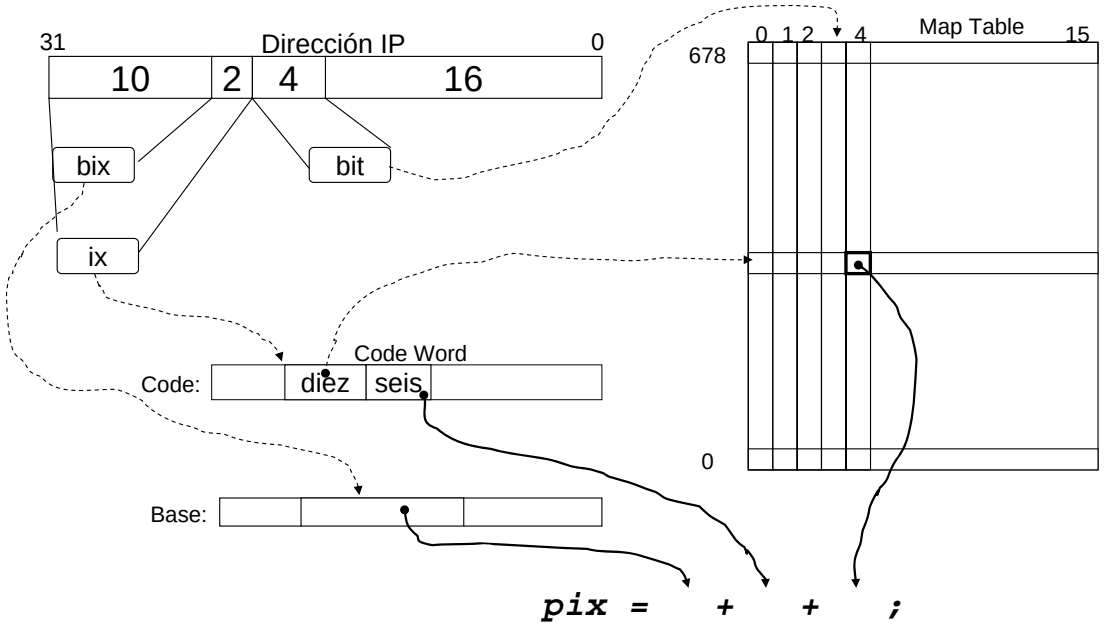


Figura 4.8: Apuntador arreglo de la Tabla de Ruteo

Para las dos secciones que faltan, la búsqueda depende de la densidad de nodos que tengan, y se decide si se realiza el mismo procedimiento que se utilizó para cortes en profundidades igual a 8 y entonces se generan los arreglos necesarios. Si no es necesario se puede utilizar el "trie" binario y hacer el recorrido como el algoritmo lo indica. A continuación se presenta la implementación del algoritmo.

4.2.2. Implementación del Algoritmo

En esta implementación se utiliza la tabla de ruteo y el árbol que se genera con el algoritmo "Trie" Binario. Se utiliza una función para completar el árbol, esta es recursiva y es un recorrido *inorden*, utilizando una pila para rellenar las hojas faltantes con la información de ruteo del prefijo que le corresponden (leaf pushing). Ya completo el árbol, se generan los primeros arreglos, el de máscara de bits, y el que contiene la tabla de ruteo con la función Genera_Vector().

```
Función Genera_Vector(RAIZ, altura, vector, arr_apunt, posición)
INICIO
CORTE=16
mi_altura=altura
    SI ( RAIZ != NULL Y mi_altura < CORTE ) ENTONCES
        Genera_Vector(RAIZ->izq, altura+1, vector, arr_apunt, posición)
        SI ( RAIZ->izq == NULO O RAIZ->der == NULO ) ENTONCES
            num_sec = CORTE - mi_altura
            sección = 2^(num_sec)
            inserta_arr_apunt(arr_apunt, RAIZ, RAIZ->prefijo)
            inserta_sec (vector, sección)
        FIN SI
        Genera_Vector(RAIZ->der, sig_alt,vector, arr_apunt, posición)
    FIN SI
    SI ( RAIZ != NULL) ENTONCES
        SI (mi_altura==CORTE) ENTONCES
            seccion=1
            inserta_arr_apunt (arr_apunt, RAIZ, RAIZ->prefijo)
            inserta_sec(vector,sección)
        FIN SI
    FIN SI
FIN
```

Teniendo el arreglo de bits, se pueden generar los arreglos code word y el base index, estos arreglos se generan con la siguiente función genera_arreglos().

```
función genera_arreglos(arr_cw, arr_base, tabla)
int offset_cw=0, base=0;
    PARA i=0 HASTA 4096 HAZ
        SI (get_mask(bit_mask,vector)) ENTONCES break FIN SI
        posit = busca_mask_maptable(bit_mask,tabla)
        offset = calcula_offset (bit_mask)
        SI (i!=0) ENTONCES
            SI (i%4 == 0) ENTONCES
                base = offset_cw
                arr_base[i/4]=arr_base[(i/4)-1] + offset_cw
                offset_cw=0
            FIN SI
        FIN SI
        arr_cw[i].ptr_maptable=posit
        arr_cw[i].offset=offset_cw
        offset_cw= offset_cw + offset
    FIN PARA
FIN
```

A esta función es necesario pasarle la tabla Map Table para hacer la búsqueda en la tabla, como la tabla no cambia, está guardada en un archivo tipo texto y es cargada a una matriz de 678 X 16.

Para la búsqueda se toman los bits de la dirección IP necesarios para el cálculo de todos los índices. Se suman todos los valores de los arreglos y se tiene el índice del arreglo que contiene la tabla de ruteo.

```
result.ptr = arr_base[num_bix] + arr_cw[num_ix].offset + off;
```

Los resultados de la simulación se presentan en la siguiente sección.

4.2.3. Resultados de la Simulación

Se generan 32,000 búsquedas controladas con direcciones compuestas por un prefijo de red, más los ceros que faltan para completar la dirección IP.

Por cada prefijo desde la longitud 9 hasta la longitud 16 se realizan 4,000 búsquedas, cada una se repite 1,000 veces para obtener un valor medio. Los resultados obtenidos en la simulación se muestran en la Figura 4.9:

El valor medio de la búsqueda en la primera sección es de 102.30264 ciclos de reloj.

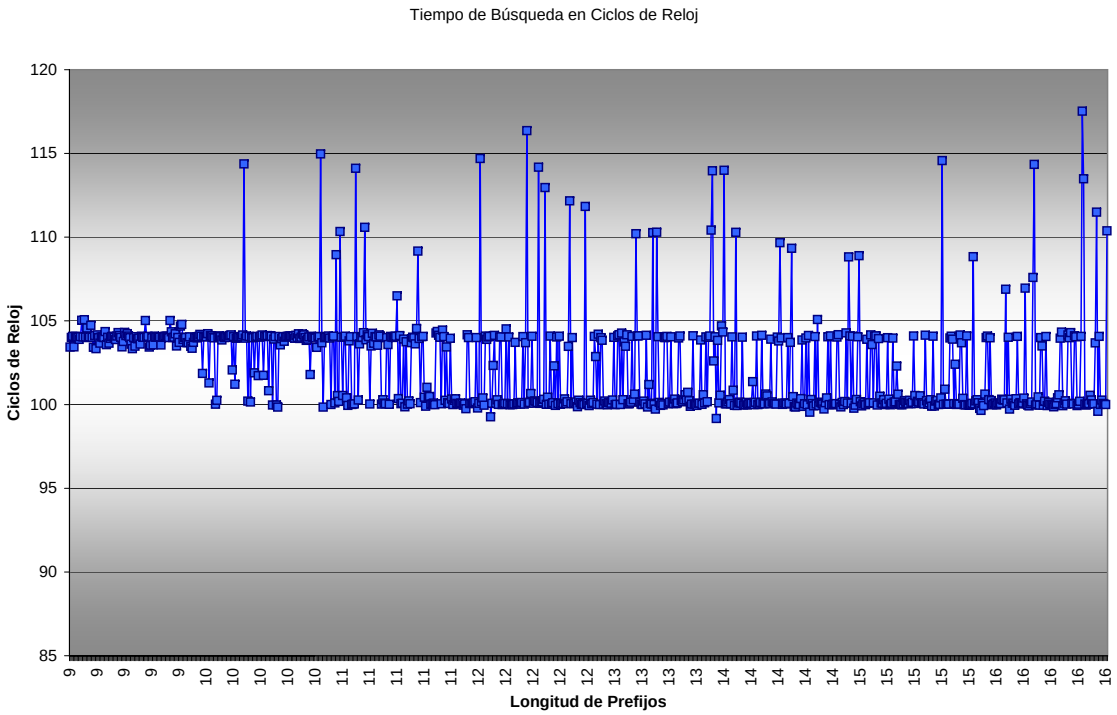


Figura 4.9: Gráfica de Resultados

4.2.4. Análisis de Resultados

Los resultados obtenidos indican los accesos realizados a memoria de los tres arreglos para encontrar el índice del arreglo donde se encuentra la información de ruteo

La búsqueda en este algoritmo para la primera sección es de 34.1 nanosegundos.

La implementación de este algoritmo es complicada, ya que se propone tener un árbol completo, y que este procedimiento se realice en cada una de las secciones. Teniendo en cuenta el tiempo de construcción de los arreglos por sección, el principal problema que tiene este algoritmo es su tiempo de actualización.

Únicamente se tiene resultados para la primer sección, ya que la búsqueda en las siguientes secciones es dependiente de la densidad de prefijos en donde se deba

buscar, con este parámetro se decide si se aplica este algoritmo o simplemente el algoritmo “trie” binario.

En la siguiente sección se realiza una comparación entre los dos algoritmos implementados.

4.3. Análisis de resultados de los algoritmos implementados

En la Figura 4.10 tomando los valores medios, se genera la comparación de tiempos de búsqueda.

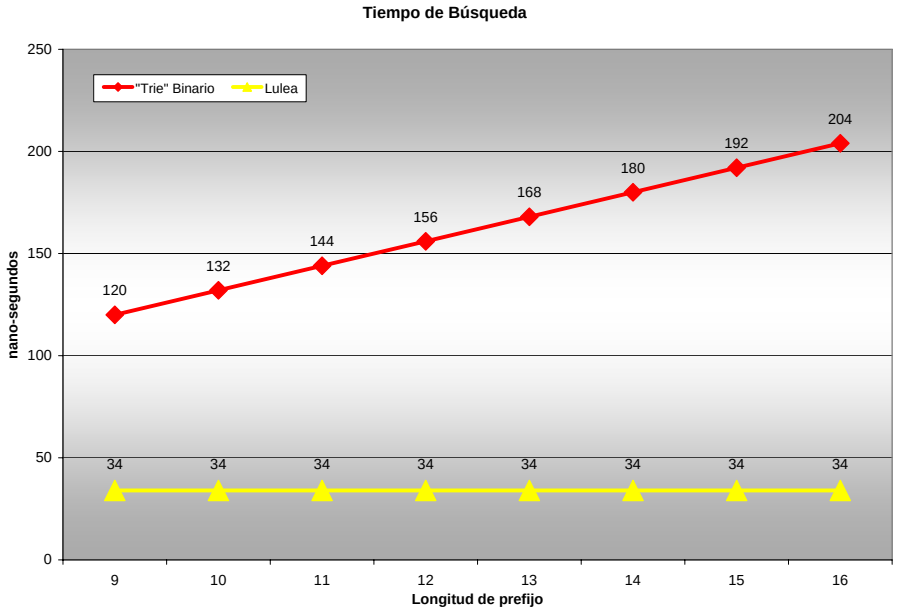


Figura 4.10: Gráfica de comparación de resultados

Estos resultados son parciales, ya que para el algoritmo de la Universidad de Lulea se generaron búsquedas solamente en la primera sección.

Es visible que el algoritmo de la Universidad de Lulea consume menos tiempo de búsqueda que el "Trie" Binario.

En la siguiente sección se explica nuestra propuesta, que funciona con arreglos como estructuras básicas para realizar búsquedas binarias. Este esquema transforma la búsqueda de prefijos en una búsqueda exacta, introduciendo redundancia en otros arreglos para encontrar la información de ruteo.

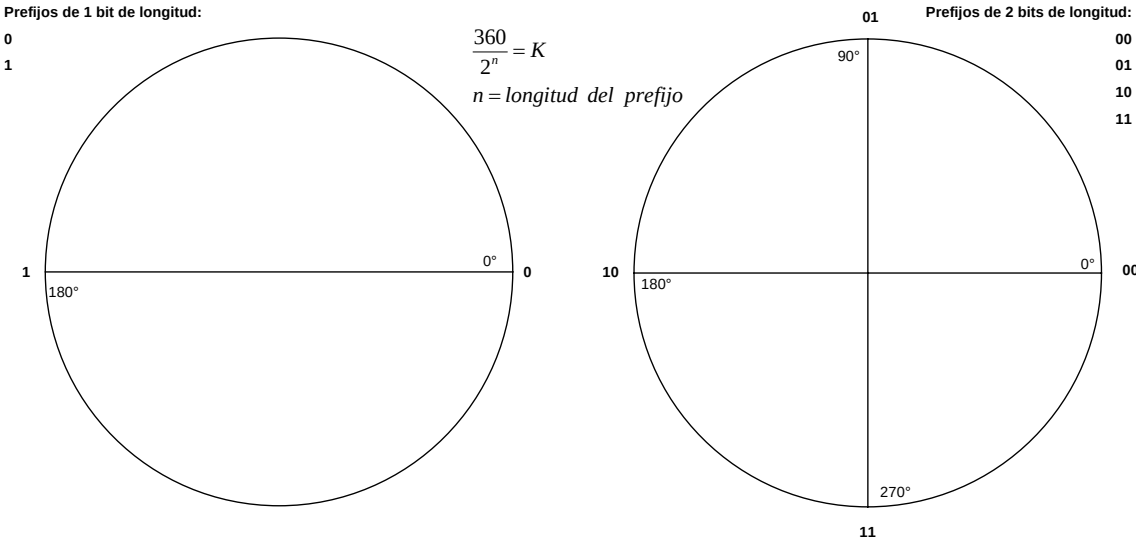
Capítulo 5

Algoritmo UAM

La idea central de este algoritmo es cambiar el tipo de búsqueda: en lugar de hacer una búsqueda directa de prefijos aplicamos una función que permita convertirla en una búsqueda exacta. Con una función se mapean los prefijos de red de la tabla de ruteo, asignándoles un valor que se puede guardar en una estructura de arreglo ordenado. El mapeo le asigna un valor a cada prefijo de red, este valor representa el ángulo que le corresponde a este prefijo de red dentro de una circunferencia, tal y como se explicará con más detalle en las secciones que siguen. Se utilizan sistemas periódicos que se pueden representar como una circunferencia, estos sistemas permiten asignar un mismo ángulo a diferentes prefijos, de longitud y valor diferente, pero con los mismos bits más significativos. Por lo tanto, cuando se realiza una búsqueda, se aplica la función de mapeo a la dirección IP y se obtiene el valor del ángulo que le corresponde dentro de la circunferencia, este ángulo se busca dentro del arreglo ordenado y se obtiene la información del prefijo de red más largo que coincide con los bits más significativos con la dirección IP. En la siguiente sección se presenta cómo se inició con esta idea, dando un preámbulo a las características del Algoritmo UAM.

5.1. Fundamentos

La idea parte de que en una circunferencia se puede mapear cualquier prefijo de red. La circunferencia se puede dividir dependiendo de la longitud del prefijo: los 360 grados de la circunferencia se dividen entre el número de posibles prefijos para una longitud dada. El valor del prefijo nos da la posición en grados que le corresponde dentro de la circunferencia. Por ejemplo, para prefijos de 1 bit de longitud tenemos dos posibilidades 0 ó 1, por lo tanto, dividimos a la circunferencia en dos partes, al prefijo 0 se asocia con el ángulo 0° y para el prefijo 1 se asocia con el ángulo 180°. La semicircunferencia superior de [0°,180°) representa a las direcciones cubiertas por el prefijo 0 y de [180°,360°) representa a las direcciones cubiertas por el prefijo 1.



$$ángulo = (valor\_prefijo)(K)$$
$$ángulo = (valor\_prefijo)(\frac{360}{2^1})$$
$$ángulo = (valor\_prefijo)(180)$$

$$ángulo = (valor\_prefijo)(K)$$
$$ángulo = (valor\_prefijo)(\frac{360}{2^2})$$
$$ángulo = (valor\_prefijo)(90)$$

Figura 5.1: Mapeo de prefijos de 1 y 2 bits de longitud. Del lado izquierdo se esquematiza una circunferencia dividida por los prefijos de un bit de longitud, en el lado derecho la circunferencia esquematiza la posición de los prefijos de dos bits de longitud.

En el caso de prefijos de 2 bits de longitud se tienen 4 prefijos posibles: 00,01,10,11; por lo tanto la circunferencia se divide entre estas cuatro posibilidades. La distancia entre prefijos es de 90°, y se asignan los ángulos de la siguiente manera: 00 se asocia con 0°, 01 se asocia con 90°, 10 se asocia con 180° y 11 se asocia con 270°. La circunferencia tiene cuatro secciones que son: [0°,90°) cubierta por el prefijo 00; [90°,180°) cubierta por el prefijo 01; [180°,270°) cubierta por el prefijo 10; y [270°,360°) cubierta por el prefijo 11. La Figura 5.1 muestra cómo se distribuyen los prefijos de longitud 1 y 2 y cómo se representan mediante ángulos en las circunferencias.

