VNIVERSIDAD NACIONAL AVTÑOMA DE MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

RECOLECCIÓN DE DATOS USANDO FILTROS DE BLOOM EN REDES DE SENSORES INALÁMBRICOS

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA DE LA COMPUTACIÓN

P R E S E N T A:

ANTONIO DE JESÚS GARCÍA DOMÍNGUEZ

DIRECTOR DE LA TESIS:
DR. JAVIER GÓMEZ CASTELLANOS

MÉXICO, D.F. **2010.**

Agradecimientos

Muchas personas me han apoyado y ayudado a lo largo de toda mi vida, todos los días aprendo cosas nuevas de esas personas que están cerca de mí, quisiera agradecer a mis amigos y familia por estar siempre conmigo y ayudarme a hacer posible esto. Además quiero también agradecer a mis maestros, los cuales al compartir sus conocimientos me ayudaron enormemente, especialmente al Dr. Javier Gómez Castellanos por la dirección de esta tesis.

Agradezco también al Conacyt por la beca brindada durante mis estudios.

De igual manera, quiero agradecer a los proyectos PAPIIT y IN106609, así como al proyecto Conacyt 105117.

Índice.

Índice de figuras.

Resumen

Las redes de sensores inalámbricos son cada vez más utilizadas en muchos lugares de nuestra vida diaria. Se han desarrollado aplicaciones en diferentes áreas de estudio que ponen en práctica esta tecnología y mediante las cuales es posible monitorear cualquier fenómeno de interés. Conforme van surgiendo nuevos avances en la tecnología, se siguen desarrollando nuevas aplicaciones para este tipo de redes, las cuales logran ofrecer buenos resultados.

Este tipo de redes cuentan con características especiales que las hacen diferentes a otras, una de estas características y la cual se considera crucial es la energía. Dentro de una red de sensores inalámbricos la energía es un recurso muy importante ya que se encuentra limitada pues ésta es provista por las baterías con las que cuente el nodo. Los nodos en este tipo de redes consumen energía al transmitir datos, al estar en estado de escucha (*listening*) o simplemente al estar en estado ocioso (*idle*). Debido a que la energía dentro de este tipo de redes es limitada, deben crearse mecanismos que ayuden a su optimización y reducción de consumo de una u otra manera, todo esto para lograr un mayor tiempo de vida de las baterías de los nodos de la red.

Como se mencionó anteriormente, uno de los procesos en los cuales se consume energía en los nodos de la red es la transmisión de datos. En este trabajo se presenta una revisión de las diferentes categorías de protocolos de enrutamiento existentes para este tipo de redes, para después hacer un análisis más a detalle de dos de ellos: *Directed Diffusion* y *Cluster Based Routing Protocol*.

La principal aportación de este trabajo es la presentación de un modelo de recolección de datos que hace uso de filtros de Bloom, con el objetivo de reducir el número de transmisiones y *bytes* transmitidos durante ese proceso. Al reducir el número de transmisiones y número de *bytes* transmitidos se estaría reduciendo por lo tanto la cantidad de energía consumida durante el proceso de recolección de datos dentro de este tipo de redes. La reducción en el consumo de energía por el modelo propuesto se comparará con los protocolos analizados que se mencionan anteriormente, para de esta manera poder observar los beneficios logrados.

Mediante la reducción en el consumo de energía en los nodos de la red durante el proceso de recolección de datos, el tiempo de vida de las baterías de los nodos aumenta, con todos los beneficios que esto trae. Se analizará también un ejemplo de consumo de energía en un escenario de pruebas para poder visualizar las ventajas y beneficios obtenidos.

Capítulo 1 Introducción.

En los últimos años, el desarrollo tecnológico ha hecho posible la aparición de nuevos conceptos y aplicaciones en todas las áreas de estudio. Una de estas áreas que se ha visto favorecida gracias a los recientes avances tecnológicos es la de las comunicaciones inalámbricas. Hace algunos años, las comunicaciones se basaban en redes y dispositivos alámbricos, en donde se necesitaba que los dispositivos estuvieran conectados físicamente entre sí para poder interactuar. Esto hacía que las redes formadas por dichos dispositivos tuvieran ciertas limitaciones. Sin embargo, hoy en día es común poder ver dispositivos que interactúan entre sí a través de una comunicación inalámbrica en casas, oficinas, escuelas, etc. El uso de computadoras o dispositivos inalámbricos continúa en aumento conforme siguen surgiendo nuevos avances tecnológicos, facilitando muchas de nuestras actividades cotidianas.

Dentro de las comunicaciones inalámbricas existe un tipo especial de redes llamadas redes de sensores inalámbricos (*WSN*, *Wireless Sensor Networks*), de las cuales aquí se presenta una introducción. Estas redes pueden estar formadas por pocos o por una gran cantidad de nodos (dependiendo de la aplicación para la que hayan sido diseñadas). Los nodos de este tipo de redes cuentan con capacidades de sensado, por ejemplo, temperatura, presión, humedad, etc. Mediante la implementación de este tipo de redes es posible monitorear eventos y fenómenos de interés, todo esto gracias a los beneficios que brindan las comunicaciones inalámbricas.

Algunas de las características de las redes de sensores inalámbricos son diferentes a las presentes en otros tipos de redes. El ejemplo principal de esto es la energía, ya que los nodos que conforman la *WSN* cuentan con una cantidad de energía limitada, normalmente provista por una batería. Los diseños que se hagan de este tipo de redes deben buscar la optimización en el consumo de energía, pues de esto dependerá en gran medida el tiempo de vida de las baterías de los nodos de la red.

Conforme su uso y popularidad crece, este tipo de redes han sido implementadas en diversas áreas donde han mostrado buenos resultados. Por mencionar algunos casos, se han desarrollado aplicaciones para la agricultura, industria, medicina, meteorología, etc. El uso de las redes de sensores inalámbricos ha estado en constante crecimiento y evolución debido a que se ha visto beneficiado por los avances tecnológicos de los últimos años. Por este motivo, es probable que este tipo de redes sigan siendo utilizadas cada vez en un mayor número de áreas de aplicación, manteniéndose en constante desarrollo y perfeccionamiento.

1.1 Definición del problema.

Una de las características más importantes de las redes de sensores inalámbricos es la energía, ya que en este tipo de redes la cantidad de energía con la que cuentan los nodos es limitada, pues normalmente se encuentra provista por unas pequeñas baterías ubicadas en el nodo. En otros tipos de redes, como en las redes *LAN* (*Local Area Network*) por ejemplo, los dispositivos se encuentran conectados directamente a la alimentación de energía, lo que les permite funcionar sin tomar en cuenta su consumo.

Debido a que las redes de sensores inalámbricos cuentan con la limitante de la energía, éstas deben de funcionar de manera que optimicen ese recurso para de esta manera alargar el tiempo de vida de las baterías de los nodos de la red. Si no se pusiera especial atención en la optimización de energía dentro de este tipo de redes, los nodos, al transmitir y recibir datos, se quedarían rápidamente sin energía, lo que provocaría que fuera necesario reemplazar las baterías continuamente. El reemplazo continuo de las baterías de los nodos puede ser algo problemático, especialmente cuando las redes cuentan con un gran número de nodos o se encuentran ubicados en zonas geográficas de difícil acceso.

El mayor consumo de energía en los nodos se presenta cuando éstos transmiten información, es por eso que es necesario buscar mecanismos que permitan realizar las tareas de sensado, recolección, transmisión y procesamiento de la información con el menor consumo posible de energía. Para la optimización de energía dentro de las redes de sensores inalámbricos se han creado protocolos de comunicación especiales que toman en cuenta todas las características de este tipo de redes.

Para lograr un bajo consumo de energía dentro de las redes de sensores inalámbricos, los protocolos de comunicación que éstas utilizan implementan diversas técnicas como disminuir el tiempo en el cual los nodos están activos. Al disminuir la cantidad de veces en las cuales los nodos entran en actividad, éstos consumirán una menor cantidad de energía y, por lo tanto, su batería tendrá un mayor tiempo de vida. Así como ésta, deben de buscarse más técnicas y mecanismos que permitan lograr la optimización de la energía en este tipo de redes. Esto traerá enormes beneficios a la red, ya que la energía es la principal limitante con la que cuentan las redes de sensores inalámbricos.

1.2 Objetivos.

En este trabajo se realizará un análisis de los protocolos de recolección de información en redes de sensores inalámbricos, especialmente *Directed Diffusion* y *Cluster Based Routing Protocol*, enfocándose en los conceptos de optimización del número de transmisiones y por lo tanto disminución del consumo de energía en la red. Después de realizar el análisis anteriormente mencionado, se buscará optimizar el proceso de recolección de datos por medio de la utilización de filtros de Bloom.

Una vez descrito el modelo que en este trabajo se propone, éste se comparará con los protocolos que se mencionan en la parte superior, para de esta manera determinar los beneficios que se obtienen en cuanto a número de transmisiones y consumo de energía en la red se refiere. De esta manera se mostrará por medio de gráficas comparativas qué tanto disminuyó el número de transmisiones en la red cuando se utiliza el modelo propuesto. También se mostrará cómo es que esto influye en el consumo de energía de la red y, por lo tanto, en el tiempo de vida de las baterías de los nodos.

