Enrutamiento en Redes Móviles Ad-hoc: Una evaluación de los protocolos DSDV, AODV y basados en Colonias de hormigas

Nicolás Castro Ponce
Universidad Técnica Federico Santa María
Valparaíso, Chile

***Resumen*—Dado los constantes cambios en la topología a la que diversas redes móviles se encuentran sujetas, en conjunto con los requerimientos de las aplicaciones, el desarrollo de algoritmos de enrutamiento eficientes es un gran desafío y se encuentran en constante evolución. En el presente informe se desarrolla un estudio de dos métodos representativos de entrutamiento en redes móviles ad-hoc: DSDV y AODV, así como de un acercamiento más moderno al problema en el que las rutas óptimas son obtenidas a través de optimización en base a colonias de hormigas. Se realizarán simulaciones para evaluar el desempeño de cada uno de los protocolos e identificar los escenarios con los cuales son compatibles, utilizando métricas para identificar la calidad del servicio entregada por estos.**

I. Introducción

ACTUALMENTE existe una gran cantidad de sistemas que requieren estar comunicados, y tanto las entidades que conforman dichos sistemas y sus aplicaciones pueden ser bastante diversos, lo que implica también una diversidad de las problemáticas y respectivas soluciones para cada caso. Particularmente, existen sistemas que requieren que las entidades que lo conforman mantengan comunicación entre ellas a pesar de que se encuentren en constante movimiento. Más aún, estos pueden necesitar actuar sin una infraestructura central. Redes vehiculares, enjambres de robots, o robots trabajando colaborativamente en zonas remotas son algunos ejemplos de aplicaciones que enfrentan estas circunstancias. Dichas redes son conocidas como Redes móviles ad-hoc, o MANETs por sus siglas en inglés.

Las MANETs representan un gran desafío en el enrutamiento de paquetes. En una MANET cada nodo que lo compone puede estar moviéndose a altas velocidades, y la topología del sistema cambia rápidamente. Aunque cada nodo puede actuar como intermediario y retransmitir paquetes, encontrar una ruta óptima entre fuente y destino no es fácil, e incluso puede dejar de existir en el tiempo que un paquete viaja entre nodos intermedios. Este problema se vuelve aún más agudo cuando se debe considerar limitaciones para los retardos, en la capacidad de procesamiento y en los costos energéticos.

El enrutamiento en MANETs es un problema que se ha gestado en conjunto con el crecimiento en la accesibilidad de los dispositivos móviles inalámbricos. Considerando la cantidad de personas con dispositivos móviles, si un grupo se encuentra en una misma habitación, resulta más simple y eficiente conectarlos directamente en vez de hacer uso de una mayor infraestructura. Por ello ya existen varios protocolos implementados para servir al enrutamiento en este tipo de redes, desarrollados ya desde los años 90. Pero dichos algoritmos presentan bastantes diferencias entre ellos, y no son adaptables a todos los escenarios, ni son escalables. Por ello es necesario caracterizar y evaluar el desempeño de los protocolos existentes.

En general, los protocolos de enrutamiento pueden separarse en activos y pasivos. Los primeros transmiten constantemente beacons para descubrir la topología de la red y actualizar las tablas de rutas. En el segundo caso, se busca la ruta en la medida que un nodo requiera establecer una conexión. Los dos tipos se diferencian principalmente por los métodos de control y el tiempo que demoran en descubrir las rutas. Dentro de los protocolos de enrutamiento de tipo pasivos, se encuentra DSDV, *Destination Sequenced Distance Vector*. En el caso de protocolos pasivos, AODV, *Ad-hoc On demand Distance Vector*, es uno de los más conocidos. Pero desde la implementación de estos protocolos ya han pasado varios años. Si bien cumplen los requerimientos básicos necesarios de una MANET, no son suficientemente buenos para cumplir los requisitos de las redes móviles modernas [1]. Por ello, utilizando desarrollos en inteligencia artificial, y particularmente en inteligencia de enjambres, es posible mejorar los protocolos existentes para dotarlos de una mayor autonomía. Especial atención generan los algoritmos inspirados en colonias de hormigas (ACO) por su factibilidad de uso en esquemas ad-hoc, ya que permiten evaluar otras características de la red al momento de generar una ruta de comunicación.