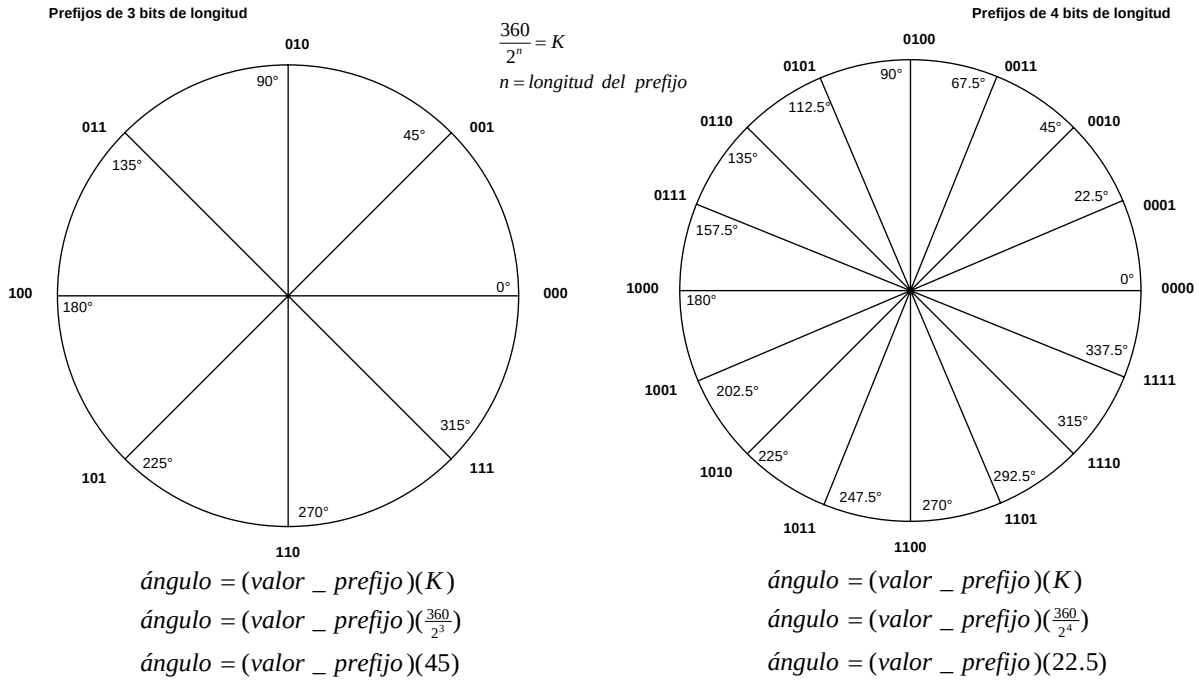


Figura 5.2: Mapeo de prefijos de 3 y 4 bits de longitud. Del lado izquierdo, la circunferencia se divide en ocho partes, la circunferencia del lado derecho se divide en dieciséis partes, de aquí se establece la relación entre la longitud y el número de partes en que se divide a la circunferencia.

Para prefijos de 3 bits de longitud el número de prefijos distintos que se pueden tener es 8 (2^3). Dividiendo la circunferencia entre estas ocho posibilidades, los

prefijos se ubican a distancias de 45°. Con prefijos de 4 bits de longitud se divide a la circunferencia en 16 (2^4) partes teniendo una distancia entre prefijos de 27.5°. En la Figura 5.2 se muestran las circunferencias que contienen a los prefijos de 3 y 4 bits de longitud.

Para 5 bits de longitud tenemos 32 (2^5) prefijos posibles, por lo tanto cada prefijo va a cubrir rangos de 11.25°. En la Figura 5.3 podemos observar a todos los prefijos de 5 bits distribuidos en la circunferencia.

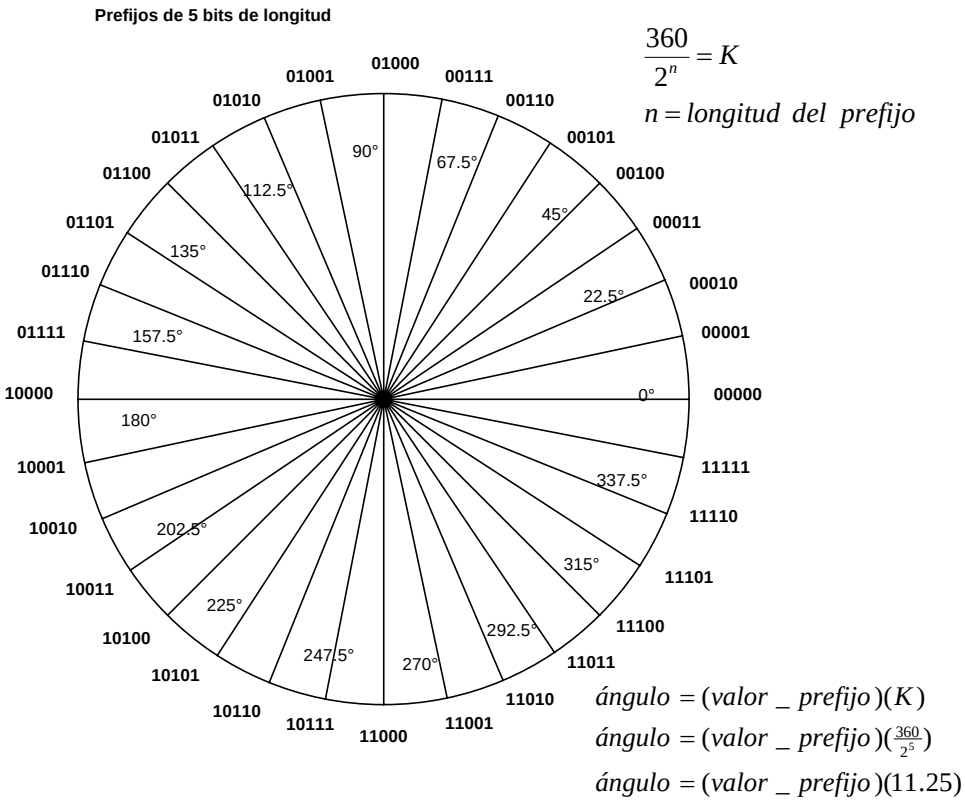


Figura 5.3: Mapeo de prefijos de 5 bits de longitud. La circunferencia está dividida en treinta y dos partes establecido por las ecuaciones que se muestran en la parte inferior del gráfico.

La ubicación en grados (ángulo) de un cierto prefijo dentro de la circunferencia se puede determinar a partir del número (n) de bits de longitud del prefijo y de su valor numérico mediante las ecuaciones 5.1 y 5.2.

$$K = \frac{360^\circ}{2^n} \tag{5.1}$$

$$ángulo = (valor)(K) \tag{5.2}$$

En lo que sigue, para describir el funcionamiento del algoritmo se consideraran prefijos de 5 bits de longitud, lo cual todavía permite ilustrarlo de manera gráfica. Las longitudes mayores dificultan la descripción gráfica, sin embargo, todo lo que se aplica a prefijos de esta longitud (5 bits) es aplicable a prefijos de longitudes mayores como los que se pueden tener en el protocolo IPv4 (hasta 32 bits). Con la dirección de 5 bits de longitud podemos generar hasta 62 prefijos de red, teniendo longitudes desde 1 hasta 5 bits. El espacio de prefijos que se puede generar se observa en la Tabla 5.1.

Tabla completa para 5 bits					
longitud	**1**	**2**	**3**	**4**	**5**
Prefijos de Red	0	00	000	0000	00000
	1	01	001	0001	00001
		10	010	0010	00010
		11	011	0011	00011
			100	0100	00100
			101	0101	00101
			110	0110	00110
			111	0111	00111
				1000	01000
				1001	01001
				1010	01010
				1011	01011
				1100	01100
				1101	01101
				1110	01110
				1111	01111
					10000
					10001
					10010
					10011
					10100
					10101
					10110
					10111
					11000
					11001
					11010
					11011
					11100
					11101
					11110
					11111

Tabla 5.1: Espacio de prefijos para longitudes de 1 a 5 bits.

La ubicación de los prefijos de 1,2,3,4 y 5 bits de longitud se esquematiza en una sola circunferencia en la Figura 5.4, y se observa que algunos ángulos mapean a más de un prefijo de red, esto es lo importante en este sistema de clasificación como se vera a continuación.

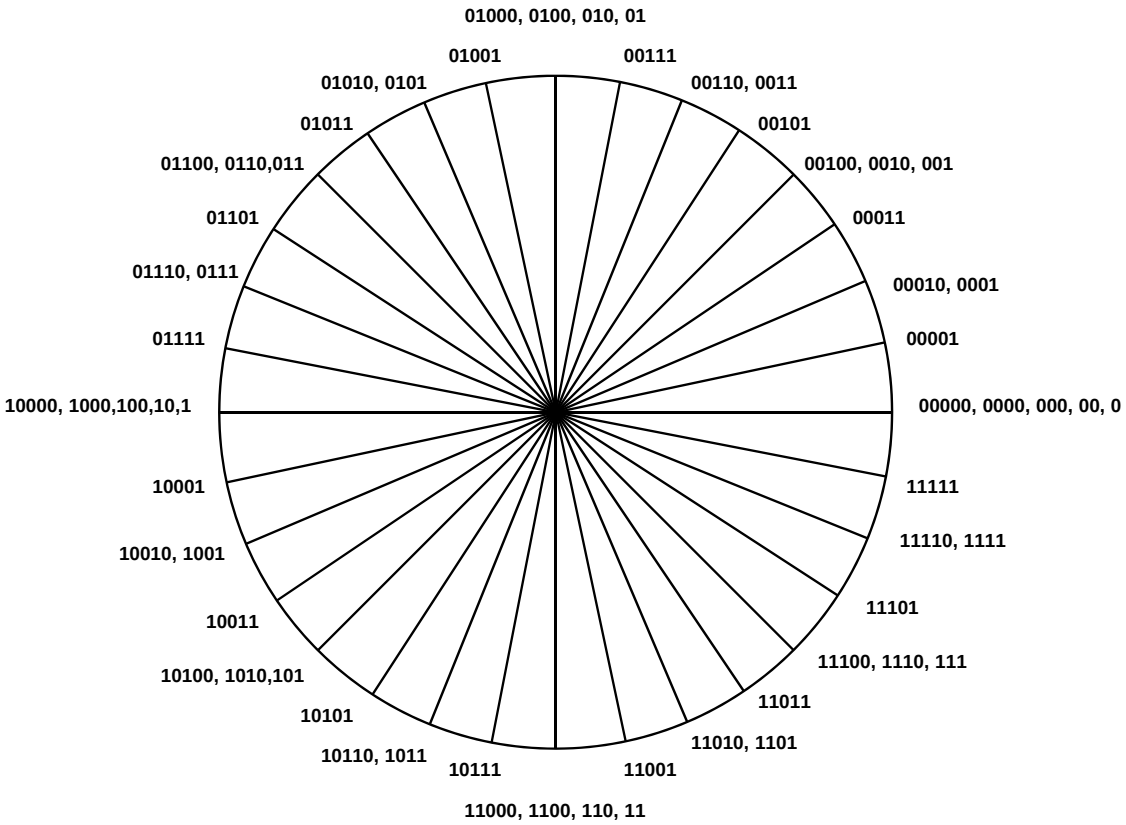


Figura 5.4: Circunferencia del espacio de prefijos de 5 bits. La circunferencia está dividida en treinta y dos partes en donde se relaciona todos los prefijos de red con su respectivo ángulo.

La tabla de ruteo se representará por una estructura donde la llave para la búsqueda es el ángulo asociado con el o los prefijos. Por lo tanto en la entrada correspondiente a cada ángulo se guarda información de ruteo de cada uno de los

prefijos que tiene asociados. Para esto se propone que cada entrada apunte a un arreglo que contiene un número de entradas igual al número de bits de la dirección IP, tal como se observa en la Figura 5.5. El arreglo no está completamente lleno por que los prefijos se distribuyen en la circunferencia con respecto a la longitud del prefijo.

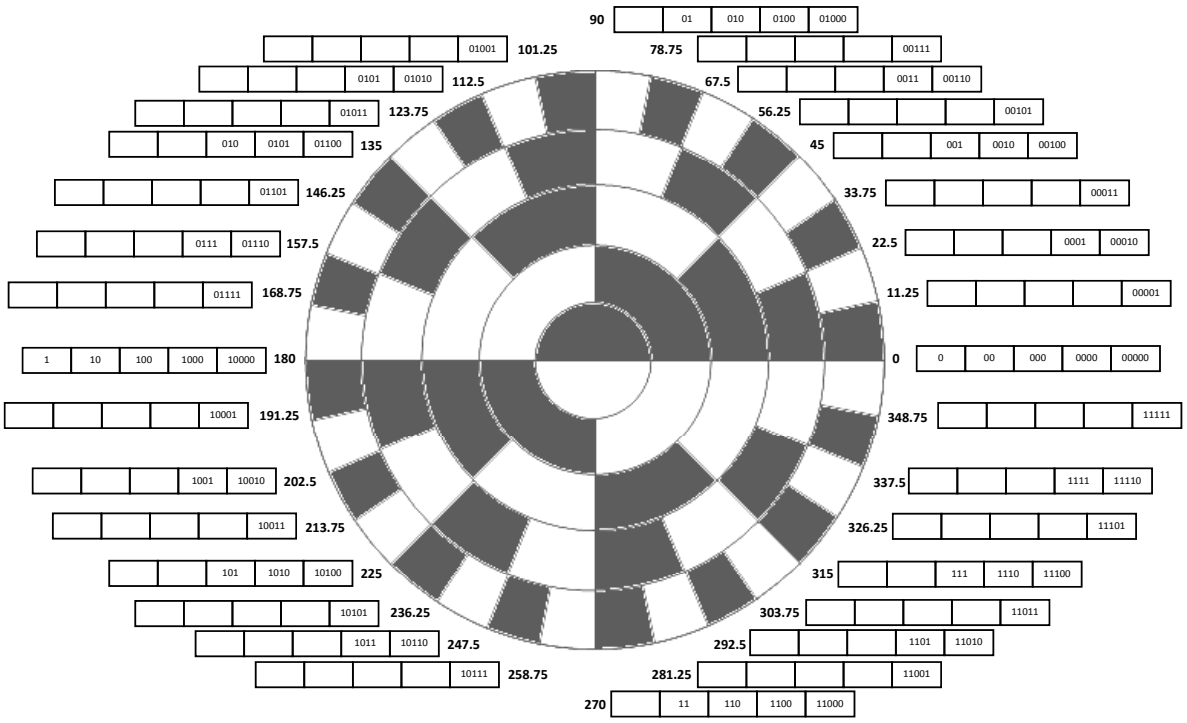


Figura 5.5: Arreglos que contienen la información de los prefijos de red. Se utiliza un arreglo que tiene un número de entradas igual a la longitud de la dirección IP, para este ejemplo cinco entradas, llamado arreglo de longitud de prefijos y es usado para guardar la información de ruteo de los prefijos.

Teniendo este esquema, la estructura para representar la tabla de ruteo puede ser un arreglo donde están los ángulos que representan a los prefijos, este arreglo es ordenado. Cada entrada de este arreglo se relaciona con una serie de arreglos donde cada entrada representa una longitud de prefijo. Esto se muestra en la Figura 5.6

Arreglos de ángulos y prefijos para el espacio de prefijos de 5 bits de longitud

Arreglo de ángulos	0	11.25	22.50	33.75	45	56.25	67.50	78.75	90	101.25	112.50	123.75	135	146.25	157.50	168.75	180	191.25	202.50	213.75	225	236.25	247.50	258.75	270	281.25	292.50	303.75	315	326.25	337.50	348.75
Prefijos de 1 bit	0																1															
Prefijos de 2 bits	00								01								10								11							
Prefijos de 3 bits	000				001				010				011				100				101				110				111			
Prefijos de 4 bits	0000		0001		0010		0011		0100		0101		0110		0111		1000		1001		1010		1011		1100		1101		1110		1111	
Prefijos de 5 bits	00000	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	10010	10011	10100	10101	10110	10111	11000	11001	11010	11011	11100	11101	11110	11111

Figura 5.6: Arreglos de ángulos y de longitud de prefijos. Los ángulos de la circunferencia son guardados en un arreglo ordenado donde el índice también se utiliza en los arreglos de longitud de prefijos, en estos arreglos es donde se guarda la información de ruteo de los prefijos, es importante observar que no está lleno el arreglo de prefijos.

Como ejemplo, para observar como se distribuyen los prefijos de red de una tabla de ruteo, veamos el caso de los prefijos listados en la Figura 5.7.

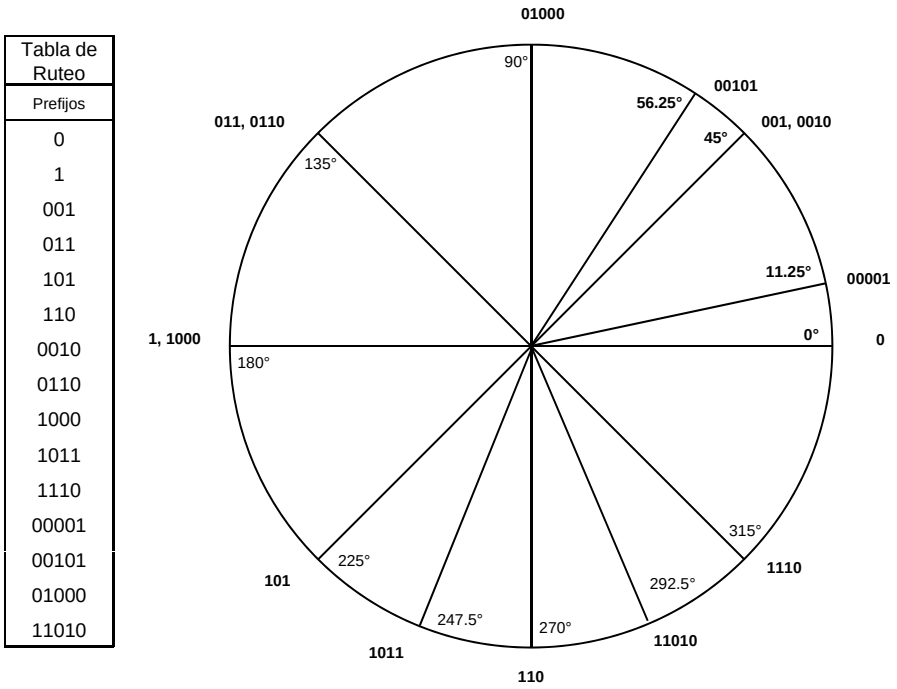


Figura 5.7: Mapeo de la Tabla de Ruteo en la Circunferencia. En esta circunferencia se observa todos los prefijos de red asociados con el ángulo que les corresponde.

Los prefijos de esta tabla se mapean en ángulos dentro de una circunferencia como se puede ver en la Figura 5.7. Se observa además que para tres ángulos se tienen asociados prefijos de diferentes longitudes: con el ángulo de 45° se asocian los

prefijos de red 001 y 0010; con el ángulo de 135° se asocian los prefijos de red 011, 0110; y con el ángulo de 180° se asocian los prefijos de red 1 y 1000. La información representada en esta circunferencia se refleja en la estructura de arreglos (mostrados como renglones) de la Figura 5.8.

