4. Algoritmos de Asignación de Rutas

En este apartado se estudiarán los diferentes algoritmos de asignación de rutas, también denominados protocolos de asignación de rutas, utilizados en las simulaciones llevadas a cabo en este estudio.

Se pueden dividir en cuatro clases los algoritmos de enrutamiento: proactivos (también conocidos como guiados por tabla), reactivos (también llamados guiados por demanda), hibridos, y basados en posición.

En los algoritmos proactivos, tales como "destination sequenced distance vector" (DSDV), los nodos mantienen actualizadas todas sus rutas aunque éstas no estén activas en dicho momento. Mensajes entre nodos son continuamente usados para el mantenimiento de dichas rutas. El principal problema que tienen estos algoritmos, es que esos mensajes de mantenimiento de rutas consumen gran parte del ancho de banda de la red, lo que provoca retardos en las entregas de mensajes. Los algoritmos proactivos no son aconsejables para redes dinámicas en escenarios urbanos como puede ser una autopista, o una carretera nacional, debido a que los nodos están entrando y saliendo continuamente de la red.

Los protocolos reactivos, tales como "Ad hoc on-demand distance vector" (AODV) o "dynamic source routing" (DSR), sólo mantienen activas las rutas que mantienen en uso, para evitar ese exceso de mensajes entre nodos. Es decir, cuando un nodo quiere comunicar con otro, si no tiene la ruta, inicializa un proceso de descubrimiento de ruta. Una vez que termina de enviar la información, y si la ruta se mantiene inactiva por un tiempo determinado, ésta se cancelará de la tabla de rutas. El principal inconveniente de este tipo de protocolos es que se crean retardos de transmisión al inicializar las comunicaciones cuando la ruta no se encuentra en la tabla de enrutamiento y un proceso de descubrimiento de ruta se tiene que llevar a cabo. La principal ventaja con respecto a los protocolos proactivos es que se ahorra ancho de banda.

4.1. Protocolos proactivos

A continuación se pasan a describir los protocolos proactivos más utilizados en redes ad hoc.

4.1.1. Destination Sequenced Distance Vector (DSDV)

Está basado en la clásica idea del algoritmo de Bellman-Ford con ciertas mejoras **¡Error! No se encuentra el origen de la referencia.**. Cada nodo mantiene en una tabla una lista con todos los destinos posibles, el número de saltos hasta alcanzar el destino y un número de secuencia asignado por el nodo destino. El número de secuencia se utiliza para diferenciar rutas antiguas, de rutas nuevas. Los nodos envían periódicamente sus tablas de enrutamiento a sus nodos vecinos. De igual forma ésta es enviada si la tabla ha sufrido algún cambio significativo. Luego se puede decir, que la actualización de las

tablas está basada en dos sucesos, uno temporal, es decir periódicamente se actualiza la misma, y otro basado en un evento que produce un cambio importante en la tabla. Por ejemplo la entrada de un nuevo nodo en la red. El envío de la tabla se puede realizar de dos formas, la primera es el envío completo de la tabla, o un envío incremental. Cuando la red se encuentra relativamente estable, los envíos totales son evitados para no saturar con extra tráfico la red. Por el contrario en redes que están constantemente cambiando, envíos completos son preferidos antes que envíos parciales.

La información sobre las rutas de comunicaciones es actualizada periódicamente como se ya se ha comentado. Cuando no se recibe información de actualización sobre una de las rutas en varios periodos de actualización consecutivos, ésta es eliminada de la tabla de enrutamiento.

DSDV fue uno de los primeros algoritmos de enrutamiento disponibles. Éste es adecuado para la creación de redes ad hoc con un número pequeño de nodos, bebido a que no existe una descripción formal de este algoritmo, no existe ninguna implementación formal del mismo **¡Error! No se encuentra el origen de la referencia.**. Por contra, muchas mejoras del protocolo se han introducido es diversos artículos.

Cada vez que se produce un cambio en la red se debe realizar un nuevo procedimiento de actualización de tablas. Esta es la principal causa que hace inadecuado DSDV para escenarios altamente dinámicos.

