Sistemas Informáticos

Curso 2007-2008

Encaminamiento Adaptativo en Redes Ad Hoc mediante el Algoritmo Ant Colony Optimization

José María Benítez Escario
Carla Díaz Martín
Víctor Nasser López

Dirigido por:

Prof. Borja Manero Iglesias
Dpto. Ingeniería del Software e Inteligencia Artificial (DISIA)

Prof. Luis Javier García Villalba
Grupo de Análisis, Seguridad y Sistemas (GASS)
Dpto. Ingeniería del Software e Inteligencia Artificial (DISIA)

Facultad de Informática

Universidad Complutense de Madrid

Resumen

This document shows a new perspective for the design of routing protocols in Ad Hoc Networks. Due to their dynamic topology it is necessary to use too many recurses to study the status in order to get the optimum route. A more global vision is adapted for dealing with not only to find the optimum route, but also to improve the network resource performance.

On first place a theoretical analysis of the problem will be taken, dealing with the consequences of a metric change while evaluating the routes and the minimum information necessaries to get a good route. The previous result is then argumented with a routing proposal based on a revised Ant Colony Optimization algorithm.

Este documento presenta un cambio de perspectiva a la hora de diseñar protocolos de encaminamiento en redes Ad Hoc. Dichas redes tienen la caracteristica de presentar una topologia dinamica, este hecho obliga a derivar un número de recursos elevado a capturar el estado de la red, la topologia, para poder realizar un cálculo de rutas óptimo. Se adopta una vision mas global que no trata exclusivamente de una búsqueda de rutas óptimas si no de una optimizacion de recursos de la red.

La primera parte del documento consiste en un analisis teorico del problema, las implicaciones de un cambio de metrica a la hora de evaluar las rutas y la información minima necesaria para efectuar un buen encaminamiento. El estudio teorico previo se incorpora a una propuesta de encaminamiento mediante una modificación del algoritmo Ant Colony Optimization.

Índice general

Capítulo 1

Introducción

1.1. Redes inalámbricas.

A medida que las redes inalámbricas vayan evolucionando, los clientes solicitarán aplicaciones de red incluso en casos donde no exista una infraestructura de red propia. Los usuarios de terminales móviles con dispositivos inalámbricos compatibles entre sí deberán ser capaces de establecer una red de corta duración que les permita satisfacer sus necesidades de comunicación en un momento determinado, es decir, deberán poder implementar una red *ad hoc*.

Pero las redes *ad hoc* no sólo tendrán utilidad como redes aisladas e independientes. Su papel será primordial al ser empleadas en los 'hotspots' o lugares de interés (hoteles, aeropuertos, centros comerciales, etc.) donde existe una alta concentración de personas con necesidad de establecer al mismo tiempo comunicaciones de voz y/o datos. Para poder cubrir las necesidades de dichos usuarios se suele usar el estándar de tecnología WLAN (Wireless Local Area Networks) denominado IEEE 802.11.

Cuando las WLAN con IEEE 802.11 operan en el modo basado en infraestructura, la red está constituida por al menos un punto de acceso conectado a la infraestructura de la red cableada y a un conjunto de estaciones inalámbricas.

En cambio, cuando las WLAN con IEEE 802.11 operan en el modo *ad hoc*, lo que tenemos es un conjunto de estaciones inalámbricas que se comunican directamente entre sí, sin la presencia de puntos de acceso, evitándose de esta forma su colapso a la hora de atender a un número tan elevado de usuarios (no todo el tráfico debe de estar dirigido al punto de acceso, tal y como sucede con las WLAN operando en el

modo basado en infraestructura).

Estas propiedades convierten a las redes *ad hoc* en redes altamente flexibles y rápidas de desarrollar, pues no necesitan para su funcionamiento una infraestructura propia.

Actualmente existen 2 tipos de variantes de las redes inalámbricas móviles:

- Las redes que tienen una infraestructura
- Las redes que no tienen una infraestructura

Las que no tienen infraestructura se conocen comúnmente como Redes *ad hoc* (MANET).

1.2. Redes *ad hoc*.

Puede definirse una red *ad hoc* como aquella que establece una comunicación espontánea entre terminales fijos y móviles o sólo móviles, siempre y cuando exista la posibilidad física de lograrlo.

Las redes *ad hoc* están formadas por dos o más dispositivos que son capaces de comunicarse entre sí sin la necesidad de recurrir a una infraestructura de red preexistente, con lo cual no son requeridas estaciones base ni cables ni routers fijos. Dichas redes pueden estar constituidas por grupos de terminales móviles independientes y basados en radio enlaces, aunque también cabría la posibilidad de que alguno de estos dispositivos estuviera conectado a un sistema celular o a una red fija.

Las redes *ad hoc* son adaptativas y están habilitadas para configurarse a sí mismas, prescindiéndose de la intervención de un administrador del sistema. En la práctica, las redes *ad hoc* podrían disponer desde decenas hasta centenares de nodos de comunicaciones capaces de cubrir alcances radio de 30 a 100 metros en interiores y de 100 hasta 300 metros en exteriores.

En las redes *ad hoc* es posible que dos nodos inalámbricos se puedan comunicar entre sí, incluso cuando se hallan fuera de su alcance radio, gracias a la presencia de nodos intermedios que actuarán como routers y reenviarán los paquetes de datos de la fuente al destino. Una red donde la comunicación entre dos estaciones se consigue mediante el reenvío de datos a través de otros nodos intermedios, recibe el nombre de red inalámbrica multisalto.

Las redes *ad hoc* se caracterizan por tener topologías dinámicas, donde los nodos se mueven libremente de manera arbitraria y en un tiempo impredecible y pueden estar constituidas por enlaces unidireccionales (comunicación en un único sentido) o bidireccionales (comunicación en ambos sentidos).

Las redes *ad hoc* presentan restricciones de ancho de banda motivadas por la capacidad variable de sus enlaces inalámbricos y, como consecuencia, en muchas ocasiones se produce congestión. De hecho, el throughput de las comunicaciones inalámbricas es mucho menor que la tasa máxima de transmisión radio debido a causas tales como la contienda en el acceso múltiple, fading, ruido y condiciones de interferencia.

1.2.1. Estructura.

- Las Redes *ad hoc* no tienen una infraestructura establecida.
- Están estructuradas sobre el estándar definido para redes inalámbricas IEEE 802.11
- En este tipo de red todos los nodos son móviles y pueden ser dinámicamente conectados de maneras arbitrarias.
- Una MANET es un sistema autónomo, que opera aislado o que puede ser conectado a través de un gateway a una red estructurada.

1.2.2. Aplicaciones.

Las redes inalámbricas *ad hoc* tienen una amplia variedad de aplicaciones:

- Juegan un papel importante en aplicaciones militares.
- Utilizadas como otra alternativa en comunicaciones marítimas.
- De gran utilidad en conferencias y congresos en los que es necesaria una red para compartir todo tipo de información sin una estación base.
- Particularmente útil para las comunicaciones entre equipos de rescate o policiales donde cada miembro del equipo lleva un equipo móvil.
- Todo tipo de redes en las que los miembros compartan información y adquieran datos en lugares de difícil acceso.

1.3. Desarrollo del Proyecto

Los primeros meses, prácticamente el primer cuatrimestre entero, estuvo dedicado a la investigación y documentación. Los primeros pasos fueron acerca de redes *ad hoc*, que son, cuales son sus características y que problemas presenta en encaminamiento. A continuación se estudiaron las soluciones existentes hasta la fecha, protocolos de encaminamiento típicos de redes *ad hoc*, decimos típicos porque al ser un campo en desarrollo se pueden encontrar infinitas propuestas de encaminamiento. Aunque muchas de ellas son modificaciones sobre uno de los típicos.

La segunda etapa de investigación consistió en documentarse acerca del algoritmo Ant Colony Optimization, concretamente en todas sus aplicaciones al encaminamiento en redes. Con todo esto disponíamos ya de una primera especificación del algoritmo que queríamos llevar a cabo.

Construimos un simulador de redes *ad hoc* en Java especifico para probar el algoritmo y comprobar que podía hacer frente al encaminamiento en un escenario sencillo. Los primeros problemas con que nos topamos fue la imposibilidad de realizar el algoritmo como lo habíamos diseñado. Nuestra primera versión destacaba sobre todo por no realizar búsquedas exhaustivas (anchura), de esta manera optimizamos recursos y es fiel al modelo del Ant Colony Optimization propuesto por Dorigo.

Como comentaremos más adelante este modelo es inviable en redes de topología dinámica debido a la lenta convergencia que presenta, finalmente conseguimos a través del trabajo de la bióloga Deborah Gordon acerca de las hormigas americanas solventar el problema, para ello utilizamos la misma estrategia (descrita en otra parte de la memoria) que utilizan las hormigas reales para la búsqueda de alimento. Con esta modificación el finalmente el algoritmo funcionaba y conservamos la idea primitiva de optimizar los recursos disponibles.

Una vez que el algoritmo encaminaba hacer frente al dinamismo, cambios en la topología, no supuso muchos problemas. La aplicación en Java nos sirvió para todo el desarrollo teórico del algoritmo y para comprobar que efectivamente el algoritmo encuentra la ruta óptima en una topología cambiante.

La última parte del proyecto consistió en la implementación en NS-2 de nuestra propuesta de encaminamiento, Ns-2 es un simulador de redes que permite trabajar con redes *ad hoc*, de esta manera podíamos comprobar si el protocolo ofrecía buenas perspectivas al encaminamiento en redes *ad hoc* o no.

Capítulo 2

Encaminamiento en redes *ad hoc*

El diseño de protocolos de encaminamiento para redes *ad hoc* se ha convertido en un importante desafío debido a la escasez de recursos y a la topología dinamica que poseen este tipo de redes.

Los algoritmos usados tradicionalmente en redes fijas no pueden ser aplicados porque el frecuente intercambio de mensajes para capturar la topología de la red requeriría un uso excesivo de los recursos de la red. Asimismo, los protocolos de encaminamiento en redes *ad hoc* deben operar con tasas de error para los enlaces elevadas y topologías muy dinámicas, en contraposición a las redes fijas.

En muchas ocasiones , la investigación de los protocolos de encaminamiento se centra en redes homogéneas, donde todos los nodos tienen los mismos recursos y capacidades; pero debería tenerse en cuenta que las redes *ad hoc* son heterogéneas .

De hecho, se deberían diseñar protocolos de encaminamiento para redes *ad hoc* que satisficieran básicamente los siguientes criterios :

- **Intercambio de información mínimo**

  El intercambio de mensajes de control para hacer frente al dinamismo operante debe reducirse al minimo, de tal manera que no consuma todos los recursos disponibles.

- **La complejidad del algoritmo de caulculo de rutas debe ser baja**

  Se requieren algoritmos con cálculos computacionales que no sean excesivamente complejos por varias razones : alargar la duracion de los dispositivos que funcionan con bateria y forman parte de la red.

Por otro lado si son dispositivos (sensores) los cuales no tienen una gran capacidad computacional no podran manejar el algoritmo.

- **Adaptación a una topología dinámica**

  El algoritmo deberá ser capaz de reaccionar eficazmente a los cambios que se producen en la topología.

Los protocolos de encaminamiento para redes *ad hoc* se pueden clasificar mediante distintos criterios. La primera clasificación está relacionada con la construcción de rutas bajo demanda o no y la segunda clasificación está vinculada al soporte de calidad de servicio. Estas dos clasificaciones no son las únicas existentes; hubiera sido posible considerar otras clasificaciones atendiendo a si los protocolos de encaminamiento son unicast o multicast, son de camino único o de camino múltiple, etc.

2.1. Protocolos de encaminamiento proactivos, reactivos e híbridos

En general, podemos clasificar los protocolos de encaminamiento en tres categorías diferentes :

- **Protocolos proactivos o globales o basados en tablas**
- **Protocolos reactivos o bajo demanda**
- **Protocolos híbridos**

En la Figura siguiente se observa esta clasificación de los protocolos de encaminamiento, mencionándose aquellos protocolos específicos más importantes.

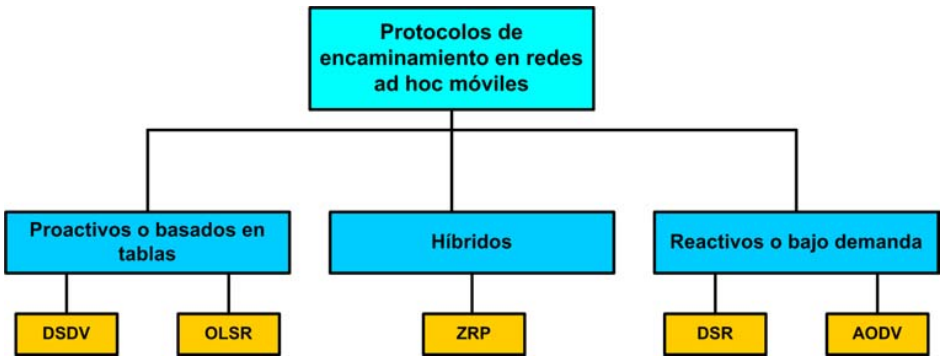


Figura 2.1: Clasificación de los protocolos de encaminamiento en redes *ad hoc*. ()

2.2. Protocolos de encaminamiento proactivos

Los protocolos de encaminamiento proactivos o basados en tablas son aquellos algoritmos que intenatn capturar el estado global de la red en cada instante. Dicha información se halla almacenada en las denominadas tablas de encaminamiento. Este tipo de aproximación presenta un problema básico : deriva recursos a una captura de información en muchos casos inutil, el hecho de ser una topología dinamica invalida la información cada cierto tiempo por tanto para realizar un buen calculo de rutas solo es relevante el estado de la red en el instante de realizar la transmision. Toda la información previamente recogida solo supone un gasto de recursos.

Este tipo de protocolos operará en redes en las cuales se necesite que el procedimiento de Descubrimiento de Ruta no sea excesivamente lento, en tiempo de convergencía. y la red pueda asumir el gasto derivado de mantener una estructura global de toda la red.

A continuación se presentan dos protocolos de encaminamiento proactivos :

- Destination-sequenced Distance Vector (DSDV)
  Basado en el algoritmo de vector distancia.
- Optimized Link State Routing (OLSR)
  Basado en el algoritmo de estado enlace.

2.2.1. Destination-sequenced Distance Vector (DSDV)

Destination-sequenced Distance Vector (DSDV) es protocolo de encaminamiento proactivo de vector distancia.

Este protocolo de encaminamiento se basa en el algoritmo clásico de vector distancia o Bellman-Ford [Bel57][FF62], el cual ha sido mejorado para evitar bucles. Sirve para encontrar a partir del algoritmo de vector distancia aquella ruta que proporciona la trayectoria más corta posible hacia un destino [IC03] Cada nodo dentro de la red *ad hoc* mantiene una tabla de encaminamiento con la información siguiente para cada destino :

- Dirección IP destino
- Número de secuencia del destino
- Próximo salto hacia el destino (dirección IP)

- Coste de la ruta hacia el destino (en número de saltos)
- Tiempo de vida: Sirve para eliminar rutas antiguas

Cada nodo envía periódicamente en modo broadcast su tabla actualizada a sus vecinos [Per01]:

- Cada nodo añade su número de secuencia cuando envía su tabla de encaminamiento
- Cuando los demás nodos reciben dicha información actualizan sus propias tablas de encaminamiento

En caso de un cambio brusco de la topología (rotura de rutas o aparición de nuevas rutas) las tablas deben ser actualizadas. En este caso, la información de actualización que viaja en los mensajes de encaminamiento es la siguiente:

- Dirección IP destino
- Coste de la ruta hacia el destino (en número de saltos)
- Número de secuencia del destino

Los nodos utilizan los números de secuencia del destino para clasificar las rutas según su antigüedad, este criterio se aplica solo a rutas hacia un mismo destinatario.Un nodo incrementa su número de secuencia cuando se produce un cambio a nivel local en la topología de sus vecinos (se crea o elimina un enlace). Aquella ruta hacia un destino que tenga asociada el número de secuencia del destino más reciente (el mayor) será la que se considerará válida. En el caso de que existan dos rutas con el mismo número de secuencia del destino hacia un destino, prevalecerá aquella cuyo número de saltos sea menor.