Las siguientes tres secciones de este documento se dedican a identificar las características principales de los protocolos de enrutamiento DSDV, AODV y aquellos basados en ACO, respectivamente. Para ello se presentará la evolución de dichos algoritmos y el funcionamiento general de cada uno. La sección V contendrá la metodología utilizada para realizar la simulación de una MANET utilizando los distintos protocolos de enrutamiento, con el fin de evaluar el desempeño de cada uno. Para ello se consideraran varias métricas como serían el overhead por control, packet delivery ratio, throughput, entre otros, orientadas a evaluar la calidad del servicio (QoS). Estos resultados son presentados en la sección VI. Finalmente, en la sección VII se presentan las conclusiones y principales desafíos que aún persisten en el enrutamiento en MANETs.

II. DSDV

El protocolo DSDV es de tipo activo, en el que cada nodo envía periódicamente su tabla de rutas, y a su vez mantiene una tabla de ruta actualizada a todos los destinos. El artículo donde se presenta fue publicado en 1994 [2] , por lo que es un protocolo que lleva bastante tiempo siendo utilizado. A continuación se presenta el funcionamiento general del protocolo.

DSDV está construido sobre el protocolo Distance Vector. En este último, cada nodo i mantiene un conjunto de distancias d_{ij}^x a cada nodo de destino x, a través de cada uno de sus vecinos directos j. Al recibir un paquete con destino x, el siguiente salto será al vecino j cuya distancia sea mínima. Cada nodo mantiene un costo estimado de cada conexión vecina, que es monitoreado constantemente. Además, periódicamente se realizan broadcasts del conjunto estimado d_{ij}^x a todos sus vecinos.

El principal problema del protocolo basado solo en vectores de distancia radica en que puede generar loops en el enrutamiento. Esto se debe a que la decisión del siguiente salto del paquete puede estar basada en información no actualizada, y por lo tanto, se intenta la comunicación sobre enlaces que ya no están disponibles. A modo de ejemplo, puede ocurrir que un nodo A esté enviando datos a un nodo C a través de B, si el nodo entre B y C se rompe, y B no notifica de esto al nodo A, este último continuará enviando datos a B, pero B reenviará datos a A para intentar encontrar a C. La mayoría de las soluciones a este problema proponen la implementación de un protocolo de coordinación, extra al enrutamiento. Pero esto es muy difícil de lograr en redes cuya topología cambia drásticamente, además de que provocaría una sobrecarga del sistema solo con el envío y transmisión de paquetes de control.

La principal característica de DSDV es la adición de un número de secuencia. Cada vez que un nodo realiza un broadcast de su tabla de rutas, en conjunto realiza un incremento del numero de su número de secuencia. En la recepción, los nodos vecinos y otros más lejanos pueden actualizar o descartar las entradas asociadas al nodo que generó la actualización dependiendo de si el número de secuencia asociado es mayor, menor, o con una métrica más pequeña. Cada vez que un nodo transmite una actualización debe enviar

- Su número de secuencia,
- dirección de destino para entradas en la tabla
- y el número de secuencia recibido desde el nodo que mantiene dicha ruta.

De esta manera, cada nodo puede decidir a cual nodo vecino reenviar un paquete. La ruta a elegir debe ser siempre la que tiene mayor número de secuencia de destino, representando la información más actual. Si dos rutas tienen igual número de secuencia, pues llegaron desde nodos vecinos diferentes, se elegirá aquella con menores saltos.

Ahora bien, para no acaparar el uso del medio inalámbrico, cada entidad en la red puede transmitir dos tipos de datos. El primero consiste en enviar la tabla completa actual del nodo. En el segundo caso solo se envían aquellas entradas que son nuevas, o han sido actualizadas. En caso de que la topología del sistema esté cambiando rápidamente, es mejor enviar la tabla completa, puesto que la cantidad de actualizaciones a las tablas de rutas son demasiadas. También se deben diferenciar aquellos cambios que son relevantes para ser enviados en el siguiente broadcast incremental, o que pueden ser descartados, o bien son importantes y se deben enviar inmediatamente. Un cambio en el número de secuencia de una conexión ya establecida no es relevante reenviarla, pero si es importante guardarlo, para mantener la secuencia actualizada. Cambios en el número de saltos se guardan para ser enviados en el siguiente broadcast. Un enlace roto o nuevo son cambios que se deben informar inmediatamente. Cabe notar que debido a esto cada nodo debe mantener dos tablas de rutas: la propia, y la que se envía a sus vecinos.