4.1.2. Tailoring Link-State Algorithm (OLR)

La idea básica de OLR es el uso de multipoint relay (MPR) nodes, para la transmisión de paquetes de forma efectiva y evitando la duplicidad de paquetes en una misma zona. Es decir, con este protocolo se intenta evitar que un grupo de nodos que están en el mismo rango de cobertura tengan que enviar los mismos paquetes a los mimos nodos, reduciendo así el flujo de información en la red.

Cada nodo i elige uno conjunto de nodos denominados MPR(i), los cuales se encuentra a un salto del nodo i. Los nodos en MPR(i) tiene la siguiente propiedad, cada nodo en la vecindad próxima de dos saltos de i, debe tener un enlace simétrico hacia MPR(i). Cuando un nodo quiere enviar un mensaje, éste envía el mensaje a todos los nodos que componen su MPR(i).

En la siguiente Figura 1 se puede observar la representación de varios MPRs. Cada nodo periódicamente inunda su MPR con mensajes de control denominados “Toplogy Control” (TC), estos mensajes son utilizados para mantener actualizada la topología del MPR. Cada TC contiene un número de secuencia, el cual se incrementa cada vez que un cambio en producido en el MPR.

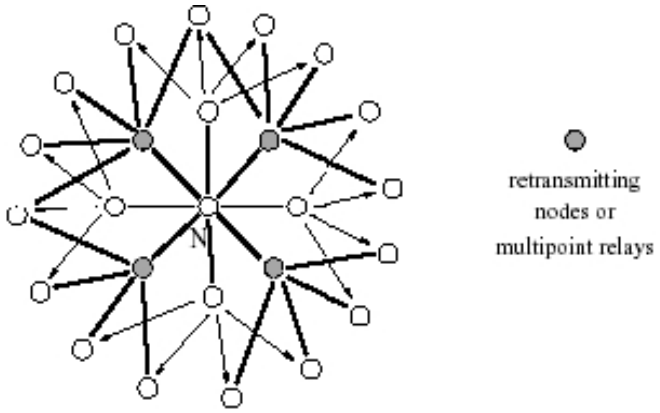

Figura 1: MPRs en OLSR.

4.1.3. Merging Distance Vector and Link State Behaviours (FSR)

FSR de las siglas Fisheye State Protocol, es un protocolo proactivo basado en la denominada “fisheye technique” (técnica de ojo de pez).

La principal aportación de este protocolo es que los nodos mantienen una información más detallada de los nodos que se encuentran en su vecindad. Mientras que la información de red es más pobre conforme nos alejamos de los nodos. Es decir, los nodos más cercanos se encuentran más actualizados de los cambios de red producidos en sus proximidades. Mientras que los cambios más lejanos son menos informados.

Esta técnica proporciona buenos resultados para redes ad hoc muy grandes. Ya que cambios en la red producidos en la lejanía de un nodo, son menos probables a afectar a la transmisión de mensajes de éste.

En la siguiente, Figura 2 se puede observar un ejemplo del protocolo FSR.

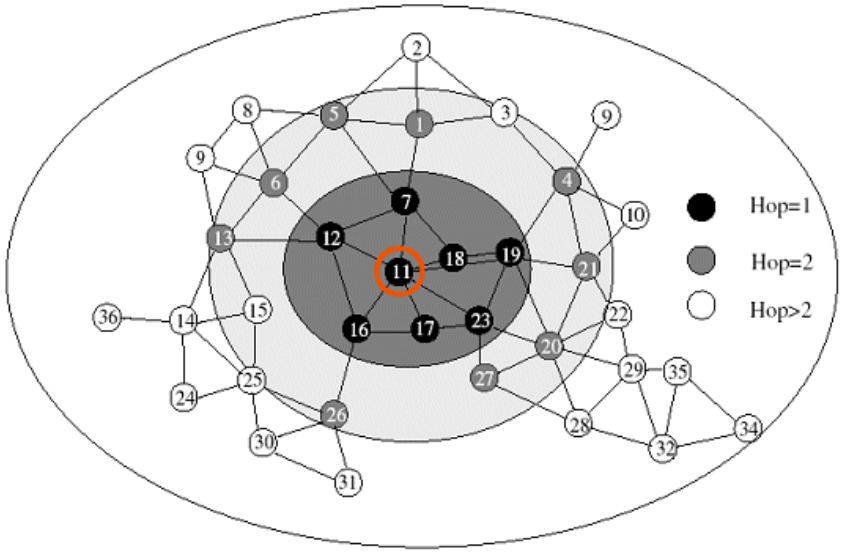

Figura 2: FSR routing protocol.