Para lograr los objetivos que en esta sección se describen, se realizarán distintas pruebas en un escenario elaborado en el simulador *NS-2*. Se estudiará el proceso de recolección de información de los protocolos anteriormente mencionados y el del modelo propuesto, para de esta manera obtener los resultados que permitan establecer los beneficios logrados.

Es importante mencionar que, dentro de las redes de sensores inalámbricos, no toda la energía de los nodos es utilizada en transmitir y recibir datos, sino que ésta también se consume cuando los nodos se encuentran en estado ocioso (*idle*) y en estado de escucha (*listening*). En este trabajó se abordará la optimización de la energía utilizada sólo en transmisiones de datos, dejando a trabajos futuros posibles optimizaciones de energía en otros procesos.

Cabe aclarar que durante todo este trabajo se hablará sobre redes de sensores inalámbricos estáticos, ya que también existen redes con nodos móviles, pero éstas cuentan con otras características las cuales no se abordan en el presente trabajo.

1.3 Estructura de la tesis.

Este trabajo se divide en 6 capítulos, dentro de los cuales, en cada uno de ellos, se describe lo siguiente:

En este capítulo introductorio se realiza un repaso de los conceptos principales en las redes de sensores inalámbricos, mencionando su forma de funcionamiento, características y aplicaciones. Además se presentan los principales tipos de protocolos de recolección de datos existentes, para después profundizar en dos de ellos: *Directed Diffusion* y *Cluster Based Routing Protocol*.

En el capítulo 2 se presentan los filtros de Bloom, estructuras utilizadas por el modelo propuesto en este trabajo para la recolección de información en la red. Aquí se presenta su definición, teoría y aplicaciones dentro del área de las redes.

En el capítulo 3 se define el modelo propuesto para la recolección de información en las redes de sensores inalámbricos. En este capítulo se describe su forma de funcionamiento y la manera en la que se introducen los filtros de Bloom.

En el capítulo 4 se muestra la forma en que se realizó el modelo de simulación en *NS-2* para implementar el modelo propuesto en este trabajo y realizar las comparaciones con los protocolos que sirvieron de base.

En el capítulo 5 se presentan las pruebas que fueron realizadas para analizar el proceso de recolección de información en la red, además se presentan los resultados obtenidos y la explicación de ellos.

En el capítulo 6 se presentan las conclusiones obtenidas de la realización de este trabajo de investigación y algunas de las posibles opciones que se tienen como trabajos futuros para continuar con este tema.

1.4 Redes de sensores Inalámbricos.

Una red de sensores inalámbricos, como se describe en [1], es una red formada de varios dispositivos autónomos, los cuales cuentan con sensores de diversos tipos, dependiendo de la aplicación para la cual haya sido diseñada la red. Dentro de la misma, los nodos se encuentran distribuidos de forma tal que puedan monitorear los eventos de interés para la aplicación de la red.

En este tipo de redes, las comunicaciones entre los nodos se llevan a cabo de manera inalámbrica, por lo que cada uno de los nodos cuenta con su respectivo módulo de comunicación inalámbrica.

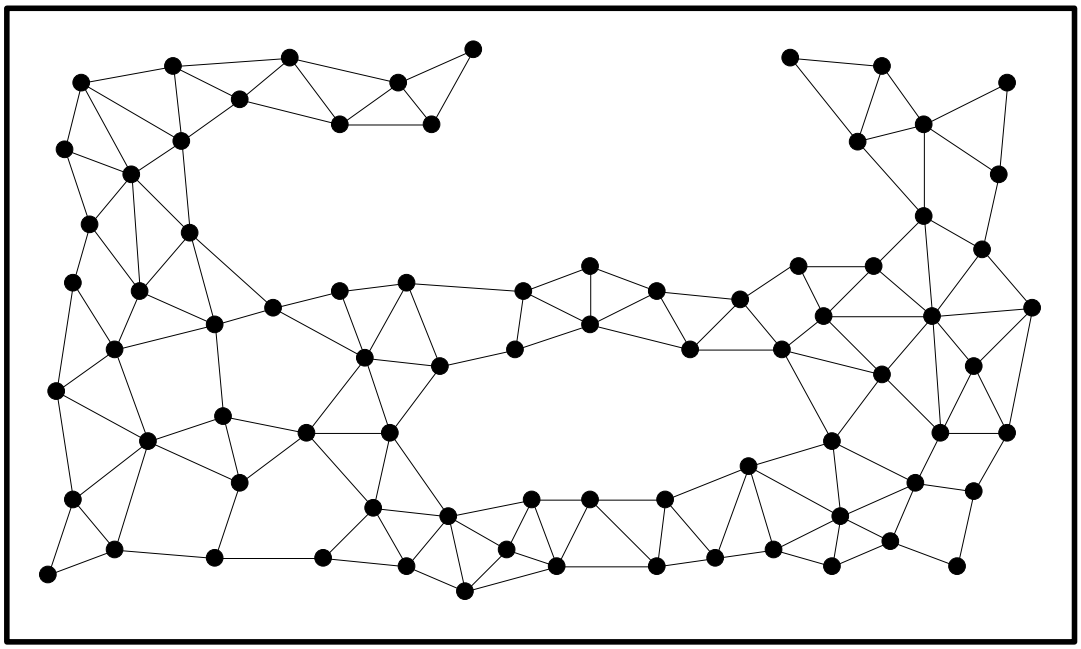


Figura 1.1 Ejemplo de una red de sensores inalámbricos.

En la Figura 1.1 se puede observar un ejemplo de una red de sensores inalámbricos, en donde cada punto representa a un nodo sensor y las líneas que los unen representan la conectividad que existe entre nodos vecinos de la red. Las redes de sensores inalámbricos son implementadas en una región en particular para cumplir las siguientes metas:

- Determinar el valor de variables físicas en algún lugar predefinido.
- Detectar la ocurrencia de eventos de interés y estimar sus parámetros.
- Clasificar un objeto detectado.
- Rastrear objetos.
- Otros.

1.4.1 Nodos sensores.

Los componentes principales de una red de sensores inalámbricos son los nodos [2], los cuales pueden contener uno o más sensores (térmico, visual, sísmico, de presión, etc.). Las principales características de los nodos sensores son su peso ligero, bajo precio y cantidad de energía limitada. La cantidad de energía disponible es la mayor limitación en este tipo de nodos. Cada nodo está compuesto de los siguientes cuatro subsistemas [2]:

- Subsistema del sensor.
- Subsistema de procesado.
- Subsistema de comunicación.
- Fuente de energía.

En algunos nodos, el sensor o subsistema de sensado viene montado directamente en la placa principal del nodo. En otros, este sensor puede ser añadido conectándolo a un puerto de expansión presente en el nodo, dicho puerto es compatible con una gran cantidad de tipos de sensores. Lo anteriormente mencionado dependerá del fabricante y del tipo de nodo con el que se esté trabajando.

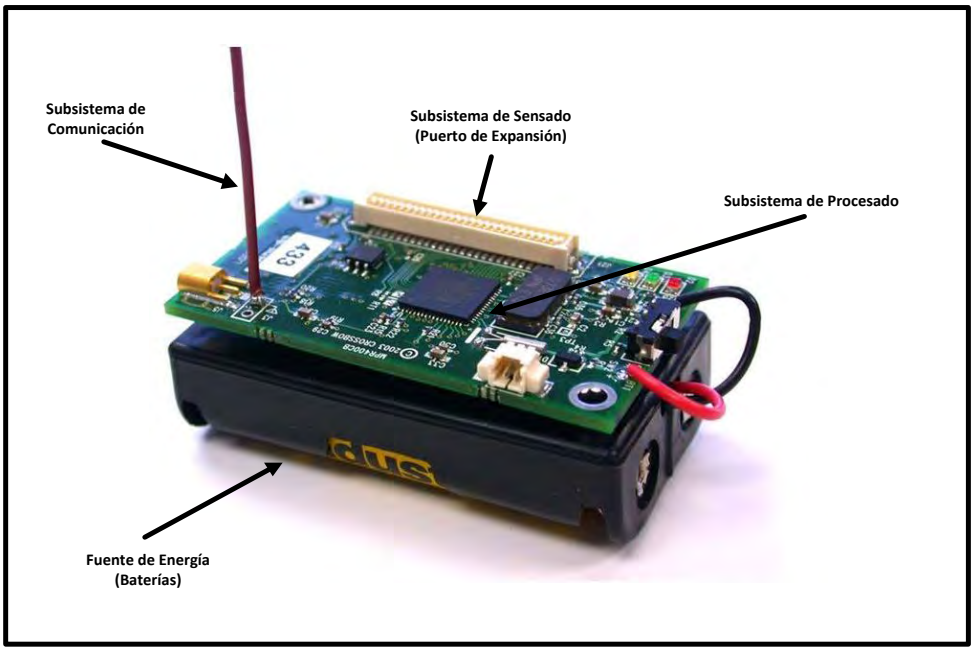


Figura 1.2 Nodo sensor *MICA2* de *Crossbow Technology Inc.* [3].