La detección de un enlace roto se puede realizar a través de la capa de enlace, o se puede inferir al no recibir actualizaciones del nodo vecino dentro de un determinado tiempo. Cuando este evento es detectado, se le asigna a la ruta en la tabla una métrica infinita, pues es imposible llegar al destino a través de ese enlace. Además, se incrementará el número de secuencia de destino, de manera que los nodos vecinos se vean obligados a actualizar la tabla de ruta. Por último, puede que un nodo vecino reciba una notificación desde el nodo que detecta el enlace roto, y otra notificación desde el destino a través de otra ruta, y ambas pueden contener el mismo número de secuencia. Para evitar esta colisión, se definen los incrementos realizados por un nodo destino como pares, y los incrementos asociados a una métrica infinita (de un enlace roto), como impares. Así, cualquier valor actual de una ruta real al destino sobrescribirá cualquier ruta con métrica infinita.

Con estas consideraciones, se garantiza que no existirán loops en la red, y que mientras existan enlaces en una vecindad, se asegura que estas estarán disponibles para nodos que no tienen conexión directa.

Si bien DSDV es un protocolo bastante simple ya que se basa en tablas de ruteo, tiene altos costos a nivel de memoria ya que debe mantener en la tabla rutas a todos los destinos en la red. Además la emisión periódica de mensajes de actualización sobrecarga los medios de transmisión, lo que puede generar problemas bastante agudos. Esto se traduce en una limitación en la escalabilidad del protocolo, por lo que no es útil para redes con una gran cantidad de nodos.

III. AODV

AODV es un protocolo basado en DSDV pero que genera búsquedas de rutas de comunicación solo cuando esta es necesaria [3]. Al establecer un esquema de comunicación bajo demanda se reducen los costos energéticos y el ancho de banda utilizado para la transmisión periódica de información. AODV cuenta con 3 características principales:

- Broadcast para solicitar ruta a destino solo bajo demanda.
- La conectividad a nivel local se maneja separadamente de la conectividad a nivel global.
- Cambios en la topología local, como nuevos vecinos o enlaces rotos son informados solo a nodos vecinos que mantengan una conexión activa.

III-A. Descubrimiento de rutas

Cuando un nodo fuente desea comunicarse con un nodo destino, no existe una dirección actual en la tabla de rutas para dirigir inmediatamente el paquete a un nodo vecino. Por ello se debe inicializar el proceso de búsqueda y establecimiento de una ruta. Esto se realiza en dos etapas: establecimiento de una ruta reversa a la fuente y establecimiento de la ruta desde la fuente al destino.

Cada vez que un nodo requiere comunicarse realiza un broadcast con un mensaje RREQ (Route Request). Cada RREQ contiene la dirección de fuente, un contador asociado al nodo fuente, similar a DSDV, el último número de secuencia del destino conocido por la fuente, y un contador de broadcasts que se incrementa en cada nuevo RREQ. Si uno de los vecinos conoce una ruta al destino, responde con un RREP (Route Replay), en otro caso, reenvía la solicitud a sus propios vecinos, añadiendo un contador de saltos (hops). Gracias al número de secuencia, un nodo puede recibir múltiples RREQ desde la misma fuente, y descartar aquellos que ya fueron respondidos. Cada nodo que retransmite un RREQ guarda información respecto de las direcciones fuente y destino, número de secuencia de fuente, tiempo de expiración de la solicitud, y número de secuencia del RREQ.

III-A1. Reverse path: El número de secuencia de la fuente es usado por los nodos intermedios para establecer la actualidad de la información de la ruta de retorno a la fuente. Además, el número de secuencia de destino (entregado por la fuente) es utilizado para decidir qué tan nueva debe ser la ruta al destino conocida por el nodo intermedio para que sea aceptada por la fuente. En la medida que el mensaje RREQ viaja por la red crea una ruta de regreso, ya que cada nodo intermedio guarda la dirección del nodo desde el que recibió el primer RREQ. Estas rutas se mantienen por un tiempo específico, suficiente para que el RREQ se propague por la red y generar una respuesta.