4.2. Protocolos reactivos

4.2.1. Ad hoc On-Demand Distance Vector (AODV)

Fue desarrollado por el centro de investigación de Nokia en la Universidad de California (Nokia Research Center of University of California), Santa Barbara y la Universidad de Cincinati. Los investigadores que desarrollaron el protocolo fueron C. Perkins y S. Das.

Se trata de uno de los protocolos de enrutamiento más utilizado para aplicaciones móviles. En concreto, para MANETs y otros tipos de redes ad-hoc inalámbricas, donde el número de nodos es elevado. El algoritmo crea una ruta entre dos nodos, sólo cuando ésta es demandada por el nodo fuente. Esto permite a los nodos tener flexibilidad para entrar y salir de la red. Las rutas permanecerán activas sólo cuando existan paquetes transmitiéndose desde el nodo fuente al nodo destino. Cuando el nodo fuente termine de enviar paquetes y cierto time out expire, la ruta entre los nodos se cancelará.

AODV soporta enrutamientos tanto unicast como multicast. Básicamente la diferencia estriba en que en las comunicaciones unicast la información parte de un sólo origen a un único destino, mientras que es las comunicaciones multicast los datos parten de un sólo origen hacia múltiples destinos. Por ejemplo el envío de un e-mail a varios destinatarios o una videoconferencia, son algunos posibles ejemplos de comunicaciones multicast.

AODV es un protocolo bajo demanda, eso quiere decir que las rutas sólo son creadas bajo demanda de los nodos fuentes de la información. Estas rutas son mantenidas mientras que éstas son necesarias. AODV es conocido como un protocolo reactivo, a diferencia de otros muchos protocolos comunes en internet que son proactivos. En los protocolos proactivos las rutas son buscadas y mantenidas, aunque éstas no estén activas, como se describió en el capítulo anterior.

4.2.1.1. ¿Cómo funciona AODV?

AODV utiliza tablas de enrutamiento para almacenar la información de red. Una tabla para rutas multicast y otra para rutas unicast. La información que mantienen dichas tablas se lista a continuación:

- Dirección de destino.
- Dirección del siguiente nodo en caso de multi-hop (Hop Address).
- Número de nodos en caso de multi-hop (Count Hop).
- Número de secuencia de destino.
- Tiempo de vida.

Existen cuatro tipos de mensajes que son utilizados entre los nodos para el descubrimiento y mantenimiento de las rutas. Estos mensajes son:

- Route Request (RREQ).
- Route Reply (RREP).
- Route Error (RERR).
- HELLO messages.

Los mensajes RREQ y RREP se utilizan para el descubrimiento de rutas, mientras que los mensajes RERR y HELLO son usados para el mantenimiento de las mismas. El siguiente diagrama Figura 3, ilustra básicamente AODV.