En la Figura 1.2 se muestra un ejemplo de un nodo sensor, el *MICA2*, en el cual podemos observar las partes de las que se compone, mencionadas en párrafos anteriores.

1.4.2 Retos de las redes de sensores inalámbricos.

Cuando se implementa una red de sensores inalámbricos hay varios retos que involucra este proceso, los cuales son determinantes para el buen funcionamiento de la misma. Entre los más importantes se encuentran los siguientes [1]:

- Topología.
- Diseminación y recolección de información.
- Consumo de energía.
- Actividades de sensado esporádicas.

Es importante que se cuente con un buen modelo de recolección de la información dentro de la red, ya que de ello dependen otros parámetros como el consumo de energía, concepto que es crucial para el tiempo de vida de las baterías de los nodos de la red.

1.4.3 Aplicaciones de la redes de sensores inalámbricos.

Aunque inicialmente las redes de sensores inalámbricos fueron pensadas exclusivamente para usos militares, hoy en día es común encontrarlas en un gran número de áreas de aplicación, objetos y lugares que usamos cotidianamente, como se describe en [4]. Gracias a ellas es posible monitorear y controlar un gran número de parámetros. El incremento de su popularidad se debe a los buenos resultados que han alcanzado y al avance de la tecnología que ha hecho posible que las redes de sensores inalámbricos se encuentren cada vez con mayor frecuencia presentes en nuestra vida diaria. La Figura 1.3 muestra un ejemplo de la aplicación de este tipo de redes en la agricultura:

Figura 1.3 Ejemplo de una aplicación de las redes de sensores inalámbricos [5].

1.5 Protocolos de enrutamiento en las redes de sensores inalámbricos.

Dentro de las redes inalámbricas, las redes de sensores inalámbricos son una categoría especial, es por ese motivo que algunos protocolos de enrutamiento han sido diseñados específicamente para ellas. Dentro de una red de este tipo, el enrutamiento es necesario para que los nodos envíen de manera adecuada la información recolectada por sus sensores hacia un nodo principal o estación base. Este nodo es el encargado de, inicialmente, realizar las consultas a la red, y posteriormente procesar los datos cuando los nodos sensores le envíen la información.

En las redes de sensores inalámbricos se cuenta con características especiales y limitaciones, como por ejemplo, la energía, el procesamiento y el almacenamiento, por lo cual se requiere un manejo

óptimo de estos recursos. La mayoría de los protocolos de enrutamiento existentes en la actualidad, como se muestra en [6], pueden ser clasificados de acuerdo a la estructura de la red como:

- Planos.
- Jerárquicos.
- Basados en la posición.

Existen otras clasificaciones que se han realizado, basándose en la operación del protocolo, éstas se estudian más adelante en el presente capítulo.

1.5.1 Protocolos de enrutamiento.

La Figura 1.4 muestra la clasificación de los protocolos de enrutamiento existentes:

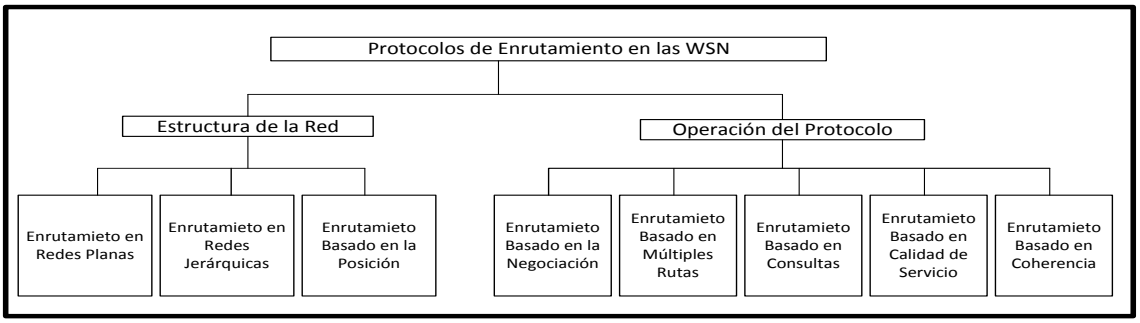


Figura 1.4 Clasificación de los protocolos de enrutamiento en las redes de sensores inalámbricos [6].

Enrutamiento en redes planas.

En este tipo de redes todos los nodos sensores son del mismo tipo o al menos con las mismas características. Las tareas de sensado se llevan a cabo con una colaboración por igual por parte de los nodos. Los nodos de este tipo de redes no tienen asignado un identificador global debido a la gran cantidad de ellos dentro de la red [6]. Un ejemplo de esto es la familia de protocolos *SPIN* (*Sensor Protocols for Information via Negotiation*), descrita en [7] y [8].

Enrutamiento en redes jerárquicas.

La idea de jerarquizar las redes de sensores inalámbricos surgió con la necesidad de optimizar el consumo de energía. Al tener clasificados los nodos de manera jerárquica conforme a su reserva de energía se les pueden asignar tareas diferentes a cada uno, por ejemplo, tareas de procesamiento y transmisión a quienes cuenten con mayor reserva de energía, y tareas de sensado a aquellos nodos que tengan menos energía disponible. Esta idea puede ayudar a la conservación de energía en la red y por lo tanto a aumentar el tiempo de vida de las baterías de los nodos [6]. Un ejemplo de esto es el protocolo *LEACH* (*Low Energy Adaptive Clustering Hierarchy*), descrito en [7] y [8].

Enrutamiento basado en la posición.

Este tipo de protocolos toman en cuenta la posición de los nodos así como la distancia entre nodos vecinos, para estimar sus coordenadas intercambian información. En algunos casos, dependiendo de la red y de la aplicación, las coordenadas pueden ser provistas directamente a los nodos por medio

de un satélite. Como en todos los tipos de protocolos, el objetivo principal es ahorrar energía, es por eso que se pone un periodo de *dormir (sleep)* a tantos nodos como sea posible, cuando no haya actividad en ciertas regiones de la red [6]. En [9] se describe el protocolo *GAF* (*Geographic Adaptive Fidelity*), el cual es un ejemplo de este tipo de protocolos.

Enrutamiento basado en múltiples rutas.

Este tipo de protocolos realiza el envío de datos dentro de la red por medio de múltiples rutas para de esta manera mejorar el rendimiento de la red. El mantenimiento de estas rutas trae como consecuencia un incremento en el consumo de energía y en la cantidad de tráfico generado, ya que se transmiten mensajes periódicamente para dicho objetivo [6]. Un ejemplo del uso de múltiples rutas dentro de una red se describe en [10].

Enrutamiento basado en consultas.

En este tipo de protocolos, la consulta o tarea de sensado es generada por un nodo de la red que actúa como estación base. Cada uno de los nodos que cumplan con los parámetros solicitados en dicha consulta envía los datos hacia la estación base. Un ejemplo de este tipo de protocolo es *Directed Diffusion* [6].

Enrutamiento basado en la negociación.

Este tipo de protocolos realiza un tipo de negociación entre los nodos antes de que la comunicación comience. De esta manera se previenen datos redundantes e información duplicada en la red que pudiera presentarse cuando se realiza la transmisión de datos en forma de *inundación (flooding)* [6]. Un ejemplo de esto es la familia de protocolos *SPIN* (*Sensor Protocols for Information via Negotiation*), descrita en [7] y [8].

Enrutamiento basado en calidad de servicio.

Dentro de las redes de sensores inalámbricos la principal limitación que se tiene es la cantidad de energía disponible en los nodos sensores. Este tipo de protocolos trata de establecer un balance entre el consumo de energía y la calidad de los datos transmitidos [6]. Un ejemplo de esto es el protocolo *SAR* (*Sequential Assignment Routing*), propuesto en [11].

Enrutamiento basado en coherencia.

Este tipo de protocolos toma en cuenta el procesamiento de los datos que se transmiten dentro de la red. Existen dos enfoques: el procesamiento coherente y el no coherente. En el enrutamiento basado en el procesamiento no coherente, antes de transmitir los datos, éstos son procesados localmente por los nodos. Esto tiene como consecuencia un aumento en el consumo de energía, pues ésta es utilizada también por las tareas independientes de procesamiento. En el enrutamiento basado en el procesamiento coherente sucede lo contrario, los datos son enviados sin procesar hacia la estación base. Esto trae considerables ahorros de energía pues solamente se realizan transmisiones y no procesamiento de datos por parte de los nodos [6]. Un ejemplo de esto es el protocolo *SAR* (*Sequential Assignment Routing*), propuesto en [11].

Algunos de los protocolos de ejemplo mencionados anteriormente pueden estar dentro de varias categorías, éstos son llamados protocolos híbridos.

1.5.2 *Directed Diffusion*.