III-A2. Forward path: Cuando el RREQ llegue a un nodo que conozca la ruta al destino, este último comparara el número de secuencia del destino con el recibido en el RREQ. Si la ruta almacenada por el nodo posee un número de secuencia menor, no debe responder, si no que reenviara el RREQ. Cuando ocurra el caso contrario, el nodo responde con un RREP de vuelta a la fuente utilizando la ruta reversa ya establecida, con información del número de secuencia del destino, cuenta de saltos, y tiempo de mantención de la ruta. A medida que el RREP viaja, cada nodo establece una conexión con el nodo previo, reinicia el contador de *timeout* de esta, y actualiza la información de números de secuencia. Si un RREP no llega a un nodo que ya estableció una ruta reversa, la entrada expirará luego de pasado el tiempo *timeout*. Un nodo puede recibir múltiples RREP con misma fuente, este replica solo el primero, aquellos repetidos son reenviados solo en el caso de que contengan un número de saltos menor. De esta manera, la fuente inicia la comunicación apenas recibe un primer RREP, pero puede actualizar a una ruta mejor al recibir un RREP con un menor número de saltos.

En la figura 1 se visualiza el descubrimiento de rutas a través del broadcast entre nodos vecinos, y también el establecimiento de la comunicación directa entre fuente y destino. Se puede observar también que aquellos nodos que no se encuentran activos dentro de la ruta eliminaran la ruta reversa (a la fuente) una vez pasado el tiempo *timeout*.

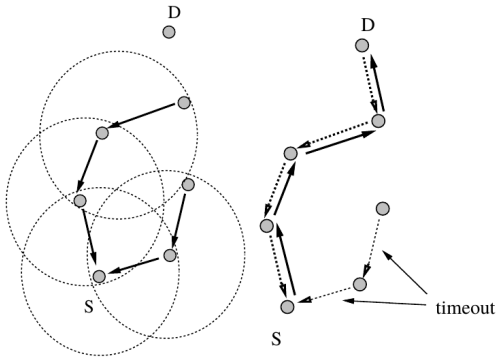

Figura 1. Izq: Descubrimiento de ruta reversa. Der: Establecimiento de la ruta de comunicación y timout de entradas en nodos que no pertenecen a la tabla de enrutamiento. Figura extraida de [3]

III-B. Manejo de la tabla de rutas

La tabla de rutas en cada nodo guarda información adicional respecto a los tiempo de expiración de las entradas. Si un nodo genera una ruta reversa a la fuente pero finalmente no se encuentra dentro del camino entre la fuente y destino, las entradas se eliminan luego de transcurrido dicho tiempo.

Se mantiene almacenado las direcciones de los nodos vecinos que se encuentran activos en una ruta, de manera que sea fácil notificarlos en caso de que algún enlace se rompa. Una ruta entre nodos se considera activa si se han recibidos paquetes desde esa dirección en el último intervalo de tiempo *active_timeout*.

III-C. Mantención de rutas

El protocolo también se encarga de mantener las rutas activas, o notificar a los nodos fuente en caso de que algún enlace ya no se encuentre disponible. Para lograr esto, cada nodo mantiene constante monitoreo de su vecindad. Si detecta un cambio en el estado de los enlaces mientras una ruta se encuentra activa, el nodo enviará a la fuente un mensaje RREP con un incremento en el número de secuencia destino y una métrica infinita (tal como en DSDV).

Cuando el nodo fuente recibe la información del enlace roto puede reiniciar la búsqueda de rutas. Incluso, si el nodo estima conveniente reconstruir completamente la ruta, puede incrementar el número de secuencia de destino conocido previamente, de tal manera que todos los nodos intermedios deban buscar rutas nuevas reenviando los RREQ.