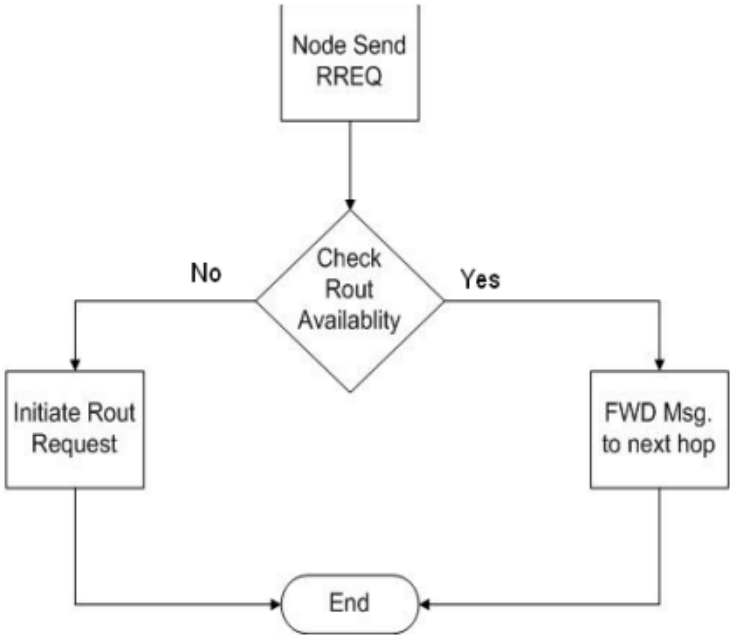

Figura 3: Funcionamiento básico de AODV.

Cuando un nodo fuente (S) desea enviar un paquete a un nodo destino (D), primero comprueba si la ruta se encuentra en su tabla de enrutamiento. Si la ruta se encuentra en la tabla, el nodo fuente envía la información al siguiente nodo en la ruta objetivo. En caso contrario, es decir la ruta no se encuentra en la tabla, el nodo inicializa el proceso de descubrimiento de rutas.

Proceso de Descubrimiento de Rutas:

AODV adquiere las rutas mediante mensajes request/reply. Por ejemplo, cuando un nodo fuente (S) desea enviar un paquete a un nodo destino (D) que no tiene en su tabla de enrutamiento, el nodo fuente envía un mensaje request a todos los nodos a su alcance. Es decir, realiza un broadcast con un mensaje tipo RREQ.

La información que contiene el mensaje enviado por el nodo (S) al nodo (D) contiene la siguiente información:

- Dirección IP del nodo destino.
- Número de secuencia del nodo origen.

- Número de secuencia del nodo destino.
- Número de identificación broadcast.

Cada vez que un nodo realiza el envío de un mensaje broadcast, el número de identificación broadcast se incrementa en uno.

Dependiendo de la información que contenga los paquetes recibidos, los nodos podrán cambiar la información de su tabla de enrutamiento. Los paquetes RREQ contienen el número de secuencia más reciente para el destino solicitado por la fuente. Un nodo que recibe un paquete RREQ puede contestar a dicho paquete con un mensaje tipo RREP, si y sólo si, es el nodo destino, o tiene una ruta para el destino solicitado por el nodo origen cuyo número de secuencia es superior al que contenía el paquete RREQ. En caso contrario no contestará el mensaje, sino que retransmitirá el paquete RREQ a todos los nodos a su alcance y almacenará la dirección IP del nodo fuente del cual ha recibido el mensaje. En caso de recibir el mismo nuevamente, éste será descartado.

Una vez que el nodo origen recibe el paquete RREP con la ruta hacia el nodo destino, el nodo almacenará dicha información en su tabla de enrutamiento. En el caso de que más tarde recibiera otra respuesta con un número de secuencia mayor, o igual pero con un número de saltos inferior, el nodo fuente volverá actualizar su tabla.

La ruta será mantenida mientras que ésta se encuentre activa. Una ruta se considera activa mientras que exista una transmisión de paquetes periódica entre el destino y la fuente de la ruta. En caso de que la fuente pare de enviar paquetes, y transcurrido un tiempo de time out, la ruta se considerará inactiva y se eliminará de la tabla. Si el enlace se rompiera por cualquier motivo mientras la ruta se encuentra activa, se generará un mensaje de error RERR hacia el nodo origen. El nodo fuente puede volver a reiniciar el descubrimiento de la ruta en caso de considerarlo necesario.

En la siguiente imagen Figura 4, se resume el proceso de descubrimiento de rutas.