Se usan dos tipos de paquetes de actualización de rutas :

- Full dump

  Transportan toda la información contenida en la tabla de encaminamiento de un nodo.

- Incremental

  Este tipo de paquetes transporta únicamente la información que ha variado en la tabla de encaminamiento de un nodo

Sin embargo, a pesar de la introducción de los paquetes 'incremental', DSDV continúa teniendo problemas debido al exceso de señalización requerida, que crece de acuerdo con $O(N^2)$, siendo N el número de nodos de la red. Por esta razón, el protocolo no es escalable.

Para evitar que un nodo anuncie un cambio de ruta para encaminar paquetes cuando existe una ruta mejor pero que todavía se está descubriendo, se precisa que cada nodo espere un tiempo fijo antes de anunciar una nueva ruta hacia el destino con un coste menor. El tiempo fijo a esperar será calculado como el tiempo medio que se tarda en conseguir todos los mensajes de actualización de una ruta. Así, un nodo recibirá todos los mensajes con información acerca de cambios en la ruta antes de propagar cualquiera de dichos cambios. De esta manera, los nodos vecinos reducen la utilización del ancho de banda y el consumo de potencia.

El funcionamiento del protocolo DSDV se ilustra con unas imágenes y tablas que son propiedad de Mari Carmen Domingo Aladrén.

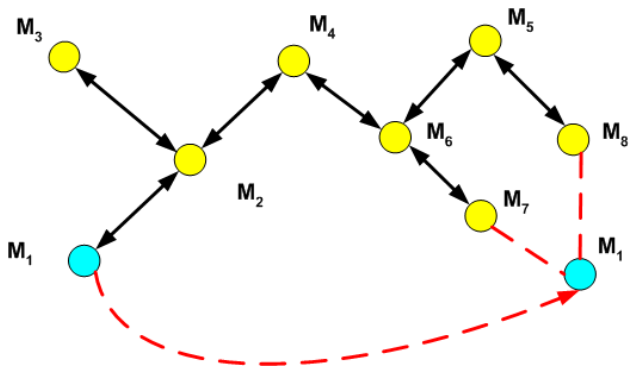


Figura 2.2: Red *ad hoc* donde existe movilidad

La información de la tabla de encaminamiento que se enviará en el mensaje de encaminamiento de actualización se ilustra en la Figura 2.4.

Destino	*Próximo salto*	*Coste*	*Número de secuencia*	*Tiempo de instalación*
M_1	M_2	2	S406_M_1	T001_M4
M_2	M_2	1	S128_M_2	T001_M4
M_3	M_2	2	S564_M_3	T001_M4
M_4	M_4	0	S710_M_4	T001_M4
M_5	M_6	2	S392_M_5	T002_M4
M_6	M_6	1	S076_M_6	T001_M4
M_7	M_6	2	S128_M_7	T002_M4
M_8	M_6	3	S050_M_8	T002_M4

Figura 2.3: Tabla de encaminamiento para el nodo M_4

Destino	*Coste*	*Número de secuencia*
M_1	2	S406_M_1
M_2	1	S128_M_2
M_3	2	S564_M_3
M_4	0	S710_M_4
M_5	2	S392_M_5
M_6	1	S076_M_6
M_7	2	S128_M_7
M_8	3	S050_M_8

Figura 2.4: Tabla de encaminamiento del nodo M_4 de actualización

Si se produce un cambio en la topología de la red y el nodo M_1 se mueve hasta alcanzar la nueva posición que se ilustra en la Fig. 2.2, la tabla de encaminamiento para el nodo M_4 y el mensaje de encaminamiento de actualización serán los que se muestran en la Tabla 2.3 y la Tabla 2.4 respectivamente.

Solamente existe una entrada de encaminamiento nueva para el nodo destino M_1, pero durante este intervalo de tiempo sí que se han recibido nuevos números de secuencia de destinos asociados a otras entradas de la tabla. El nodo M_4 deberá enviar un mensaje de encaminamiento 'incremental' para informar a sus

vecinos de la variación en la entrada del nodo destino M_1 para que puedan enterarse sin necesidad de esperar a que se envíe el siguiente paquete 'full dump' con la actualización completa de la tabla.

También se incluyen las variaciones de los números de secuencia del resto de entradas, con lo que al final, como ha variado cada entrada de la tabla, es como si se hubiera enviado toda la tabla completa.

Destino	*Próximo salto*	*Coste*	*Número de secuencia*	*Tiempo de instalación*
M_1	M_6	3	S516_M_1	T810_M4
M_2	M_2	1	S238_M_2	T001_M4
M_3	M_2	2	S674_M_3	T001_M4
M_4	M_4	0	S820_M_4	T001_M4
M_5	M_6	2	S502_M_5	T002_M4
M_6	M_6	1	S186_M_6	T001_M4
M_7	M_6	2	S238_M_7	T002_M4
M_8	M_6	3	S160_M_8	T002_M4

Figura 2.5: Tabla de encaminamiento para el nodo M_4 (actualizada).

Destino	*Coste*	*Número de secuencia*
M_4	0	S820_M_4
M_1	3	S516_M_1
M_2	1	S238_M_2
M_3	2	S674_M_3
M_5	2	S502_M_5
M_6	1	S186_M_6
M_7	2	S238_M_7
M_8	3	S160_M_8

Figura 2.6: Tabla de encaminamiento del nodo M_4 de actualización enviada en el mensaje de encaminamiento incremental.

2.3. Protocolos de encaminamiento reactivos

Los protocolos de encaminamiento reactivos [AWD04] o bajo demanda son aquellos en los cuales los algoritmos de encaminamiento crearán rutas únicamente en el caso de que un nodo fuente necesite enviar información a un nodo destino. Así, se utilizan los recursos de red tales como la energía o el ancho de banda de forma más eficiente que en los protocolos de encaminamiento proactivos, aunque, por otro lado, aumenta el retardo de Descubrimiento de Ruta.

Durante el proceso de Descubrimiento de Ruta, si un nodo fuente desconoce una ruta hacia el destino envía un mensaje de petición de ruta (Route Request) en modo broadcast para obtenerla y recibirá un mensaje de respuesta de ruta (Route Reply), que contendrá la ruta buscada.

Si los enlaces son bidireccionales y por tanto el mensaje que contiene la ruta buscada (Route Reply) puede utilizar la misma ruta que el mensaje de petición de ruta o Route Request, entonces la señalización introducida en el proceso de Descubrimiento de Ruta crece en el peor caso con $O(M+N)$, donde N representa el número de nodos en la red y M el número de nodos en el camino de vuelta con la respuesta de ruta; para enlaces unidireccionales la señalización introducida crece con $O(2N)$.

Los protocolos de encaminamiento reactivos se pueden dividir en dos grupos:

- **Basados en la fuente (source-based)**

  Cada paquete de datos transporta en su cabecera la ruta completa de la fuente al destino, es decir, las direcciones de cada nodo intermedio a lo largo de la ruta desde la fuente al destino. Cada nodo intermedio consultará la cabecera del paquete que le llega para saber por dónde debe reenviarlo. Por lo tanto, ya no hace falta que cada nodo intermedio mantenga una tabla de encaminamiento con información actualizada continuamente mediante el envío periódico de mensajes de encaminamiento, como sucedía con los protocolos proactivos.

  Como contrapartida, en las redes *ad hoc* grandes, la probabilidad de que un enlace se rompa crece con el número de nodos y, además, al aumentar con mayor probabilidad el número de nodos intermedios Protocolos de encaminamiento en redes *ad hoc* aisladas a lo largo de la ruta, crece también la cabecera del paquete. En consecuencia, los protocolos de encaminamiento basados en la fuente no son recomendables en redes de gran tamaño con muchos saltos y alta movilidad debido a sus dificultades para escalar.

- **Salto a salto (hop-by-hop) o punto a punto (point-to-point)** El paquete lleva en su cabecera únicamente la dirección del destino y la dirección del próximo salto, de forma que cada nodo intermedio a lo largo de la ruta en dirección al destino deberá consultar su tabla de encaminamiento para decidir por donde debe reenviar el paquete.

  La ventaja de utilizar este tipo de encaminamiento es que cada nodo intermedio actualiza su tabla de encaminamiento continua e independientemente, de forma que cuando le llega un paquete decide encaminarlo según el estado actual de la red y así las rutas pueden adaptarse más fácilmente a la topología dinámica de este tipo de redes. La desventaja de utilizar este protocolo es la necesidad de que cada nodo intermedio a lo largo de la ruta mantenga su tabla de encaminamiento permanentemente actualizada mediante el intercambio periódico de mensajes de actualización con sus nodos vecinos.

Seguidamente se presentan los dos protocolos de encaminamiento bajo demanda más conocidos y aceptados por toda la comunidad científica debido a sus particulares méritos.

- Dynamic Source Routing (DSR)
  Está basado en la fuente.
- Ad-hoc On-Demand Distance Vector (AODV)
  Funciona salto a salto.

2.3.1. Dynamic Source Routing (DSR)

DSR [JM96] es un protocolo de encaminamiento reactivo basado en la fuente. Es un protocolo de encaminamiento bajo demanda, lo cual significa que se crearán rutas únicamente en el caso de que un nodo fuente necesite enviar datos a un nodo destino.

El protocolo esta compuesto de dos características :

- Route Discovery
- Route Maintenance

El Route Discovery es el mecanismo mediante el cual un nodo origen busca la ruta disponible para alcanzar un nodo destino. Mientras que el Route Maintenance es el mecanismo mediante el cual un nodo es capaz de detectar si la topología de la red ha cambiado y por tanto su ruta deja de ser valida. Un nodo puede

almacenar múltiples rutas en su cache, ademas ambos procedimiento de búsqueda y mantenimiento están designados para trabajar en condiciones de unidireccionalidad de los enlaces.

Todos los aspectos del protocolo operan bajo demanda, permitiendo que el flujo de paquetes de enrutamiento se escale automáticamente con las necesidades de la red.

Cada paquete de datos incluye en su cabecera una lista ordenada de nodos por los que el paquete debe pasar, impidiendo así la aparición de ciclos en las rutas y no necesitando una actualizando por parte de los nodos que atraviesa. A su vez los nodos por los que viaja el paquete pueden almacenar en su cache la ruta del mensaje.

Asunciones

- El protocolo asume que todos los nodos de la red pueden transmitir paquetes.
- El protocolo asume que el diámetro de la red será casi siempre pequeño (de 10 a 5 saltos) pero mayor que 1 salto
- Los nodos que forman la red tiene la habilidad de detectar y descartar mensajes corruptos
- La movilidad de los nodos será moderada respecto a la velocidad de transmisión de la red
- Los nodos son capaces de activar el modo promiscuo en su interfaz de red
- Cada nodo selecciona una dirección IP única mediante la cual será conocido en la red.

Descubrimiento de Rutas Cuando un nodo necesita manda un paquete a otro nodo destino y no dispone de una ruta para alcanzarlo, origina un *'Route Request'*.

Cada *'Route Request'* identifica al nodo originario de la petición y posee un identificador único, poseen una lista de los nodos que ha ido atravesando. EL nodo origine inicializa la lista como vacía y los transmite a todos sus vecinos. Cuando un nodo recibe un *'Route Request'* y es el destino del Route Discovery, entonces emite un *'Route Reply'* con un copia de la lista acumulada en el *'Route Request'*. Cuando en el nodo origen recibe un *'Route Reply'* almacena la ruta en su cache.

Si un nodo recibe un *'Route Request'* inmediatamente de haber recibido otro proveniente del mismo origen y con el mismo identificador, entonces lo descarta. También se puede producir un descarte si el nodo que recibe el *'Route Request'* se encuentra ya en la lista. Por el contrario si no se dan las situaciones anteriores,

un nodo que reciba un *'Route Request'* añadirá su propia dirección a la lista y retransmitirá la petición a todos sus vecinos.

En la respuesta (*'Route Reply'*) el nodo generador de la respuesta mirara en su cache para ver si tiene alguna ruta para alcanzar al nodo origen, de otro modo generar otro *'Route Request'* hacia el nodo origen del primer *'Route Request'* pero incluyendo el *'Route Reply'* para evitar recursiones infinitas entre ambos nodos. Otra opción es utilizar la ruta recién descubierta pero esto solo es recomendable en las redes bidireccionales.

Cuando un nodo inicia un Route Discovery almacena en su bandeja de salida los mensajes pendientes del descubrimiento de ruta, si pasado un tiempo no se ha descubierto todavía una ruta puede enviar otro route discovery incrementando el temporizador de una manera exponencial, evitando así colaras la red con peticiones de ruta.

Almacenamiento en cache de la escucha de información de rutas Un nodo que retransmita un paquete o que escuche a otro nodo cercano la retransmisión de un paquete puede incluir esa información en su cache de rutas, existe el riesgo almacenar en cache rutas imposibles si los caminos son unidireccionales, pues en la escucha el nodo supone una dirección entre el y el vecino al que esta escuchando que puede no ser correcta.

Contestando a *'Route Request'* mediante rutas en cache Un nodo recibe un *'Route Request'* del cual el no es el destino pero tiene en su cache una ruta hasta el destino, puede emitir un *'Route Reply'* concatenando la ruta del *'Route Request'* con su información en cache.

Se debe comprobar que la ruta resultante de la concatenación es valida, se entiende por valida :

- No contiene duplicados
- Contiene al nodo actual, el que realiza la composición.

Prevención de tormentas *'Route Reply'* El uso de respuesta basadas en caches puede producir un fenómeno de tormenta, que resulta de que todos los vecinos de un nodo contesten a la vez basandose en su cache de rutas produciendo colisiones en la red. Para evitar esta situaciones se utiliza un temporizador y el modo promiscuo para escuchar si un nodo con mejor ruta (menor numero de saltos) esta transmitiendo el *'Route Reply'*. EL temporizador de cada nodo esta basado en la longitud del camino a responder, de dicha manera se escalonan los *'Route Reply'* en medida de la longitud de su ruta

Limite de saltos para el *'Route Request'* Se trata de limitar el numero de saltos disponibles para un ROUTE REQUESST mediante un contador que se decrementa en cada salto, si el contador llega a cero antes de encontrar el destino, se retransmite un 'nopropagation' ROUTE DISCOVERY. Si no se recibe ningún *'Route Reply'* entonces se envía un *'Route Request'* sin limite.

Mantenimiento de Rutas Todo nodo que recibe un paquete y lo retransmite envía una confirmación al nodo anterior cuando el nodo siguiente ha recibido efectivamente el paquete, utilizando instrumentos provistos por capas inferiores (ACK) si están disponible y si no implementando en el propio DSR.

Los paquetes a retransmitir se almacenan en un buffer en espera de la confirmación, si la confirmación excede de un tiempo prefijado el paquete se vuelve a retransmitir.

Si un paquete es transmitido el máximo numero de veces permitido y no ha recibido confirmación, ese nodo emite un *'Route Error'* al originario del mensaje, indicando que parte de la route esta rota. El nodo origen retransmitirá el paquete por otra ruta si dispone de ella en la cache o bien generara un Route Discovery.

Salvamento de paquetes Cuando un paquete recibe un *'Route Error'*, se puede salvar el contenido del paquete y enviarlo por una ruta alternativa que exista en el nodo actual, en vez de descartar el paquete entero. Si se toma esta decisión el paquete se marca como salvado para evitar ciclos en la transmisión de paquetes salvados varias veces.
Se aplican las mismas reglas de composición de rutas descritas anteriormente.

Optimizaron de rutas Las rutas pueden ser automáticamente optimizadas cuando algunos nodos intermedios no son necesarios. El mecanismo para obtener esta información se basa en que si un nodo es capaz de escuchar a otro transmitir un paquete que le incluye a el en un salto posterior, puede actualizar su cache de rutas eliminando los nodos no necesarios y emitiendo un *'Route Reply'* informando al originario del mensaje de la nueva ruta.