III-D. Conectividad local

Para mantener actualizado el estado de los enlaces es esencial que cada nodo que forma parte de una ruta activa monitoree su vecindad. Para lograr esto, los nodos participantes envían regularmente mensajes broadcast *hello* (con TTL = 1), para notificar a sus vecinos de su presencia. Así, si un nodo

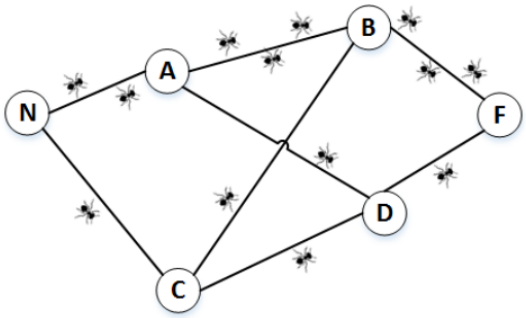

Figura 2. Hormigas virtuales recorriendo un grafo, reforzando la ruta superior. Figura tomada de [1]

activo deja de escuchar mensajes *hello* del nodo siguiente en la ruta, puede notificar a todos en la ruta que el enlace se rompió.

El protocolo AODV es relativamente antiguo (año 2000), pero logra solucionar bastantes problemas de DSDV. Aún cuando presenta un evidente mayor overhead para encontrar la ruta del primer paquete, en el largo plazo utiliza menos recursos y resulta más eficiente.

IV. Protocolos de enrutamiento con optimización basada en colonias de hormigas

El protocolo AODV tiene una implementación bastante simple, y además satisface los requerimientos básicos de enrutamientos en MANETs, por lo que es bastante popular. Pero esto no es suficientemente bueno para algunas aplicaciones modernas, con mayores restricciones en el consumo de energía, conectividad, seguridad, entre otras. Es por ello que el desarrollo de algoritmos de optimización basados en funciones objetivos ayudan en la búsqueda de rutas óptimas sujetas a mayores restricciones. Especialmente algoritmos meta-heurísticos permiten encontrar aproximaciones a soluciones óptimas de problemas complejos que no pueden ser resueltos en tiempo polinomial (NP-hard), sin aumentar demasiado el overhead de búsqueda. Dentro de los algoritmos meta-heurísticos uno de los más estudiado es el de colonias de hormigas, o ACO, y el cual genera especial interés para su uso en el enrutamiento en MANETs, pues se centra en encontrar rutas óptimas a problemas que pueden ser planteados como grafos, que es el caso directo en encontrar las mejores rutas entre nodos móviles.

IV-A. Algoritmo de colonia de hormigas

La meta heurística ACO se inspiró en el comportamiento de las hormigas en la naturaleza. Capaces de crear caminos para conseguir su alimento sin ninguna especie de organización centralizada. Las hormigas utilizan feromonas que dejan en su camino, que otras hormigas pueden seguir para poder encontrar el mismo camino. Si el camino lleva a una zona de alimento, más hormigas recorrerán el camino, añadiendo más feromonas. Similarmente, en un problema de optimización complejo existen varios caminos a seguir para llegar a una solución, pero en este caso, la permanencia de la feromona en el camino depende directamente de la calidad de la solución encontrada. Cada vez que el mismo camino a una solución sea recorrido habrán más feromonas, y por lo tanto, más hormigas virtuales lo transitarán.

En la práctica, el algoritmo ACO consiste en inicializar una hormiga en un nodo, eligiendo al principio al azar entre las posibles salidas del nodo. En cada nodo se realizará el mismo proceso de elección, hasta que se llegue al nodo solución deseado o se haya transitado por un número de nodos límite. Al llegar a la solución, la hormiga puede evaluar la calidad de la solución en base a diversos costos y actualizará un valor de feromona para los caminos que recorrió. La actualización puede aumentar o disminuir la cantidad de feromona. Como se ve en la figura 2, una mayor cantidad de hormigas caminan por la ruta superior, ya que representa una mejor solución y ya hay más hormigas recorriéndola.

Las feromonas, y otras heurísticas asociada a cada enlace (i.g distancia física), son utilizadas para modificar la probabilidad con que una hormiga elegirá una ruta u otra para salir de cada nodo. Similar a la naturaleza, en la medida que más hormigas transiten en un ruta, mayor será el refuerzo para que otras hormigas la recorran. Aquellas rutas que fueron recorridas pero no reforzadas, es decir, pocas o ninguna hormiga la transito, disminuirá el nivel de feromonas progresivamente hasta llegar a 0.