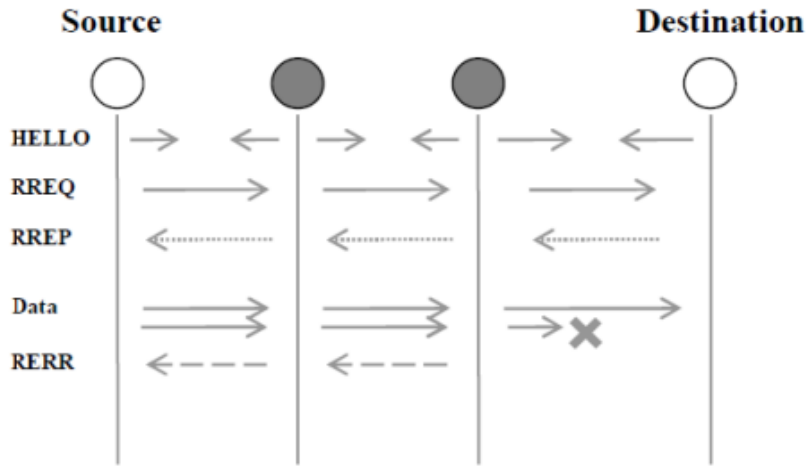

Figura 4: Descubrimiento de rutas en AODV.

El proceso de descubrimiento para rutas multicast es muy parecido al descrito anteriormente para rutas unicast. Para enfatizar más en el proceso de descubrimiento de rutas en AODV, es cual es uno de los aspectos más importantes del protocolo, el siguiente ejemplo detalla las múltiples posibilidades del proceso de descubrimiento, Figura 5.

1. El nodo S necesita una ruta al nodo D.
2. El nodo S crea un mensaje de petición de ruta (RREEQ). En el paquete se introduce la dirección IP de D, el número de secuencia, dirección IP de S, y número de saltos hasta D.

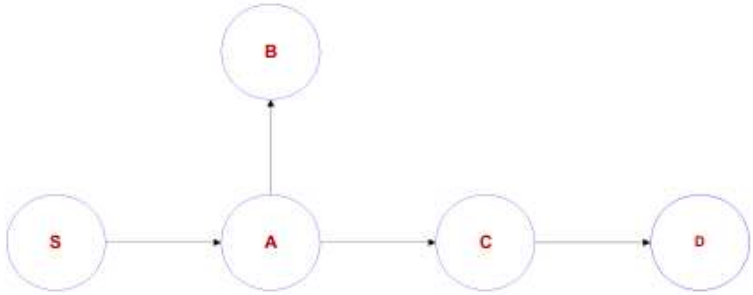

Figura 5: Descubrimiento de ruta AODV I.

3. El nodo S broadcast el mensaje RREQ a sus nodos vecinos, Figura 6.

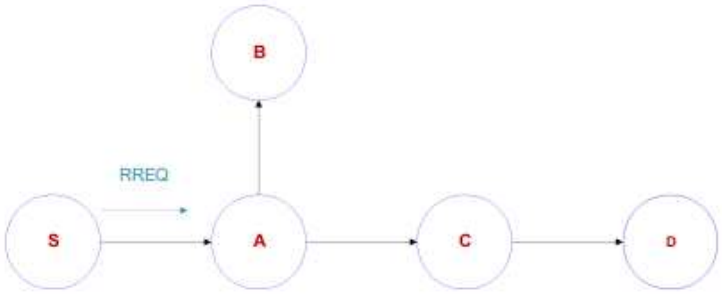

Figura 6: Descubrimiento de ruta AODV II.

4. El nodo A recibe el mensaje RREQ, Figura 7.

- Hace una entrada invertida en su tabla a la solicitud de S. Introduce destino S, siguiente salto S, número de cuenta igual a 1, ya que este es el primer salto.
- Como este no tiene ruta para el destino D, vuelve a retransmitir el paquete a todos sus vecinos.

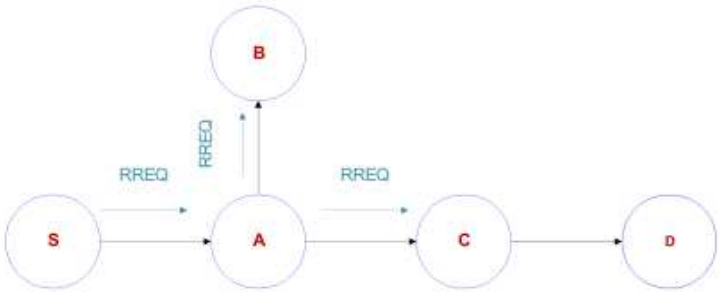

Figura 7: Descubrimiento de ruta AODV III.