Propagación de errores Cuando un nodo recibe un *'Route Error'*, este nodo propaga dicho *'Route Error'* a sus vecinos en el próximo *'Route Request'* de dicho modo no se generan nuevas rutas que contengan el mismo nodo problemático. Se puede ampliar a enviar un *'Route Error'* en el camino inverso del paquete que ha generado el error.

Almacenamiento de errores en cache Los nodos pueden almacenar en cache información sobre enlaces rotos de tal manera que no se generan *'Route Reply'* con rutas potencialmente problemáticas. La información negativa lleva un temporizador que una vez expirado elimina dicha información de la cache.

El funcionamiento del protocolo DSR se ilustra mediante unas imágenes que son propiedad del profesor Nitin Vaidya.

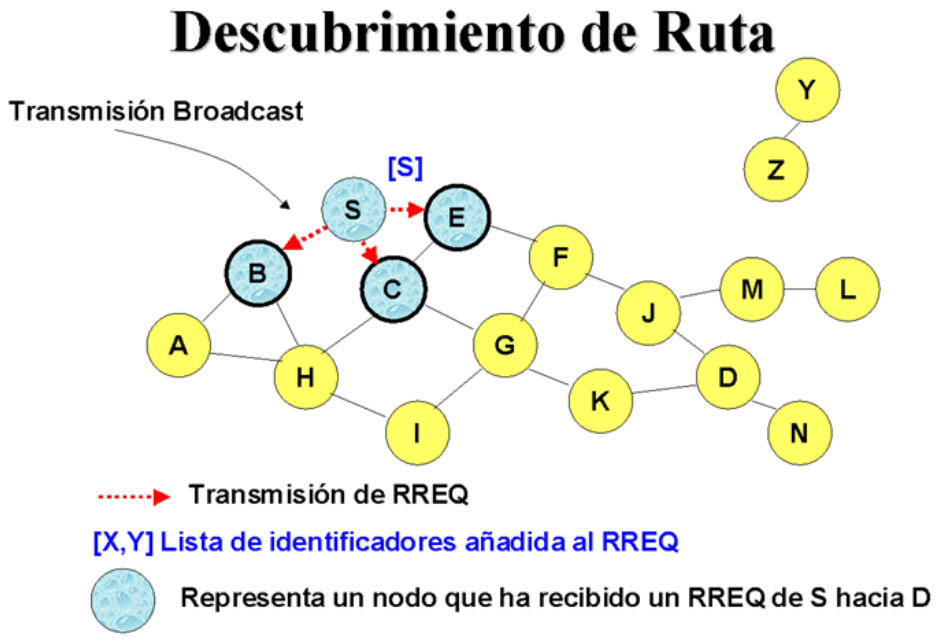


Figura 2.7: Descubrimiento de Ruta en una red *ad hoc* que utiliza el protocolo de encaminamiento DSR para enviar datos desde un nodo origen S hasta un nodo destino D (1).

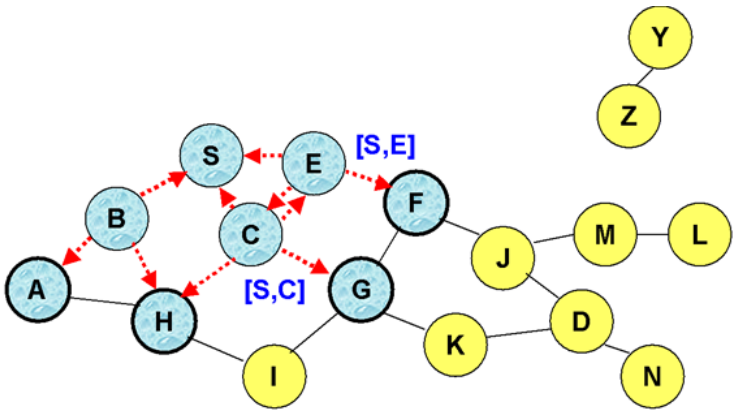
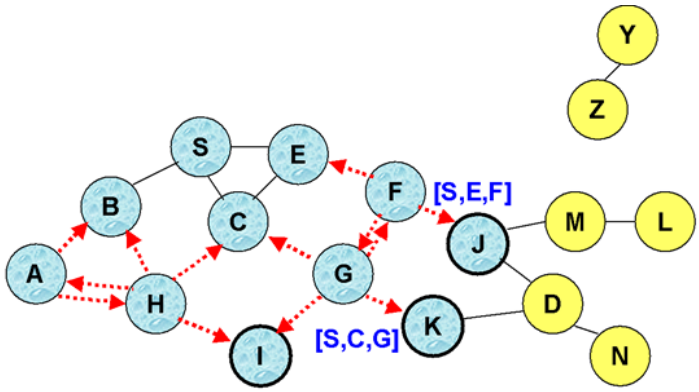


Figura 2.8: Descubrimiento de Ruta (2).



• El nodo C recibe los RREQ de G y H, pero no hace una retransmisión porque ya ha sido hecha una vez.

Figura 2.9: Descubrimiento de Ruta (3).

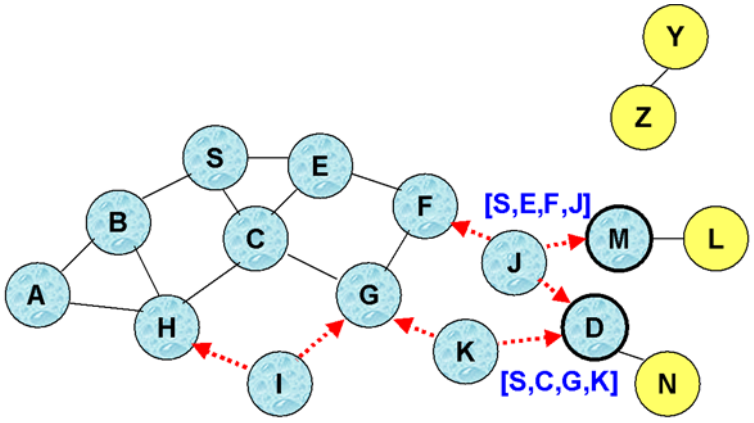


Figura 2.10: Descubrimiento de Ruta (3).

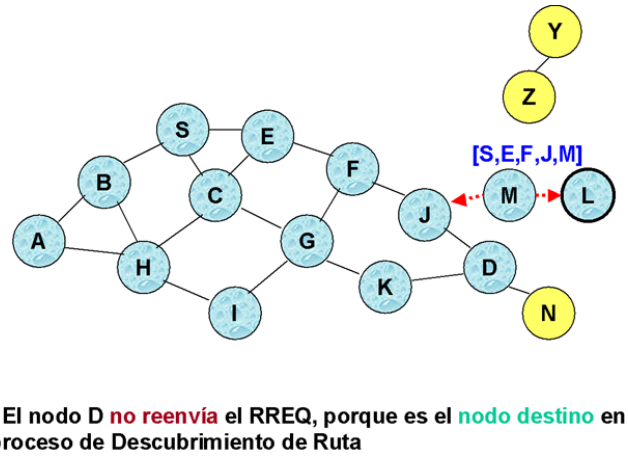


Figura 2.11: Descubrimiento de Ruta (4).

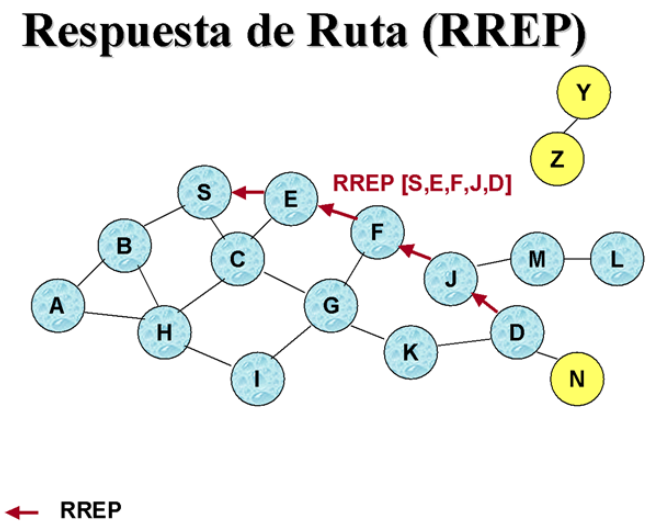


Figura 2.12: Descubrimiento de Ruta (5).

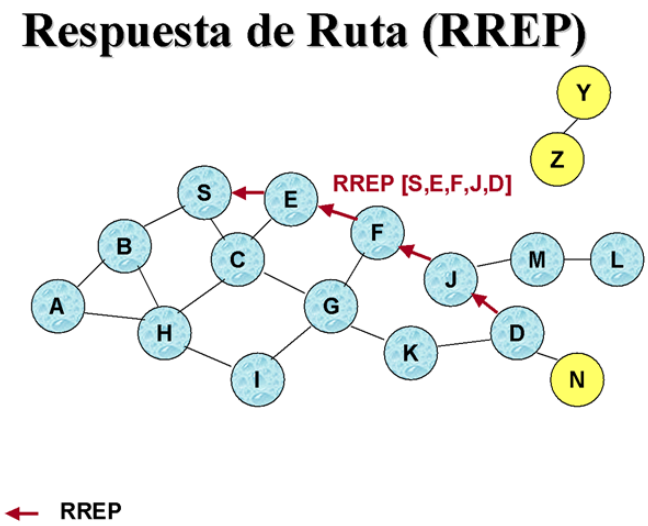


Figura 2.13: Descubrimiento de Ruta (6).

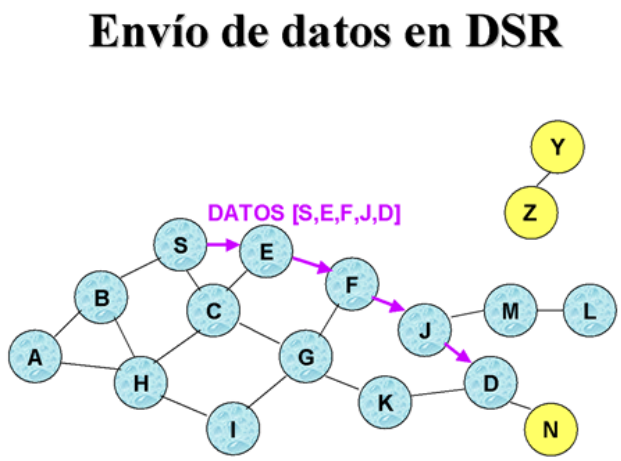


Figura 2.14: Descubrimiento de Ruta (7).

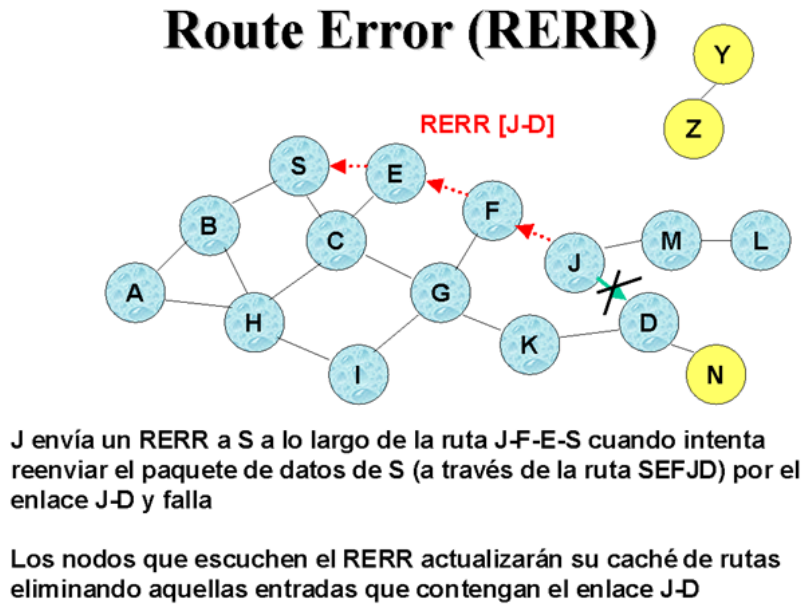


Figura 2.15: Mantenimiento de Ruta en una red *ad hoc* que utiliza el protocolo de encaminamiento DSR para enviar datos desde un nodo origen S hasta un nodo destino D.

2.3.2. *ad hoc* On-Demand Distance Vector (AODV)

AODV [Per97] es un protocolo de encaminamiento reactivo salto a salto.

Es un protocolo de encaminamiento que establece rutas bajo demanda. Esto significa que en AODV no se mantienen permanentemente actualizadas las rutas de cada nodo a cada nodo de la red, sino que se descubren y mantienen solamente cuando son necesarias.

Las rutas son descubiertas durante el proceso de Descubrimiento de Ruta, una fase en la cual un nodo fuente busca una ruta hacia un nodo destino para poder enviarle información. Este proceso termina cuando al nodo fuente se le retorna la ruta buscada.

AODV presenta las siguientes características:

- AODV esta basado en un encaminamiento mediante vector de distancia.
- El mantinimiento de rutas se realiza unicamente sobre aquellas en las que exista una comunicación activa con los nodos destino de dichas rutas. Esto ayuda a reducir la sobrecarga en la red.
- La tabla de rutas mantiene un tiempo de expiración para las entradas disponibles en la tabla.
- Las entradas son asociadas con un numero de secuencia para evitar la formación de bucles que a su vez sirve para poder distinguir la información antigua de la moderna.
- El correcto funcionamiento de AODV depende de que cada nodo mantenga actualizado su propio número de secuencia.

Con el fin de prevenir bucles cada nodo mantiene un número de secuencia (también llamado número de secuencia del destino) que sirve para evaluar la vigencia de la información de encaminamiento asociada y aumenta en uno cada vez que un nodo envía una nueva petición de ruta, RREQ (Route Request). Si un nodo recibe un RREQ destinado a él mismo antes de generar el mensaje de petición de respuesta, RREP (Route Reply) debe actualizar su número de secuencia $NumSeq_D$ al valor máximo entre su número de secuencia actual $NumSeq_{D\_actual}$ y el número de secuencia del destino que se halla contenido en el RREQ (NumSeq.RREQ) más uno:

$$NumSeq_d = Max(NumSeq_{D\_actual}, RREQ.NumSqe + 1)$$

Los números de secuencia sirven para que siempre se seleccione la ruta más reciente hacia un destino. Si se da el caso de que un nodo fuente o nodo intermedio recibe dos rutas que contienen el mismo número

de secuencia del destino, escogerá una ruta de las dos utilizando una métrica como puede ser aquella que contenga el menor número de saltos hacia el destino.

AODV utiliza para su correcto funcionamiento unas tablas de encaminamiento donde se almacena:

- La dirección IP del destino
- La dirección IP del próximo salto
- El número de secuencia del destino
- Tiempo de vida (tiempo requerido para eliminar la ruta)
- Contador de saltos (número de saltos para alcanzar el destino)

Si se sobrepasa el tiempo de expiración de la entrada esta se borra de la tabla. Los tiempos de expiración son renovados si se recibe un mensaje "Hello' o si la entrada es utilizada.

Descubrimiento de Ruta (Route Discovery)

- Cuando un nodo fuente desee enviar paquetes a otro nodo destino, lo primero que debe hacer es consultar su tabla de encaminamiento para comprobar si ya existe una ruta actualizada hacia dicho destino. En caso de que sí que exista, el nodo fuente la usará para enviar los paquetes al próximo salto en dirección hacia el destino. En caso de que no exista, el nodo iniciará un procedimiento de Descubrimiento de Ruta enviando un paquete denominado petición de ruta, RREQ (Route Request) en modo broadcast. Dicho paquete contiene la siguiente información :
  - Dirección IP nodo origen
  - Número de secuencia origen
  - Dirección IP nodo destino
  - Número de secuencia destino
  - ID (identificador broadcast) del RREQ
  - Contador del número de saltos
- Cuando un nodo transmite un mensaje RREQ, almacenará durante un tiempo el identificador de dicho mensaje, así como su nodo origen, para evitar retransmitirlo.