IV-B. ACO aplicado a enrutamiento de MANETs

La breve descripción del algoritmo ACO permite visualizar la similitud entre encontrar una solución óptima en un grafo a encontrar una ruta en una red. Es por ello que se han propuesto e implementado diversas variaciones de este algoritmo para enrutamiento en MANETs.

Uno de los protocolos más simples es ARA [4], el cual consiste en enviar hormigas (FANTs) desde cada nodo en búsqueda de distintos destinos. Cada FANT tiene asociada un número de secuencia, de manera de detectar aquellas repetidas. Si una FANT llega al destino, se reenvía una hormiga de regreso (BANT) utilizando la ruta reversa. En este caso particular, ARA mantiene conexiones reversas a través del envió de datos, para no generar mayor overhead a causa de las hormigas producidas periodicamente.

Existe una gran diversidad de algoritmos basados en ACO. Algunos utilizan un esquema de envío de hormigas periódicamente, otros solo bajo demanda, o métodos híbridos [1]. Pero hasta aquí, pareciera ser una combinación de DSDV y AODV, utilizando métodos más complejos y con otras heurísticas definidas para cada enlace. La principal ventaja de utilizar ACO para el enrutamiento es la posibilidad de añadir otras heurísticas al descubrimiento de las rutas, de manera de lograr un objetivo de optimización mayor, más que tan solo encontrar una ruta con menos salto al destino. Algunos de estas optimizaciones que son de interés pueden ser:

- Calidad de servicio: Parámetros como el retardo, saltos y otros pueden ser considerados al momento de evaluar

la calidad de la solución para actualizar los valores de feromonas. Así se buscan caminos que optimicen la calidad del servicio.
- Consumo de energía: En general, nodos que se encuentran en rutas más cortas tienen un mayor consumo energético. Por ello, un objetivo puede ser balancear este consumo al elegir rutas más largas que eviten nodos con bajo nivel de batería. Nuevamente, las hormigas pueden tomar en consideración esto al momento de actualizar los niveles de feromonas.
- Posición global de los nodos: Muchos dispositivos cuentan con información de su posición global (GPS, GLONASS, etc.), la cual puede ser utilizada para elegir caminos predeterminados para las hormigas según distintas zonas. De esta manera se reduce la cantidad de hormigas a enviar, y se puede recudir la búsqueda de rutas a una zona en particular.
- Búsqueda de rutas seguras: A través de distintos mecanismos de cyber seguridad, se puede evaluar el nivel de confiabilidad de un determinado enlace. Esta métrica permite asociar un nivel de calidad a determinadas rutas.

Además, en un algoritmo ACO pueden considerarse más de uno de estos elementos. Lo que permite generar una gran adaptabilidad a los distintos sistemas que lo requieran. Aunque esto último aún es un tema de investigación [1].

En general, los protocolos que consideran el algoritmo ACO en sus implementaciones pueden conseguir mejores desempeño a restricciones más realitas de aplicaciones en redes móviles. Son una opción que añade una capa de complejidad al sistema de enrutamiento, pero el overhead causado por ello no necesariamente implica una gran reducción en el desempeño general al compararlos con los beneficios de sus alcances.

V. Simulación

Existe bastante diversidad en las propuestas para enfrentar la problemática del enrutamiento en MANETs, por ello es importante notar, ante todo, las diferencias escenciales enunciadas en las secciones anteriores entre los diversos métodos. La explicación sobre el papel permite intuir la dirección en que cada uno de los métodos es ventajoso respecto a los otros, o también identificar sus principales falencias. En la práctica, el desempeño de cada uno de estos dependerá de la dinámica del sistema, el número de nodos y la cantidad de transmisiones realizadas. Por ello, resulta necesario evaluar en que escenarios es necesario implementar algun algoritmo de alta complejidad ACO o bastaría con DSDV. Para realizar la evaluación se utilizará la biblioteca NS3, simulando varios escenarios.

V-A. NS3

Dado el interés en evaluar y probar protocolos en redes por parte de investigadores, es que se pueden encontrar herramientas disponibles para simular redes. NS3 es una de ellas, y será la que ayudará a evaluar el desempeño de las redes.