5. El nodo C recibe el mensaje RREQ.

- Hace una entrada invertida para la petición de S. Destino S, Siguiente salto A. Número de cuenta 2, ya se han producido dos saltos.
- Éste tiene una ruta para D, y el número de secuencia es mayor que el que se encuentra en el mensaje RREQ, Figura 8.
- C crea un mensaje de réplica a la petición de ruta RREP. Introduce la dirección IP de D, el número de secuencia, la dirección IP de S, y el número de salto lo pone a 1.
- Crea un mensaje unicast de vuelta hacia A.

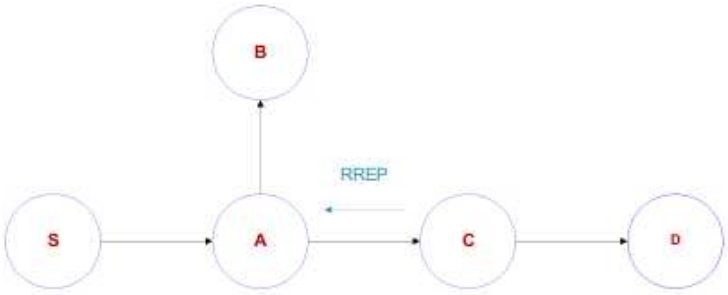

Figura 8: Descubrimiento de ruta AODV IV.

6. El nodo A recibe el mensaje de replica RREP, Figura 9.

- Realiza una entrada a D. Destino D. Siguiente nodo C. Número de saltos 1.
- Crea un mensaje unicast de vuelta hacia S

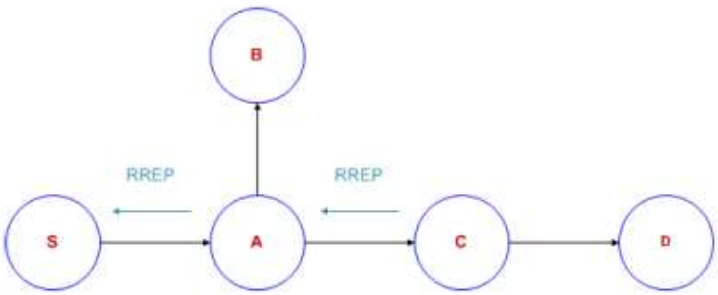

Figura 9: Descubrimiento de ruta AODV V.

7. El nodo S recibe el mensaje de réplica de ruta RREP, Figura 10.

- Realiza una entrada a D. Destino D. Siguiente nodo A. Número de saltos 3.
- La ruta ha sido ya establecida, luego se puede comenzar a enviar datos desde S a D.

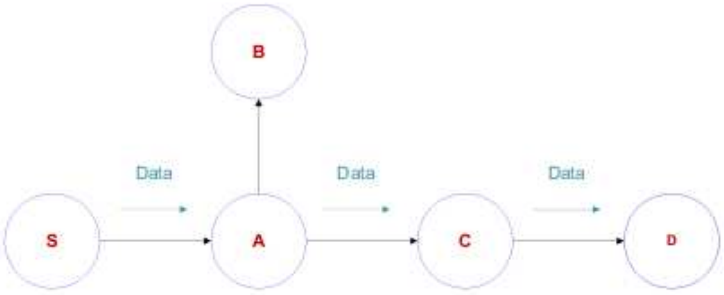

Figura 10: Descubrimiento de ruta AODV VI.

4.2.2. Dynamic Source Routing (DSR)

La principal característica de DSR es el uso de fuentes de enrutamiento. Esto es, el emisor conoce la ruta completa de saltos hasta llegar al receptor **¡Error! No se encuentra el origen de la referencia.**. Estas rutas son almacenadas en memoria cache. Los paquetes contienen la ruta hacía la fuente en su cabecera. Cuando un nodo intenta enviar un paquete a un nodo, del cual no conoce su ruta, éste inicializa un proceso de descubrimiento de ruta, con el propósito de determinar dinámicamente la ruta a tomar. El descubrimiento de rutas consiste nuevamente en el envío de paquetes RREQ. Cuando los nodos reciben estos paquetes vuelven a reenviar éstos a sus vecinos, a no ser que ellos sean el destino, o tengan una ruta hacia el destino solicitado por la fuente. La respuesta a los paquetes RREQ es nuevamente un paquete RREP. La información referente a la nueva ruta creada es almacenada en memoria cache, tanto para los nodos de ida, es decir ruta desde el origen al destino, como para la vuelta, es decir nodos desde el destino a la fuente. Cuando un enlace se rompe, dicho evento se informa a los nodos mediante mensajes de error RERR. El nodo origen pues, eliminará dicha ruta de su memoria cache. Una de las principales diferencias entre DSR y AODV, es que es el primero varias rutas entre un origen y un destino pueden ser almacenadas en la memoria cache. Mientras que en el segundo sólo una entrada entre origen y destino es introducida en la tabla de enrutamiento.