- Un nodo que recibe un RREQ debe crear o actualizar una ruta hacia el nodo vecino que lo ha retransmitido.
- A continuación comprueba si es un mensaje duplicado, en cuyo caso no realiza ningún procesamiento mas.
- Si no es duplicado el nodo crea o actualiza una ruta inversa hacia el origen del mensaje RREQ. Si ya exista dicha ruta se tiene que actualizar su numero de secuencia con el del mensaje RREQ si este ultimo es mayor. El salto siguiente será el vecino del que se ha recibido el mensaje. Gracias a esta ruta se podrá retransmitir la respuesta a la petición de ruta.
- Si el nodo que ha recibido el RREQ no esta en condiciones de generar un RREP deberá retransmitir el RREQ, actualizando antes el numero de secuencia para el destino del mensaje con el que el tiene guardado si este ultimo es mayor.
- Cuando un nodo recibe el mensaje RREQ generara una respuesta RREP si es el destino o si tiene información actualizada para llegar a el (el que sea actualizada o no lo dirán los números de secuencia).
- El RREP ya no será inundado, sino que se enviara en unicast al nodo que origino el proceso de búsqueda de ruta. El mensaje será encaminado correctamente gracias a las rutas inversas que fueron creadas conforme se inundaba el RREQ.
- Si el nodo que genera el RREP es el destino deberá incrementar en una unidad su numero de secuencia si ese fuera el valor anunciado por el RREQ, justo antes de enviar el RREP.
- Si fuera un nodo intermedio el que lance el RREP deberá poner en el el numero de secuencia que posee para el destino.
- Los nodos que procesan un RREP crean o actualizan la ruta hacia el vecino que se lo ha enviado. Además crean o actualizan la ruta directa hacia el destino, es decir, hacia el emisor del RREP. De esta manera podrán enrutar los paquetes que vayan dirigidos a el.

Cuando un nodo fuente recibe un RREP puede comenzar a usar la ruta contenida para encaminar paquetes de datos. Si se da el caso de que un nodo fuente recibe múltiples RREPs seleccionará aquella ruta con el número de secuencia del destino mayor y el número de saltos hacia ese destino menor.

Cada nodo monitoriza el estado de los enlaces que le comunica con el próximo salto a través de una ruta que está siendo utilizada en ese momento (ruta activa). En el caso de que detecte la rotura del enlace, este nodo invalida en su tabla de encaminamiento todas aquellas entradas hacia destinos que no están en estos momentos disponibles debido a la rotura del enlace.

Además, el nodo enviará un paquete de error de ruta, RERR (Route Error) hacia el nodo fuente para informarle acerca de este suceso. En el RERR se informa acerca de aquellos destinos que ya no pueden alcanzarse.

Cuando un nodo recibe un paquete de RERR comprueba que efectivamente el nodo que envió dicho mensaje sea su próximo salto hacia alguno de los destinos y si esa así el nodo invalida estas entradas de ruta en su tabla de encaminamiento y reenvía el paquete de RERR hacia la fuente. Cuando finalmente el paquete de RERR llega a la fuente, ésta puede decidir iniciar un nuevo proceso de Descubrimiento de Ruta si lo considera necesario.

Los nodos que forman parte de rutas activas pueden enviar información de conectividad a sus vecinos mediante mensajes HELLO. Estos serán generados por un nodo cuando en un período determinado de tiempo no ha transmitido ningún mensaje broadcast.

Los mensajes HELLO son realmente RREPs con un TTL de un salto para que solo sean recibidos por el vecindario. Cuando un vecino procesa un HELLO debe crear o actualizar la entrada de la tabla de rutas cuyo destino es el origen del mensaje.

Si algún vecino recibe un HELLO del nodo y tras un período de espera no recibe ningún otro mensaje de el, dará el enlace por perdido.

El funcionamiento del protocolo AODV se ilustra mediante unas imágenes que son propiedad del profesor Nitin Vaidya.

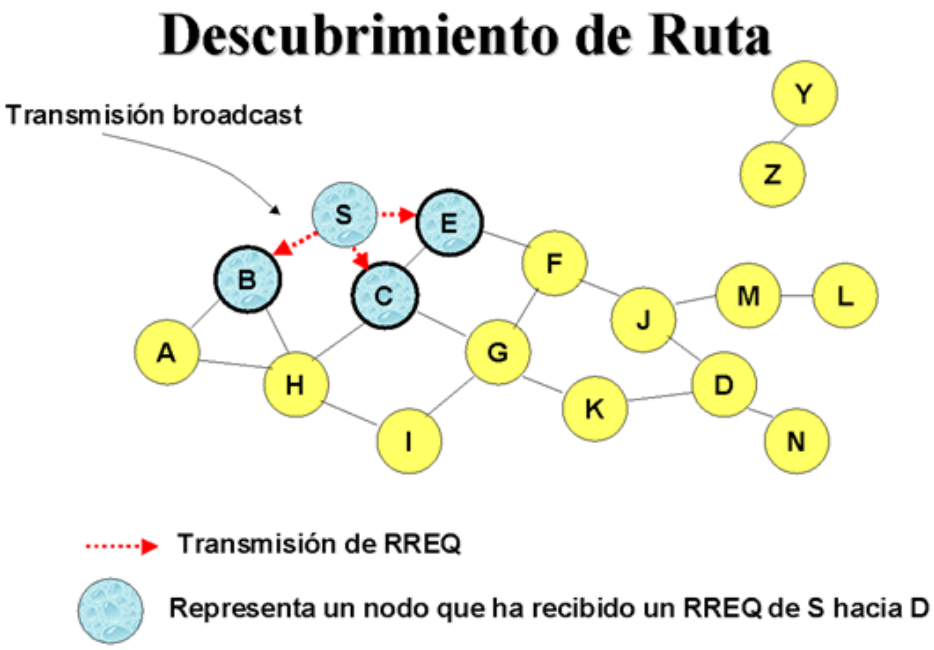


Figura 2.16: Descubrimiento de Ruta en una red *ad hoc* que utiliza el protocolo de encaminamiento AODV para enviar datos desde un nodo origen S hasta un nodo destino D (1).

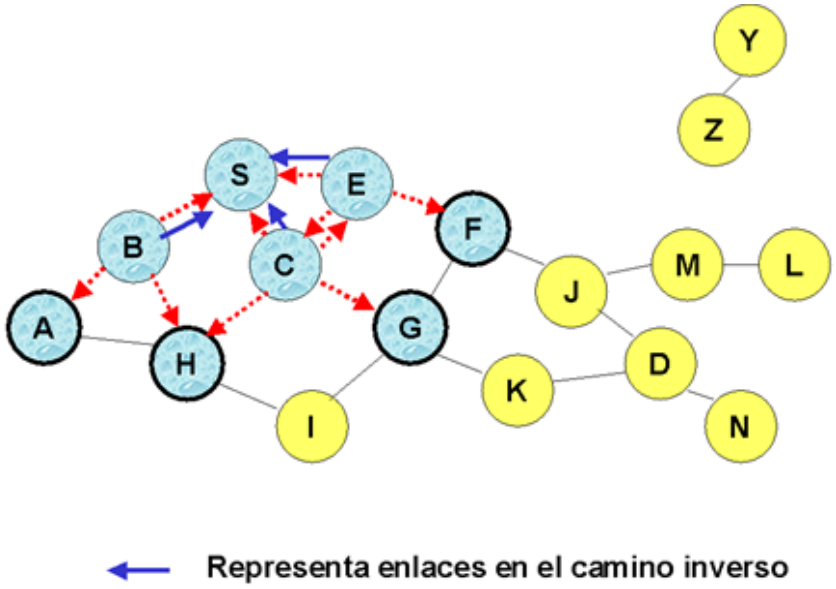


Figura 2.17: Descubrimiento de Ruta (2).

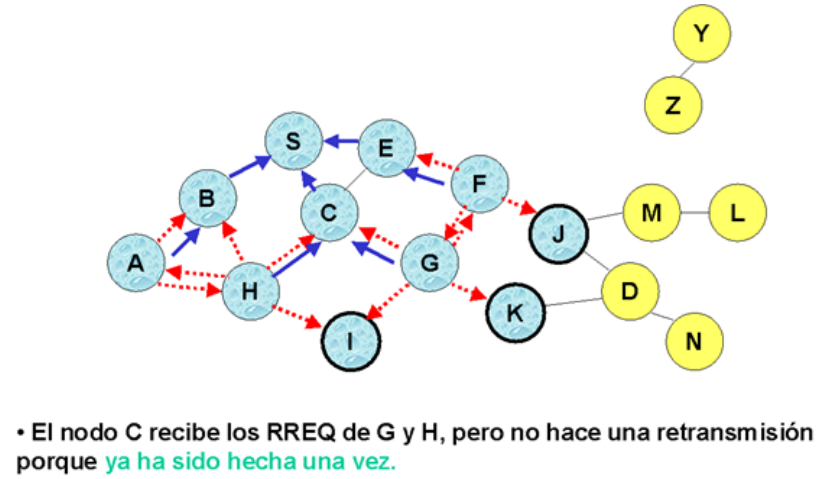


Figura 2.18: Descubrimiento de Ruta (3).

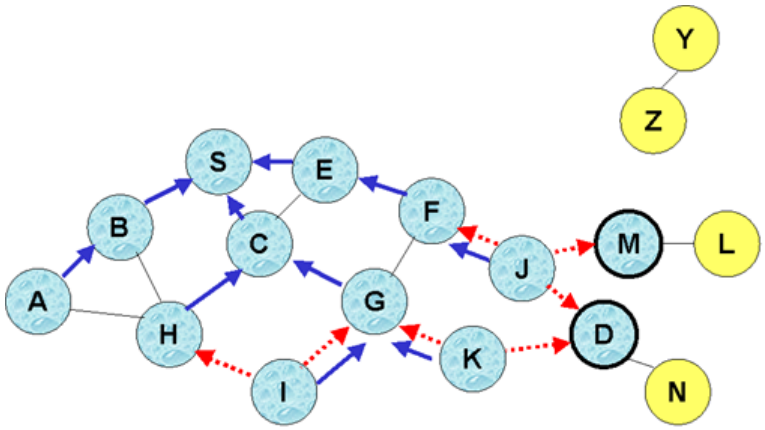


Figura 2.19: Descubrimiento de Ruta (4).

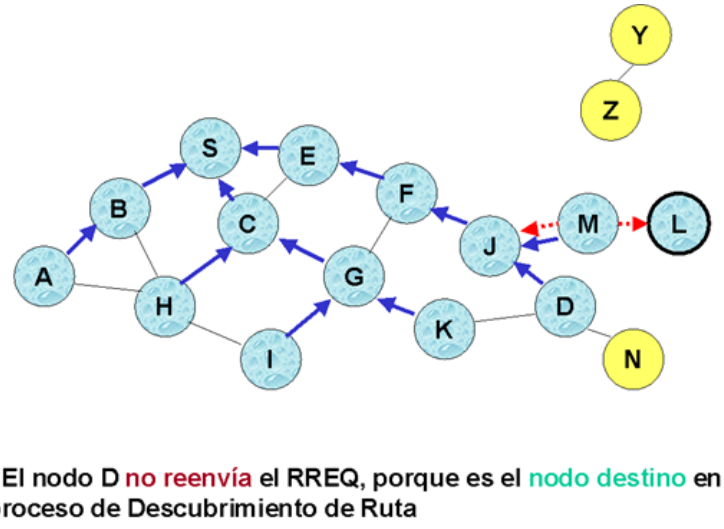


Figura 2.20: Descubrimiento de Ruta (5).

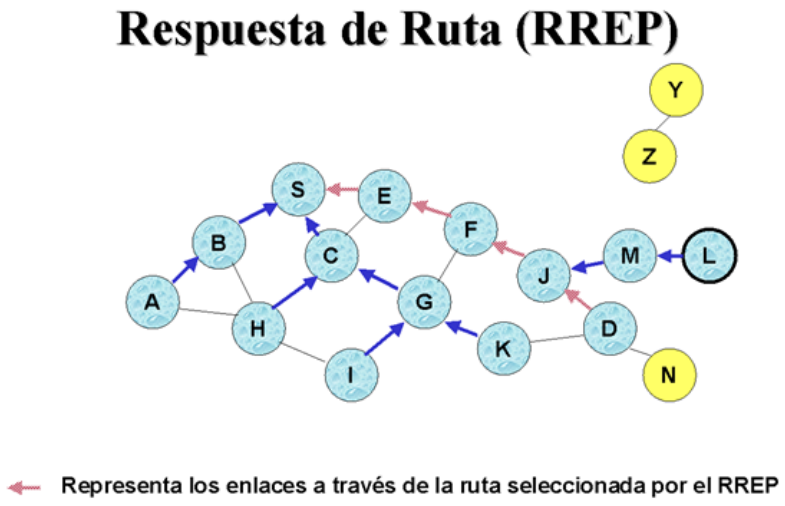


Figura 2.21: Descubrimiento de Ruta (6).

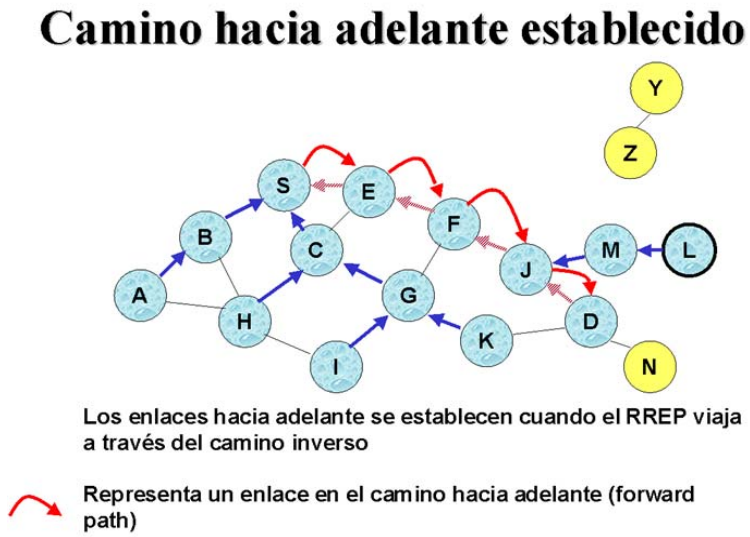


Figura 2.22: Descubrimiento de Ruta (7).

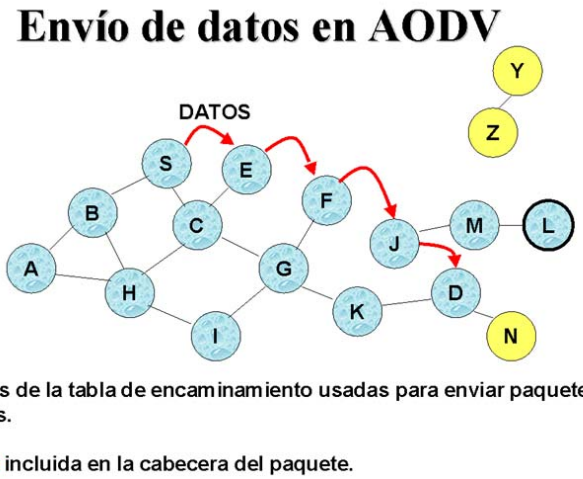


Figura 2.23: Descubrimiento de Ruta (8).

2.4. Desacuerdo con los protocolos de encaminamiento actuales

Encaminamiento Proactivo : la idea subyacente bajo este tipo de protocolos es realizar el calculo de rutas mientras que no haya peticiones de transmisión, de tal manera que cuando se solicite una transmisión la ruta este disponible. Este planteamiento es bastante ineficiente imaginemos el siguiente escenario : una red *ad hoc* con una topología variable cada minuto pero con una necesidad de transmisiones realmente baja, pongamos una cada 10 minutos. Este planteamiento consumirá recursos innecesarios intentando tener la ruta disponible, mediante una captura de la topología de la red, los 9 minutos anteriores a que se produzca una transmisión y la información recogida sea útil. El error consiste en orientar la reacción del protocolo a la topología, lo mas importante de una red son las transmisiones, el factor principal que gobierne el comportamiento de la red y del protocolo de encaminamiento debe ser la necesidad de transmisión. En caso contrario es absurdo pero podemos llegar a una situación donde hemos consumido todos los recursos de la red sin haber realizado una sola transmisión, es un caso excepcional, pero con este planteamiento podría darse.