NS3 es una biblioteca de C++ y Python, que haciendo uso de la programación orientada a objetos permite implementar protocolos, generar topologías de red, para varios tipos de interfaces. Además, para simular MANETs, permite la definición de la cinemática de los nodos. Para lograr esto, cuenta con cuatro componentes escenciales:

- **Nodos**: Componentes principales de la red, en una red con infraestructura corresponden a hosts, pero en MANETs, la palabra nodo es más adecuada.
- **Interfaces**: Una interfaz corresponde al tipo de conexión física entre los nodos, como ethernet o wifi.
- **Canal**: Un canal establece una conexión entre dos interfaces compatibles, y define la capacidad, ruido, etc.
- **Aplicación**: Con el fin de probar la capacidad de la red, se pueden correr aplicaciones para generar flujo de datos entre los nodos.

Con estos elementos, es posible correr simulaciones simples, o tan complejas como la capacidad del computador nos permita. Además, con la capacidad de C++, es posible heredar clases y así modificar algoritmos existentes o implementar los propios. NS3 permite registrar los eventos ocurridos a lo largo de la simulación, generando archivos que pueden ser analizados con Wireshark u otras herramientas.

V-B. Metodología

Para evaluar el desempeño de cada protocolo se medirán características de la calidad de servicio (QoS) de cada uno:

- Throughput: Cantidad de paquetes transmitidos entre todos los nodos por instante de tiempo.
- Goodput: Cantidad de paquetes con información relevante para las aplicaciones transmitidos entre todos los nodos, por cada instante de tiempo.
- Overhead: De la totalidad de paquetes transmitidos, cuantos de ellos, en porcentaje, corresponden a paquetes utilizados para el control y redireccionamiento de paquetes.
- Retardo: Se medirá el retardo promedio entre la transmisión desde la fuente hasta su recepción en el destino.

Las simulaciones se realizaran dentro de una región de 500×500 metros, con los nodos móviendose en una caminata aleatoria dentro de esta, y cada vez que llegan a un lugar fijo, esperan 1 segundo para moverse al siguiente lugar. En todas las pruebas habrán 15 pares de nodos con comunicación establecida.

La primera prueba será con 50 nodos, moviéndose a 20[m/s], solo modificando el protocolo. Luego, para cada protocolo, se harán pruebas variando el número de nodos, y manteniendo una velocidad fija de 20[m/s], y otra variando la velocidad de los nodos, manteniendo fijo el número de nodos en 50.

VI. Resultados

A continuación se mostrarán los principales resultados obtenidos de las simulaciones, lamentablemente no se alcanzó a realizar la prueba con variación en la mobilidad, ni la simulación de algoritmos ACO, dada la complejidad de su implementación en NS3. Los resultados mostrados son para DSDV y AODV.

VI-A. *Prueba simple*

En la tabla I se observan los resultados para la primera prueba con 50 nodos moviéndose a 20[m/s] para 15 pares de nodos transmitiendo datos. Se puede observar que en este caso existe una diferencia leve entre ambos protocolos en los que respecta a througput, y bastante diferencia en cuanto al delay.

Protocol	Delay [ms]	Throughput	Goodput	Overhead
AODV	330	557	114	0.88
DSDV	17,1	421	105	0.91

Tabla I
RESULTADOS PARA PRIMERA PRUEBA SIMPLE

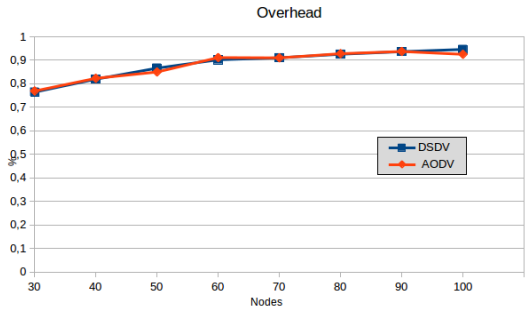

Figura 5. Overhead para AODV y DSDV variando la cantidad de nodos.

VI-B. *Pruba con variación en el número de nodos*

En las figuras 3 a 5 se observan gráficos con los resultados para cada una de las características de QoS, exceptuando el delay, pues este estará directamente relacionado al número de saltos que da un paquete, y no estrictamente relacionado a la cantidad de nodos presentes en el sistema. Es bastante evidente la similitud entre ambos protocolos para la cantidad de nodos simulados, inclusive, en algunos casos DSDV es ligeramente mejor.