Directed Diffusion es un protocolo de enrutamiento para redes de sensores inalámbricos en donde los datos se representan por medio de pares atributo-valor, como se muestra en [12]. Este protocolo se encuentra dentro de la categoría de protocolos basados en consultas, aunque también forma parte de la categoría de enrutamiento en redes planas. El funcionamiento principal de este protocolo es el siguiente: la estación base propaga la tarea de sensado a la red por medio de mensajes llamados *intereses*. Este proceso de propagación de la consulta o tarea de sensado crea *gradientes*, los cuales son direcciones de envío de datos creadas por cada nodo hacia el nodo vecino del cual recibió el mensaje *interés*. Después, el o los nodos que respondan a la tarea de sensado comienzan a enviar la información a la estación base a través de múltiples rutas. La red solamente refuerza una o un pequeño grupo de ellas para que las transmisiones se hagan por allí. *Directed Diffusion* consiste de tres fases principales, a continuación se describe cada una de ellas [12]:

- Diseminación del mensaje *interés.*

  La estación base es la encargada de generar el mensaje *interés* y de propagarlo a la red a través de sus nodos vecinos, además lo continua reenviando periódicamente por cierto periodo de tiempo, como se describe en [12]. Todos los nodos de la red tienen una memoria en donde almacenan los mensajes *interés* que les llegan. Cuando un nodo recibe un mensaje, revisa en su memoria si ya había recibido ese mensaje *interés*, si aún no lo había recibido lo almacena en su memoria y lo reenvía a sus nodos vecinos. Si en su memoria ya contiene una entrada para ese mensaje *interés*, simplemente deshecha el mensaje que le acaba de llegar. De esta manera se previene la transmisión de mensajes duplicados y la creación de ciclos de transmisión infinitos. Durante el proceso de propagación del *interés* se establecen los *gradientes*, que son direcciones de envío de datos. Cada nodo establece un *gradiente* en dirección a su nodo vecino del cual recibió el mensaje *interés* [12].

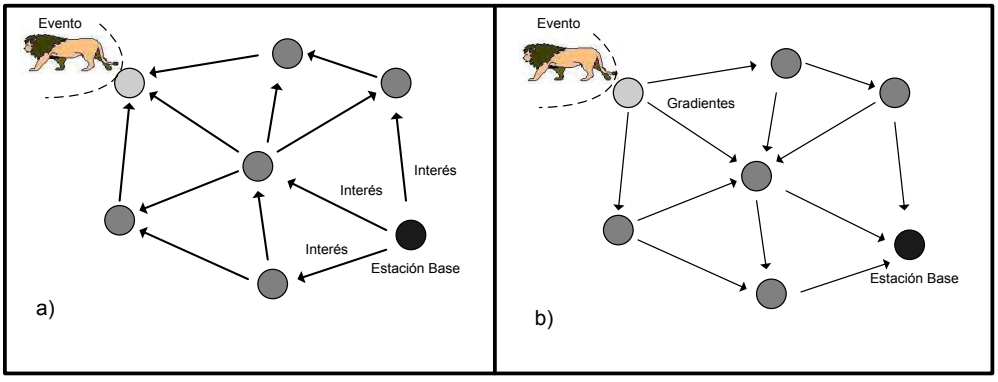


Figura 1.5 a) Propagación del mensaje *interés* a través de la red. b) Establecimiento de los *gradientes* [12].

- Respuesta exploratoria.

  Todos los nodos que reciben el mensaje *interés* sensan el medio para recolectar información y determinar si existen parámetros que concuerden con los solicitados por la estación base a través del *interés*. Aquellos o aquel nodo que encuentra los datos solicitados envía la información por medio de un mensaje de exploración. Este mensaje contiene los datos, los cuales viajan a través de los *gradientes* establecidos durante la propagación del *interés*. Dentro de cada nodo también existe una memoria en donde almacena los datos que le llegan y funciona de la misma manera que lo hace la memoria para los *intereses*. Es decir, cuando llegan datos el nodo verifica en la memoria de datos si ya existen, si aun no los tiene los almacena y los reenvía. Si ya existe una entrada para esos datos simplemente los elimina. De esta manera también se evita la transmisión de datos duplicados, lo que ayuda a la conservación de energía en la red [12].

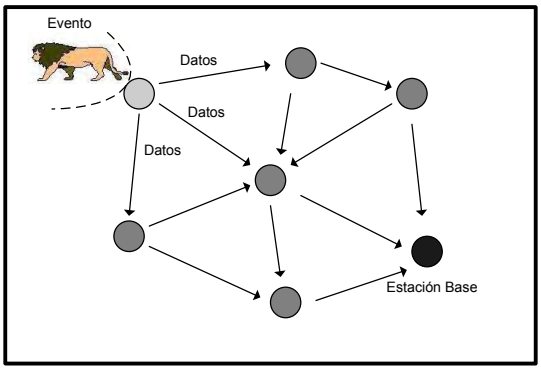


Figura 1.6 Transmisión de los datos hacia la estación base [12].

- Refuerzo de la ruta.

  Inicialmente la estación base recibe los datos por medio de diferentes rutas, pero solamente escoge un nodo vecino por el cual aceptará las futuras transmisiones. Para hacer esta elección se puede basar en cuál nodo vecino le transmitió los datos antes que los demás. Ese nodo hace lo mismo con los nodos que le transmitieron los datos a él y así sucesivamente hasta llegar al nodo que mandó la información originalmente, de esta manera se refuerza una ruta desde el nodo fuente hasta la estación base. Las futuras transmisiones ya no se propagarán a todos los vecinos de la red, sino que sólo por la ruta que se reforzó anteriormente [12].

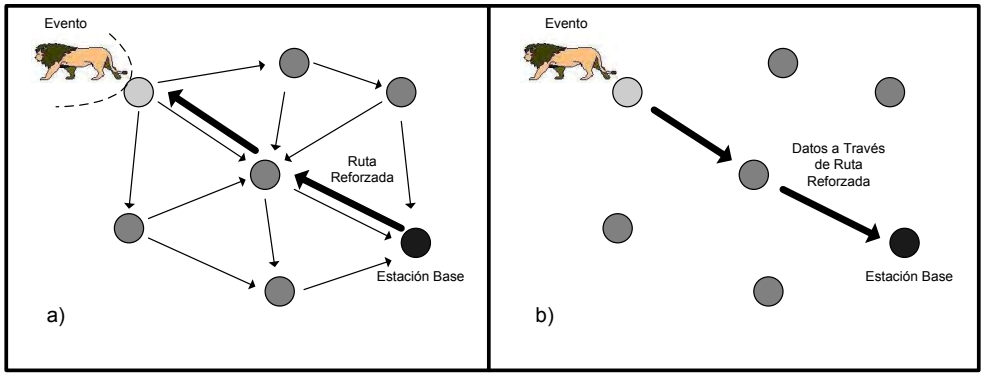


Figura 1.7 a) Refuerzo de una ruta desde la estación base hasta el nodo que envía los datos. b) Transmisión de los datos a través de la ruta reforzada [12].

1.5.3 *Cluster Based Routing Protocol*.

Un *cluster*, como se muestra en [13], es un conjunto de nodos sensores agrupados de tal manera que existe un nodo central dentro de él, a través del cual se llevan a cabo las transmisiones hacia el exterior del *cluster*. En el protocolo de enrutamiento basado en *clusters*, el cual pertenece a la categoría de enrutamiento jerárquico, los nodos que conforman la red son divididos en *clusters*, los cuales pueden estar disjuntos o traslapados. Como se mencionó anteriormente, existe un nodo central en cada uno de los *clusters*, llamado *cluster head*, el cual se encarga de las tareas de enrutamiento de los datos. Una de las características de este protocolo es que dos *cluster heads* no pueden estar conectados entre sí, sino que deben estar comunicados a través de otro nodo llamado *gateway* o nodo *entrada*. Un nodo *gateway* es un nodo que tiene como vecinos a uno o más *cluster heads*, también puede tener como vecino a otro nodo *gateway*, cuando los *clusters* son disjuntos [13]. La Figura 1.8 muestra un ejemplo de una red con este tipo de configuración.

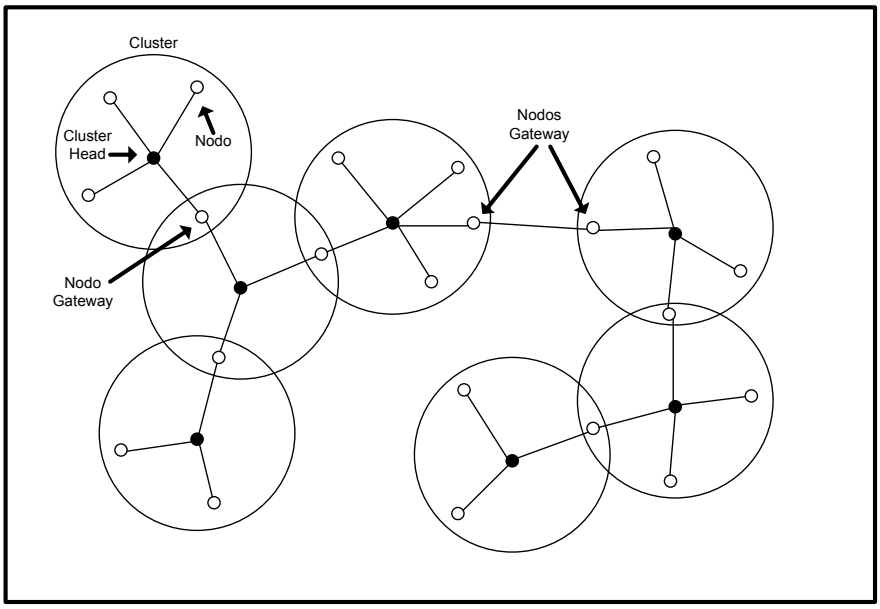


Figura 1.8 Ejemplo de una red configurada en forma de *clusters*.