Mantenimiento de Rutas : la tarea de mantenimiento de rutas en nuestra opinión es innecesaria, esto se explica en nuestra propuesta de protocolo pero como adelanto diremos que es necesario eliminar este componente de los algoritmos de encaminamiento para reducir la complejidad de estos algoritmos, eliminar sobrecarga de paquetes en la red y destinar los recursos de este proceso a un mejor cálculo de rutas. Para eliminar el mantenimiento de rutas hay que eliminar el concepto de ruta, tradicionalmente una ruta es el

camino que seguirá un paquete para alcanzar desde un nodo origen de la transmisión al nodo destino de dicha transmisión. Nosotros proponemos otra concepción : una ruta es el camino que ha seguido un paquete para alcanzar desde nodo origen a un nodo destino, la ruta es el resultado de la dinámica del paquete en la red (de su movimiento). De esta manera solo existen rutas cuando hay transmisión, la tarea de mantenerlas se elimina del sistema puesto que sin transmisión las rutas no tienen sentido.

Búsquedas en Anchura : las búsquedas en anchura introducen una limitación excesiva : no son escalables, con redes grandes producen una sobrecarga excesiva de paquetes y lo que es peor introducen un tiempo de espera entre búsqueda y búsqueda que limita de sobremanera la captura de la topología de la red. Es decir no se puede capturar eficientemente una topología dinámica mediante búsquedas en anchura por el volumen de nodos involucrados en dicha captura (toda la red). Las búsquedas deben realizarse por profundidad, de esta manera podemos realizar búsquedas en paralelo con una menor carga en la red y muestrear la topología de la red mas rápidamente, de tal manera que dispondremos de un mejor modelo topologico de la red.

Capítulo 3

QoS en redes *ad hoc.*

Las redes *ad hoc* son una tecnología emergente que, creemos, serán de capital importancia en un futuro próximo. Es por ello, y porque aún se encuentra en etapas tempranas de desarrollo, que no se ha profundizado en el área de QoS.

3.1. Introducción

El término QoS se acuñó para describir la garantía de recibir un servicio fiable de transmisión a través de una red.

"The capability to control traffic-handling mechanisms in the network such that the network meets the service needs of certain applications and users subject to network policies" – ***Networking Quality of Service and Windows Operating System, Yoram Bernet, New Riders 2001.***

Cuando se desarrollaron las primeras redes de computadores estas carecían de la capacidad para proveer garantías de calidad de servicio (QoS) debido a la falta de potencia de los routers. Es por eso que simplemente se conseguía un nivel básico de QoS, denominado best effort. Actualmente existe gran necesidad de superar la calidad de servicio ofrecida por best effort en redes ad-hoc.

Las redes ad-hoc carecen de una infraestructura fija y sus nodos son a la vez routers, fuentes y destinos de la comunicación. Debido a la naturaleza dinámica de las redes *ad hoc*, así como a las limitaciones en ancho de banda y a los problemas derivados del medio de transmisión, resulta muy complicado dar buena calidad de servicio.

3.2. Parámetros de QoS

Hay varios factores que afectan a QoS. Entre estos factores los principales son: pérdida de paquetes, retrasos (o *delays*), variación del retardo, entrega fuera de orden, disponibilidad de la red y ancho de banda.

Pérdida de paquetes. Un *router* puede fallar al entregar un paquete a su destinatario, provocando importantes retrasos en la comunicación e, incluso, haciéndola imposible.

Retrasos (o *delays*). Puede que un paquete tarde mucho en alcanzar su destino debido a diversos factores como quedar atrapado en un *buffer*, coger una ruta alternativa para evitar congestiones, etc. Cierto tipo de comunicaciones pueden verse arruinadas como consecuencia de estos retrasos.

Variación del retardo. Este problema afecta sobretodo a las comunicaciones que requieren un flujo constante de información como las comunicaciones audio y video multimedia. Dos paquetes no tardan el mismo tiempo en llegar desde la fuente al destino de la comunicación, lo que puede hacer que al usar una aplicación de las ya citadas observemos "saltos" o información inconsistente. Este fenómeno es conocido también como *jitter*.

Disponibilidad de la red. La ruta entre la fuente y el destino de la comunicación puede no ser utilizable en un momento dado, provocando problemas de transmisión entre nodos.

Entrega fuera de orden. Cuando una colección de paquetes relacionados se dirige de una fuente a un destino puede que no todos sigan la misma ruta debido a saturaciones u otros problemas en la red, provocando que los *delays* de los paquetes sean diferentes y estos lleguen al destino en un orden distinto al que fueron enviados. Esto conlleva inconsistencias severas en la comunicación.

Ancho de banda. En cierto tipo de comunicaciones es fundamental tener un caudal mínimo de información, como por ejemplo en servicios o aplicaciones multimedia. Si no gozamos de este caudal es imposible llevar a cabo tal comunicación. Este factor es especialmente relevante en redes inalámbricas debido a la sobrecarga (*overheading*) que sufren algunos de los más populares protocolos de enrutado.

3.3. Mecanismos y modelos QoS en redes IP

Existen dos mecanismos distintos para obtener QoS en una red IP: **sobredimensionar la red y aplicar métodos para administrar los recursos de la red.** El primero de estos mecanismos es una solución costosa y nada escalable, por lo que no se trata de una buena solución para garantizar el QoS. En cuanto a la segunda opción, se basa en la gestión de los recursos de la red y, a pesar de requerir de un esfuerzo mayor que el mecanismo anterior, proporciona buenos resultados.

Dentro del segundo mecanismo, **aplicar métodos para administrar los recursos de la red**, podemos hablar de dos enfoques distintos. Por un lado, reservar recursos de red, es decir, los recursos de red son

asignación en función de las peticiones de QoS, y, por otro lado, un mecanismo de ordenación del tráfico basado en clasificar el tráfico de red y darle un tratamiento o no preferente en función de qué tipo de tráfico se trate.

Siguiendo la línea del segundo enfoque, tenemos dos modelos básicos:

- **IntServ (Integrated Services).** Se basa en la reserva de recursos necesarios para hacer viable una comunicación entre un nodo fuente y un destino. El usuario especifica qué recursos requiere la comunicación mediante una serie de parámetros QoS, los cuales se gestionan de forma individual. Este modelo necesita de un protocolo de reserva y gestión de los recursos, como por ejemplo RSVP (*Resources reSerVation Protocol*).
- **DiffServ (Differenciated Services).** Este modelo no requiere la reserva de recursos para un flujo de datos, sino que ofrece unas clases diferentes de servicios que soportan distintos tipos de aplicaciones. Hace uso del campo TOS (ipv4) y el TC (ipv6) para indicar qué tipo de servicio debe recibir cada uno.

IntServ es capaz de ofrecer garantías estrictas de QoS para cada flujo que gestione, pero no es muy escalable debido a que mantiene variables de estado por cada flujo de la red, mientras que DiffServ es fácilmente escalable, pero no facilita unas garantías estrictas ya que simplemente asegura un trato diferenciado por clases.

3.4. QoS en redes Ad-Hoc

Proporcionar QoS en redes *ad hoc* (o MANETS) se convierte en una tarea bastante complicada de acometer por una razón fundamental, para soportar QoS debemos tener información del estado de los enlaces, con todas las variables que esto conlleva (retardo, ancho de banda, coste, tasa de perdidas, etcétera). Debido a la propia naturaleza de la red (entrada y salida de nodos aleatorios, recursos limitados, movilidad, etc.), es muy difícil obtener y, por tanto, gestionar esta información.

Aprovisionar de QoS a una red *ad hoc*, conlleva el reto de definir un modelo adecuado que defina la arquitectura que nos permitirá ofrecer servicios que operen mejor que el actual "Best effort" actual en MANETS. Aquí habría que tener en cuenta. Las características propias de la naturaleza de las redes Ad-Hoc que se comentaron más arriba. Vamos a comentar por qué no podemos aplicar los dos modelos vistos anteriormente y a estudiar una propuesta llamada FQMM (Flexible QoS Model for MANETS) propuesta por Xiao, Chua y Lo, en un documento del año 2000.

- IntServ: Este modelo no es aplicable dado que las MANETS sufren una carestía fundamental de recursos. Hay varios factores específicos que imposibilitan su uso:
- Grandes requisitos de almacenamiento y "overhead" de procesamiento elevado en cada nodo.
- Los procesos de reserva y mantenimiento consumen recursos de la red ya que son procedimientos "out-

of-band" que se explicará más adelante. Por ahora basta saber que un proceso "out-of-band" utiliza paquetes distintos a los de datos o control, que consumen bastante del ya por si escaso ancho de banda que tienen las redes inalámbricas.

- Una parte fundamental de un modelo completo de QoS es la clasificación y planificación dentro de la red. Esto requiere una considerable cantidad de ancho de banda, que acabaría por consumir el ya escaso ancho que nos quedaría tras aplicar el punto anterior.
- DiffServ: Es un modelo "Lightweight", esto es, la información de estado viaja asociada a los paquetes normales del protocolo en lugar de haber paquetes propios. Esto alivia considerablemente el esfuerzo de los routers, no obstante este modelo no es aplicable tampoco dado que para dirigir un caudal de datos de flujo hay que tener claramente definido que routers son de envío de datos ("Ingress"), de paso de datos ("Core") o de recepcion de datos ("Egress") y esto en una red ad-hoc donde cada nodo es un router y un agente, es imposible de fijar. Además como DiffServ crea clases en función de las características de los servicios, para que esto fuera viable, debería haber un "ISP" que nos proporcionara servicios a nosotros como clientes, pero, dado que esto en una red ad-hoc no es así ya que no existe el concepto de proveedor de servicios y cliente, tampoco podemos hacer uso de la característica fundamental de este modelo.
- FQMM: Es el primer modelo de QoS propuesto explícitamente para MANETS. Su idea principal es combinar el conocimiento que tenemos de soluciones que funcionan para redes cableadas y aplicarlas a MANETS teniendo en cuenta sus características propias. Hace uso del control de flujo de IntServ y la diferenciación de servicios de DiffServ de manera que la prioridad más alta se maneje por control de flujo y el resto por clases. Esto se puede aplicar a sabiendas de que no todos los paquetes de la red están buscando la máxima prioridad. Para poder usar las facetas de DiffServ, FQMM también define los tres tipos de nodos que se definieron anteriormente, solo que en este caso la localización física no importa para nada a la hora de asignar un tipo a un nodo, sino que se le asigna el tipo en función de su estado actual (emitiendo, recibiendo, forwarding).

3.5. Señalización de QoS

La señalización se usa en redes con QoS para reservar y liberar recursos. Para ello es necesario que haya una transferencia fiable de señales entre routers y que todos los nodos de la red entiendan el significado de las señales.

La transferencia de señales se puede dividir en "in-band" y "out-of-band". "In-band" significa que todos los paquetes de control se encapsulan en paquetes de datos haciendo la señalización fácil y ligera de llevar a cabo para los nodos. "Out-of-band", en cambio, usa paquetes explícitos para llevar a cabo sus funciones.

Una MANet no puede tolerar un protocolo de señalización complejo por sus características propias, así que deseamos una aproximación que sea lo más "in-band" posible.
El primer protocolo de señalización realizado para MANETS es el llamado INSIGNIA, propuesto por Lee y Campbell. Proporciona una reserva rápida de flujo y esta provisto de algoritmos de restauración y adaptación que están específicamente diseñados para proporcionar servicio adaptativo y en tiempo real en entornos de MANETS.
Se caracteriza como un protocolo "in-band" ya que encapsula su infamación en los paquetes de datos IP.

3.6. QoS en *Ant Colony Optimization*

Nuestro proyecto ha estado encaminado a crear una aproximación algorítmica y computable de un comportamiento biológico para intentar modelizar un sistema de enrutado eficiente y fiable. Es por ello que no nos hemos preocupado de la calidad del servicio en profundidad, sino que hemos hecho una aproximación "Best-effort" para dedicar los recursos de que disponíamos a desarrollar el algoritmo de enrutado. Seria interesante ampliar este protocolo en el futuro para dotarlo de un sistema de QoS que lo hiciera aún más favorable para ser usado.

Capítulo 4

Propuesta Encaminamiento en Redes ad hoc

4.1. Inteligencia Artificial

4.1.1. Justificación de uso

El propio análisis del problema nos responde a la cuestión aquí planteada. La búsqueda de caminos mínimos en un grafo es un problema clasificado como Np-hard, si intentamos aplicar una solución estándar nos vemos en la tesitura de que aunque en casos donde el grafo sea pequeño y dichas soluciones sean suficientes, a medida que aumentamos la complejidad del problema quedan patentes las limitaciones de estas soluciones.

El algoritmo de Dijkstra por ejemplo, cuanto mayor sea el grafo mas coste nos llevara resolver el problema con este método, hasta que llegue un momento en que no haya capacidad de computo disponible para llevarlo a cabo. Si a esto le unimos la componente dinámica del sistema que obliga a realizar una búsqueda constante, no parece muy aconsejable realizar búsquedas exhaustivas o ciegas.

Para poder tratar con el problema necesitamos reducir su complejidad de tal manera que sea abordable computacionalmente, tanto en recursos como en tiempo. Aquí es donde entra en juego la inteligencia artificial, necesitamos técnicas que nos permitan encontrar la solución sin necesidad de explorar todo el espacio de búsqueda, todo el grafo.

Una posible solución es utilizar una heurística que guíe la búsqueda y por tanto reduzca la complejidad, justamente en lo que las técnicas basadas en Inteligencia Artificial consisten.

4.1.2. Elección de la técnica

La técnica que utilizamos es lo que algunos denominan la nueva Inteligencia Artificial[Bro90] cuya idea fundamental es definir entidades simples que se agrupan dando como resultado comportamientos complejos, producto de la interacción de las entidades entre si y con el entorno. Es el núcleo fundamental de la arquitectura de subsuncion formulada por R.Brooks, el cual sostiene que el procesamiento simbólico de la información proveniente del entorno es innecesario. Dicha información ya se encuentra subsumida en el entorno y lo único que hay que hacer es interactuar con el, es decir un esquema de estimulo-acción que funciona constantemente. La construcción de distintos módulos que sean sensibles a diferentes estímulos del entorno y actúen en consecuencia se agrupa en capas, de este modo el comportamiento resultante del sistema es producto de todos los comportamientos simples definidos en las distintas capas y lo que es mas importante el control del sistema se deja enteramente al entorno, no es necesario un control como tradicionalmente se había venido haciendo en Inteligencia Artificial.

Las ideas de Brooks encajan muy bien con el comportamiento de los animales reales. Si nos paramos por un momento a observar la vida de un animal -cualquiera un animal en un zoo-, nos acercamos a su jaula y este nos mira, si le hacemos creer o le damos alimento se acercara a nosotros, al igual que ocurre cuando el cuidador entra en su jaula; pero si simplemente nos quedamos parados, mirándole, al cabo del rato el animal volverá su atención hacia otro observador o dormitara -si no hay mas observadores-, podemos también hacer un ruido o movimiento brusco y el animal centrara toda su atención en nosotros. Esto es explicado Ortega y Gasset en su ensayo El hombre y la Gente[Gas57], Ortega lo explica de la siguiente manera : en uno y otro caso son los objetos y acaecimientos del contorno los que gobiernan la vida del animal, el no rige su existencia, sino que está siempre atento a lo que pasa fuera de el. Es lo que Ortega llama estado de alteración.

Es la misma idea expresada por Brooks respecto a como realizar el control en este tipo de sistemas, un control similar al que gobierna a lo animales en el mundo real. Nuestra planteamiento es utilizar sistemas basados en estas ideas básicas : comportamientos emergentes y control en manos del entorno.