Arreglos de ángulos y prefijos para el ejemplo de 5 bits

Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.50	270	292.50	315
Prefijos de 1 bit	0						1					
Prefijos de 2 bits												
Prefijos de 3 bits			001			011		101		110		
Prefijos de 4 bits			0010			0110	1000		1011			1110
Prefijos de 5 bits		00001		00101	01000						11010	

Figura 5.8: Arreglo de ángulos y arreglos de prefijos para el ejemplo de 5 bits de longitud. En estos arreglos se esquematizan los ángulos que existen para la tabla de ruteo y los arreglos de longitud de prefijos en donde se guarda la información de ruteo.

El número de arreglos necesarios para IPv4 son 33, ya que el número de bits de la dirección IP es 32 y, como se tiene un arreglo por cada longitud, habrá un arreglo para los prefijos de red desde 1 hasta 32 bits de longitud, más otro arreglo que guarda la información de la existencia de ángulos en la circunferencia.

5.2. Búsqueda

El Algoritmo UAM está hecho para encontrar el prefijo más largo coincidente con los bits más significativos de la dirección IP de destino. Para lograr esto, la dirección IP de destino se mapea mediante las ecuaciones 5.3 y 5.4 de la misma forma que los prefijos, con la diferencia de que no se toma en cuenta el valor numérico de los 32 bits de la dirección sino que solamente se considera el valor numérico de la dirección IP truncada a los m bits más significativos (donde m es la longitud máxima de prefijo existente en la tabla de ruteo), así se obtiene el ángulo que le corresponde a cada dirección IP en la circunferencia.

$$K_{max} \ = \ \frac{360}{2^m} \ , \ m \ = \ Longitud\ máxima\ de\ prefijo \tag{5.3}$$

$$ángulo = (valor\ numérico\ de\ la\ dirección\ IP\ truncada)(K_{max}) \tag{5.4}$$

La búsqueda del prefijo que corresponde a una dirección IP dada puede corresponder con uno de los tres casos particulares que a continuación se detallan.

5.2.1. Caso 1

El mejor caso es que al realizar la búsqueda se encuentra que en la circunferencia existe la asociación de uno o más prefijos de red de la tabla de ruteo para el ángulo que se calculó para la dirección IP. Entonces, si existen varios, tomamos el prefijo de mayor longitud que esté asociado a este ángulo ya que éste es el que mejor coincide con esta dirección IP.

Ejemplo 1:

Si se busca la dirección 00100, utilizando las ecuaciones 5.3 y 5.4 (con $m = 5$ ya que tenemos prefijos de hasta 5 bits de longitud) vemos que le corresponde el ángulo de 45°. En la figura 5.9 se muestra la posición que tiene en la circunferencia.

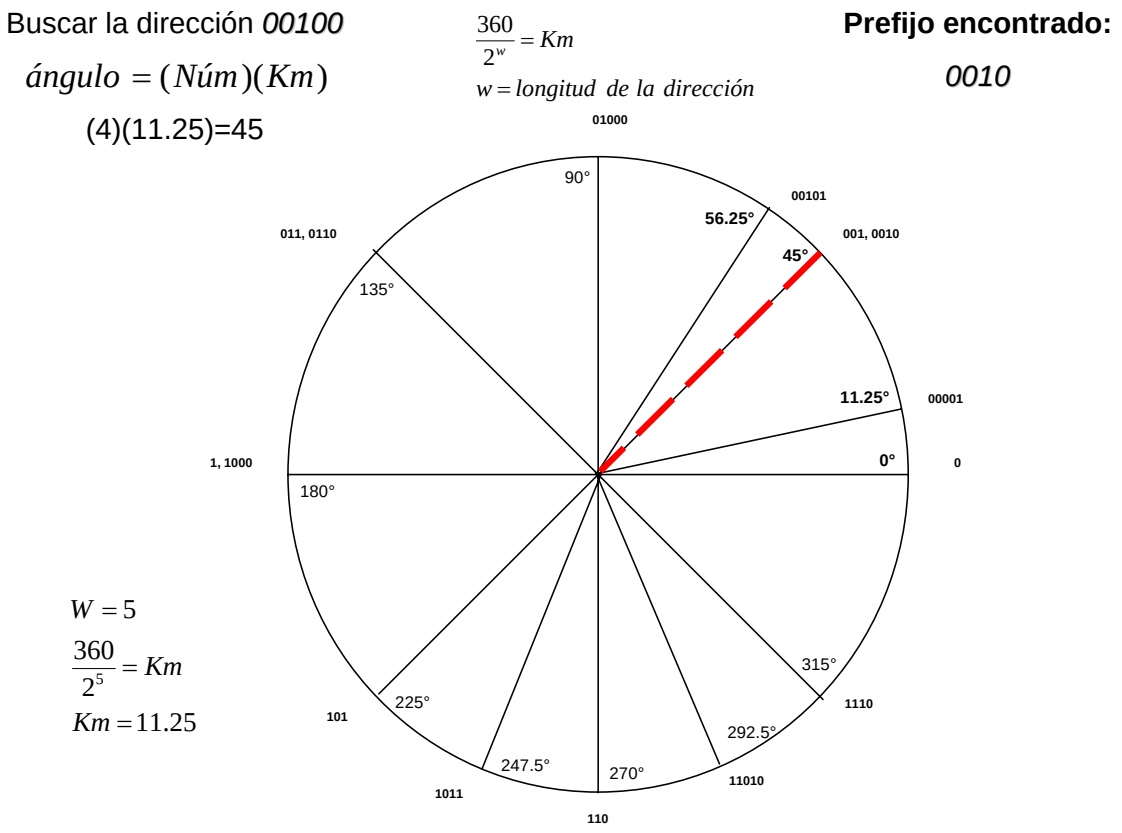


Figura 5.9: Localización de la dirección IP 00100. En esta figura se busca el prefijo 00100 que se asocia con el ángulo de 45°, como existe este ángulo, se reporta como solución el prefijo de mayor longitud asociado al ángulo.

El ángulo de 45° tiene asociados dos prefijos, por lo tanto, se elige el más largo. Así, se puede concluir que el prefijo 0010 es el prefijo de red de mayor longitud en la tabla de ruteo que coincide con esta dirección IP.

5.2.2. Caso 2

Puede suceder también que el ángulo que corresponde a la dirección IP que se busca no esté mapeado con ningún ángulo (prefijo de red). En este caso se recorre la circunferencia buscando el primer ángulo menor inmediato, es decir, se busca en sentido de las manecillas del reloj el ángulo más cercano y se realiza la búsqueda solamente entre los prefijos que estén asociados a este ángulo. Sin embargo, a diferencia del caso anterior, NO se puede simplemente tomar el prefijo de red de mayor longitud ya que no fue éste el ángulo que le corresponde directamente, por lo tanto no está garantizado que el prefijo más largo que tiene asociado coincida con la dirección IP. Ahora se debe realizar una comparación de la dirección IP con todos los prefijos asociados a este ángulo: desde el prefijo de mayor longitud hasta el prefijo de menor longitud hasta encontrar el prefijo de red que corresponde.

Ejemplo 2: Si ahora se buscara la dirección IP: 01110, al aplicar las ecuaciones 5.3 y 5.4 se tiene que el ángulo que le corresponde es 157.5°, pero este ángulo no tiene asociada información de ningún prefijo de red por lo que se debe recorrer la circunferencia hasta encontrar el primer ángulo menor que exista.

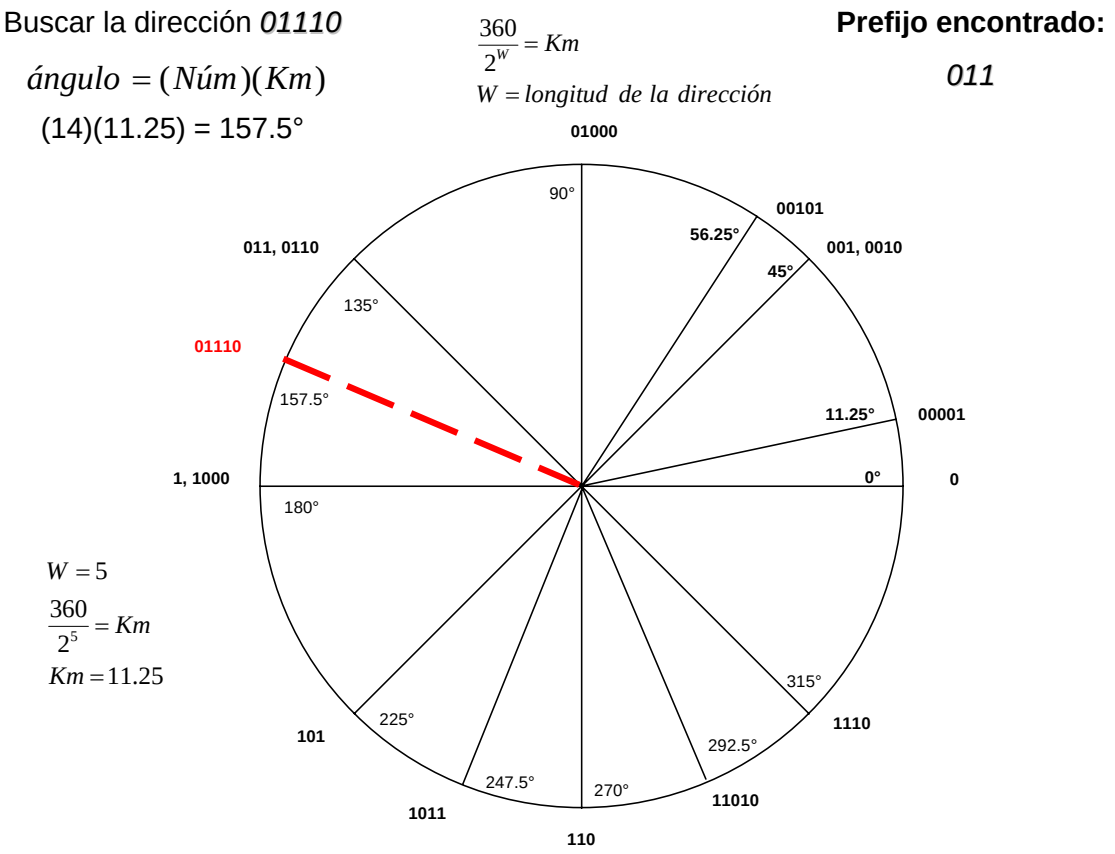


Figura 5.10: Localización de la dirección IP 01110. En la figura se busca el prefijo 01110 que se mapea con el ángulo 157.5°, este ángulo no existe entonces recorremos la circunferencia en sentido de las manecillas del reloj hasta encontrar un ángulo que corresponda con algún prefijo y buscamos en este ángulo un prefijo que sea solución.

El ángulo próximo menor es el ángulo de 135° (ver figura 5.10) y está asociado con dos prefijos de red: el 0110 y el 011. Si simplemente reportáramos el prefijo más largo cometeríamos un error ya que 0110 no corresponde con la dirección IP 01110, es por esto que se comparan todos los prefijos desde el más largo hasta el más corto. Para este ejemplo solamente resta comparar con el prefijo de tres bits y resulta que éste es el prefijo más largo coincidente con la dirección IP.

5.2.3. Caso 3

El peor caso, desde el punto de vista de los tiempos de búsqueda, es cuando no existen prefijos asociados con el ángulo que le corresponde a la dirección IP y además, al buscar en los prefijos asociados con el ángulo menor inmediato (de acuerdo a lo explicado en el caso 2) no se encuentra una solución.

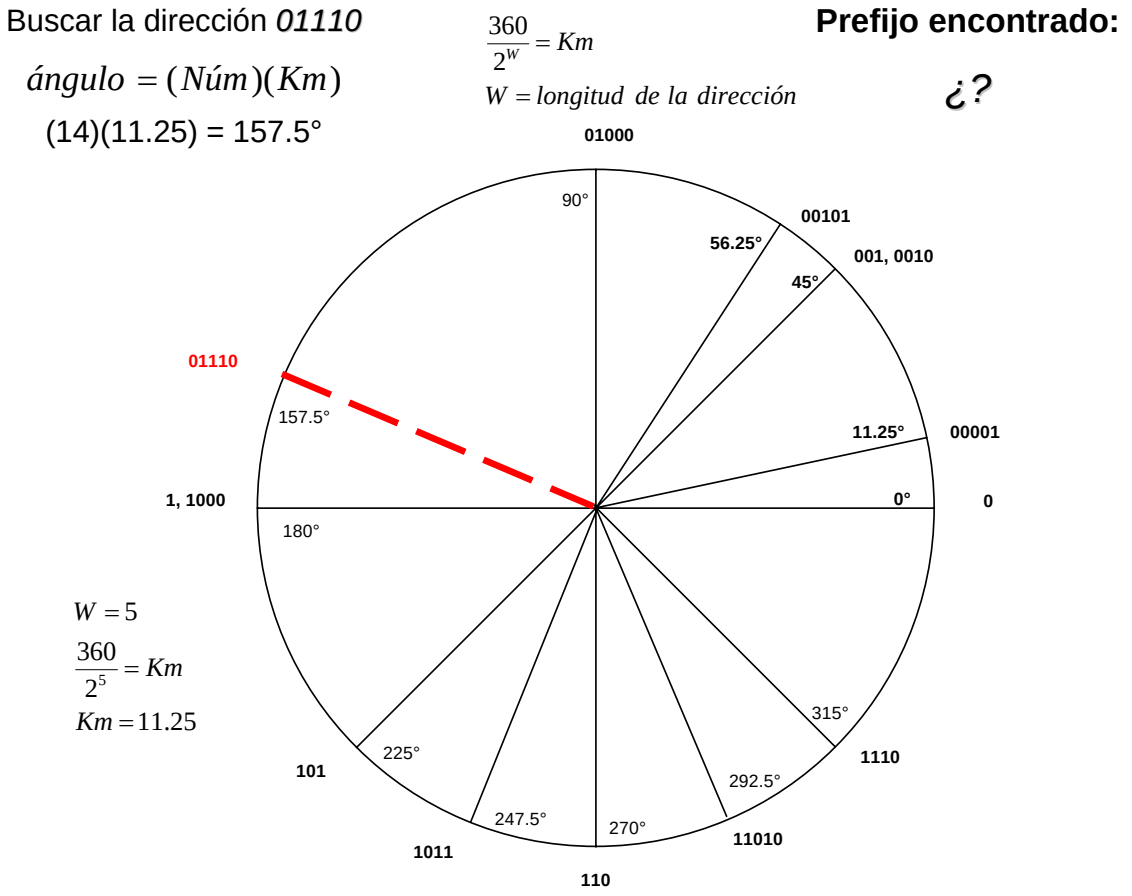


Figura 5.11: Error en la búsqueda de la dirección IP 01110. En la figura se busca el prefijo 01110 pero ahora no hay información que sea solución en el primer ángulo que se encuentra, pero existe el prefijo 0 que es la única solución pero en este esquema no se tiene información de este prefijo.

Ejemplo 3: Para dar un ejemplo de esta situación suponemos que no existen prefijos asociados con el ángulo de 135° y que se busca la misma dirección del ejemplo 2, la cual es: 01110. De acuerdo a lo explicado hasta aquí para los casos 1 y 2, como

el ángulo que corresponde con esta dirección no está asociado con ningún prefijo, se recorre la circunferencia hasta encontrar el ángulo próximo menor asociado con algún prefijo, que en este ejemplo es el ángulo de 90° (ver la figura 5.11). En este ángulo se tiene la información del prefijo 01000 que no coincide con la dirección 01110 que se está buscando. De la figura 5.11 se puede ver que el único prefijo de red que coincide con la dirección IP es el prefijo 0 (de un bit de longitud) que está asociado con el ángulo de 0 grados dentro de la circunferencia, sin embargo, no hay ninguna información en el ángulo de 90° que indique esto. Por lo tanto, para remediarlo, debemos hacer que para el ángulo de 90° también se tenga disponible la información de los prefijos de menor longitud, así que utilizamos el arreglo con un número de entradas igual al número de bits de la dirección IP donde cada entrada representa una longitud de bits del prefijo de red. Este arreglo estará totalmente lleno para algunos ángulos pero en otros ángulos tendrá espacios vacíos, estos espacios vacíos es donde falta la información de prefijos de menor longitud.

Para los casos 1 y 2, el mapeo genera toda la información necesaria ya que el arreglo de ángulos y los arreglos de prefijos bastan para encontrar una solución. Para el caso 3 es necesario hacer un procedimiento extra para que cada entrada del arreglo de ángulos contenga también la información de los prefijos de menor longitud que están mapeados a ángulos de menor valor. En la siguiente sección se explica la solución adoptada para atacar este problema al introducir la noción de “cobertura”.

5.2.4. Cobertura de los prefijos

Para que cada una de las entradas del arreglo de ángulos contenga información de los prefijos de red de menor longitud que pudieran terminar como resultados de la búsqueda, se propone que al insertar cada nuevo prefijo de red se realice un procedimiento para propagar esta información en las entradas de los ángulos que deben considerar la existencia de éste nuevo prefijo. Así, se define a la cobertura de un prefijo como el intervalo en grados (ángulos) en donde debe ser considerado como una posible solución.

La cobertura es el intervalo que tiene como límite inferior cerrado al ángulo que le asignó el mapeo con la ecuación 5.5 y el límite superior abierto es la suma de este ángulo más la constante K de la ecuación 5.8. Como se explica a continuación, a través de este atributo se podrá saber, para cada prefijo, a cuáles ángulos deben copiar su información.

$$limite\_inferior = ángulo \tag{5.5}$$

$$limite\_superior = ángulo + K \tag{5.6}$$

$$Donde \quad K \ = \ \frac{360}{2^n} \tag{5.7}$$

$$n \ = \ Longitud\ del\ prefijo \tag{5.8}$$

En la Figura 5.12 se esquematizan las circunferencias que se generan para todos los prefijos con longitudes desde uno hasta cinco bits que utilizamos en estos ejemplos. Estas circunferencias se representan concéntricas y de diferentes radios, siendo la circunferencia de menor radio la que representa a los prefijos de red de 1 bit de longitud y la de mayor radio representa a los prefijos de red de 5 bits de longitud.