Cuando un nuevo nodo es insertado en la red inicial, toma el estado *indeciso*, lo que significa que este nodo todavía no pertenece a ningún *cluster*. La primera acción que realiza es mandar en forma de *broadcast* mensajes *HELLO*. Si un *cluster head* recibe estos mensajes, éste envía mensajes de respuesta y cuando el nodo recién agregado los recibe cambia su estado a *miembro*. Esto significa que este nodo es un miembro del *cluster* cuyo *cluster head* es el que le envió los mensajes de respuesta [13].

Para el funcionamiento de este protocolo de enrutamiento, cada uno de los nodos mantiene dos estructuras: la tabla de *clusters* adyacentes (*CAT*) y la base de datos de topología de dos saltos. La tabla de *clusters* adyacentes almacena información sobre los enlaces existentes hacia los *clusters* que se encuentran conectados directamente, esto es, información sobre si existen enlaces unidireccionales o bidireccionales entre los *clusters*. La base de datos de topología de dos saltos

contiene información acerca de los nodos que se encuentran a un máximo de 2 saltos. Esta base de datos se construye a partir de la información recibida por medio de los mensajes *HELLO* [13].

El proceso de enrutamiento de los datos se describe a continuación:

El nodo fuente debe enviar los datos hacia el nodo *sink* o destino. Primeramente el nodo fuente envía solicitudes de ruta a los *cluster heads* vecinos. Cuando un *cluster head* recibe la solicitud verifica en su tabla si el nodo destino se encuentra en su *cluster*, de ser así, el *cluster head* envía directamente la solicitud hacia el nodo *sink* o destino. Cuando el nodo destino no se encuentra en el *cluster*, la solicitud se envía a los *cluster heads* adyacentes. Todos los *cluster heads* graban su dirección física (*MAC*) en el paquete, de esta manera cuando un *cluster head* recibe un paquete en donde viene almacenada su propia dirección, lo descarta.

Cuando la solicitud de ruta alcanza finalmente el nodo destino, éste envía un paquete de respuesta donde incluye la ruta que se recorrió. Cuando el nodo fuente no recibe esta respuesta después de un tiempo determinado, éste nuevamente envía otra solicitud de ruta [13].

1.6 Descripción de los filtros de Bloom.

Un filtro de Bloom es una estructura de datos utilizada para almacenar un conjunto de elementos utilizando una menor cantidad de espacio en comparación con el necesario para almacenar el conjunto completo [14]. Por medio de los filtros de Bloom es posible almacenar listas de elementos y hacer consultas sobre ellos por medio de operaciones simples. En el capítulo siguiente se mencionan de manera amplia todos los conceptos de los filtros de Bloom, así como su forma de operación y aplicaciones más comunes.

Este trabajo se basa en el uso de los filtros de Bloom por medio de la implementación de un modelo de recolección de información en una red de sensores inalámbricos que haga uso de ellos. En este modelo los filtros de Bloom servirán como herramienta para lograr disminuir el número de transmisiones necesarias dentro de la red así como la cantidad de información que se transmite durante el proceso de recolección de información, lo que traerá como beneficio una disminución en la energía consumida en la red.

1.7 Contribuciones.

Como se ha mencionado en las secciones anteriores, las redes de sensores inalámbricos han estado en constante desarrollo y son usadas cada vez en más áreas de aplicación para diferentes fines. Este tipo de redes cuentan con características que las hacen especiales y distintas a otras, una de ellas es la energía, ya que los nodos que las conforman cuentan con una cantidad de energía limitada. Debido a esto, la conservación y optimización de energía es un reto fundamental al momento de implementar una red de este tipo, ya que de ello depende la vida útil de las baterías de los nodos.

En este trabajo se presenta un modelo de recolección de información en redes de sensores inalámbricos el cual hace uso de los filtros de Bloom. Mediante este modelo se logra disminuir el número de transmisiones necesarias para dicho proceso, así como la cantidad de información transmitida. Para realizar las pruebas y comparaciones de la eficiencia del modelo propuesto, se analizan también dos protocolos conocidos: *Directed Diffusion* y *Cluster Based Routing Protocol*.

Los modelos se comparan para de esta manera mostrar las ventajas que tiene el uso del modelo propuesto en este trabajo.

Al disminuir tanto el número de transmisiones necesarias para llevar a cabo la recolección de la información como el número de información transmitida, se logra por lo tanto una disminución de la energía consumida por los nodos de la red. Esto trae como consecuencia un aumento en el tiempo de vida de las baterías de los nodos. Es importante resaltar es que esto se logra con la ayuda de los filtros de Bloom, estructura de datos utilizada en muchas otras áreas por distintas aplicaciones. La contribución de este trabajo es significativa ya que, como se ha venido mencionando, la optimización de la energía es un concepto fundamental en las redes de sensores inalámbricos.

1.8 Conclusiones.

Las redes de sensores inalámbricos han tenido un gran crecimiento en años recientes y cada día son más utilizadas en diversas áreas de aplicación. La investigación de este tema continúa y seguramente se seguirán encontrando más aplicaciones en donde las redes de sensores inalámbricos puedan brindar una buena solución. Conforme esta área evolucione los retos actuales seguirán y se espera que cada día haya mejores maneras de enfrentarlos. Estos retos son la optimización de energía, desarrollo de nuevos sensores, formas de comunicación entre los nodos, procesamiento y descubrimiento de nuevas aplicaciones de las redes de sensores inalámbricos.

En los capítulos siguientes se introducen los filtros de Bloom, así como la forma en que éstos se implementarán en el modelo de recolección de información que se propone en este trabajo. El modelo propuesto también se describe detalladamente. En los capítulos finales se presentan las pruebas realizadas así como los resultados obtenidos.

Capítulo 2 Filtros de Bloom.

Un filtro de Bloom es una estructura de datos utilizada para representar un conjunto de elementos y operaciones de consulta sobre él [14], enseguida se presentará una introducción a este concepto. Esta estructura de datos busca reducir el espacio necesario para el almacenamiento del conjunto. Una característica de este tipo de estructuras de datos es la de permitir falsos positivos durante las consultas, es decir, puede responder de manera positiva ante la consulta de la existencia de un elemento cuando en realidad ese elemento no está presente dentro del filtro de Bloom.

La idea fue introducida por Burton Bloom en los años 1970's, como se describe en [15], y durante todo este tiempo el número de aplicaciones que hacen uso de este concepto ha crecido. Esto hace de los filtros de Bloom una idea interesante para seguir buscando nuevas aplicaciones, debido a que han mostrado buenos resultados. Los filtros de Bloom han sido adaptados y modificados para ser aplicados dentro de las redes de computadoras con buenos resultados en diversas aplicaciones, como se describe en [14]. El éxito del uso de los filtros de Bloom se debe a que éstos ofrecen una manera distinta de representar un conjunto o una lista de elementos. Dentro de las redes existen muchas aplicaciones donde es común mantener o transmitir un conjunto de elementos, es aquí donde entra el concepto de los filtros de Bloom. Su uso ayuda a disminuir el espacio necesario para almacenar ese conjunto de elementos y mejorar el rendimiento de la red, con el costo de los ya mencionados falsos positivos. Algunas aplicaciones de los filtros de Bloom son presentadas en [14].

Un filtro de Bloom puede ser representado por un arreglo de un tamaño determinado, donde inicialmente cada una de las posiciones del arreglo son *bits* puestos en 0. Además de esto, deben de existir funciones *hash* de las cuales se hará uso para introducir los elementos al filtro y hacer las consultas cuando sea necesario [16].

Una función *hash* es un método para representar en forma de una clave al elemento de entrada a dicha función. Este tipo de funciones tienen como posibles resultados un conjunto finito de información, normalmente un subconjunto de números. Cuando se quiere obtener la clave de algún elemento para cierta función *hash*, éste se introduce a dicha función. El resultado obtenido es la clave o llave del elemento introducido. Como el rango de posibles resultados de las funciones *hash* es finito, es posible que elementos de entrada diferentes produzcan resultados iguales. De igual manera, un mismo elemento de entrada a la función no puede producir resultados diferentes, esto se debe a las propiedades de correspondencia de la función.