4.2. Hipótesis de Trabajo

El encaminamiento en una red *ad hoc* puede verse como un problema de optimización, donde lo que hay que optimizar son los recursos de la red frente al problema del aumento de entropía experimentado por la

red a medida que esta aumenta su tamaño. Con entropía nos referimos al concepto manejado en teoría de la información, introducido por Shannon en el articulo .ª Mathematical Theory of Communication"[Sha48]. La entropía hace referencia al tamaño de información transmitida. A medida que la red crece y necesitamos más información para poder realizar el cálculo de rutas de manera efectiva, crece la entropía. Conforme la entropía crece, aumenta el gasto en recursos que debemos utilizar para seguir manteniendo la red operativa, tanto en la transmisión de dicha información como en su procesamiento, hasta el momento en que no disponemos de recursos suficientes para hacer frente a ese aumento entrópico y la red deja de ser operativa.

El límite del tamaño de la red o, lo que es lo mismo, la cantidad de entropía capaz de ser manejada, viene en gran medida determinado por el algoritmo de encaminamiento y el buen uso que éste haga de los recursos disponibles. El diseño de un protocolo de encaminamiento se concibe, tradicionalmente, como un cálculo de rutas mínimas en un grafo que modela la red. Este planteamiento es muy razonable cuando la topología de la red es estática. El problema surge cuando introducimos una topología dinámica. En este caso, el problema se divide en dos tareas: por un lado averiguar cuál es la topología de la red en un instante de tiempo y, según esa topología, encontrar la mejor ruta posible para una transmisión. Si introducimos las transmisiónes en simultáneo nos damos cuenta de la complejidad del problema.

Abordar dicha tarea desde la perspectiva únicamente de la calidad de las rutas trae como consecuencia una limitación excesiva del tamaño de la red que podemos manejar porque cuanto más calidad queramos obtener en las rutas de transmisión, mayor gasto deberemos destinar a la captura del estado de la red para disponer, de manera fiable, de la topología de la red. No se trata por tanto únicamente de un cálculo de rutas, si no más bien de un equilibrio entre el gasto que requiere el cálculo de rutas frente al beneficio que se obtiene. No siempre la ruta más óptima es la mejor. Por el contrario, en ocasiones la mejor es aquella que requiere menos gasto para su construcción, es decir aquella que más reduce la entropía de la red.

El propósito de la red es otro factor limitante de su tamaño, pues una red que precise velocidades de transmisión muy altas necesita de rutas con el menor retraso posible, por las razones mencionadas anteriormente.

4.2.1. Elección de la métrica

Las soluciones adoptadas hasta el momento fijan la cuestión en encontrar el camino mínimo en número de saltos. ¿Es en realidad esto lo que estamos buscando? Un número menor de saltos implica que existen menos nodos involucrados en la transmisión de un mensaje, pero no garantiza el camino mínimo, pues éste depende

de la velocidad de transmisión de los nodos, y la red no es homogénea. Aun siendo una red homogénea, las velocidad de transmisión pueden variar según la carga de trabajo que experimente cada nodo.

Además esta métrica trae otro problema añadido: que puede ocasionar una fragmentación de la red. Supongamos que la red se conforma de dos núcleos unidos únicamente por dos caminos posibles, uno más largo y otro mas corto. Si aplicamos una métrica basada en saltos, toda la comunicación desde un núcleo a otro viajara por el camino más corto sobrecargando esta zona de la red, pudiendo incluso producir un colapso. El colapso se deriva del fenómeno de congestión. Dicho fenómeno produce además una inyección de paquetes extra en la red por retrasos en las comunicaciones o por perdidas de paquetes, con lo que la entropía global de la red iría aumentando paulatinamente.

La hipótesis aquí planteada es independiente del cálculo de rutas y el protocolo de encaminamiento utilizado es producto de la métrica escogida.

Se necesita un cambio de métrica, es necesario utilizar una métrica basada en el tiempo. La elección de esta métrica ofrece varias ventajas: por un lado, refleja el problema que estamos buscando, es decir, las rutas más rápidas serán aquellas con tiempos de transmisión mas bajos; por otro lado, realiza una distribución homogénea del tráfico.

4.2.2. Implicaciones del cambio de métrica

Lo que en realidad se plantea utilizando este tipo de métricas es simplificar el control necesario para escoger que ruta utilizar en cada momento. Realizamos un control basado en el entorno. Este tipo de control fue propuesto originalmente por Rodney Brooks. Volvamos al ejemplo anterior de la red con dos núcleos. Supongamos homogeneidad en los nodos y un estado de partida para cada nodo como libre de carga. A medida que realicemos transmisiónes, el canal de comunicación más corto se irá sobrecargando, llegando un momento en que, debido a la carga de trabajo, el canal más largo tenga un tiempo de transmisión menor. Si nuestro algoritmo de encaminamiento prioriza las rutas según su tiempo de transmisión, en el momento que empiece ese fenómeno de congestión, desviará el tráfico hacia el camino largo, libre de carga, permitiendo que la red continúe funcionando. Por esta razón, se dice que el control es realizado por el entorno: la decisión de qué ruta escoger es externa, nos viene dada por el propio estado de la red, lo único que hace el protocolo de encaminamiento es actuar según el estado de la red. Como bien señala Brooks [Bro90, Bro91] :

> . . . La observación clave es que el mundo es el mejor modelo de si mismo. Siempre esta exactamente actualizado. Siempre contiene todo detalle que puede ser conocido. El truco esta en percibirlo lo suficientemente a menudo y apropiadamente. . .

El problema de elección de rutas para conseguir la ruta más rápida posible y un reparto homogéneo del tráfico queda reducido a un problema de percepción del estado de la red.

4.3. Ant Colony Optimization

4.3.1. Descripción

De hormigas reales a hormigas artificiales

La percepción visual de muchas especies de hormigas s muy reducida. De hecho, investigaciones recientes acerca del comportamiento de las hormigas revelan que la mayor parte de la comunicación que se realiza entre individuos o entre individuos y su entorno, esta basada en el uso de sustancias químicas producidas por las hormigas. Dichas sustancias son llamadas feromonas.

Los senderos de feromonas, particularmente importante para ciertos tipo es de hormigas, son un tipo especial de feromona que es usada para marcar caminos en el suelo. Guiadas por los restos de feromonas depositadas en el suelo las hormigas recolectoras pueden seguir el camino hacia a la fuente de alimento descubierta por otras hormigas anteriormente. Este comportamiento colectivo de creación de senderos y seguimiento de senderos es lo que inspira el desarrollo del algoritmo Ant Colony Optimization (ACO).

El experimento del doble puente

El experimento del doble puente fue desarrollado por Goss, Aron, Deneubourg y Pasteels ([GADP89]) los cuales usaron un doble puente conectando una colonia de hormigas de la especie Argentina I. humilis y una fuente de comida. Realizaron los experimentos variando el ratio $r = \frac{l_1}{l_2}$ que es la longitud entre los dos ramales del puente. l_1 es la longitud del mas largo y l_2 la del mas corto.

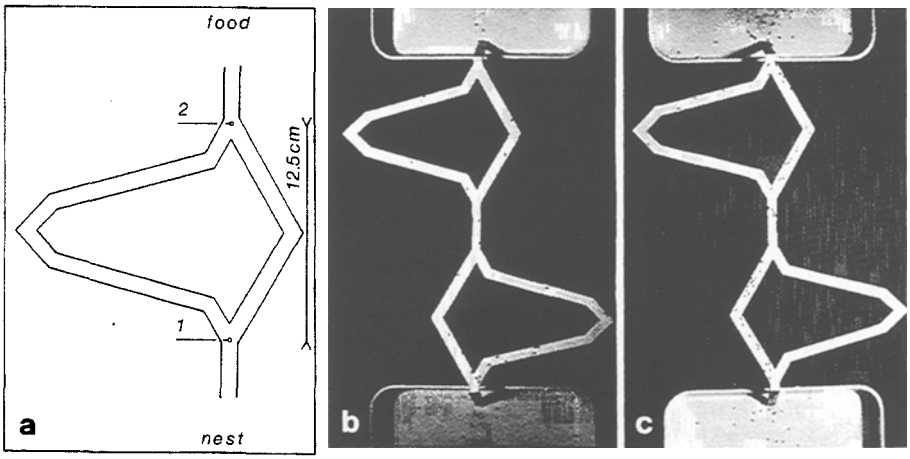


Fig. 1. A colony of *I. humilis* selecting the short branches on both modules of the bridge; a) one module of the bridge, b) and c): photos taken 4 and 8 min after placement of the bridge

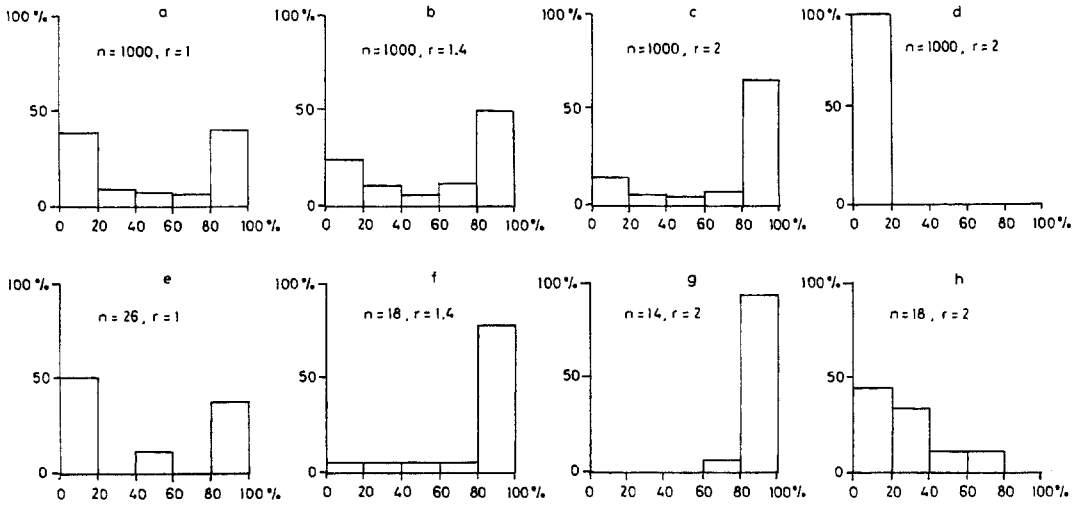


Fig. 2. The distribution of ants that chose the short branch (r = ratio of long branch length to short branch, n = no. of trials). Monte Carlo simulations, with $\Phi = 0.5$ ant s^{-1}, traffic ($\Sigma\Phi$) counted between the 501st and 1000th ant crossing the bridge: a) $r = 1.0$, $n = 1000$, $\Sigma\Phi = 500$, b) $r = 1.4$, $n = 1000$, $\Sigma\Phi = 500$, c) $r = 2.0$, $n = 1000$, $\Sigma\Phi = 500$, d) $r = 2.0$, $n = 1000$, $\Sigma\Phi = 500$ (short branch added after 1000th crossing, traffic counted between 1501st and 2000th crossing). Experiments were performed on 11 different *I. humilis* colonies. Traffic was counted between 30th and 40th min after placement of the bridge (Fig. 1): e) $r = 1.0$, $n = 26$, $\Sigma\Phi = 425 - 1043$ (as $r = 1$, short branch = the branch on the left leaving the nest), f) $r = 1.4$, $n = 18$, $\Sigma\Phi = 329 - 696$, g) $r = 2.0$, $n = 14$, $\Sigma\Phi = 407 - 912$, h) $r = 2.0$, $n = 18$, $\Sigma\Phi = 317 - 846$ (short branch added 30 min after the experiment began, traffic measured between 20 and 30 min later). For the smallest total traffic ($\Sigma\Phi = 317$), the choice is significantly nonrandom if the ants choosing a branch are outside the range 44 – 56 % ($p < 0.05$)

Figura 4.1: Experimento del doble puente de ([GADP89])

De los experimentos que realizaron dos son de gran interés :

1. En el primer experimento, un ramal es mas largo que el otro y ambos están disponibles desde el principio para las hormigas. El resultado al cabo de un tiempo fue que el ramal corto fue escogido preferente en defecto del largo.

2. En el segundo experimento solo se le presento el ramal largo y pasado un tiempo se introdujo el corto. El resultado fue que el ramal largo fue el escogido frente al corto.

La explicación del primer experimento es la siguiente : el ramal mas corto será recorrido mas rápidamente, si una hormiga deposita feromona en cada viaje, el corto tendrá una mayor concentración. Por tanto cuando una hormiga tiene que tomar la decisión acerca de que camino seguir, el corto presentara una mayor carga de feromona inclinando la decisión por este ultimo y al cabo de un tiempo la gran mayoría de hormigas circulara por el ramal mas corto.

Por el contrario en el segundo experimento solo hay un camino disponible, el cual tendrá toda la concentración de feromona Cuando se introduce el nuevo ramal mas corto, este no tiene ninguna marca porque no ha sido recorrido ninguna vez, por lo que en el momento de decidir que camino tomar la cantidad de feromona disponible en el largo frente a la del corto inclinara la decisión en favor del primero.

La conclusión que se obtiene de ambos experimentos es : la decisión tomada en conjunto por toda la colonia acerca de que camino es el mejor para alcanzar la comida, lleva tiempo y es tomada una única vez, hasta que las hormigas vuelven al hormiguero y las marcas depositadas en el suelo se borran. Por tanto solo deciden entre los elementos que tengan disponibles, en el segundo experimento el ramal corto aparición una vez escogido el camino, por lo que no se tomo en cuenta. A modo de introducción este problema que deja en evidencia el experimento del doble será uno de los mayores retos del ACO y es lo que se llama Evaporación de Feromona Artificial.

Simple Ant Colony Optimization (S-ACO)

A partir del experimento del doble puente, Dorigo crea un algoritmo recreando el comportamiento de las hormigas para encontrar el camino mínimo en un grafo.

Comienza considerando un grafo estático G=(N,A) donde N es el conjunto de vértices del grafo y A es el conjunto de aristas no dirigidas que los conecta. Los dos puntos entre los que se quiere establecer el camino mínimo los llama fuente y destino o análogamente con las hormigas reales hormiguero y fuente de alimento.

El primer problema que aparece es que mientras las hormigas construyen la solución pueden aparecer ciclos, como consecuencia del mecanismo de actualización de feromona a medida que la hormiga avanza los ciclos tenderían a ser mas y mas atrayentes para las hormigas y estas quedarían atrapadas dentro. Pero incluso si las hormigas pudiesen escapar de los ciclos, la distribución global de feromona ya no se vería favorecía la ruta mínima entre la fuente y el destino. Ya que este problema es debido a la actualización de feromona a medida que avanza la hormiga, parece razonable eliminar la actualización de feromona a medida que la hormiga avanza, pero si se suprime este mecanismo el sistema deja de funcionar, incluso para el caso mas simple del doble puente.

Lo que Dorigo explica remitiendo a los experimentos de Goss([GADP89]), este punto hay que explicarlo con mas detalle. La razón es muy simple si volvemos a las hormigas reales, como dijimos anteriormente la orientación básica de una hormiga son los rastros dejados en el suelo, de tal manera que cuando encuentra una fuente de comida debe volver a la colonia, si no ha marcado su rastro con feromona no es capaz de volver a la colonia, porque *no recuerda* el camino que ha seguido para alcanzar la fuente de comida. Es la misma situación que el cuento infantil de Hansel y Gretel , tienen que marcar el camino para salir del bosque a medida que avanzan por el para que cuando quieran volver sean capaces de identificar el camino.

Volviendo al S-ACO, Dorigo extiende la capacidades básicas de las hormigas artificiales dotándolas de una memoria primitiva capaz de almacenar la ruta que han seguido, así como el coste de la misma. Gracias a esta memoria las hormigas artificiales son capaces de implementar una serie de comportamientos que les permiten construir eficientemente una solución :

1. Construcción probabilista de la solución basada en rutas de feromonas, sin mecanismo de actualización de feromona a medida que la hormiga avanza.
2. Recorrido inverso determinista con eliminación de ciclos a la vez que se actualizan las rutas de feromonas.
3. Evaluación de la calidad de la solución generada, en base a dicha evaluación se determina la cantidad de feromona que deberá ser depositada.