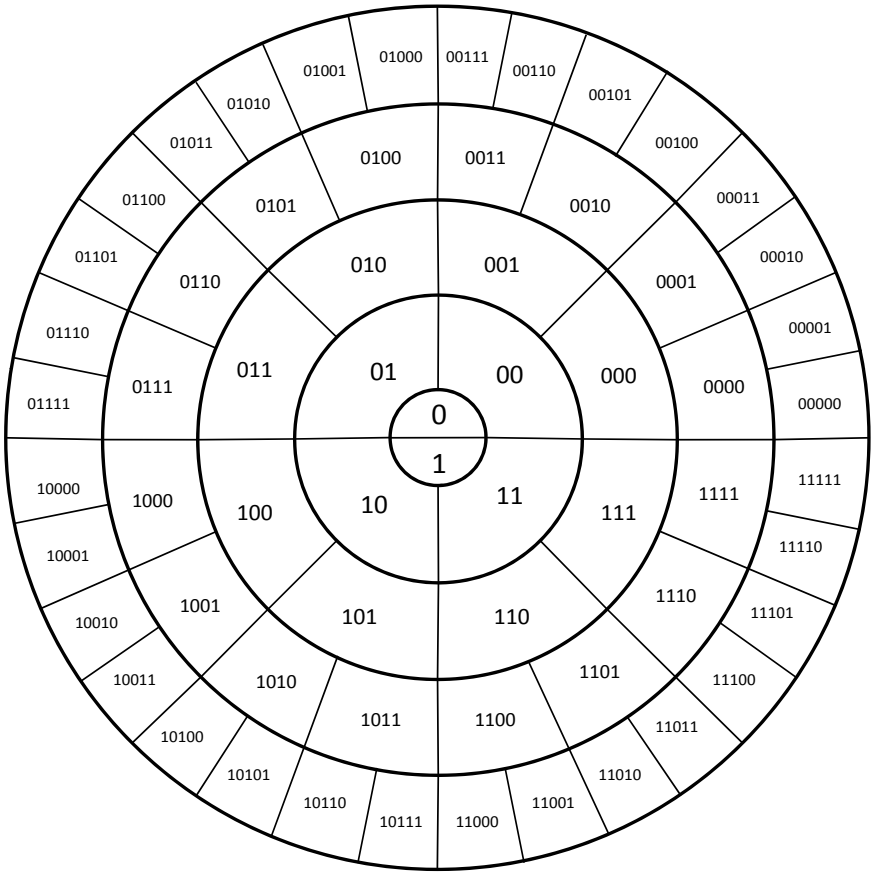


Figura 5.12: Circunferencias concéntricas de diferente radio. En la figura se esquematiza las cinco circunferencias concéntricas con diferente radio, de donde se infiere el concepto de cobertura.

Las rebanadas en cada circunferencia representan las coberturas de los prefijos. Para tener mayor claridad, en la figura 5.13 se representan las coberturas de los prefijos y para distinguirlos mejor se utilizan los colores blanco y gris.Un ángulo relacionado a una dirección IP dada debe de considerar la información de prefijos de circunferencias de todas las longitudes, la analogía es hacer un corte desde el cento de la circunferencia hasta el límite de la circunferencia externa, el corte pasa por coberturas de prefijos de menores longitudes, estos prefijos deben ser considerados como solución en caso de que existan.

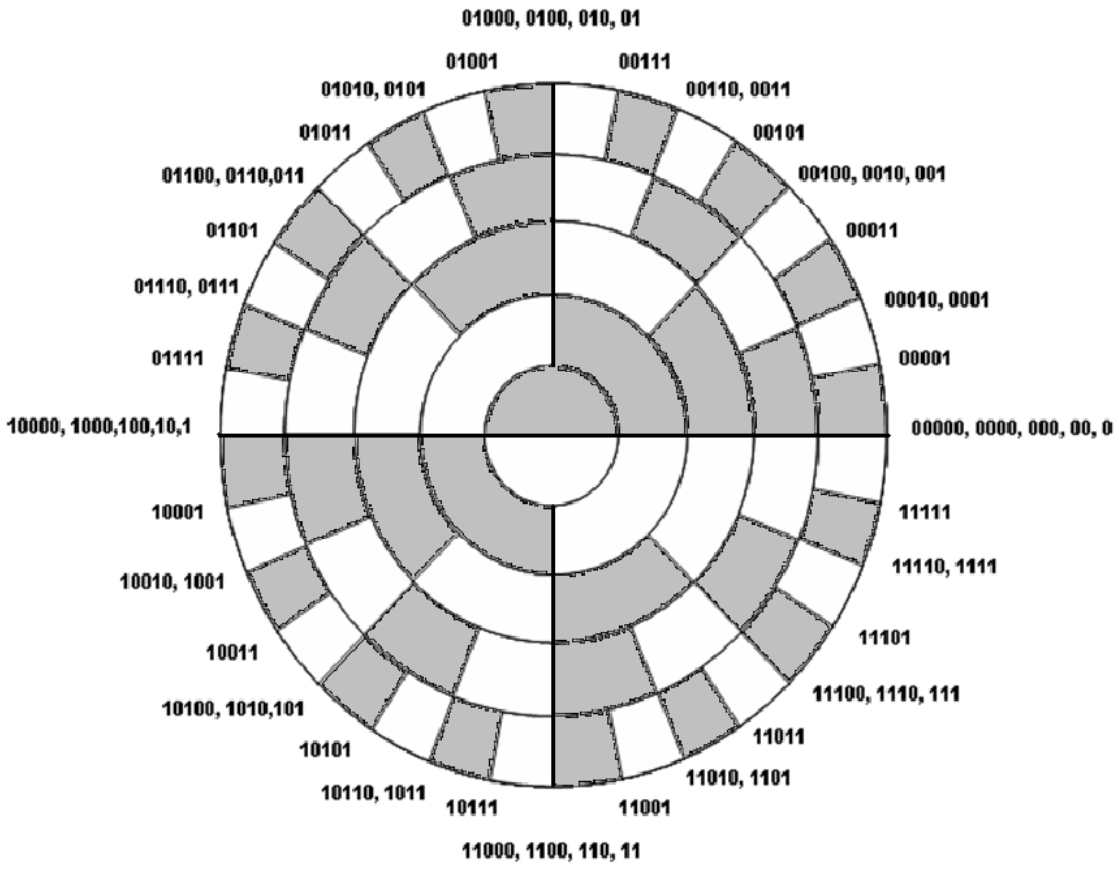


Figura 5.13: Representación de coberturas. En esta figura se representan las coberturas por medio de colores, la idea es observar hasta donde un prefijo debe ser considerado por prefijos de mayor longitud.

En la Figura 5.14 se esquematiza cómo un prefijo debe ser copiado en las secciones de la circunferencia, estas secciones representan los arreglos de prefijos que se utilizan para guardar la información de ruteo. Con esta consideración usamos los arreglos de prefijos en las localidades que estaban vacías.

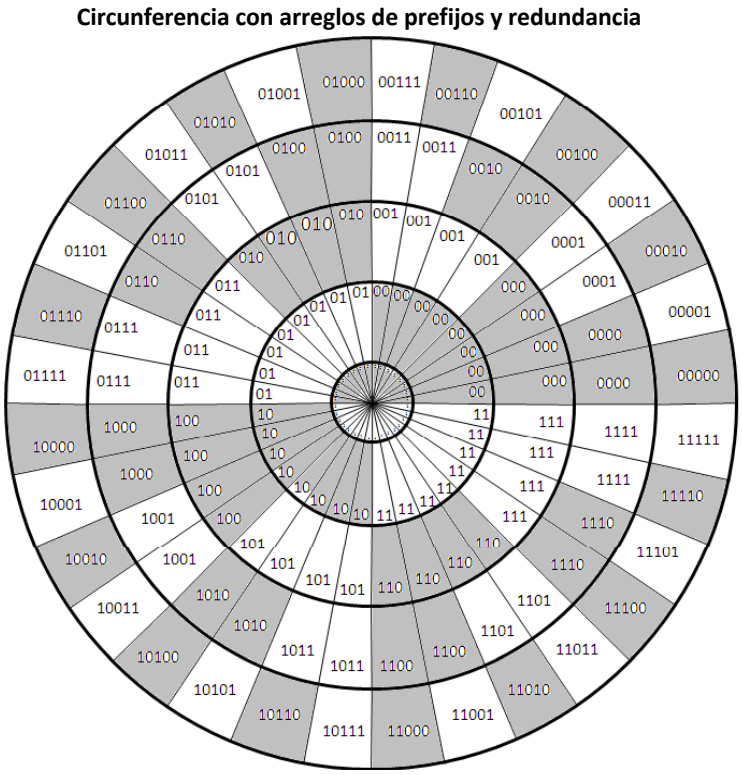


Figura 5.14: Circunferencia con información redundante. En la figura se esquematiza que para cada sección de la circunferencia se puede tener información de todas las longitudes de prefijos, se copia la información del prefijo en todas las entradas.

Para el manejo de la información utilizamos un sistema redundante, es decir, cada prefijo copia su información de ruteo y la información de su límite superior en cada entrada de los arreglos de longitudes de prefijos en relación de su longitud que le corresponde esto lo realiza en la cobertura de ángulos que debe abarcar.

Si tomamos cada sección de la circunferencia y la representamos visualmente en forma lineal tenemos los arreglos que se muestran en la Figura 5.15, donde podemos ver que cada información de ruteo de cada prefijo se debe copiar a otros ángulos.

Arreglos de ángulos y de prefijos con redundancia para el espacio de prefijos de 5 bits de longitud

Arreglo de ángulos	0	11.25	22.50	33.75	45	56.25	67.50	78.75	90	101.25	112.50	123.75	135	146.25	157.50	168.75	180	191.25	202.50	213.75	225	236.25	247.50	258.75	270	281.25	292.50	303.75	315	326.25	337.50	348.75
Prefijos de 1 bit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Prefijos de 2 bits	00	00	00	00	00	00	00	00	01	01	01	01	01	01	01	01	10	10	10	10	10	10	10	10	11	11	11	11	11	11	11	11
Prefijos de 3 bits	000	000	000	000	001	001	001	001	010	010	010	010	011	011	011	011	100	100	100	100	101	101	101	101	110	110	110	110	111	111	111	111
Prefijos de 4 bits	0000	0000	0001	0001	0010	0010	0011	0011	0100	0100	0101	0101	0110	0110	0111	0111	1000	1000	1001	1001	1010	1010	1011	1011	1100	1100	1101	1101	1110	1110	1111	1111
Prefijos de 5 bits	00000	00001	00010	00011	00100	00101	00110	00111	01000	01001	01010	01011	01100	01101	01110	01111	10000	10001	10010	10011	10100	10101	10110	10111	11000	11001	11010	11011	11100	11101	11110	11111

Figura 5.15: Arreglos completos del ejemplo de 5 bits de longitud. Se muestran todos los arreglos que estaban contenidos en la circunferencia de manera lineal.

En la Figura 5.16 se representan todos los posibles prefijos de red en seis arreglos para el ejemplo de cinco bits de longitud de la dirección IP, el primero contiene todos los ángulos que representan a los prefijos de red y los otros 5 tienen la información del límite superior e información de ruteo para cada prefijo de red.

Arreglo de ángulos	0	11.25	22.50	33.75	45	56.25	67.50	78.75	90	101.25	112.50	123.75	135	146.25	157.50	168.75	180	191.25	202.50	213.75	225	236.25	247.50	258.75	270	281.25	292.50	303.75	315	326.25	337.50	348.75
Prefijos de 1 bit	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	0 180	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360	1 360
Prefijos de 2 bits	00 90	00 90	00 90	00 90	00 90	00 90	00 90	00 90	01 180	01 180	01 180	01 180	01 180	01 180	01 180	01 180	10 270	10 270	10 270	10 270	10 270	10 270	10 270	10 270	11 360	11 360	11 360	11 360	11 360	11 360	11 360	11 360
Prefijos de 3 bits	000 45	000 45	000 45	000 45	001 90	001 90	001 90	001 90	010 135	010 135	010 135	010 135	011 180	011 180	011 180	011 180	100 225	100 225	100 225	100 225	101 270	101 270	101 270	101 270	110 315	110 315	110 315	110 315	111 360	111 360	111 360	111 360
Prefijos de 4 bits	0000 22.5	0000 22.5	0001 45	0001 45	0010 67.5	0010 67.5	0011 90	0011 90	0100 112.5	0100 112.5	0101 135	0101 135	0110 157.5	0110 157.5	0111 180	0111 180	1000 202.5	1000 202.5	1001 225	1001 225	1010 247.5	1010 247.5	1011 270	1011 270	1100 292.5	1100 292.5	1101 315	1101 315	1110 337.5	1110 337.5	1111 360	1111 360
Prefijos de 5 bits	00000 11.25	00001 22.5	00010 33.75	00011 45	00100 56.25	00101 67.5	00110 78.75	00111 90	01000 101.25	01001 112.5	01010 123.75	01011 135	01100 146.25	01101 157.5	01110 168.75	01111 180	10000 191.25	10001 202.5	10010 213.75	10011 225	10100 236.25	10101 247.5	10110 258.75	10111 270	11000 281.25	11001 292.5	11010 303.75	11011 315	11100 326.25	11101 337.5	11110 248.75	11111 360

Figura 5.16: Arreglos completos con prefijos y límite superior. En estos arreglos agregamos la información del límite superior de la cobertura del prefijo de red, ya que éste es el que utilizamos cuando se realice la búsqueda para saber si está dentro de la cobertura donde debe tomarse en cuenta un prefijo de red.

Estos arreglos son las estructuras de datos donde se realizan la búsqueda para obtener el BMP. Se realizan dos tipos búsquedas:

1. La búsqueda en el primer arreglo que corresponde a los ángulo se realiza una búsqueda binaria por que es la búsqueda básica que se realiza en arreglos ordenados y tiene un rendimiento en función del tamaño del arreglo, se busca el valor de un ángulo, es por esto que debe ser ordenado, el resultado de la búsqueda es un índice, en donde se realiza la segunda búsqueda.

2. El índice que se obtiene en la búsqueda binaria se usa para realizar la búsqueda en los arreglos de longitud de prefijos, es decir, este índice es la columna donde se realiza la siguiente búsqueda. Para el caso base se realiza una búsqueda lineal sobre la columna, se compara el ángulo de la dirección IP con el valor del límite superior, y se realiza de la entrada del prefijo de mayor longitud hacia la entrada del prefijo de menor longitud. Lo que se busca es el primer límite superior que sea estrictamente mayor que el ángulo que se le asignó a la dirección IP, esto garantiza que la información que tiene esta entrada corresponde al prefijo de red más largo que existe en la tabla de ruteo.

El algoritmo UAM genera 33 arreglos para IPv4 donde tenemos direcciones de 32 bits de longitud; en un arreglo contiene todos los ángulos ordenados que representan uno o más prefijos, cada entrada tiene el índice de los arreglos de longitud del prefijo donde está la información de ruteo y la información del límite superior que en conjunto con el ángulo determinan la cobertura; estas entradas representan la información de todas las circunferencias para direcciones IP de IPv4.

Finalizamos esta sección reiterando que la cobertura tiene un límite inferior y un límite superior, pero el dato de límite superior es el que utilizamos para realizar las comparaciones, es decir, hasta que ángulo llega la representación del prefijo. En la siguiente sección se muestran ejemplos del procedimiento de búsqueda del algoritmo UAM.

5.2.5. Procedimiento de Búsqueda

En esta sección se muestra el método de búsqueda aplicado a la dirección IP de 5 bits de longitud. Iniciamos generando la estructura de datos donde se realiza la búsqueda, podemos ver en la Figura 5.17 la información de cuáles son los ángulos, prefijos y las coberturas donde deben ser tomados en cuenta los prefijos de red.

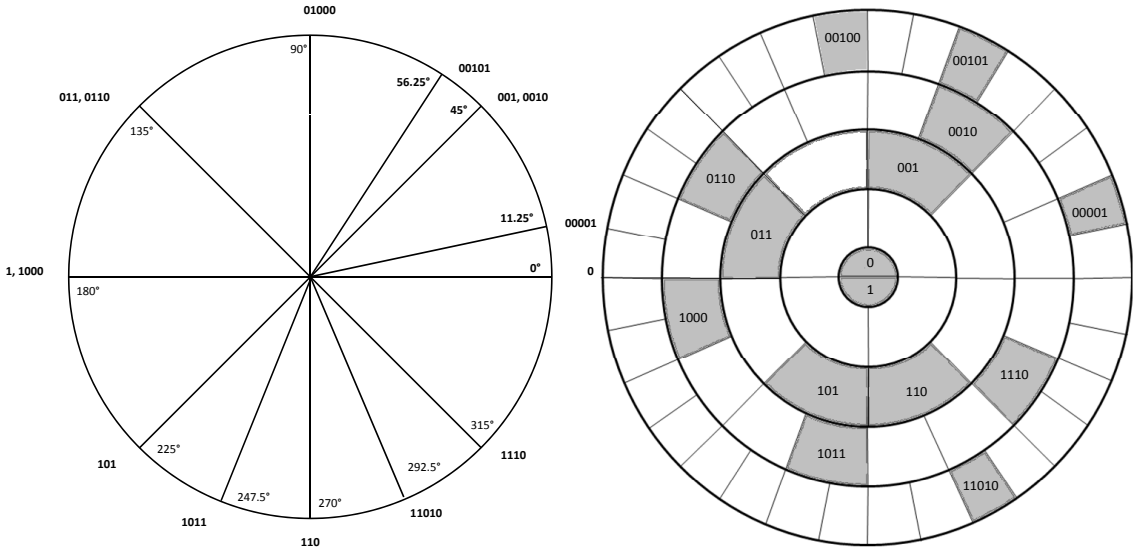


Figura 5.17: Tabla de ruteo de 5 bits representada por coberturas. Para el ejemplo se 5 bits de longitud, en la circunferencia del lado derecho se observa los ángulos, los prefijos y la cobertura donde deben ser tomados en cuenta los prefijos.

Se generan los seis arreglos, donde en el arreglo de ángulos se almacenan todos los ángulos que aparecen en la circunferencia, en los arreglos de longitudes de prefijos se almacena en la posición de la longitud del prefijo que se esté ingresando la información de ruteo y el valor del límite superior de la cobertura, que es hasta donde debe ser tomado en cuenta el prefijo, así los demás ángulos conocen la información de los prefijos de menor longitud.

Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Figura 5.18: Tabla de Ruteo y Cobertura en la circunferencia para el ejemplo de dirección de 5 bits de longitud. En el primer arreglo se guarda la información de los ángulos, es un arreglo ordenado, y en los siguientes arreglos se guarda la información de ruteo y la informaciones del límite superior, este dato se visualiza por que lo utilizamos para realizar las búsquedas.

La información de ruteo es guardada en el arreglo de longitud de prefijos junto con el valor del límite superior de la cobertura del prefijo, pero para los siguientes gráficos solamente se esquematiza el valor del límite superior, para que en los gráficos podamos distinguir que se está buscando o comparando.

Los siguientes dos ejemplos muestran gráficamente el método de búsqueda del algoritmo.

Primer ejemplo

La búsqueda se realiza para la dirección IP 00100, aplicando la ecuación 5.3, se obtiene el ángulo de 45° que le corresponde al prefijo, con el valor del ángulo se realiza la búsqueda binaria en el arreglo de ángulos, el ángulo de 45° existe en el arreglo de ángulos y la búsqueda da como resultado el índice 2.

Buscar la dirección 00100

ángulo = (4)(11.25)=45

$$\frac{360}{2^5} = Km$$

$$Km = 11.25$$

Prefijo encontrado:

0010

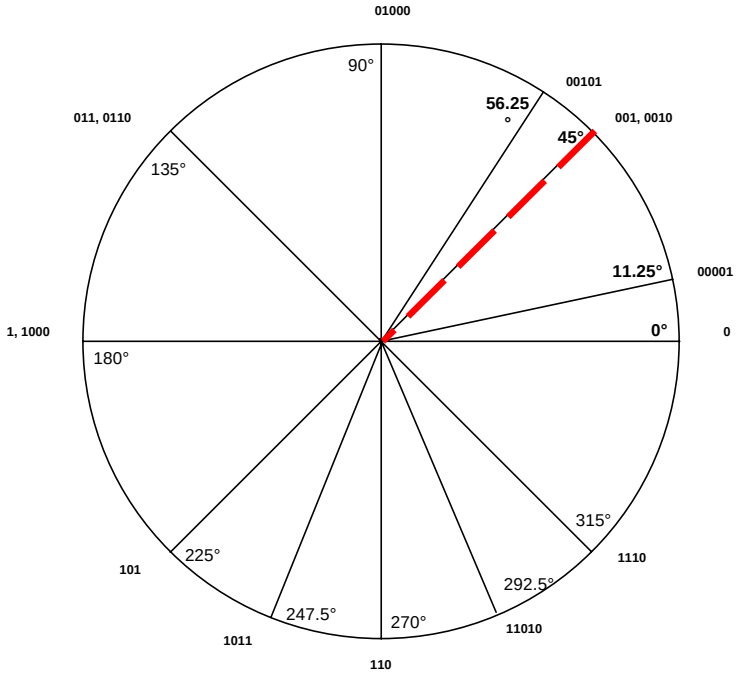


Figura 5.19: Buscar la dirección IP 00100 en la tabla de ruteo. El ángulo que se asocia a la dirección IP 00100 es 45°, y se observa en la circunferencia.

Con el índice que se obtiene de la búsqueda binaria, se inicia una nueva búsqueda en los arreglos de longitud de prefijos desde la entrada de 5 bits de longitud de prefijo hasta la de 1 bit de longitud de prefijo, haciendo la comparación del ángulo con el límite superior de la cobertura que esté registrado en la entrada, buscando que el límite superior de la cobertura sea estrictamente mayor al ángulo. Para este caso el ángulo es 45° y en la entrada de 5 bits no existe información del prefijo, por lo tanto se continúa la búsqueda en la entrada de 4 bits de longitud, y se obtiene un límite superior de la cobertura de 67.5° que indica que este prefijo debe considerarse hasta este ángulo y cubre el ángulo de 45°, entonces el algoritmo se detiene y reporta la información del prefijo 0010 como el más largo que existe en la tabla de ruteo.

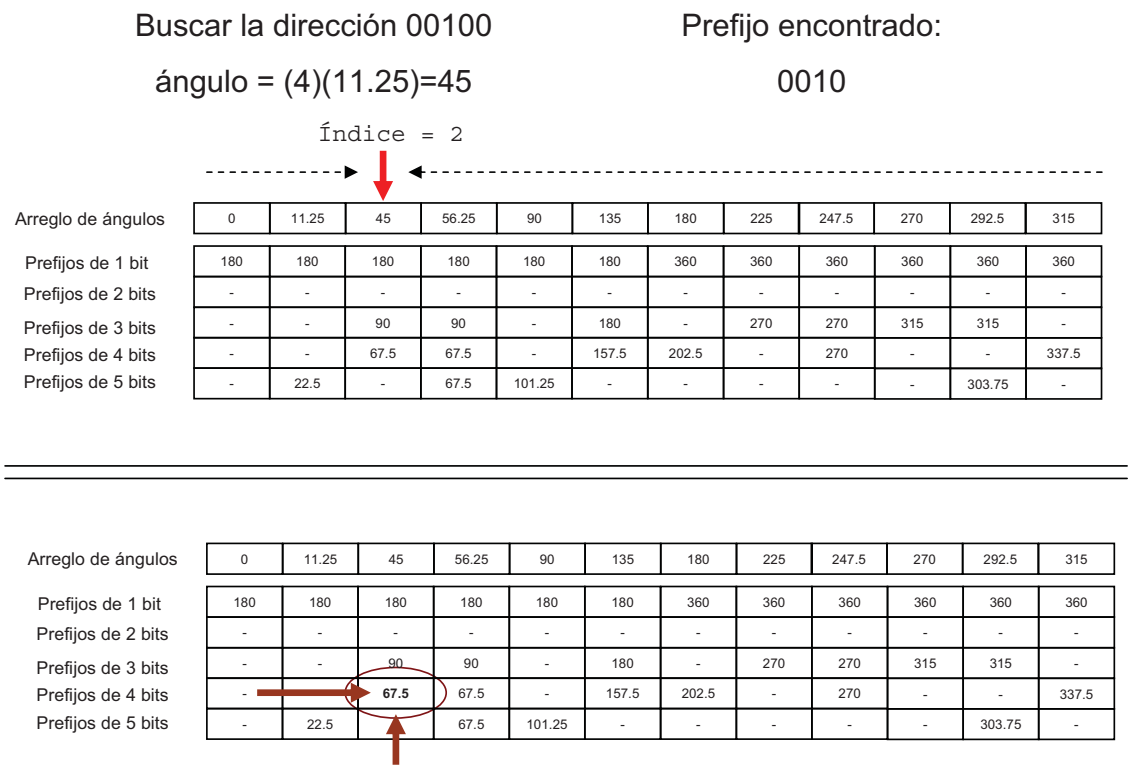


Figura 5.20: Encontrando la información de ruteo. La búsqueda binaria en el arreglo de ángulos regresa el índice con valor 2, con este índice se busca en los arreglos de longitud de prefijos, encontrando que la solución es el prefijo 0010 de 4 bits de longitud.

Segundo Ejemplo

Buscamos la dirección 01110 para ejemplificar la posibilidad donde no existe el información de prefijos mapeados en el ángulo que se busca. El valor que se obtiene para el ángulo es 157.5°. En el arreglo de ángulos se realiza la búsqueda binaria pero para este prefijo no se encuentra una entrada que coincida exactamente con este ángulo. La búsqueda devuelve el índice con valor 5 que representa la entrada con el ángulo menor inmediato que en este caso es el ángulo de 135°.

El índice que se obtiene en la búsqueda binaria es usado para realizar una nueva búsqueda en los arreglos de longitud de prefijos. Se compara el ángulo con la cobertura registrada en las entradas desde la que representa 5 bits de longitud, hasta la

Buscar la dirección 01110

(14)(11.25) = 157.5°

$$\frac{360}{2^5} = Km$$

$$Km = 11.25$$

Prefijo encontrado:

011

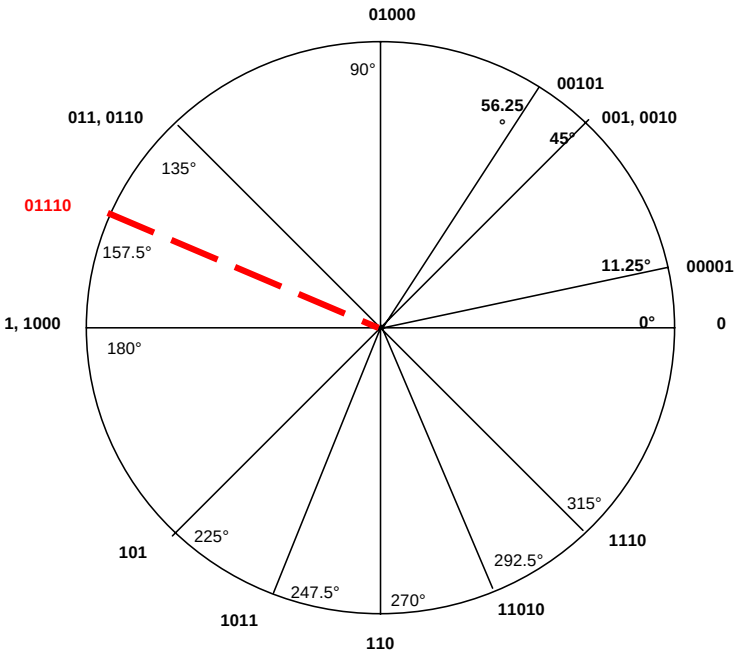


Figura 5.21: Buscar la dirección IP 01110 en la tabla de ruteo. En el gráfico se muestra que para el prefijo 01110 la posición en la circunferencia está en el ángulo 157.5°.

que representa la de 1 bit de longitud. El algoritmo se detiene cuando se encuentra un rango estrictamente mayor al ángulo y reporta la información de esta entrada.

En este caso, el ángulo es 157.5° y en el arreglo de longitud de prefijos existe la información para el prefijo de 4 bits de longitud pero su límite superior es 157.5° que no es estrictamente mayor, es decir, no cubre el ángulo de 157.5°, por esto la búsqueda continúa en la siguiente entrada de 3 bits de longitud donde el límite superior es de 180° y es estrictamente mayor que 157.5°,quiere decir que este prefijo tiene de cobertura hasta el ángulo de 180° y cubre el ángulo de 157.5°, por lo tanto, éste es el prefijo de red más largo para esta dirección IP. Para este caso el prefijo encontrado es 011.

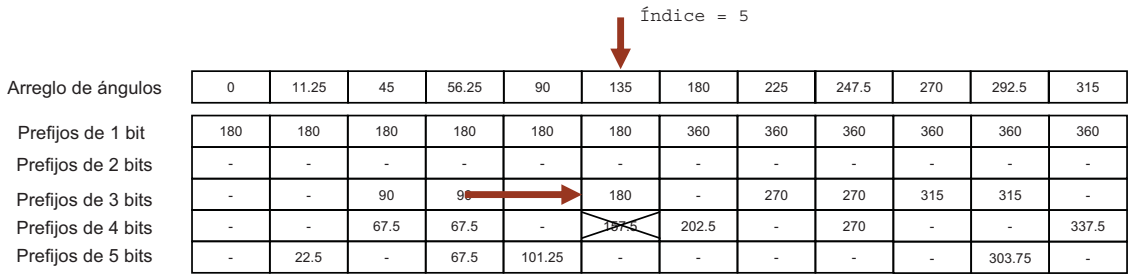


Figura 5.22: Encontrar la información de ruteo. Se realiza la búsqueda binaria en el arreglo de ángulos y el índice devuelto es 5, con este índice se comienza a buscar sobre los arreglos de longitud de prefijos. En la figura se esquematiza que el límite superior para el prefijo de longitud 4 es de 157.5°, por lo tanto el ángulo está fuera de la cobertura de este prefijo. Continúa la búsqueda y la siguiente entrada del prefijo de 3 bits de longitud tiene un límite superior con valor de 180°, éste límite indica que este prefijo abarca el ángulo de 157.5° y es el prefijo que le corresponde.

En la siguiente sección se muestra el pseudocódigo de la función para realizar la búsqueda.

5.2.6. Implementación de la búsqueda

Para la búsqueda de direcciones IP aplicando el algoritmo UAM se realizó con la siguiente función:

```
Función Busca_Prefijo(Arreglo_Angulos, Arreglo_Longitud,Dirección)
    encontrado=FALSE;
    longitud_IP=32;
    indice=0;
    K=(360/(2^longitud_IP));
    angulo=valor_Direccion*K;
    rango=angulo+K;
    indice=Busqueda_Binaria_Angulo(Arreglo_Angulos, angulo, encontrado);
    Info_Ruteo = Busca_Arreglo_Rangos(Arreglo_Longitud, indice, rango);
```

La función Busqueda_Binaria_Angulo se encarga de realizar la búsqueda binaria sobre el arreglo de ángulos regresando dos valores, el índice en donde debe insertarse la información de ruteo y una bandera que indica si se encontró exactamente el ángulo que se busca. La función Busca_Arreglo_Rangos realiza el recorrido en el arreglo de longitud de prefijos y regresa la información de ruteo del prefijo encontrado.

La complejidad en el peor caso es O(log(M)+W) donde M es el número de prefijos de red en la tabla de ruteo y W es la longitud de la dirección IP, que son la composición de la búsqueda binaria en el arreglo de ángulos y la búsqueda lineal en el arreglo de longitud de prefijos.

En la siguiente sección se explica como el algoritmo UAM soporta los cambios que pueden suceder en los ruteadores de backbone.

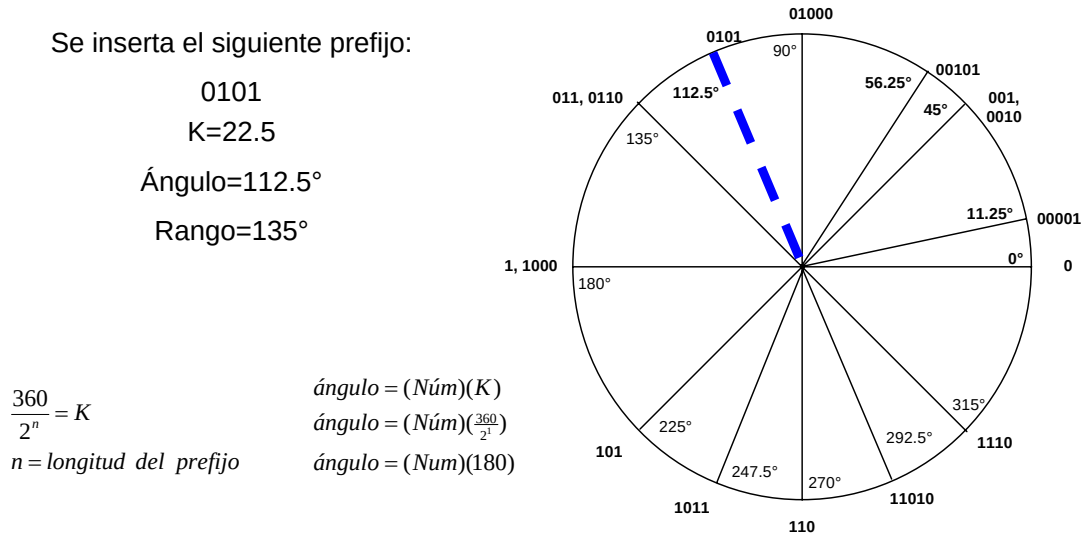
5.3. Actualización

Los ruteadores están inmersos en un sistema distribuido que cambia en todo momento, en esta sección se describe el funcionamiento del algoritmo que faltaba por cubrir, que son las operaciones de inserción, modificación y borrado correspondiente a un prefijo de red.

5.3.1. Inserción de un prefijo de red

Cuando aparece una nueva red a la que el ruteador puede acceder, es necesario insertar la información de ruteo del prefijo de esta nueva red en la estructura de datos que representa a la tabla de ruteo. Para insertar un prefijo de red, calculamos el ángulo para este prefijo mediante las ecuaciones de mapeo. El valor encontrado sirve para determinar la ubicación (índice) que le corresponde dentro del arreglo de ángulos. Con el mismo índice también se agregan las entradas correspondientes en los arreglos de longitud. La información de ruteo del prefijo es almacenada en la localidad que le corresponde dentro del arreglo de longitud acorde a su longitud.