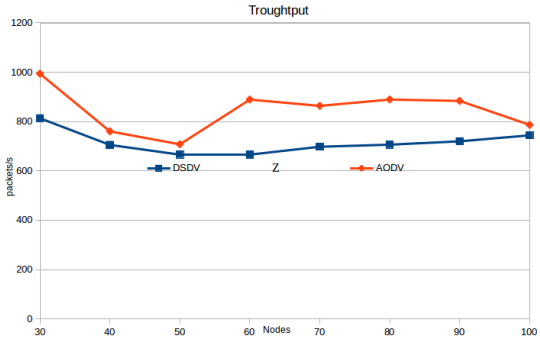

Figura 3. Throuhput para AODV y DSDV variando la cantidad de nodos.

Figura 4. Goodput para AODV y DSDV variando la cantidad de nodos.

VII. CONCLUSIÓN

Se presentaron las bases lógicas detrás de los algoritmos escenciales en el estudio de enrutamiento en MANETs, entendiendo sus funcionalidades y limitaciones. Comprendiendo las diferencias entre los protocolos es posible deducir cuales son las ventajas de unos versus otros, las que pueden ser listadas:

- DSDV, al ser activo, consume una mayor cantidad de energía. También genera un mayor flujo de paquetes de control, dado que realiza broadcast de las actualizaciones constantemente. Esto último traerá mayores problemas si el sistema tiene un alta cantidad de nodos o es altamente dinámico.
- AODV consume una menor cantidad de energía y no genera una saturación de la red, ya que solo envia paquetes de control bajo demanda o de manera local. Por esto mismo el protocolo presenta un mayor retardo en la conexión, pues necesita solicitar y establecer la ruta cada vez que se quiere establecer la conexión.
- Algoritmos basado en ACO presentan mayor complejidad en la implementación, y existe una gran variedad para cumplir distintos objetivos. Esta versatilidad hace que sean protocolos de interés, ya que permiten tomar en consideración diversas caracteristicas del sistema, como energía, seguridad, localización u calidad del servicio. Si bien muchas de estas características pueden añadirse como parte de la métrica en AODV o DSDV, no resulta fácil realizar la evaluación de todo el camino al destino como en ACO, ya que solo se mantienen en tabla direcciones asociadas al destino y al siguiente salto para llegar a dicho destino.

A pesar de las conclusiones lógicas que se pueden extraer de la descripción de los protocolos, a través de las simulaciones es posible concluir que la diferencia muchas veces no es tan significativa como para justificar una complejización de soluciones ya existentes. Si bien AODV resulta teóricamente más viable para un mayor número de nodos, al menos hasta 100 nodos tiene un desempeño similar a DSDV, lo que radica la pregunta ¿Cuántas aplicaciones requieren más de 100 nodos móviles?.

Finalmente, la decisión de utilizar uno u otro protocolo dependerá de la complejidad en la topología de la red (velocidad, número de nodos, número de conexiones) y la eficiencia con la que se quiera realizar la comunicación. Algoritmos ya implementados y disponibles alrededor del internet permiten, hoy en día, una solución simple y limpia para sistemas de baja embergadura, como lo son la mayoría hoy en día. Para sistemas más complejos, con dinámicas más rápidas, nodos

más lejanos y en ambientes más propensos a errores, aún es un desafío el enrutamiento y sigue siendo un tema de bastante investigación y en el que aún se espera lograr avances, en cuanto a desempeño, robustez y eficiencia.

REFERENCIAS

[1] H. Zhang, X. Wang, P. Memarmoshrefi, and D. Hogrefe, "A survey of ant colony optimization based routing protocols for mobile ad hoc networks," *IEEE Access*, vol. PP, no. 99, pp. 1–1, October 2017.

[2] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," *SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 234–244, Oct. 1994. [Online]. Available: http://doi.acm.org/10.1145/190809.190336

[3] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA '99. Second IEEE Workshop on*, Feb 1999, pp. 90–100.

[4] M. Gunes, U. Sorges, and I. Bouazizi, "Ara-the ant-colony based routing algorithm for manets," in *Proceedings. International Conference on Parallel Processing Workshop*, 2002, pp. 79–85.