A continuación se resumen las principales características del protocolo de red DSR:

- Los paquetes de enrutamiento que envían los nodos contienen toda la información de la ruta que éstos deben seguir. Los nodos intermedios pueden adquirir esta información para un posterior uso.
- DSR no requiere paquetes periódicos de mantenimiento de rutas. Lo que significa, que funciona totalmente bajo demanda “On Demand”.

- Cuando todos los nodos se encuentran en un estado estacionario, no existen mensajes de red entre ellos, lo cual evita embotellamientos. Solamente cuando se producen cambios, debidos a movimientos en la red, se producen mensajes de la capa de red.
- Un nodo puede almacenar diferentes rutas para un mismo destino. Esto es una ventaja cuando se rompe el enlace de comunicación en uso. La ventaja se produce por el hecho de que no se tiene que iniciar inmediatamente un nuevo proceso de descubrimiento de rutas. Esto es una ventaja con respecto al protocolo AODV, aunque también puede ser un inconveniente en algunos casos, debido a que se pueden utilizar rutas no actualizadas.

4.2.2.1. Proceso de Descubrimiento de Rutas

Cuando un nodo origen quiere comunicar con un nodo destino, el primero inicia un proceso de descubrimiento de rutas. En primer lugar, éste busca en su memoria cache si tiene alguna ruta para el nodo destino. Es caso afirmativo, envía el paquete al siguiendo nodo en la ruta seleccionada, dicho paquete contiene en su cabecera toda la información de todos los nodos por los que el paquete debe pasar. En caso negativo, el nodo origen envía un mensaje broadcast para intentar conseguir una ruta para el nodo destino. La siguiente Figura 11, ilustra el proceso de descubrimiento cuando un nodo A intenta descubrir una ruta hacia el nodo E. El primer paso es la transmisión del mensaje ROUTE REQUEST a todos sus nodos vecinos. Cada mensaje ROUTE REQUEST, contiene información sobre el nodo fuente, y un identificador único request id. Los mensajes ROUTE REQUEST, contienen también una lista de nodos por los que han sido retransmitidos. Esta lista en comenzada por el nodo fuente.

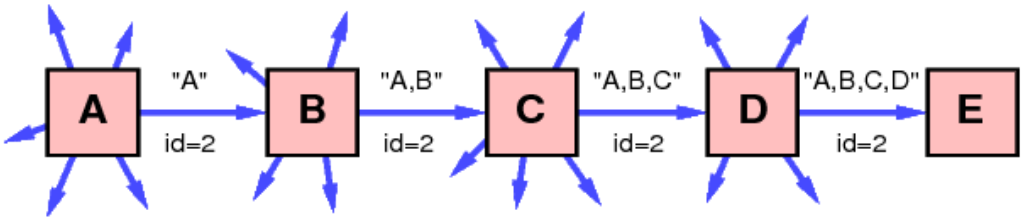

Figura 11: Proceso de descubrimiento de rutas en DSR.

Cuando el nodo destino recibe en mensaje ROUTE REQUEST, éste transmite al nodo fuente un mensaje ROUTE REPLY. Este mensaje contiene todo la información de los nodos por los que pasó el mensaje ROUTE REQUEST. Una vez recibido el mensaje ROUTE REPLY por el nodo origen, éste almacena dicha información en su memoria cache. El nodo E buscará en su memoria cache una ruta de vuelta para el nodo A. En caso que éste no tuviera ninguna ruta de vuelta para el nodo A, éste podría utilizar la información que contiene el paquete ROUTE REQUEST.