Después de insertar la información de ruteo, se procede a propagar la información de la existencia del prefijo hacia los ángulos mayores al suyo. Además, a su vez, este nuevo prefijo/ángulo debe tener información de los prefijos/ángulos que lo alcanzan con su cobertura. Nótese que estos procedimientos se realizan únicamente cuando la entrada del ángulo no existe en el arreglo de ángulos ni en el arreglo de longitud de prefijo. Recordemos que un mismo ángulo puede corresponder a más de un prefijo, por lo que, cuando se determina el ángulo del prefijo, es posible que la entrada del ángulo para el nuevo prefijo ya exista. Por tanto, únicamente se debe actualizar los

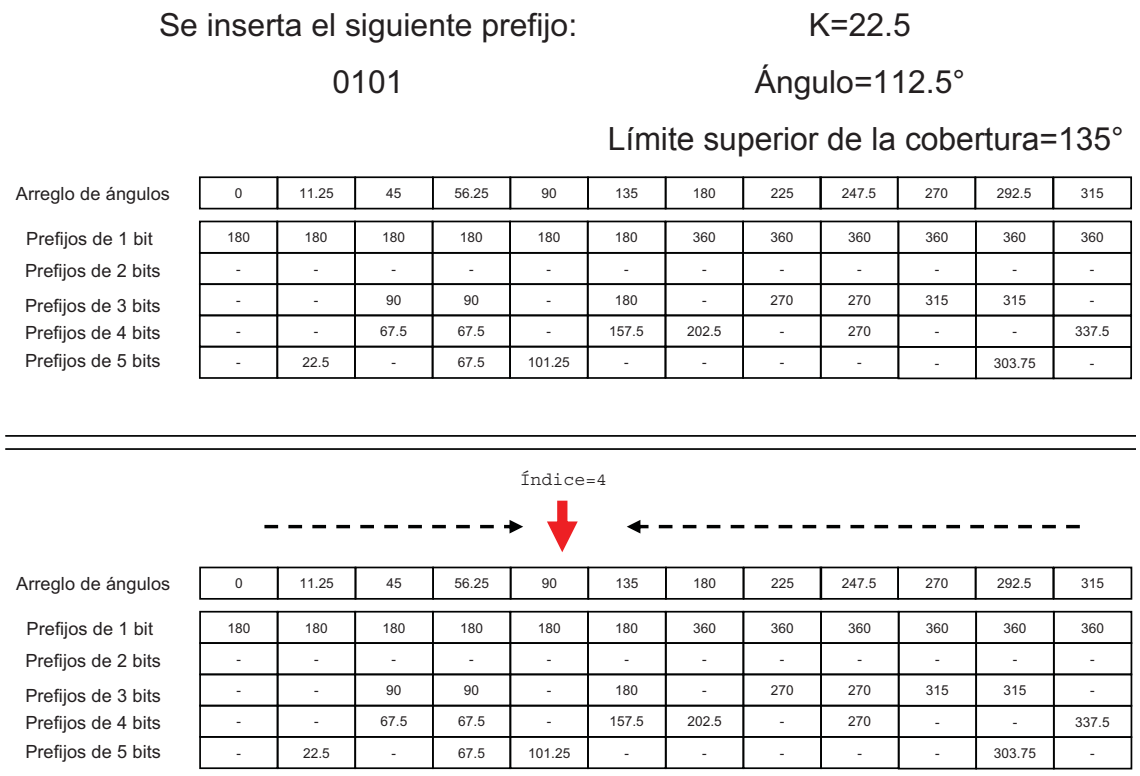


Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Figura 5.23: Insertar el prefijo 0101. Insertar el prefijo 0101 se asocia con el ángulo y el límite superior de su cobertura, que son 11.5° y 135° respectivamente, se observa en la circunferencia del lado derecho la posición del prefijo en la circunferencia.

datos del prefijo en la entrada del arreglo de longitud de prefijo donde le corresponde su longitud y en seguida copiar la información de ruteo y su límite superior a los ángulos mayores que él y que queden dentro de su cobertura. El procedimiento de copiar la información del primer ángulo menor que él en este caso no es necesaria.

En la Figura 5.18 se observa cómo están formados los arreglos para la tabla de ruteo que estamos manejando en los ejemplos.



Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Figura 5.24: Búsqueda binaria en el arreglo. La búsqueda se realiza primero en el arreglo de ángulos, esta búsqueda es binaria y para este caso regresa el índice con valor 4, apuntando a 90°, así que se inserta una nueva columna.

En los arreglos de longitud de prefijo observamos que para el prefijo 0 se le asigna el ángulo 0° y su límite superior es 180°, así que todos los ángulos que son estrictamente menores a 180° deben tener información del prefijo 0, por lo tanto en los ángulos 11.25°,45°,56.25°,90° y 135° también se copia la información del prefijo 0 en los arreglo de longitud de prefijo en la entrada de 1 bit de longitud.

Por ejemplo para insertar el prefijo 0101 se tiene el siguiente procedimiento. Primero, tomando en cuenta su valor y su longitud, se obtiene el ángulo que le corresponde al prefijo. En seguida se calcula cual es su cobertura. Para este caso el ángulo que se va a insertar es 112.5 y el límite superior de la cobertura es 135° (Figura 5.23).

Se realiza la búsqueda binaria del ángulo para saber el índice en el arreglo de ángulos. Se busca el ángulo que sea igual, si existe o en otro caso el inmediato inferior (Figura 5.24). En este caso el ángulo no existe en el arreglo, por lo tanto se inserta una entrada en el arreglo de ángulos y en todas las entradas en el arreglo de longitud de prefijos entre el ángulo de 90° y el ángulo de 135°, en la Figura 5.25 observamos que insertamos una nueva columna para este ángulo.

En la nueva entrada del arreglo de ángulos se registra el nuevo ángulo. La información de ruteo del prefijo y el límite superior de la cobertura del prefijo se registra en la entrada del arreglo de longitud correspondiente. Para que todos los ángulos dentro de la cobertura tengan información de este prefijo, esta información se propaga (copia) en las entradas dentro del arreglo de longitud con ángulos que sean mayores que el nuevo ángulo insertado, pero estrictamente menores al límite superior. Para este caso, la propagación de la información en realidad no se produce ya que el límite superior es 135° y el siguiente ángulo en los arreglos es 135°, por lo tanto éste no es afectado ya que no es estrictamente menor que el límite superior de cobertura.

Por último se copia el arreglo de longitud de prefijos del ángulo inmediato inferior siempre que los límites superiores (cobertura) sean mayores al nuevo ángulo. Si el ángulo es menor, esto indica que está dentro del rango de cobertura del prefijo de esa longitud. Para este ejemplo, en la entrada de 1 bit de longitud el límite es de 180° y es mayor que el nuevo ángulo 112.5° por lo tanto se copia la información de esta entrada. Para la longitud de 5 bits también existe información, pero el límite superior es de 101.25° que es menor que el ángulo que se está insertando, por lo

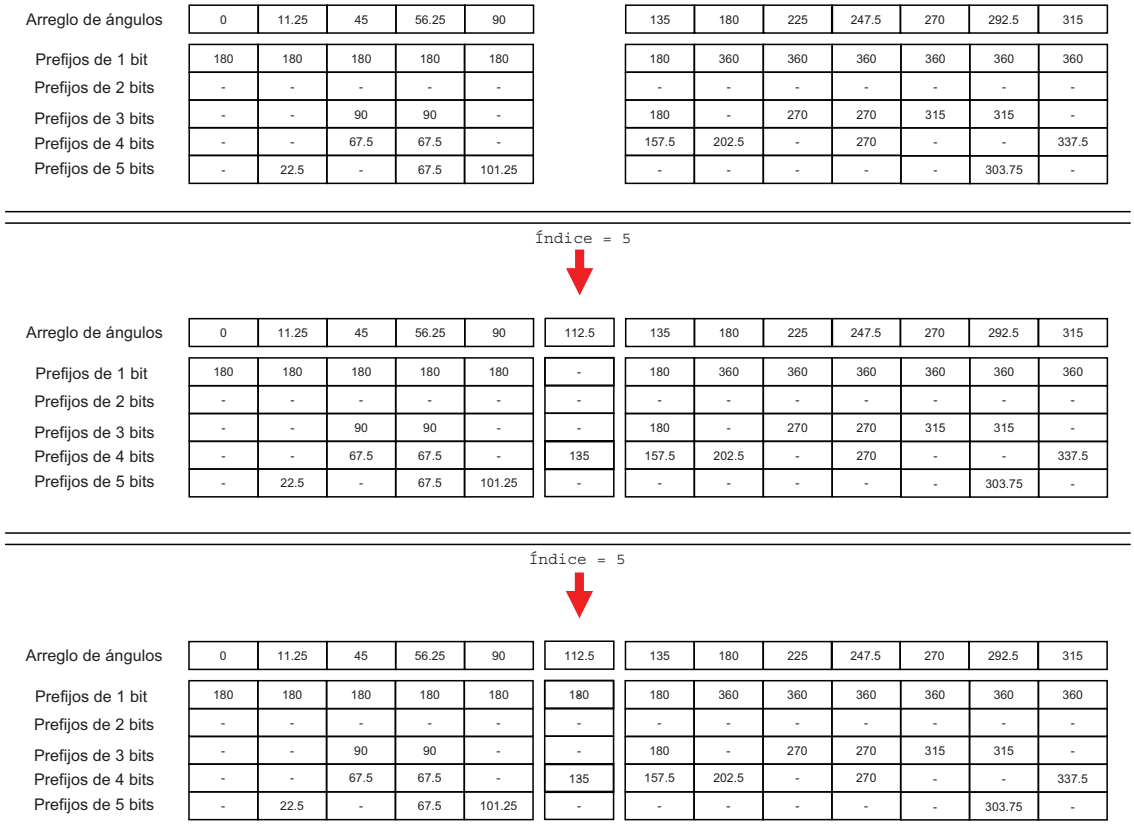


Figura 5.25: Completando arreglos. Cuando se inserta la columna se agrega la información del prefijo 0101 en el arreglo prefijos de 4 bits de longitud, y se procede a propagar su información. En este caso, el límite superior es 135° y el siguiente ángulo es 135° así que aquí termina la propagación ya que no es un ángulo estrictamente menor. Después se copia la información de ruteo y límites de cobertura del ángulo inmediato inferior, tomando en cuenta que esté dentro de la cobertura de los prefijos, para este caso se copia la información del prefijo de longitud 1.

tanto esta información no se copia. Terminando así la inserción de un prefijo.

La complejidad computacional de insertar un prefijo en el peor caso es O($log(M)+M$), donde M es la cantidad de prefijos en la tabla de ruteo. En la siguiente sección se muestra la función de inserción de prefijos en pseudocódigo.

5.3.2. Implementación de la inserción de un prefijo

Para construir la estructura de datos se implemento la función para insertar los prefijos de red de la tabla de ruteo.

```
Función Inserta_Prefijo(Arreglo_Angulos, Arreglo_Longitud,Prefijo,longitud)
    encontrado=FALSE;
    indice=0;
    K=(360/(2^longitud));
    angulo=valor_prefijo*K;
    rango=angulo+K;
    indice=Busqueda_Binaria_Angulo(Arreglo_Angulos, angulo, encontrado);
        SI encontrado = FALSE ENTONCES
            Inserta_nodo(Arreglo_Angulos, Arreglo_Longitud, indice);
            Arreglo_Angulos[indice]=angulo;
        FIN SI
    Rellena_Arreglo_Rangos(Arreglo_Longitud, indice, rango);
```

La función Busqueda_Binaria_Angulo es la misma que se describe en la sección de búsqueda. La función Inserta_nodo funciona cuando el ángulo no existe en el arreglo y se debe insertar una entrada en el arreglo de ángulos y también una entrada en los arreglos de longitud de prefijos. La función Rellena_Arreglo_Rangos se utiliza para propagar la información ruteo y el valor del límite superior del rango a las entradas donde le corresponde en función de la longitud del prefijo en los ángulos que estén dentro del rango del prefijo. El arreglo se debe completar haciendo que los elementos que estén dentro del rango de una dirección también tengan la información de este prefijo.

A continuación se presenta como modificar la información de ruteo de un prefijo.

5.3.3. Modificación un prefijo de red

La modificación de la información de ruteo sucede cuando el enlace que se conoce como mejor ruta cambia por otro, es decir, cambia la información de ruteo. Para modificar un prefijo debe existir el ángulo, la información de ruteo y el dato del límite superior del rango en la entrada que le corresponde en los arreglos de longitud para la longitud de este prefijo.

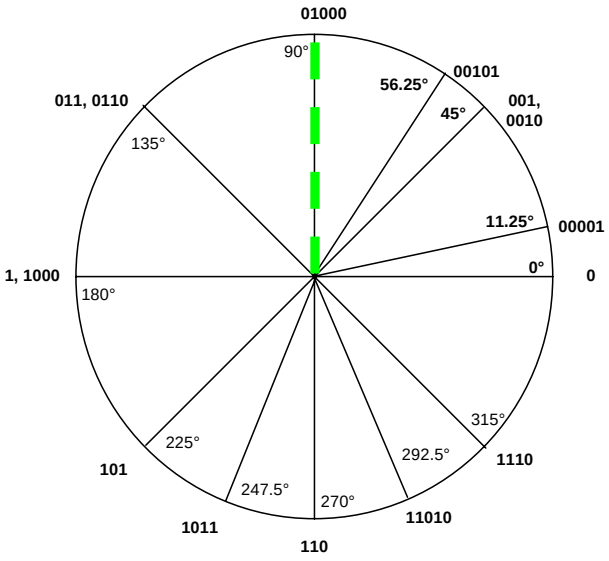
Se modifica el siguiente prefijo:

01000

K=11.25

Ángulo=90°

Rango=101.25°



Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Figura 5.26: Modificar la información del prefijo 01000. En la figura se observa la posición que le corresponde al prefijo 01000 que es el ángulo de 90°

Lo primero es obtener el ángulo que le corresponde al prefijo de red y el valor del límite superior de su cobertura. Después se realiza la búsqueda binaria exacta con el valor del ángulo. Teniendo el índice para el arreglo de longitud de prefijos, se actualiza la entrada correspondiente a la longitud del prefijo. Por último se propaga

la información a los ángulos mayores que están incluidos en la cobertura de este prefijo.

Para la tabla de ejemplo que se ha estado manejando, modificamos la información de ruteo del prefijo 01000. Se obtiene el ángulo de 90° y el límite superior de 101.25°, el procedimiento de obtener el ángulo y el valor del límite de la cobertura se observa en la Figura 5.26.

Se modifica el siguiente prefijo: K=11.25

01000 Ángulo=90°

Límite superior de la cobertura=101.25°

Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

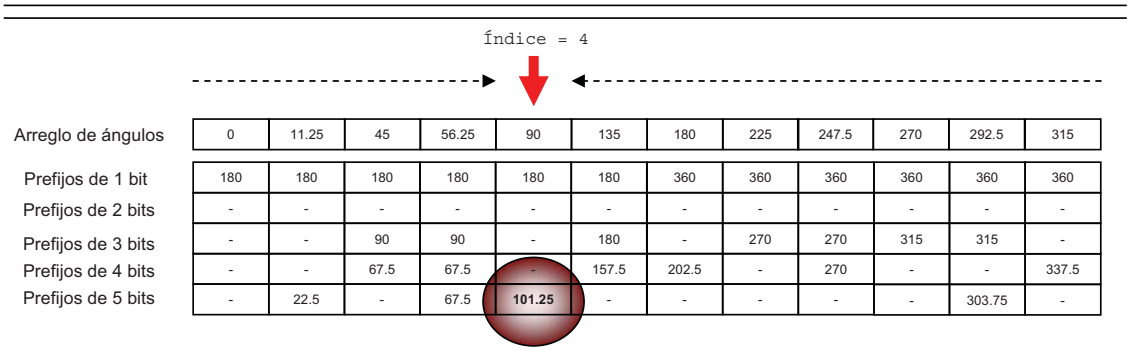


Figura 5.27: Cambiar la información en el arreglos de longitud de prefijos. La búsqueda binaria realizada en el arreglo de ángulos dando como resultado el índice 4, y se modifica la entrada en el arreglo prefijos de 5 bits de longitud.

La búsqueda es exacta y se tiene el índice para modificar la información de ruteo en el arreglos de longitud de prefijos. En el ejemplo se tiene que modificar la columna del ángulo de 90° y el arreglo de prefijos de 5 bits de longitud, este procedimiento se ilustra en la Figura 5.27. Por último se propaga la nueva información de ruteo

a los ángulos mayores que están incluidos dentro de la cobertura del prefijo que se modifica. Para este ejemplo el límite superior es de 101.25 y el siguiente ángulo es de 135° por lo tanto no está dentro de la cobertura y no se copia la información.

La modificación de un prefijo tiene una complejidad computacional en el peor caso de O($log(M)+ M$), donde M es el número de prefijos en la tabla de ruteo.

5.3.4. Borrado de un prefijo de red

Cuando no se tiene comunicación con un nodo dentro de Internet se debe quitar la información de este prefijo, por lo tanto borrar un prefijo de red es la operación inversa a la inserción de un prefijo.

Para borrar un prefijo, se obtiene el ángulo que le corresponde y el límite superior de la cobertura, con esta información se realiza la búsqueda exacta en el arreglo de ángulos para encontrar el índice en los arreglos de longitud.

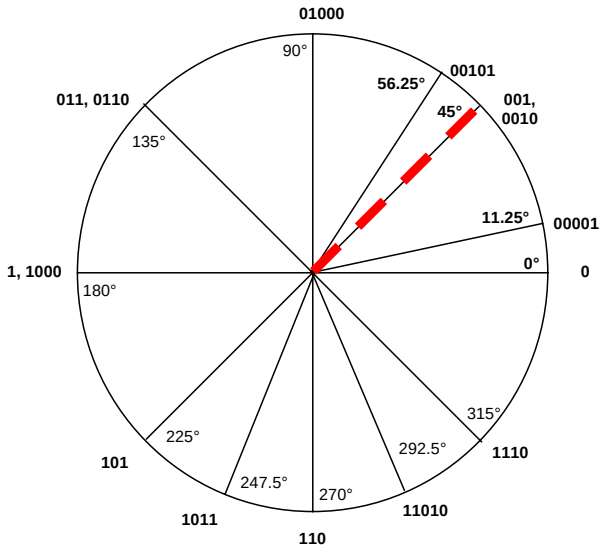
Se borra el siguiente prefijo:

001

K=45

Ángulo=45°

Rango=90°



Arreglo de ángulos	0	11.25	45	56.25	90	135	180	225	247.5	270	292.5	315
Prefijos de 1 bit	180	180	180	180	180	180	360	360	360	360	360	360
Prefijos de 2 bits	-	-	-	-	-	-	-	-	-	-	-	-
Prefijos de 3 bits	-	-	90	90	-	180	-	270	270	315	315	-
Prefijos de 4 bits	-	-	67.5	67.5	-	157.5	202.5	-	270	-	-	337.5
Prefijos de 5 bits	-	22.5	-	67.5	101.25	-	-	-	-	-	303.75	-

Figura 5.28: Borrando la información del prefijo 001. El prefijo 001 tiene asociado el ángulo 45° y en la circunferencia se muestra la posición del prefijo.

En los arreglos de longitud de prefijos se borra la información de ruteo y cobertura dentro de la entrada que le corresponde de acuerdo a su longitud del prefijo. Se borra la información que se propago, es decir, se borra la información de ruteo de la entrada

en el arreglo de longitud de prefijos con respecto a la longitud de los ángulos mayores al ángulo del prefijo que se borra, que sean estrictamente menores al límite superior de cobertura.