Los procesos de inserción y consulta en el filtro se describen a continuación:

Para insertar un elemento del conjunto dentro del filtro de Bloom se introduce dicho elemento a todas las funciones *hash*. El resultado de cada una de las funciones para dicho elemento se toma como índice de la posición en la cual se pondrá un 1 en el arreglo que representa al filtro de Bloom. De esta manera, si se tienen k funciones *hash*, se modificarán k posiciones diferentes en el filtro, como se describe en [14].

Para realizar las consultas sobre la existencia de cierto elemento dentro del filtro, se procede de manera similar, es decir, se introduce el elemento a las funciones *hash* y se obtienen los resultados. Después, se revisan esas posiciones en el filtro, si en todas las posiciones resultantes hay un 1, significa que el elemento sí está dentro del filtro, aunque en esta afirmación hay cierta probabilidad de error (probabilidad de falsos positivos). Si en alguna de las posiciones resultantes hay un 0 significa que el elemento definitivamente no se encuentra en el filtro [15].

Los llamados falsos positivos pueden presentarse al momento en que se esté verificando si cierto elemento pertenece al conjunto. Si al momento de verificar los resultados de las funciones *hash*, éstos apuntan a posiciones donde se encuentren números 1, pero colocados allí debido al resultado de introducir otros elementos, no el elemento en cuestión, se tendría un falso positivo. En este caso se daría por hecho que el elemento sí pertenece al conjunto, cuando en realidad no pertenece, como se describe en [14]. En la Figura 2.1 se muestra un ejemplo de un filtro de Bloom.

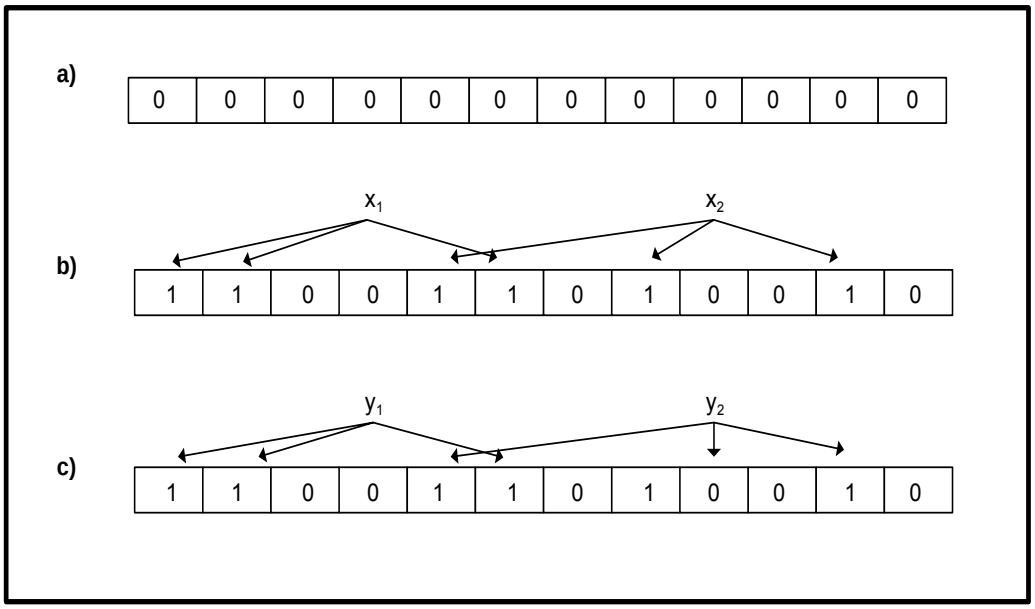


Figura 2.1 Ejemplo de un filtro de Bloom [14].

En la Figura 2.1 a) se puede observar que el arreglo inicial del filtro Bloom contiene 0´s en todas sus posiciones. En 2.1 b) cada uno de los elementos x_i se pasa por las k funciones *hash* (en este caso $k=3$) y como resultado de cada función *hash* se obtiene la posición en el arreglo en la cual se colocará un 1. En 2.1 c) para verificar si un elemento está en el conjunto, éste se introduce a las k funciones *hash* y se verifica que sus resultados apunten a posiciones en donde haya solamente 1´s. En este caso el elemento y_2 definitivamente no se encuentra en el conjunto pues uno de sus resultados apunta a una posición donde se encuentra un 0, mientras que el elemento y_1 puede sí estar en el conjunto, con cierta probabilidad de que haya ocurrido un falso positivo. Para muchas aplicaciones, los falsos positivos pueden ser aceptados si la probabilidad de que ocurran es pequeña.

Probabilidad de Falsos Positivos.

Un filtro de Bloom sirve para contener un conjunto de *n* elementos. Se utilizan *k* funciones *hash*, y este filtro es de tamaño *m*. Se asume que cada función *hash* selecciona las posiciones del arreglo todas con la misma probabilidad. Por lo tanto, la probabilidad de que cierto *bit* no sea seleccionado por una función *hash* es [14]:

$$1 - \left(\frac{1}{m}\right)$$

Y la probabilidad de que cierto *bit* no sea seleccionado por ninguna de las *k* funciones *hash* es:

$$\left(1 - \left(\frac{1}{m}\right)\right)^k$$

Como se insertan *n* elementos en el filtro, la probabilidad de que cierto *bit* siga siendo 0 después de la inserción de todos los *n* elementos es [14]:

$$(1 - (1/m))^{kn} \approx e^{-(km/n)}$$

Y la probabilidad de que sea 1 es:

$$1 - (1 - (1/m))^{kn} \approx 1 - e^{-(km/n)}$$

Se produce un falso positivo si las *k* funciones *hash* encuentran *bit*s en 1 (para un elemento que no pertenece a la lista), es decir, la probabilidad de que cierto *bit* sea 1 (calculada anteriormente) elevada a la *k* (*k* funciones *hash*) [14].

$$f = (1 - (1 - (1/m))^{kn})^k \approx (1 - e^{-(km/n)})^k$$

Se observa que la probabilidad de falsos positivos disminuye cuando el tamaño de arreglo (*m*) aumenta, y aumenta cuando el número de elementos a insertar (*n*) se incrementa. La eficiencia de un filtro de Bloom se puede definir determinando el número de *bits* (*m*) necesarios en el arreglo. Para dos valores dados de *m* y *n*, el valor óptimo de *k* (valor que minimiza la probabilidad de falsos positivos) es:

$$k = \frac{m}{n} \ln 2 \approx 0.7 \frac{m}{n}$$

Lo que da una probabilidad de falsos positivos de [14]:

$$f = 2^{-k} = (1/2)^k \approx 0.6185^{\frac{m}{n}}$$

Funciones *hash* y operaciones sobre filtros de Bloom.

Desde hace tiempo, las funciones *hash* han sido utilizadas en diversas aplicaciones que hacen uso de conjuntos de elementos. De cierta manera, un filtro de Bloom es un derivado de las funciones *hash*, pues éste hace uso de ellas para el mismo propósito. Un filtro de Bloom puede ser equivalente a una función *hash* ordinaria si el filtro cuenta solamente con una de ellas, como se describe en [14].

La forma en la que están formados los filtros de Bloom hace posible la realización de algunas operaciones sobre ellos. Por ejemplo, es posible obtener, a partir de dos filtros de Bloom que representen dos conjuntos, un tercer filtro que represente la unión de ambos conjuntos, el cual estaría conformado por el resultado de aplicar la operación "*OR"* a los dos vectores que representan a los dos conjuntos. Los filtros de Bloom también pueden ser usados para aproximar la intersección entre dos conjuntos, como se describe en [14].

2.1 Aplicaciones de los filtros de Bloom.

En los últimos años, la atención que se ha puesto en los filtros de Bloom ha venido en aumento debido a que han comenzado a ser usados en un gran número de sistemas y aplicaciones. A continuación se presentan algunas de las aplicaciones en las que se han venido utilizando los filtros de Bloom desde hace tiempo [14].

Cuando surgió el concepto de filtros de Bloom, una de las primeras aplicaciones que se les dio fue en diccionarios. En este ámbito los filtros de Bloom ayudan a mantener una lista de palabras con una menor cantidad de espacio necesario en comparación con el que se necesitaría para almacenar la lista completa, como se describe en [14].

Los filtros de Bloom también se han venido utilizando en las bases de datos. En este campo, los filtros de Bloom han brindado buenas soluciones a las necesidades de almacenamiento de información. Cuando las bases de datos son demasiado grandes, es posible mantener un filtro de Bloom que contenga almacenados los elementos de interés, de esta manera el espacio necesario para almacenar la base de datos será menor. Así mismo, si existe la necesidad de transmitir elementos de una base de datos a otra que se encuentre en un equipo distinto, en lugar de transmitir todos los elementos, se puede transmitir un filtro de Bloom que los contenga. Haciendo las acciones anteriormente mencionadas se reduce considerablemente la cantidad de información transmitida, con todos los beneficios que ello trae, como menor consumo de ancho de banda, menor tiempo de transmisión, etc. [14].