Construcción probabilista de la solución Las hormigas del S-ACO pueden ser concebidas con dos modos de funcionamiento: hacia delante y hacia atrás (a partir de ahora por similitud con los textos ingleses nos referiremos a ellas como forward y backward). Se encuentran en modo forward cuando se están moviendo desde el hormiguero hacia la fuente de comida y en modo backward cuando regresan de la fuente hacia el nido.

Cuando una hormiga en modo forward alcanza la fuente de comida, cambia su modo de comportamiento a modo backward y comienza su viaje de regreso hacia la colonia. En S-ACO las hormigas forward construyen una solución escogiendo de manera probabilista el nodo siguiente al que moverse entre los vecinos del nodo donde se encuentren actualmente. (Dado un grafo G= (N,A), dos nodos $i, j \in N$ son vecinos si existe un arco(i,j) $\in$ A que los conecte) La Elección probabilista esta pesada por la feromona depositada previamente por otras hormigas depositada en ese arco. Las hormigas en modo forward no depositan ninguna cantidad de feromona mientras se mueven. Esto junto con el comportamiento determinista de las hormigas en modo backward evita la aparición de ciclos.

Recorrido inverso determinista El uso de una memoria explícita permita a una hormiga volver por el camino que ha le ha llevado hasta la fuente de comida. S-ACO mejora el rendimiento de las hormigas, eliminado los ciclos que se encuentren en el camino que les haya llevado a la solución. Mientras se mueven hacia atrás actualizan la feromona presente en los arcos que recorren.

Actualización de feromona basada en la calidad de la solución En S-ACO la hormiga memoriza el camino que le ha llevado hasta la solución a la vez que el coste de los arcos que ha recorrido. De tal modo que pueden evaluar el coste de la solución obtenida y utilizar dicho coste para modular la cantidad de feromona que depositan en cada arco que forma el camino recorrido.

Haciendo que la función que genera la cantidad de feromona a depositar, una función variable en función del coste del camino se consigue que se deposite mas cantidad en los caminos mas cortos, de tal manera que la búsqueda de caminos mas cortos converja mas rápido hacia la mejores soluciones.

Evaporación de Feromona En las hormigas reales la intensidad de la feromona presente en el medio decrece en función del tiempo. En el S-ACO dicha evaporación es simulada aplicando una regla de reducción de feromona.

La evaporación de feromona reduce la influencia de la feromona depositada en las primeras, cuando las hormigas construyen las peores soluciones. Aunque este mecanismo no influye en las hormigas reales, la evaporación es muy lenta comparada con el movimiento de las hormigas, en los algoritmos basados en hormigas juega un papel muy importante.

Detalles del S-ACO

Búsqueda de caminos : cada hormiga construye, comenzando desde el nodo fuente una solución al problema aplicando una política de decisión local. En cada nodo, la información que este almacena acerca de la zona que le es mas próxima (sus vecinos), es percibida por la hormiga y usada de una manera estocástica para decidir a que nodo debe visitar a continuación.

Al comienza de cada proceso de búsqueda, una cantidad constante de feromona es asignada a todos los arcos. Cuando la hormiga k se encuentra en el nodo i, usa el sendero de feromona τ_0 para calcular la probabilidad de escoger el nodo j de la siguiente manera :

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in N_i^k} \tau_{ij}^\alpha} & \text{si} \quad j \in N_i^k \\ 0 & \text{si} \quad j \in N_i^k \end{cases}$$

donde N_i^k es la lista de nodos disponibles para la hormiga k cuando se encuentra en el nodo i. En S-ACO la vecindad de un nodo contiene a todos los nodos que conectan directamente con dicho nodo, exceptuando aquel nodo del que procede la hormiga, es decir del que procede. De esta manera se evita que las hormigas vuelvan sobre el nodo inmediatamente visitado. Solo en el caso en que la vecindad de un nodo sea el conjunto se permite a la hormiga volver sobre sus pasos. Hay que señalar que esta política puede inducir fácilmente a la generación de caminos cíclicos en el grafo.

Un hormiga salta repetidamente de nodo a nodo usando la política de decisión previamente descrita hasta que alcanza el nodo destino. Es claro que debido a las diferencias entre los caminos que siguen unas hormigas u otras, existirán diferencias en el tiempo que tardan en alcanzar el nodo objetivo, las que vayan por los caminos mas cortos finalizaran antes.

Trazado de rutas y actualización de la feromona : Cuando una hormiga alcanza su destino, cambia de comportamiento, de forward a backward y comienza a reconstruir el camino que ha hecho en sentido inverso paso a paso. Antes de comenzar el regreso a la fuente, la hormiga elimina los ciclos que pueda haber en el camino que ha construido mientras buscaba el nodo objetivo. El problema de los ciclos es que pueden recibir cargas de feromonas varias veces mientras la hormiga realiza su viaje de retorno a la fuente, generando un fenómeno de auto reforzamiento de los ciclos frente a los caminos no cíclicos.

Mientras la hormiga regresa deposita una cantidad fija de feromona en los arcos que ha visitado. En particular, si una hormiga k en modo backward atraviesa el arco (i,j) cambia la cantidad de feromona que

marca al arco de la siguiente manera :

$$\tau_{ij} \leftarrow \tau_{ij} + \triangle\tau^k$$

Con esta regla una hormiga que use el arco que conecta j e i incrementa la probabilidad de que el resto de las hormigas usen ese arco en el futuro.

La elección de la cantidad de feromona depositada se realiza en función de la calidad de la solución que la hormiga ha encontrado. En particular el caso mas simple es que sea una cantidad constante.

Evaporación de las marcas de feromona : la evaporación de feromona puede ser vista como un mecanismo que evita la rápida convergencia de todas las hormigas hacia una ruta suboptima. De hecho el decrecimiento de la cantidad de feromona que se encuentra en los caminos favorece la exploración de nuevas rutas durante el proceso de búsqueda global. Señala Dorigo que en las hormigas reales este mecanismo esta presente de igual manera, pero no juega un papel fundamental en las hormigas reales.

En nuestra opinión hay un cierto malentendido acerca de esta cuestión. Es falso que no juegue un papel fundamental en la vida de una colonia de hormigas reales, si no hubiese evaporación de feromona todas las mañanas las hormigas recorrían los mismo camino una y otra vez, día si y día también. Esto no ocurre, las hormigas cambian progresivamente de zonas de exploración gracias a que la feromona se evapora o se destruye por otros elementos del entorno de las hormigas.

El malentendido radica en la compresión del problema, el problema fundamental al que se enfrentan las hormigas reales es el de la supervivencia (como cualquier otro ser vivo). Varios autores (biólogos incluidos) sostienen que la ya mencionada evaporación de feromona no juega un papel importante porque es muy lenta comparada con la búsqueda de rutas óptimas. Cuando en realidad es justo el mecanismo que las permite sobrevivir y explorar zonas nuevas cada día, porque es importante mantener la perspectiva de que lo único que realmente buscan las hormigas es sobrevivir.

Resumiendo hay que entender la evaporación de feromona desde la perspectiva de las hormigas y el papel que para ellas juega. Curiosamente desde la perspectiva de las hormigas tanto el mecanismo de evaporación de feromona artificial como el natural juegan un papel clave, sin dichos elementos el sistema no funcionaria.

4.3.2. Ant Colony Optimization : Aplicaciones en redes *ad hoc*

Concretamente es interesante la aplicación que Dorigo llama "AntNet", que es la aplicación al encaminamiento en redes estáticas. El encaminamiento en estos algoritmos es emergente, surge de la propia dinámica del algoritmo. Esto es muy atractivo ya que ahorra recursos, no es necesario un exhaustivo cálculo de rutas, mantenimiento de rutas e intercambio de información.

El cálculo de rutas surge de manera emergente mediante la construcción de rutas a partir de otras ya conocidas, de tal manera que si vinculamos los paquetes de datos a transmitir con hormigas que realicen la exploración, el resultado es que de la propia transmisión de información resulta la ruta óptima. Esto contrasta con el planteamiento tradicional de realizar por un lado la construcción de rutas para después transmitir.

No es necesario un mantenimiento de rutas porque las rutas son construidas por saltos que se realizan de un nodo a otro. No existe una información global que actualizar: en el caso de que una ruta sufra modificaciones, se actualiza el salto en concreto que ha sufrido los cambios.

Respecto al intercambio de información necesaria para el cálculo de rutas, hay que tener presente que estos algoritmos trabajan mediante "stimergy", que significa comunicación indirecta a través del entorno, con lo que la información reside en los nodos de manera local y las hormigas acceden a ella cuando visitan el nodo.

Por todo esto, este tipo de algoritmos parece un candidato claro al encaminamiento en redes *ad hoc* ya que reducen la entropía generada por el cálculo de rutas.

Problema de una aplicación directa del Ant Colony Optimization

La aplicación directa del algoritmo tal y como esta descrito por Dorigo no es viable en redes ad hoc debido a la lenta convergencia que ofrece. La propuesta que el ofrece y que llama "AntNet"se fundamenta en una topología estática de la red, por tanto todas las rutas posibles son conocidas de antemano, lo único que realizan las hormigas es la elección de la ruta en función de la carga de tráfico que experimente cada ruta. Además estas rutas son validas siempre, en redes ad hoc esto no ocurre, la topología es desconocida, por tanto la aplicación directa del ACO traería consigo una sobrecarga de información en la red y un tiempo de convergencia lento.

Una topología dinámica implica que desconocemos las rutas disponibles para realizar una comunicación, el primer paso sería realizar una exploración para capturar la topología de la red, es decir las rutas disponibles.

En principio esto no debería ser un problema porque es la misma situación que experimentan las hormigas reales tras un día de lluvia. Todos sus senderos marcados con feromona han sido borrados, cuando las hormigas abandonan la colonia no tienen ningún rastro que seguir y comienza su búsqueda , a ciegas, del alimento. En un tiempo relativamente corto habrán reconstruido sus senderos y volverá a existir un flujo entre la colonia y la fuente de comida. Ahora bien el radio de exploración de las hormigas suele ser pequeño, por esta razón no les resulta problemático un borrado total de feromona, existen hormigas suficientes en la colonia para reconstruir los senderos y volver a recolectar alimento antes de que la existencia de la colonia peligre.

En la aplicación del ACO a las redes ad hoc esto no sucede, proporcionalmente el espacio de búsqueda que deben cubrir las hormigas artificiales es muy grande en base al tiempo de respuesta que se le exige a un protocolo de encaminamiento, para cumplir este requisito deberíamos destinar muchos recursos para una rápida captura de la topología de la red. Pero si se destinan muchos recursos a acelerar la captura de la topología de la red, aumenta el intercambio de información, de tal modo que podemos llegar a colapsar la red solamente intentando explorar las rutas disponibles. Aquí es donde radica el problema si no queremos sobrecargar la red aumentamos el tiempo de convergencia del ACO y si queremos disminuir el tiempo de convergencia sobrecargamos la red Hay que añadir que mientras en las hormigas reales el borrado total de feromona es esporádico (desde el punto de vista de las propias hormigas) en una red ad hoc la topología dinámica puede invalidar la topología capturada rápidamente.

Por todas estas razones la aplicación directa del Ant Colony Optimization a las redes ad hoc no es viable, es necesario una modificación de su funcionamiento para poder aplicarlo de tal modo que acelere su tiempo de convergencia sin peligro de sobrecargar la red.

Modificación del ACO para su aplicación en redes ad hoc

Ya existen aplicaciones de las ideas de Dorigo al campo de las redes *ad hoc*, pero todas en mayor o menor medida incurren en el mismo error: utilizar búsquedas en anchura. Este error es producto de que, para que los algoritmos de hormigas funcionen, es necesario que todas las posibles soluciones al problema estén disponibles para las hormigas. En una red estática esta implícito, la topología no cambia y el algoritmo funciona muy bien. Pero en una red *ad hoc* esto no ocurre. Por tanto, la solución adoptada por otros autores es realizar una primera búsqueda por anchura para tener todas las rutas disponibles; y una vez que están presentes todas las rutas, se aplica el ACO. Existen otros autores que disfrazan esta búsqueda exhaustiva, en el proceso de exploración de rutas, construyendo un modelo topológico de la red.

En la propuesta original de Dorigo no existen búsquedas exhaustivas, ni tampoco son realizadas por las hormigas de manera natural, todas las búsquedas son por profundidad.

El error reside en lo mencionado anteriormente: para que los algoritmos de hormigas funcionen es necesario que todas las posibles soluciones al problema estén disponibles para las hormigas. Esto es cierto, no obstante, el error consiste en entenderlo como un requisito para el espacio de búsqueda que se le presente al ACO. Dicho espacio no tiene por qué ser el espacio de búsqueda global, puede ser una porción más o menos grande de dicho espacio. Lo exigible es únicamente que contenga la solución buscada. Este es el comportamiento que exhiben las hormigas reales, las cuales no exploran el espacio total, sino solo un fragmento, como señala Deborah Gordon en "Ants at Work" [Gor00], la exploración realizada por las hormigas recolectoras es dirigida. Gordon señala lo siguiente: al comienzo de cada día un tipo de hormiga, que Gordon llama "patrulla", sale en diversas direcciones a explorar los alrededores de la colonia. Este tipo de hormiga, al volver a la colonia, indica si ha encontrado o no una fuente de alimento. En caso positivo, estimula de alguna manera a las hormigas recolectoras de tal manera que éstas salen en la dirección indicada por la patrulla hacia la fuente de comida y crean así un flujo de tráfico entre la fuente y el hormiguero. La obtención de la ruta mínima es realizada por las hormigas recolectoras, pero solo en la zona que previamente ha marcado la patrulla. El problema del cálculo de ruta se presenta después de haber realizado una exploración espacial para determinar la zona a explorar.

Este mismo comportamiento se puede aplicar en el ACO. Se crea un mecanismo previo que sitúa espacialmente al objetivo. Así se consigue que la exploración realizada por el ACO se reduzca a una zona local donde se encuentra el objetivo. Por tanto, dicha zona local se puede cubrir mediante búsquedas en profundidad. Para evitar que esta zona local sea muy grande, se mantiene el mecanismo de patrulla descrito por Gordon para ir afinando más el radio de búsqueda. De esta manera se acelera la convergencia del algoritmo y se reduce el coste de la búsqueda.

4.4. Propuesta de Algoritmo

4.4.1. Mecanismo de cálculo de rutas

Exploración de rutas

En esta propuesta de algoritmo existen dos mecanismos de exploración: uno, inicial, que actúa en la situación en la que el nodo responsable de la transmisión no conozca ninguna ruta hacia el nodo destino; otro, que actúa en paralelo con la transmisión, que se encarga de capturar la topología de la red en la zona

donde se lleva a efecto la transmisión. Al primero lo llamamos "mecanismo de descubrimiento de ruta" y al segundo "presentación de rutas".

Descubrimiento de ruta: este mecanismo actúa facilitando al nodo fuente una ruta disponible para realizar la transmisión. Se realiza mediante una hormiga de búsqueda ($Request\_Ant$) la cual realiza una exploración en profundidad con control de ciclos hasta que alcanza el nodo destino. En ese momento la hormiga de búsqueda se destruye y a partir de ella se genera una hormiga de retroceso ($Backward\_Ant$), la cual recorre el camino inverso seguido por la hormiga de exploración hasta llegar al nodo fuente. En su recorrido, inicializa la ruta de transmisión hacia el nodo destino, en los nodos que va visitando. En el momento que alcanza al nodo fuente, se inicializa la tabla de rutas correspondiente al nodo destino de la transmisión y comienza la transmisión.

Presentación de ruta: este mecanismo actúa una vez que se ha iniciado la transmisión entre el nodo fuente y destino. Se generan de manera proporcional hormigas de exploración($Forward\_Ant$) con un tiempo de vida máximo cuyo valor se asigna según la mejor ruta de transmisión que disponga el nodo fuente. Estas hormigas realizan una búsqueda en profundidad con control de ciclos, si expira su tiempo de vida son destruidas. Si alcanzan el nodo destino generan una hormiga de retroceso ($Backward\_Ant$) que provee al nodo origen de una nueva ruta de transmisión.