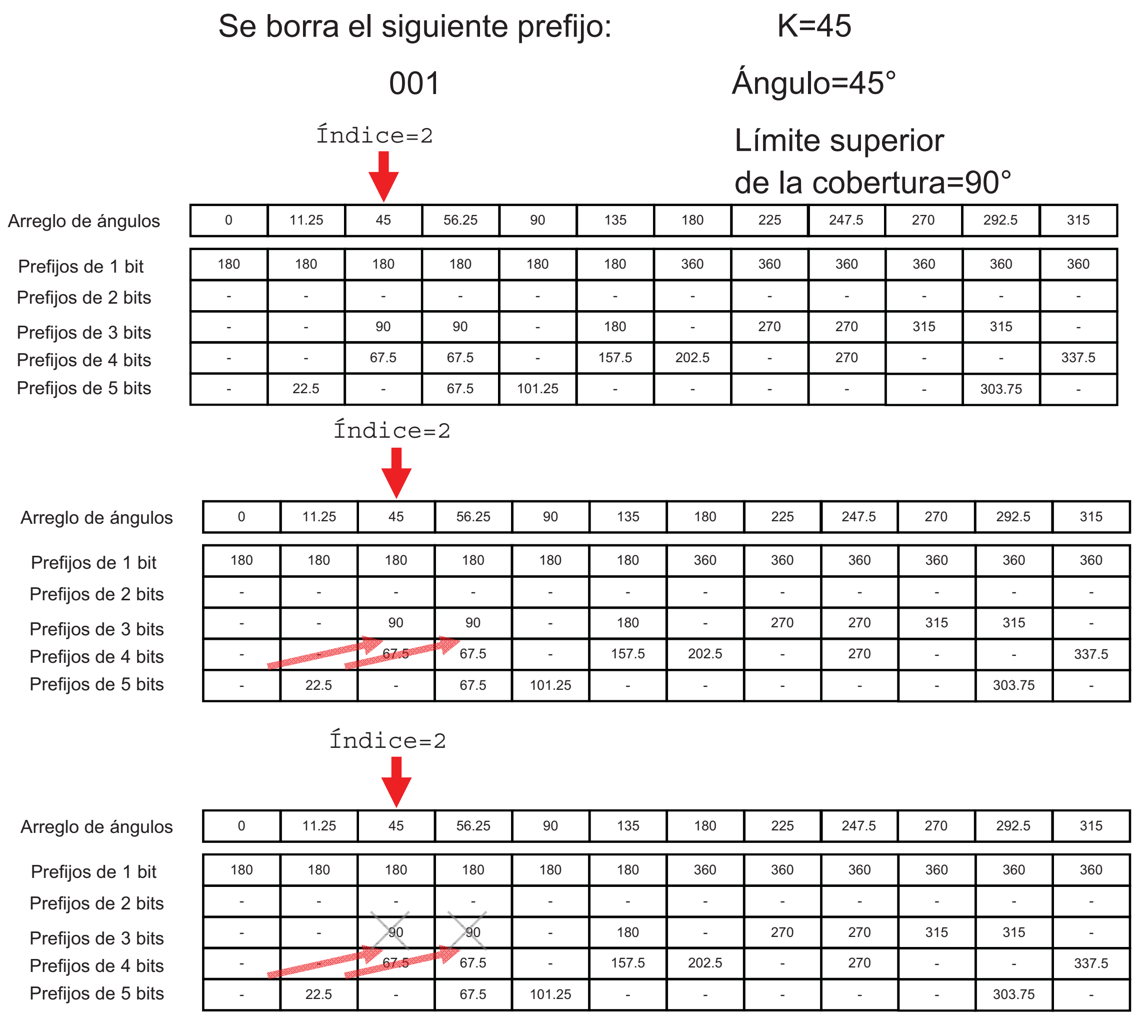


Figura 5.29: Borrar la información que se propagó. La búsqueda binaria regresa el índice de valor 2, se quita la información de ruteo y cobertura del arreglo de 3 bits de longitud, también se borra la información en el índice 5 del arreglo de longitud 3 ya que el ángulo 56.25° está dentro del límite de la cobertura del prefijo que es de 90°.

Por último, se compara el arreglo de longitud de prefijos del ángulo del prefijo a borrar, con los arreglos de longitud de prefijos del ángulo inmediato inferior. Si es exactamente igual, indica que esta entrada es redundante y debe eliminarse. Si es diferente, los arreglos de longitud de prefijos tiene información referente a otro prefijo

de diferente longitud asociado a este ángulo inferior, por lo tanto no se elimina.

Para el ejemplo de borrar el prefijo 001, se obtiene el ángulo y cobertura, se realiza la búsqueda binaria exacta para encontrar el índice. Para finalizar la eliminación del prefijo, se compara el arreglo de longitud de prefijos del ángulo de 45° con los arreglos de longitud de prefijos del ángulo inmediato inferior, para este caso 11.25°. En este caso son diferentes ya que en el ángulo de 45° se tiene información del prefijo 0010, por lo tanto no se elimina esta entrada.

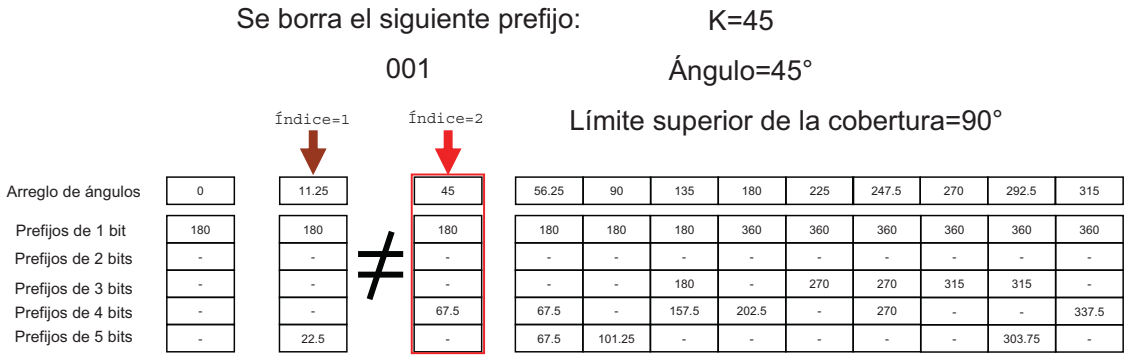


Figura 5.30: Borrar la información redundante. Por último se compara la columna del ángulo menor inmediato, para ver si existe diferencias. En este caso son diferentes por lo tanto no se borra la columna del ángulo 45°.

El borrar la información de un prefijo tiene una complejidad computacional en el peor caso de O($log(M)+ M$) donde M es el número de prefijos en la tabla de ruteo, ya que se debe quitar la información que se propago a los ángulos que deben de considerarse dentro del límite superior de la cobertura.

5.4. Desempeño del Algoritmo UAM

El desempeño del algoritmo se midió realizando búsquedas de los prefijos correspondientes a distintas direcciones IP. La búsqueda de direcciones IP los siguientes resultados. Se generan direcciones IP de 32 bits aleatorias controladas, es decir, las direcciones IP se forman utilizando los prefijos de red que están en la tabla de ruteo, así sabemos que prefijo debe encontrar, y se inyectan a la función de búsqueda. Esta función regresa la información de ruteo que está relacionada con el prefijo de mayor longitud que coincide con los bits más significativos de la dirección IP, que existe en la tabla de ruteo.

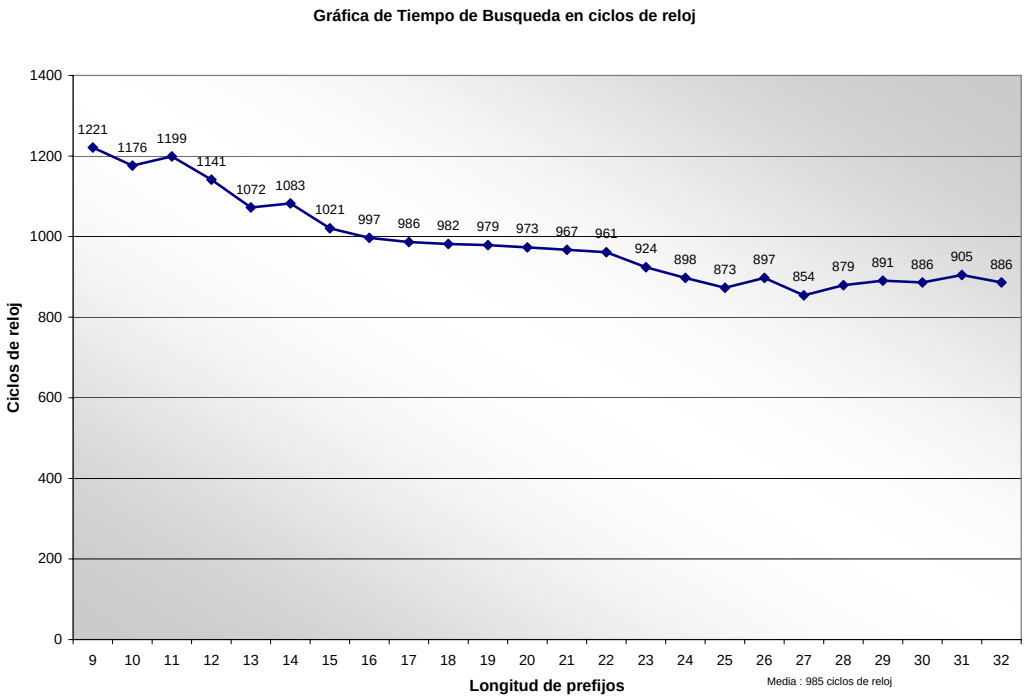


Figura 5.31: Gráfica de resultados en ciclos de reloj

Controlando cómo se construyen las direcciones IP, se sabe de antemano el re-

sultado de la búsqueda, así se puede comprobar que los resultados que se obtienen del algoritmo UAM son los correctos.

La gráfica de la Figura 5.31 muestra los ciclos de reloj que le toma al algoritmo encontrar el prefijo más largo para una dirección IP, esto con respecto a la longitud del prefijo de red que se encuentra.

La gráfica de la Figura 5.32 indica el tiempo que se tarda en encontrar el BMP en nanosegundos, cambiando los ciclos de reloj con respecto a las características del equipo de cómputo utilizado.

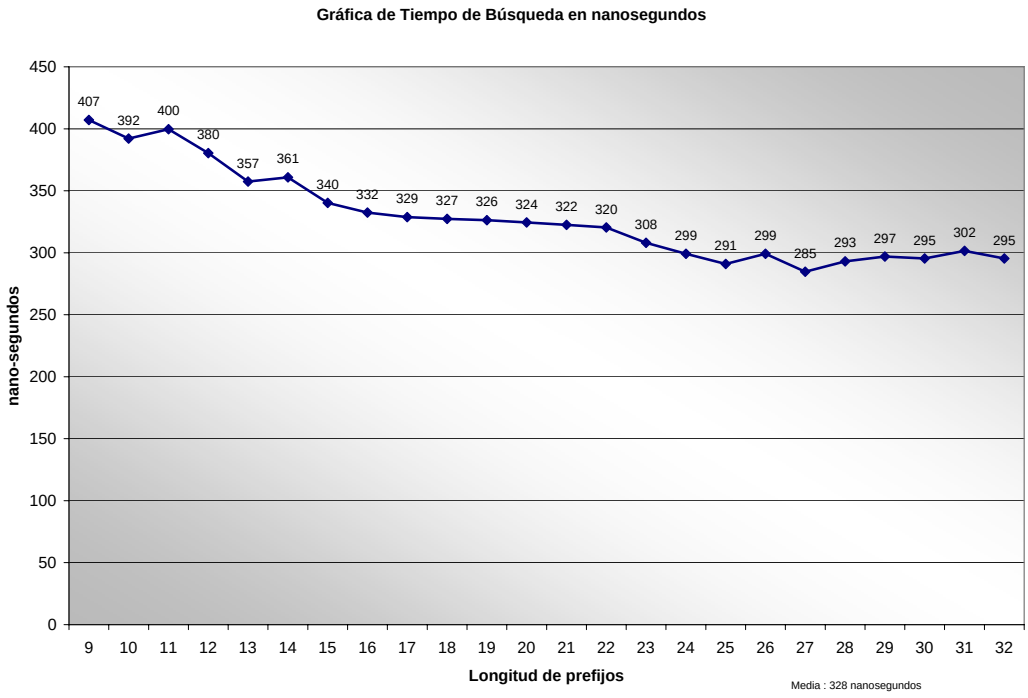


Figura 5.32: Gráfica de resultados en nanosegundos

Se observa que es menor el tiempo en encontrar la solución en prefijos de mayor longitud, ya que por la estructura del algoritmo, se indicó que comience la búsqueda

en los arreglos de longitud de prefijos desde las entradas que representan a los prefijos más largos hacia las entradas que representan a los prefijos de menor longitud.

El tiempo que el algoritmo UAM se tarda en encontrar la solución es determinado por las iteraciones que realiza la búsqueda binaria más las iteraciones de búsqueda lineal en los arreglos de longitud de prefijos. El número de iteraciones de la búsqueda binaria es de 18 y en el peor de los casos la búsqueda lineal realiza 32 iteraciones por lo tanto, en el peor caso el algoritmo UAM realiza 50 iteraciones, que son las iteraciones de la búsqueda binaria, más las iteraciones de búsqueda lineal en los arreglos de longitud de prefijos, ya que las iteraciones sobre los arreglos realizan las mismas operaciones por iteración.

Podemos calcular el promedio con respecto al retardo de las iteraciones y tenemos que los accesos a memoria en este algoritmo se tardan 15 nano-segundos, por lo tanto, en el peor caso debe tardarse 750 nano-segundos. Todos estos cálculos se realizaron con 1,000 búsquedas de la misma dirección IP ya que se calcula el promedio en forma iterativa, y el promedio tiene una varianza estable en 1,000 iteraciones, esto de describe en el Apéndice “A”.

Se observa que para la cantidad de prefijos en la tabla de ruteo que es 213,078, el total de entradas en el arreglo de ángulos es de 134,043, la cantidad de entradas dentro del arreglo de ángulos es menor a la cantidad de prefijos de la tabla de ruteo.

En el siguiente capítulo se compara el desempeño de los tres algoritmos que se implementaron en esta tesis.

Capítulo 6

Análisis de los Algoritmos Implementados

En este capítulo comparamos los resultados de las simulaciones de los algoritmos "Trie" binario, el algoritmo de la universidad de Lulea y el algoritmo UAM.

El tiempo que tarda un algoritmo en encontrar el prefijo de red más largo coincidente con los bits más significativos de la dirección IP es la característica principal que comparamos en la Gráfica 6.1, en donde las líneas representan cuanto se tarda en promedio cada algoritmo, el tiempo siempre depende de la longitud de prefijo de red que se encuentra. El resultado de estas simulaciones está dado en ciclos de reloj.

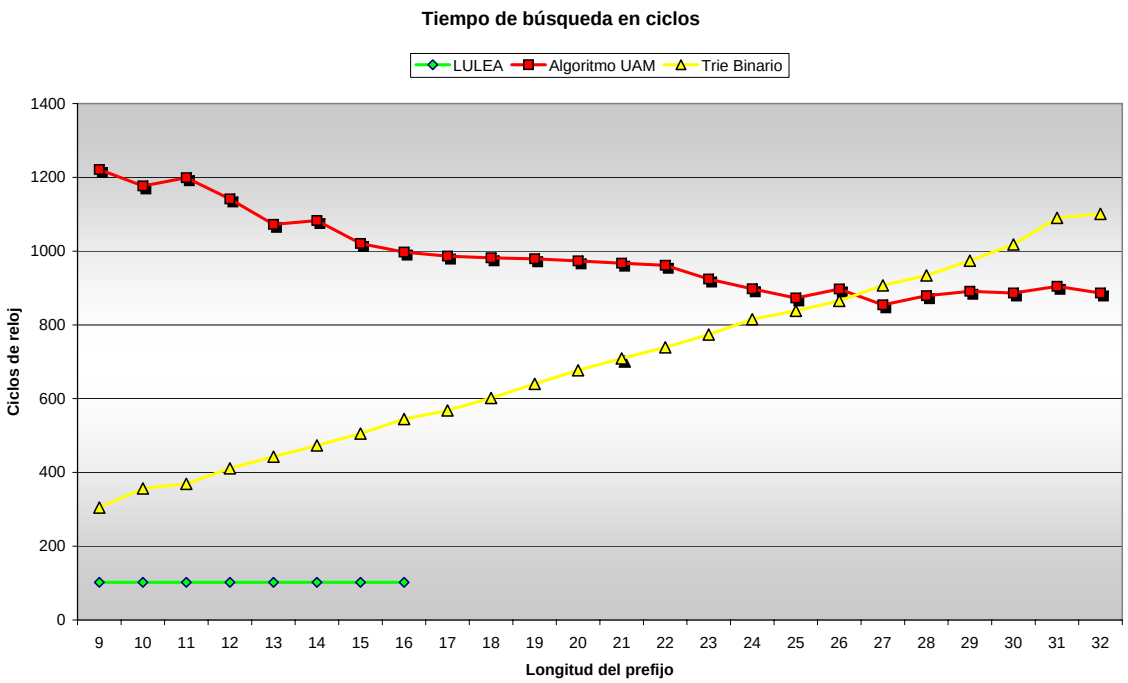


Figura 6.1: Gráfica de Resultados en ciclos de reloj

Observamos que para el algoritmo de la Universidad de Lulea reportó el menor costo, pero solamente se tienen la simulación para la primera sección, el problema para continuar con las otras dos secciones es la implementación de este algoritmo, para preparar la estructura donde se lleva a cabo la búsqueda, se realizan procedimientos costosos en tiempo de procesamiento. La segunda y tercera sección tiene este mismo problema.