2.1.1 Aplicaciones en red.

Dentro del área de las redes, los filtros de Bloom han sido aplicados para distintos propósitos, a continuación se presenta un resumen de estas aplicaciones.

Las redes *P2P* (*Peer-to-peer*), como se describe en [14], son utilizadas en aquellas aplicaciones que demandan una gran cantidad de recursos, además en este esquema es común que los equipos compartan grandes cantidades de información de manera directa. De manera similar a lo que ocurre con las bases de datos, los equipos en las redes *P2P* pueden almacenar su información en filtros de Bloom y transmitirlos en lugar de transmitir la información completa. Cuando los equipos compartan objetos y recursos, y sea necesario que cada uno de estos equipos mantenga una lista con los elementos almacenados en todos los otros nodos de la red, puede preferirse que cada uno de los nodos almacene un filtro de Bloom con la información necesaria. Almacenar la lista completa

implicaría un mayor espacio de almacenamiento, lo que puede resultar más costoso. Es por eso que la solución que brindan los filtros de Bloom normalmente ofrece buenos resultados.

2.1.1.1 Aplicaciones de enrutamiento de recursos.

Los filtros de Bloom también han sido aplicados desde hace tiempo en el área de enrutamiento de recursos, como se describe en [14], a continuación se presentan algunos ejemplos de lo que se ha realizado en este campo.

Cuando dentro de una red se esté realizando el proceso de enrutamiento de recursos, es probable que existas nodos principales, los cuales mantengan listas de los recursos que poseen los nodos que estén conectados a ellos. Estos nodos principales son los encargados de realizar el correcto enrutamiento de las solicitudes de recursos que se presenten. Normalmente, cada uno de estos nodos mantendría una lista con los recursos que el propio nodo posee y aquellos que son accesibles mediante uno de los nodos que están conectados a él. De esta manera, mediante una revisión a la lista, se puede decidir hacia dónde hacer el enrutamiento de las solicitudes de recursos.

El protocolo de enrutamiento descrito anteriormente es muy sencillo y la idea simplemente es mencionar cómo es que los filtros de Bloom podrían ayudar a mejorar el rendimiento del mismo. Las listas de recursos de las que se habló podrían estar representadas por filtros de Bloom, lo que traería como beneficio una disminución en el espacio de almacenamiento necesario. Además, las consultas se podrían seguir realizando sobre el filtro cuando una solicitud de enrutamiento se presente. Un falso positivo podría ocasionar que cierto nodo aceptara una solicitud de recurso cuando en realidad no cuenta con él, lo que causaría un enrutamiento incorrecto [14].

2.1.1.2 Enrutamiento de paquetes.

Dentro del área de enrutamiento de paquetes se han propuesto diversos usos de los filtros de Bloom. A continuación se describen algunas de las propuestas que se han presentado en esta área.

Detección de ciclos.

Whitaker y *Wetherall* [17] sugirieron utilizar un pequeño filtro de Bloom para prevenir ciclos en el envío de datos en protocolos *unicast* y *multicast*. Actualmente, la detección de los ciclos durante el envío de paquetes se realiza mediante el campo *TTL* (*Time-To-Live*), el cual es básicamente un contador que se incrementa conforme el paquete va viajando a través de los nodos. Si el contador crece en demasía se asume que el paquete ha caído en un ciclo y por lo tanto se elimina. En este ámbito, los filtros de Bloom brindan otro enfoque de detección de ciclos. La idea es que los paquetes contengan un filtro, en el cual se almacenen los nodos por los cuales va pasando. Debido a la forma en la que funcionan los filtros de Bloom, este filtro irá variando conforme pasa por cada nodo diferente, de manera que cuando el filtro ya no cambie se podrá decir que el paquete ha caído dentro de un ciclo, pues ya no se están insertando nuevos elementos en él.

La idea anteriormente mencionada podría brindar buenos resultados especialmente para ciclos pequeños, los cuales el campo *TTL* podría pasar por alto [14].

2.1.1.3 Medición de tráfico.

Dentro del área de las redes, una aplicación interesante es la referente a la medición de tráfico, es decir, determinar la cantidad de tráfico que pasa por un determinado punto, tipo de tráfico, etc. Debido al gran incremento de aplicaciones que transmiten datos dentro de las redes, la cantidad de información que pasa a través de los enrutadores es enorme, lo que da como consecuencia que el proceso de medición y obtención de estadísticas sea costoso. Los filtros de Bloom podrían hacer este proceso más sencillo, al ofrecer una solución para realizarlo de tal manera que su costo se reduzca [14].

2.1.1.3.1 Registro de flujos pesados.

Dentro de un enrutador pasa una gran cantidad de información de diferentes tipos, algunos tipos de información son más demandantes en cuanto a recursos se refiere. Es por eso que es importante poder identificar esos flujos, denominados flujos pesados, para que el enrutador les pueda dar el trato necesario.

Estan y *Varghese* [18] expusieron una idea que hace uso de los filtros de Bloom para la medición del tráfico que pasa dentro de un enrutador. De esta manera se podrían determinar los flujos pesados con un menor costo, basándose en las características y los beneficios que tiene el uso de los filtros de Bloom. La idea es colocar un filtro de Bloom de conteo dentro de cada enrutador, cuando cada paquete vaya pasando se introducirá éste al filtro mediante las funciones *hash* correspondientes. Pero en lugar de incrementar en 1 el contador, se incrementará con el número de *bytes* del paquete, de esta manera se lleva un registro del número de *bytes* asociados a cada paquete que han pasado por el enrutador. Si este registro crece en demasía se puede determinar que el flujo que se está transmitiendo es un flujo pesado.

2.1.1.3.2 Rastreo *IP*.

Actualmente, como se describe en [14], es importante tener un mecanismo efectivo de rastreo de paquetes para poder detectar rutas de paquetes maliciosos y de esta manera poder prevenir futuros problemas. Además de efectivo, este mecanismo debe de ser sencillo en su implementación y que no consuma demasiados recursos. Es en este punto donde los filtros de Bloom pueden ayudar a lograrlo.

Snoeren et al. [19] propusieron la idea de la utilización de los filtros de Bloom para el almacenamiento de la información necesaria para poder llevar a cabo el rastreo de paquetes. La idea es que cada enrutador mantenga un filtro de Bloom en donde almacene información de los paquetes que pasan por él. De esta manera si se desea conocer la ruta que siguió un determinado paquete basta con consultar los filtros de los enrutadores para determinar por cuáles de ellos fue transmitido dicho paquete y poder recrear la ruta que siguió. Esta idea permite a los enrutadores almacenar la

información de los paquetes mediante una cantidad de espacio pequeño, pues lo realiza con la ayuda de los filtros de Bloom.

2.2 Conclusiones.

El filtro de Bloom es una estructura de datos sencilla en su concepto, la cual permite almacenar objetos de un conjunto con facilidad y recuperarlos por medio de la aplicación de ciertas operaciones. Este proceso se describió en las primeras secciones de este capítulo. La ventaja de usar este tipo de estructuras es considerable en cuanto al espacio de almacenamiento se refiere. El costo de mantener el filtro de Bloom que represente los elementos del conjunto es considerablemente menor a comparación del costo que representaría almacenar el conjunto de elementos completo.

Como se pudo apreciar en las aplicaciones presentadas en este capítulo, los filtros de Bloom ofrecen una buena solución para sistemas en los que se requiere almacenar o enviar información en forma de listas, y para los cuales la cantidad de espacio para almacenar esa información es limitada, o simplemente en donde el objetivo sea reducir la cantidad de información almacenada o transmitida. Es posible utilizar esta estructura en múltiples aplicaciones, para las cuales seguramente logrará buenos resultados.

Capítulo 3 Sistema de Recolección de Datos.

En este trabajo de investigación se presenta la implementación de unas estructuras de datos llamadas filtros de Bloom para la recolección de la información en una red de sensores inalámbricos. Uno de los principales objetivos de esta implementación es reducir el número de transmisiones necesarias para la recolección de información en las *WSN*, en comparación con los protocolos normalmente utilizados. De esta manera se obtiene un mayor tiempo de vida de la batería de los nodos de la red, pues éste se basa en el consumo de energía que tienen los nodos, el cual se verá disminuido. En las siguientes secciones de este capítulo se describe de manera detallada esta idea y la manera en la que funciona.

3.1 Propagación del mensaje *interés*.

Como se mencionó anteriormente en el capítulo 1 cuando se habló sobre los protocolos de recolección de información en redes de sensores inalámbricos, el primer paso que se realiza en el protocolo *Directed Diffusion*, en el cual se apoya este trabajo, es propagar la consulta a toda la red por medio de un mensaje llamado *interés*. Este mensaje es creado por el nodo *sink* y lo transmite a sus nodos vecinos, los cuales continuarán transmitiéndolo. De esta manera se propagará a toda la red en forma de *inundación (flooding)*, como se describe en [12]. La Figura 3.1 muestra un ejemplo de la propagación del mensaje *interés* en una red de sensores inalámbricos.