Los nodos origen almacenan una copia del mensaje ROUTE REQUEST, en un buffer denominado Send Buffer. Se almacena también el tiempo en el que dicho mensaje se

introdujo en el buffer. Una vez recibido un mensaje de respuesta del nodo destino, este mensaje es borrado del buffer. En el tiempo en el que el mensaje se mantiene en el buffer, otros procesos de descubrimiento hacia el nodo destino pueden ser inicializados. Aunque esto se debe evitar, debido a que produce problemas de acumulación de tráfico. Es por ello, que existe un tiempo de "back-off", para limitar la inicialización continuada de procesos de descubrimiento.

4.2.2.2. *Proceso de Mantenimiento de Rutas*

Cuando se lleva a cabo un proceso de descubrimiento, cada nodo es responsable de confirmar que el siguiente nodo en la ruta ha recibido correctamente el mensaje. Cada paquete puede ser retransmitido hasta un número determinado de intentos. En la siguiente Figura 12, el nodo A ha originado un paquete para el nodo E. El paquete deberá transmitirse a través de los nodos B, C y D. En este caso el nodo A es responsable de la recepción del paquete por parte de B. El nodo B es responsable de la transmisión al nodo C, y así sucesivamente hasta que el nodo D es responsable de que el nodo destino E, reciba correctamente la información originada en el nodo A.

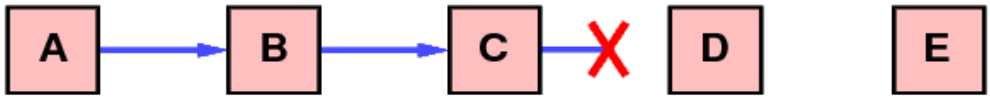

Figura 12: Proceso de mantenimiento de rutas en DSR.

Cuando un mensaje no puede ser retransmitido al siguiente nodo en la ruta, se genera un mensaje de error ROUTE ERROR. Este mensaje identifica en que punto de la ruta se ha roto el enlace. Por ejemplo en la figura anterior, el nodo C no puede conectar con el nodo D. Por este motivo el nodo C genera un mensaje de error ROUTE ERROR, que será retransmitido hacia el nodo origen A. Cuando este mensaje de error es recibido por el nodo A, éste elimina dicha ruta de su memoria cache.

4.2.3. Temporally Ordered Routing Algorithm (TORA)

El protocolo TORA pertenece a la familia de algoritmos conocidos como "link reversal". Este algoritmo es diseñado para reaccionar de manera efectiva antes cambios en la red y particiones de la misma. El nombre de este protocolo proviene de la asunción de tener relojes sincronizados en los nodos, esto se puede obtener vía GPS.

El principal objetivo de TORA es obtener rutas estables, las cuales pueden ser rápidamente y localmente reparadas, en caso de ruptura de enlaces. El protocolo construye un grafo acíclico directo de rutas hacia el nodo destino, denominado "Direct acyclic graph". El DAG se obtiene asignando una dirección a cada enlace entre nodos, y un peso, o nivel de referencia a cada nodo. El grafo DAG tiene la siguiente propiedad, sólo existe un nodo destino, mientras que los restantes nodos tienen al menos un enlace de salida.

El funcionamiento del protocolo TORA puede ser dividido en tres fases:

- Descubrimiento de ruta.
- Mantenimiento de ruta.
- Cancelación de ruta.

Durante la fase de descubrimiento los nodos pueden ser vistos como tuberías, y el agua que circula por las tuberías representaría los paquetes, esta corriente circularía hasta el nodo destino.

El objetivo del mantenimiento de rutas es el de mantener actualizado el gráfico DAG. Una de las principales ventaja que aporta TORA es que cuando un enlace se rompe en un nodo que tiene más de un enlace de salida, ningún procedimiento de cambio es efectuado en la red, ya que la comunicación continúa de manera ordinaria.

Por el contrario cuando un nodo detecta que no tiene enlaces de salida, este evento es propagado por la red a todos sus nodos antecesores. En consecuencia el gráfico DAG es modificado.