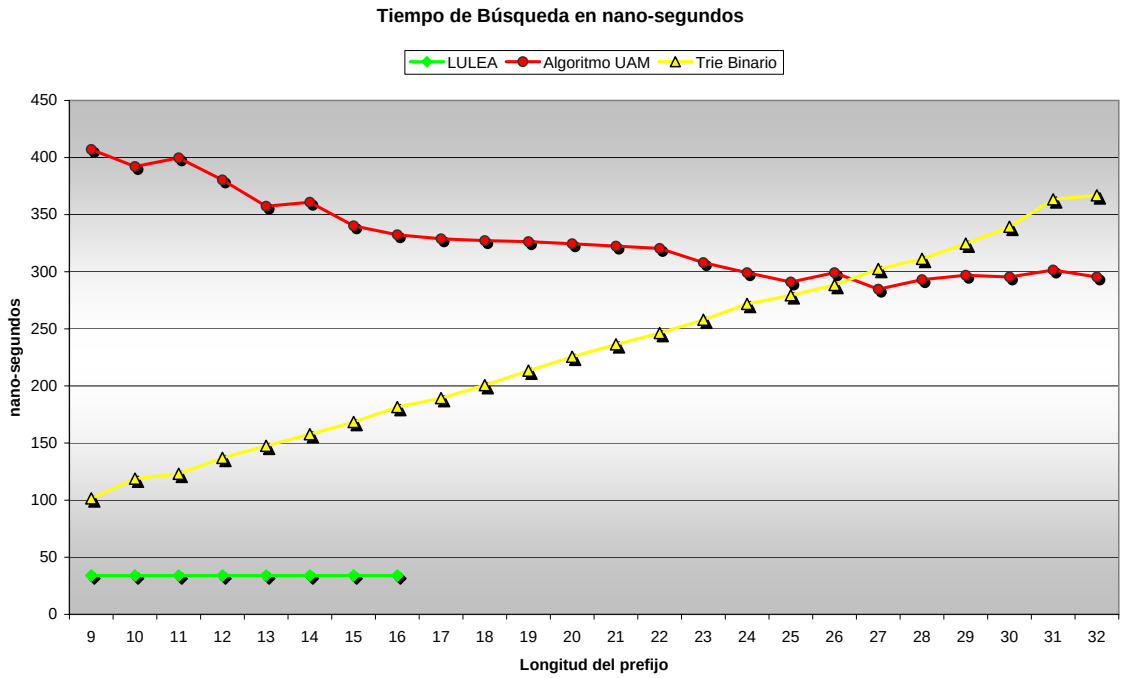


Figura 6.2: Gráfica de Resultados en nano-segundo

El algoritmo "Trie" Binario se tarda de 304 a 1100 ciclos de reloj en encontrar el BMP y el tiempo depende que tan largo es el prefijo qué se encuentra.

El algoritmo UAM se tarda de 886 a 1221 ciclos de reloj en encontrar el BMP de lo que se tarda en realizar una búsqueda binaria sobre el arreglo de ángulos y después una búsqueda lineal en el arreglo de ángulos.

En la Gráfica 6.2 se muestra la conversión con respecto a las características del equipo de cómputo que utilizamos, donde la búsqueda tarda de 295 a 407 nanosegundos.

El algoritmo con el mejor rendimiento es el de la universidad de Lulea, pero solamente se implemento para la primera sección del árbol. El método de compresión que utiliza es costoso en tiempo y compleja para generar las estructuras de búsqueda en cada sección.

El algoritmo “Trie” Binario tiene un rendimiento medio, que representa el recorrido que se realza en el árbol, donde el tiempo esta relacionado directamente con la longitud del prefijo.

El algoritmo UAM tiene un retardo mayor en prefijos de 9 bits de longitud hasta 24 bits de longitud después tiene un rendimiento semejante con respecto al “Trie” binario con prefijos de 25 y 26 bits de longitud y de 27 a 32 bits de longitud el algoritmo UAM tiene un mejor rendimiento que el “Trie” Binario, cabe mencionar que esta propuesta es la inicial, y cuenta apenas con un funcionamiento básico.

Los resultados de la primera propuesta son alentadores, el algoritmo funciona con una estructura muy sencilla, y se puede comparar con los algoritmos propuestos.

Estos resultados pueden mejorar, ya que esta primera propuesta es general. Cómo mejorarlo se explica en la sección de trabajo futuro.

Los prefijos de red que son comunes en Internet son de 24 bits de longitud, y en el resultado que genera nuestra propuesta está cercano al algoritmo “Trie” binario para esta longitud de prefijo.

En la siguiente sección se presentan las conclusiones de esta tesis y se propone el trabajo que a futuro se debe realizar para optimizar al algoritmo UAM.

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Conclusiones

El algoritmo UAM encuentra el prefijo BMP que es el que tiene el mayor número de bits que coinciden con los más significativos de la dirección IP.

Para clasificar un prefijo de red dentro de la circunferencia se utilizan dos formulas para obtener el ángulo que le corresponde y el limite superior de su cobertura. Este mapeo depende del valor y la longitud del prefijo. El mapeo sirve para convertir la búsqueda de prefijos en una búsqueda exacta.

El algoritmo UAM tiene una implementación sencilla, las estructuras de datos que utiliza son arreglos ordenados, uno se utiliza para guardar el valor de los ángulos que existen en la circunferencia, para la información de ruteo y los limites superiores de la cobertura. Se utilizan también tantos arreglos como el número de bits que forman la dirección IP, en estos arreglos se guarda el límite superior de la cobertura junto con la información de ruteo acorde a la longitud del prefijo de red. Así, las dos estructuras en donde se realizan las búsquedas son arreglos ordenados, en el primer

arreglo se realiza una búsqueda binaria y con el índice encontrado de está búsqueda se realiza en seguida la búsqueda lineal sobre los arreglos de longitud de prefijos.

Los resultados de la simulación no están alejados de los resultados de los algoritmos clásicos que implementamos. Además es posible optimizar el funcionamiento del algoritmo UAM para tener mejores resultados en tiempo, ya que la búsqueda lineal que se hace en los arreglos de longitud de prefijos, se puede mejorar por que no en todas las entradas en los arreglos de longitud de prefijos se debe buscar.

Generalmente el número de entradas en el arreglo de ángulos es menor que el número de prefijos en la tabla de ruteo. Esto se debe a que un ángulo puede tener información de más de un prefijo.

El tiempo de Búsqueda del algoritmo UAM está relacionado directamente con la cantidad de prefijos en la tabla de ruteo, no importando la longitud de la dirección IP. La implementación para IPv6 donde la dirección IP es de 128 bits tiene el rendimiento relacionado únicamente con la cantidad de prefijos en la tabla de ruteo.

La principal aportación de este trabajo es la propuesta del mecanismo que permite convertir la búsqueda de prefijos en una búsqueda exacta dentro de un arreglo. El tiempo de búsqueda no es dependiente de la longitud de la dirección IP, depende del número de prefijos en la tabla de ruteo. El tiempo para actualizar la información de los prefijos de red está normalizado con respecto al número de prefijos en la tabla de ruteo. Las estructuras de datos utilizados son sencillas ya que se trata de arreglos ordenados y estructurados.

Los sistemas digitales que necesitan compartir información, cada vez son más pequeños, y se pueden utilizar en cualquier parte, esto lleva al problema de que

tendremos más sistemas que necesiten enviar información a través de las redes, se trabaja en redes de sensores que también tendrán la necesidad de atravesar redes, probablemente de menor extensión pero de igual densidad de nodos. El número de entradas en la tabla de ruteo será grande y se usará mayor tiempo de búsqueda. Así, proponer un algoritmo que sea menos costoso en tiempo de búsqueda es necesario para el futuro.

7.2. Trabajo Futuro

El algoritmo UAM se divide en dos búsquedas: la búsqueda en el arreglo de ángulos y la búsqueda en los arreglos de longitud de prefijos.

Primera propuesta para trabajo futuro:

En la búsqueda del arreglo de ángulos se puede utilizar árboles "b" para mejorar el tiempo. Este tipo de árboles es utilizado para realizar búsquedas cuando se tiene un conjunto de datos masivo. La actualización de información de los prefijos y la búsqueda en este tipo de árboles también tiene un comportamiento logarítmico y ésta estructura es comúnmente utilizado en bases de datos y sistemas de archivos.

Segunda propuesta para trabajo futuro:

La búsqueda binaria que se realiza sobre el arreglo de ángulos regresa una bandera, si se encontró el ángulo que se busca en el arreglo, lo único que se debe hacer es reportar la información del prefijo más largo que esté registrado en los arreglos de longitud de prefijo, este es el mejor caso, la complejidad computacional sería de $O(\log(M))$ con M igual al número de prefijos en la tabla de ruteo.

El otro caso es; si no existe el ángulo en el arreglo, entonces se busca en los

arreglos de longitud de prefijos sabiendo que no puede ser el prefijo de la máxima longitud ya que no es igual al ángulo. Por medio de una diferencia entre el ángulo que se busca relacionado con la dirección IP y el ángulo que se obtiene de la búsqueda binaria sobre el arreglo de ángulos se puede obtener un índice hacia el arreglo de longitud de prefijos en donde se debe comenzar a buscar reduciendo el número de iteraciones sobre los arreglos de longitud de prefijos.

Tercera propuesta de trabajo futuro:

El algoritmo UAM puede segmentar los espacios de búsqueda, ya que con el esquema de clasificación del algoritmo se pueden tomar diferentes circunferencias, es decir, se pueden tomar diferentes longitudes de prefijos y realizar búsquedas parciales con respecto a la longitud de los estos prefijos. Se debe observar el comportamiento del algoritmo, tomando en cuenta para la longitud de la búsqueda, la frecuencia con que se presentan los prefijos de red.

Cuarta propuesta de trabajo futuro:

Se requiere implementar el algoritmo en un sistema dedicado y someterlo a trazas de ruteadores de alto rendimiento para medir el tiempo real de la búsqueda. Se propone que se utilice circuitos lógicos programables en específico FPGA para esta implementación, es decir, realizar un ruteador con este sistema y probar su rendimiento real.

Apéndice A

Justificación de la cantidad de experimentos realizados

Los resultados que se presentan en esta tesis como promedios son obtenidos mediante la formula A.1, que es una forma para calcular el promedio cada vez que se realiza una simulación.

$$\overline{X}_{i+1} \;=\; \overline{X}_i + \frac{x_{i+1} - \overline{X}_i}{i+1}, i > 2 \tag{A.1}$$

Esta función es iterativa, al final de la simulación obtenemos el promedio del tiempo que tarda el algoritmo en regresar la información de ruteo. Cada vez que se realiza una búsqueda se registra el tiempo de inicio y cuando termina se registra el tiempo final, los cálculos para obtener otra dirección a buscar, registrar el resultado, calcular el promedio y la varianza no tienen influencia en el proceso de búsqueda.

Con esta forma de calcular el promedio también se aplica la formula A.2, que obtiene la varianza de forma recursiva por cada simulación.

$$S^2_{i+1} = (1 - \frac{1}{i})S^2_i + (i+1)(\overline{X}_{i+1} - \overline{X}_i)^2, i > 2 \quad \text{(A.2)}$$

Con los resultados de la varianza, podemos decir hasta donde es posible detener la simulación.

Podemos observar que para mil simulaciones, es posible reportar el promedio del tiempo que tarda en obtenerse la información de ruteo ya que la varianza se ha estabilizado.

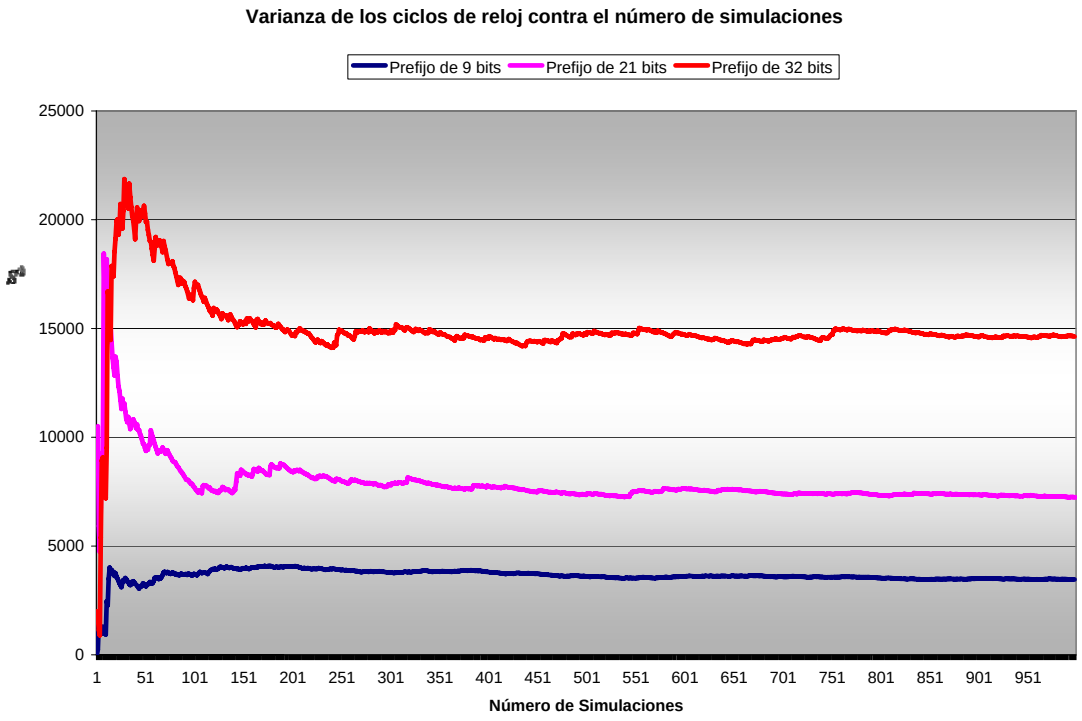


Figura A.1: Gráfica de varianza muestral para simulaciones de 9,21 y 32 bits

En la gráfica A.1, se muestran los valores de la varianza para las simulaciones de longitudes de 9, 20 y 32 bits, en la simulación del "trie"binario.

En las gráficas A.2, se muestra los valores de la varianza para las simulaciones de longitudes de 9, 20 y 32 bits, en la simulación del algoritmo UAM.

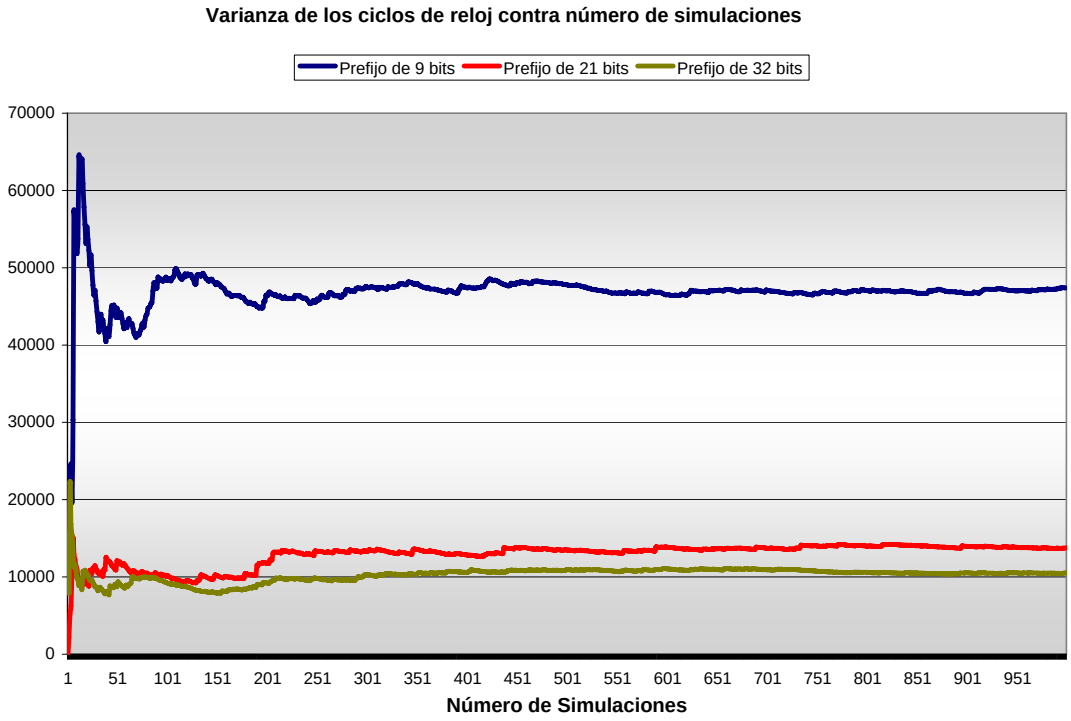


Figura A.2: Gráfica de varianza muestral para simulaciones de 9,21 y 32 bits

En las gráficas se observa como la varianza es estable en las 1000 simulaciones, por lo tanto las simulaciones pueden detenerse en este punto.

Bibliografía

[1] DARPA INTERNET PROGRAM, *RFC971, Internet Protocol*, Defense Advanced Research Projects Agency, Septiembre 1981.

[2] E. Gerich, *RFC1466, Guidelines for Management of IP Address Space*, Mayo 1993.

[3] V. Fuller, T. Li, J. Yu, K. Varadhan *RFC1519, Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*, BARRnet, cisco, MERIT, OARnet, Septiembre 1993.

[4] Miguel Á. Ruiz-Sánchez, Ernst W. Biersack, Walid Dabbous, *Survey and Taxonomy of IP address lookups algorithms*, IEEE Network, March/April 2001

[5] Mikael Degermark, Andrej Brodnik, Svante Carlsson and Stephen Pink, *Small Forwarding Tables for Fast Routing Lookups*, Departament of Computer, Science and Electrical Engineering,ACM SIGCOMM 1997

[6] V. Srinivasan, George Varghese, *Faster IP Lookups using Controlled Prefix Expansion*, SIGMETRICS 1998 Madison USA, ACM

[7] Butler Lampson, Venkatachary Srinivasan and George Varghese, *IP Loookups Using Multiway and Multicolumn Search*, IEE / ACM Transactions on Networking, Vol. 7 No3, Junio 1999

[8] Marcel Waldvogel, George Varghese, Jon Turner and Bernhard Plattner, *Scalable High Speed IP Routing Lookups*, SIGCOMM 1997 Cannes Francia

[9] Philip Smith, *"BGP Routing Table Analysis Reports"*, BGP Routing Table Analysis Reports TELSTRA, http://bgp.potaroo.net/v6/as1221/

[10] *"RIS RIPE NCC"*, Réseaux IP Européens Network Coordination Centre, http://www.ris.ripe.net/

[11] *"The CAIDA Web Site"*,Cooperative Association for Internet Data Analysis (CAIDA), http://www.caida.org/

Algoritmos de Búsqueda y Actualización de Información para ruteadores IP

Para obtener el grado de
Maestro en Ciencias
(Ciencias y Tecnologías de la Información)

P R E S E N T A

Ing. Joel Yazbek Buendía Gómez

A S E S O R E S

Dr. Miguel Ángel Ruíz Sánchez
Dr. César Jalpa Villanueva

S I N O D A L E S

PRESIDENTE: Dr. Javier Gómez Castellanos
SECRETARIO: Dr. César Jalpa Villanueva
VOCAL: Dr.Ricardo Marcelín Jiménez
VOCAL: Dr. Miguel López Guerrero

30 de Julio de 2009