La diferencia fundamental que existe con el ACO es este segundo mecanismo de presentación de soluciones. Este mecanismo nos permite realizar un uso eficiente de los recursos, de tal manera que los recursos derivados a la captura de la topología de la red son proporcionales al beneficio que obtengamos de realizar un encaminamiento más o menos óptimo. Este es el mismo principio que opera en las hormigas reales. Como dice D.Gordon, la recolección de alimento supone un gasto para la colonia, por tanto solo se realiza cuando la colonia necesite alimento. En nuestro caso, la necesidad de un encaminamiento mejor o peor viene determinado por la longitud del mensaje, en número de paquetes a transmitir, así cuanto más larga sea una transmisión más recursos destinaremos a capturar la topología de la red, pues mayor beneficio obtendremos. Por el contrario, en transmisiónes cortas no compensa tanto el gasto en recursos y se realiza un gasto menor.

Este mecanismo presenta otra ventaja: la de la convergencia. A medida que se realiza una exploración, la zona de búsqueda local disminuye, pues cada vez encontraremos rutas mejores. De esta manera se minimiza la entropía del sistema, pues el tiempo de vida de las hormigas de exploración se reduce a la par que se reducen los tiempos de transmisión, ya que mejoramos la ruta calculada. Al tratarse de búsquedas en profundidad que actúan solo a nivel local, se crean zonas independientes unas de otras permitiendo una mayor carga de

trabajo de toda la red y un mayor número de transmisiónes a simultáneo.

4.4.2. Modelo de rutas

Para capturar la información se utilizan las hormigas de retroceso ($Backward\_Ant$) que se generan cuando un paquete de datos o una hormiga de exploración ($Forward\_Ant$, $Request\_Ant$) alcanza su destino, estas hormigas implementan un contador de tiempo desde el momento de su creación hasta alcanzar el nodo origen de la transmisión. Para representar esta información se utiliza un sencillo modelo estadístico por transmisión, que resume la información de transmisión entre un nodo origen y un nodo destino. Dicho modelo almacena la media de los tiempos de transmisión, el mejor tiempo de transmisión y el número de envíos realizados, se actualiza de la siguiente manera:

x = tiempo de transmisión

- Media (μ) $\rightarrow \mu_{i+1} = \mu_i + (\frac{x-\mu_i}{N_i+1})$
- Tiempo mejor (W) $\rightarrow W_{i+1} = \text{mín}(W_i, x)$
- Numero de envíos (N) $\rightarrow N_{i+1} = N_i + 1$

A partir de este modelo podemos evaluar la calidad de una ruta frente al resto :

td = tiempo de transmisión, de la ruta a evaluar, desde el nodo actual hasta el nodo d.
μ = media de transmisiónes desde el nodo actual hasta el nodo d.

$$f(td, \mu) = \left\{ \begin{array}{cc} 0 & \text{si } \frac{\mu}{td} < 1 \\ 1 - \frac{1}{e^{\frac{\mu}{td}-1}} & \text{e.o.c} \end{array} \right\}$$

4.4.3. Elección de Ruta

Para la elección de la ruta que seguirá un paquete de información, se aplica el ACO a la manera clásica descrita por Dorigo. Los paquetes se encaminan siguiendo una elección al azar ponderada de la tabla de rutas correspondiente al nodo destino de la transmisión. Dicha tabla contiene el siguiente salto que el paquete debe realizar. Cuando un paquete de información alcanza su objetivo genera una hormiga de retroceso (*Backward_Ant*) que informa del estado actual de la ruta seguida por el paquete de información.

La actualización de las tablas se realiza de la siguiente manera:

- Para la ruta que ha seguido el paquete :
  $\Gamma_{id}^{+} = \Gamma_{id} + f(td, \mu)(1 - \Gamma_{id})$
- Para el resto de las posibles rutas que contenga la tabla :
  $\Gamma_{jd}^{+} = \Gamma_{jd} - f(td, \mu)\Gamma_{jd}$

donde

- Γ_{id} = probabilidad de alcanzar el nodo d a través del nodo i.
- td = tiempo de transmisión desde el nodo i hasta el nodo d.
- μ = media de transmisiónes desde el nodo actual hasta el nodo d.
- $f(td, \mu)$ es la función que evalúa la calidad de una ruta previamente descrita.

4.5. Dinamismo del entorno

Aparición de una ruta mejor: Si la red cambia y aparece un camino mejor, éste estará dentro del alcance fijado por nuestro límite (el camino mejor encontrado hasta el momento). Por tanto, el mecanismo de presentación de soluciones lo reconocerá y lo incorporará a la lista de rutas disponibles para su posterior evaluación por el mecanismo de elección.

Empeoramiento de la ruta mejor: Si la red cambia y la ruta que estamos utilizando se degrada, no vera aumentada su probabilidad pero se seguirá usando y se rechazaran las otras, ya que nuestra información sobre la media y el camino mejor estará anticuada. La solución que se adopta es: los modelos estadísticos se reinician después de un número determinado de muestras; de esta manera se actualiza el rango de búsqueda para la presentación de soluciones y las probabilidades volverán a funcionar. Si el sistema no cambia y reiniciamos el modelo estadístico, la ruta que reinicia el modelo será justamente aquella que era la mejor, por lo que el comportamiento general no sufre cambios.

Desconexión de un nodo participante de la ruta de transmisión: Si la ruta se rompe, necesitamos encontrar una ruta alternativa, pero a la vez debemos optimizar recursos. La solución que se plantea es la misma que utilizan las hormigas en la vida real: cuando pierden el alimento, comienzan una búsqueda local en la zona donde se encuentran. Y esto por una razón muy simple: los objetos se mueven poco a poco, no a grandes saltos, por tanto es mas probable que el objeto que buscan se encuentre en las cercanías de donde estaba antes. Eso mismo hacemos nosotros, realizamos una búsqueda desde el nodo que ha detectado la rotura. Y ello mediante el mismo procedimiento con que se realiza la búsqueda de rutas nuevas. Una vez que el nodo ha encontrado de nuevo el destino, envía los paquetes que estaban en espera de ruta.

Con el resto de la red, el camino que seguían los paquetes con anterioridad a la rotura, no se hace nada, porque solucionamos el problema como hemos mencionado en el párrafo anterior. En realidad la solución no es completa, se trata mas bien de una transformación o redefinición del problema. Se trabaja sobre el problema no sobre una solución: se transforma un problema de perdida de ruta en uno de empeoramiento de ruta, para el cual ya hemos descrito anteriormente como afrontarlo. Esta solución es sencilla pero eficaz, no requiere de mensajes de error ni de comportamientos nuevos en el algoritmo.

Capítulo 5

COMPARATIVA PROTOCOLOS ENCAMINAMIENTO

5.1. Resultados de simulaciones.

5.2. Análisis de resultados.

5.3. Ventajas y desventajas.

Capítulo 6

Conclusiones

Lo más importante es el cambio de perspectiva a la hora de diseñar protocolos de encaminamiento para redes ad hoc. El objetivo último debe ser mantener un control entrópico del sistema. Al tratarse de sistemas complejos, las situaciones que deberíamos prever, para poder controlar la entropía del sistema, son demasiadas para realizar un control explícito. Por esta razón, optamos por un control adaptativo, es decir, que sea el propio sistema quien se controle.

6.1. Posibles Problemas

Desde un punto de vista teórico pueden existir problemas en el siguiente caso

- Únicamente se envía una sola *Request_Ant* para encontrar ruta, quizás debería parametrizarse de manera que se envíen varías para ayudar a la rapidez de convergencia del algoritmo y a su vez para evitar que una perdida de alguna de estas inicializaciones se pierda y bloquee el algoritmo. Al mandar varías necesitamos un mecanismo de elección entre las tres ya que sino la ultima que llegue reiniciaría el algoritmo cuando lo que interesa es que sea la primera. Esto requiere un mecanismo de control que no debería ser muy complicado y aunque no suponga ningún problema es necesario pensarlo con detalle para que encaje con el resto del algoritmo.

Actualmente no estamos en condiciones de realizar un análisis exhaustivo de los problemas que pueden surgir porque se están realizando las pruebas necesarias para analizar el resultado del algoritmo. Para el problema descrito anteriormente no tenemos información de cuanta dimensión tiene en casos prácticos.

Capítulo 7

Trabajo Futuro

Ahora mismo se están realizando las pruebas necesarias para evaluar su rendimiento. Las dos líneas principales de su desarrollo actual son:

Muestreo de paquete: El algoritmo necesita muestrear la ruta que siguen los paquetes de información para tener un conocimiento actualizado del estado de la red. Según la estadística, una muestra representativa contiene la misma información que la población entera. Es decir: no será necesario recuperar información de todas las rutas que han seguido los paquetes para construir el modelo estadístico, antes bien, con una muestra significativa valdría. De tal manera se reduce los recursos necesarios para el cálculo de rutas.

Mecanismo de Presentación de Soluciones: Se está estudiando qué relación aplicar entre la longitud de transmisión a realizar y los recursos destinados a ella. Actualmente dicha relación es puramente lineal, pero probablemente una relación logarítmica actúe bastante mejor, pues se acerca más al modelo de hormigas reales.

Bibliografía

[AWD04] M. Abolhasan, T. Wysocki, and E. Dutkiewicz. A review of routing protocols for mobile ad hoc networks. *Ad Hoc Networks*, 2(1):1–22, January 2004. ISSN 1570-8705. URL `http://dx.doi.org/10.1016/S1570-8705(03)00043-X\%20`.

[BDT99] E Bonabeau, M Dorigo, and G Theraulaz. Swarm intelligence: From natural to artificial systems. Oxford University Press, 1999.

[Bel57] Richard Ernest Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[BMJ+98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Mobile Computing and Networking*, pages 85–97, 1998. URL `citeseer.ist.psu.edu/broch98performance.html`.

[Bou02] Imed Bouazizi. Ara - the ant-colony based routing algorithm for manets. In *ICPPW '02: Proceedings of the 2002 International Conference on Parallel Processing Workshops*, page 79. IEEE Computer Society, Washington, DC, USA, 2002. ISBN 0-7695-1680-7.

[Bro90] Rodney A. Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6(1&2):3–15, June 1990. URL `citeseer.ist.psu.edu/188611.html`.

[Bro91] Rodney A. Brooks. Intelligence without representation. Number 47 in Artificial Intelligence, pages 139–159. 1991. URL `citeseer.ist.psu.edu/brooks91intelligence.html`.

[CD] G. Di Caro and M. Dorigo. AntNet: a mobile agents approach to adaptive routing. Technical Report IRIDIA/97-12, Université Libre de Bruxelles, Belgium. URL `citeseer.ist.psu.edu/article/dicaro97antnet.html`.

[Cor] Gord Fedoriw Corey. Application of swarm intelligence solving the shortest route problem. URL `citeseer.ist.psu.edu/411369.html`.

[Csy] Zeinalipour-Yazti Demetrios Csyiaztics. A glance at quality of services in mobile ad-hoc networks. URL `citeseer.ist.psu.edu/573350.html`.

[DCG99] Marco Dorigo, Gianni Di Caro, and Luca M. Gambardella. Ant algorithms for discrete optimization. *Artif. Life*, 5(2):137–172, 1999. ISSN 1064-5462.

[DCG05] F Ducatelle, G Di Caro, and L M Gambardella. Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks. *International Journal of Computational Intelligence and Applications (IJCIA*, (5):169–184, 2005.

[DDG05] G. Di Caro, F. Ducatelle, and L.M. Gambardella. Swarm intelligence for routing in mobile ad hoc networks. In *Proceedings of the IEEE Swarm Intelligence Symposium (SIS 2005)*. Pasadena, CA, USA, June 2005.

[Dor04] M. Dorigo. *Ant Colony Optimization*. MIT Press, 2004.

[FF62] L. R. Ford and D. R. Fulkerson. *Flows in Network*. Princeton University Press, 1962.

[GADP89] S. Goss, S. Aron, J.L. Deneubourg, and J.M. Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76(12):579–581, 1989.

[Gas57] Jose Ortega Gasset. El hombre y la gente. In *Revista de Occidente*, 1957.

[GGT07] Simon Garnier, Jacques Gautrais, and Guy Theraulaz. The biological principles of swarm intelligence. *Swarm Intelligence*, 1(1):3–31, June 2007. URL `http://dx.doi.org/10.1007/s11721-007-0004-y`.

[Gor00] D.M. Gordon. *Ants at Work: How an Insect Society Is Organized*. WW Norton & Company, 2000.

[Gro01] Temporally-ordered routing algorithm (tora) version 1. IETF MANET Working Group, July 2001.

[HH96] C. Hochberger and R. Hoffmann. Solving routing problems with cellular automata, 1996. URL `citeseer.ist.psu.edu/hochberger96solving.html`.

[IC03] J. Liu I. Chlamtac, M. Conti. Mobile ad hoc networking: imperatives and challenges. In *Ad-Hoc Networks Journal 1 (Inaugural Issue, 1)*, 2003.

[JM96] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996. URL `citeseer.ist.psu.edu/johnson96dynamic.html`.

[JMC+01] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *Proceedings of the 5th IEEE Multi Topic Conference (INMIC 2001)*, 2001. URL `citeseer.ist.psu.edu/jacquet01optimized.html`.

[KE01] James Kennedy and Russell C. Eberhart. *Swarm intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001. ISBN 1-55860-595-9.

[LAZC00] Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell. INSIGNIA: An IP-based quality of service framework for mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, 60(4):374–406, 2000. URL `citeseer.ist.psu.edu/lee00insignia.html`.

[Mil] Peter Miller. Swarm theory. URL `http://ngm.nationalgeographic.com/2007/07/swarms/miller-text`.

[MMPS00] Frank McSherry, Gerome Miklau, Don Patterson, and Steve Swanson. The performance of ad hoc networking protocols in highly mobile environments, 2000. URL `citeseer.ist.psu.edu/mcsherry00performance.html`.

[PB94] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *SIGCOMM '94: Proceedings of the conference on Communications architectures, protocols and applications*, pages 234–244. ACM, New York, NY, USA, 1994. ISBN 0-89791-682-4.

[Per97] C. Perkins. Ad-hoc on-demand distance vector routing, 1997. URL `citeseer.ist.psu.edu/article/perkins97adhoc.html`.

[Per01] Charles E. Perkins. *Ad hoc networking: an introduction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001. ISBN 0-201-30976-9. 1–28 pp.

[RMLG] A. Rizzoli, R. Montemanni, E. Lucibello, and L. Gambardella. Ant colony optimization for real-world vehicle routing problems. *Swarm Intelligence*. URL `http://dx.doi.org/10.1007/s11721-007-0005-x`.

[RS] Sundaram Rajagopalan and Chien-Chung Shen. A routing suite for mobile ad hoc networks using swarm intelligence. URL `citeseer.ist.psu.edu/rajagopalan04routing.html`.

[Sha48] C E Shannon. A mathematical theory of communication. *Bell System Technical Journal*, (27): 379–423, 1948.

[SJ05] Chien-Chung Shen and Chaiporn Jaikaeo. Ad hoc multicast routing algorithm with swarm intelligence. *Mob. Netw. Appl.*, 10(1-2):47–59, 2005. ISSN 1383-469X.

[SSY+03] John Sum, Hong Shen, G. Young, Jie Wu, and Chi-Sing Leung. Analysis on extended ant routing algorithms for network routing and management. *J. Supercomput.*, 24(3):327–340, 2003. ISSN 0920-8542.

[Whi97] T. White. Routing with swarm intelligence. Technical Report SCE-97-15, 1997. URL `citeseer.ist.psu.edu/white97routing.html`.

[XSLC] H. Xiao, W. Seah, A. Lo, and K. Chua. A flexible quality of service model for mobile ad-hoc networks. URL `citeseer.ist.psu.edu/315282.html`.