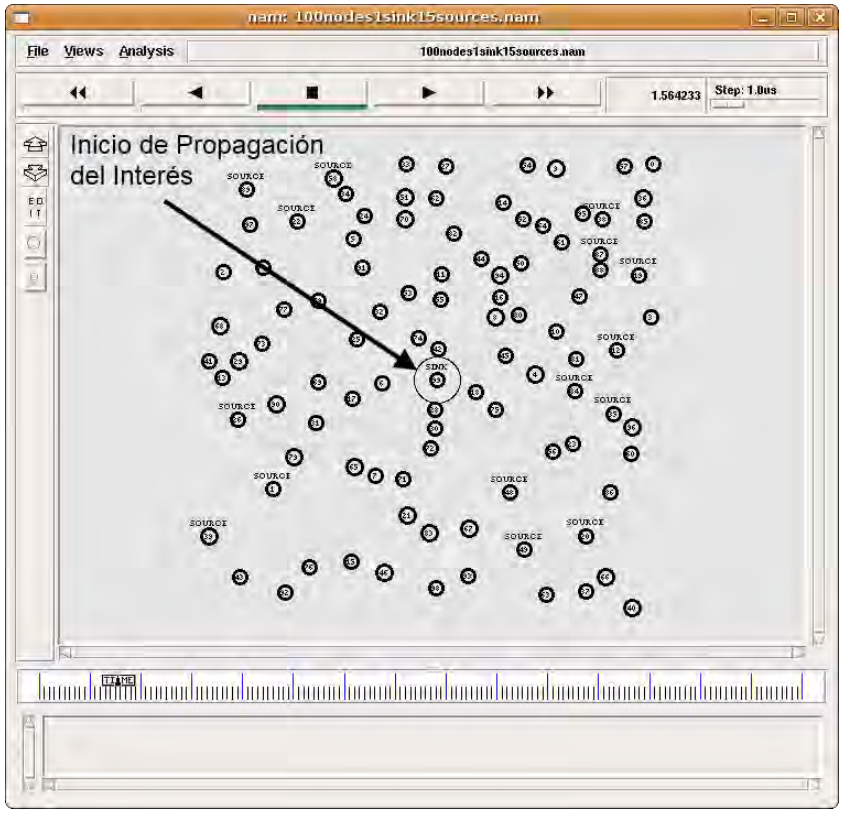


Figura 3.1 Inicio de envío del mensaje *interés*.

En la Figura 3.1 se puede observar un escenario que representa un ejemplo de una red de sensores inalámbricos. El nodo *sink* se encuentra en el centro y es el que realiza la consulta a la red por medio de un mensaje *interés*. En la Figura 3.1 se nota que este nodo está iniciando la transmisión del mensaje a sus nodos vecinos, los cuales harán lo mismo para continuar expandiendo el *interés* por toda la red en forma de *inundación (flooding)*.

La variación que en este trabajo de investigación se presenta es la agregación de un nuevo campo en ese mensaje *interés*. Este campo servirá para representar un filtro de Bloom en el cual los nodos puedan insertar los datos que quieran reportar al *sink* conforme este mensaje *interés* se propaga a través de la red. Durante esta misma propagación del *interés* en la red, cada uno de los nodos tiene que establecer una ruta de regreso de datos. Es decir, establecer a cuál de sus nodos vecinos le enviará los datos al momento de llevar a cabo el regreso de información hacia el nodo *sink*. Tanto la inserción de datos en el filtro de Bloom como la creación de las rutas de regreso de datos se detallan a continuación.

3.1.1 Creación de rutas de regreso de datos.

Cada uno de los nodos que conforman la red debe estar asociado a una ruta de regreso de datos para poder llevar a cabo ese proceso cuando sea el momento de enviar la información recolectada hacia el nodo *sink*. Estas rutas de regreso de información se irán formando conforme el mensaje *interés* se propague por toda la red en forma de *inundación (flooding)*. Durante el proceso de propagación del *interés*, cada uno de los nodos que conforman la red recibirá este mensaje proveniente de alguno de sus nodos vecinos. Cada nodo tomará como ruta de regreso de datos a aquel nodo vecino del cual le llegó el mensaje *interés*. Si un nodo recibe de varias fuentes el mensaje *interés*, dicho nodo establecerá la ruta de regreso de datos hacia su nodo vecino del cual le llegó primero el mensaje, y descartará los demás, tal y como funciona propagación en forma de *inundación (flooding)* en una red.

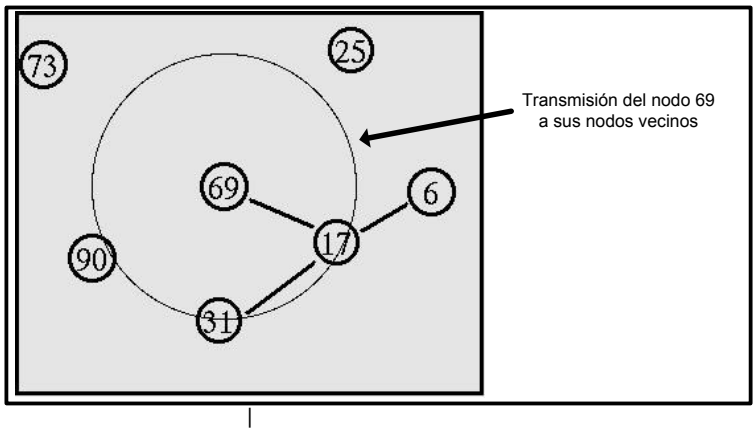


Figura 3.2 Transmisión del mensaje *interés* a los nodos vecinos.

En la Figura 3.2 se observa que el nodo 69 está transmitiendo el mensaje *interés* a sus nodos vecinos. Si es el primer mensaje que reciben, éstos establecerán como su ruta de regreso de datos el

nodo 69 del cual recibieron el mensaje y repetirán el proceso, transmitiendo el mensaje a sus nodos vecinos. Los nodos vecinos harán lo mismo y así sucesivamente hasta que el mensaje se propague en toda la red. Se puede apreciar en la Figura 3.2 que como los nodos 69 y 31 recibieron el mensaje del nodo 17, ya hay una ruta establecida de regreso de datos hacia el nodo 17. En este caso el único nodo vecino del nodo 69 que aún no ha recibido ningún mensaje *interés* es el nodo 90. Al recibirlo, este nodo establecerá su ruta de regreso de datos hacia el nodo 69. El nodo 90 será el único nodo que siga transmitiendo el mensaje, según la forma de funcionamiento de la transmisión por *inundación (flooding)*. La Figura 3.3 muestra el proceso anteriormente mencionado.

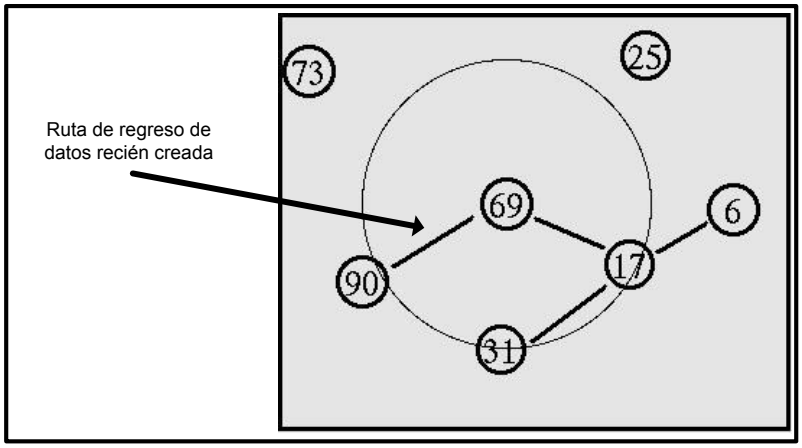


Figura 3.3 Establecimiento de rutas de regreso de datos.

Es importante mencionar que las líneas que representan los enlaces en las figuras anteriores no son en realidad enlaces cableados. Éstas son solamente representativas para así poder apreciar de una mejor forma el establecimiento de rutas de regreso de datos en la red, debido a que todas las comunicaciones son de manera inalámbrica.

3.1.2 Inserción de datos en el filtro de Bloom.

El mensaje *interés* que se propaga por la red en forma de *inundación (flooding)* contendrá un campo adicional que represente al filtro de Bloom, en donde los nodos puedan insertar la información que va dirigida al nodo *sink*. Los nodos que insertarán datos son aquellos que responden o cumplen con la petición realizada por el *sink*, la cual se propaga por toda la red. Es necesario que estos nodos inserten en el filtro de Bloom la información sensada e información adicional, como su *ID* por ejemplo. Esto permita identificar qué información fue introducida por cada nodo una vez que se procesen los datos recibidos en el nodo *sink*. Si no se insertara esta información acerca del nodo fuente, al momento de procesar los datos en el *sink* se obtendrían solamente las mediciones o los datos recolectados en la red, pero no información de qué nodos fueron los que los produjeron.

Para poder tener en el filtro de Bloom tanto los datos recolectados en la red como la información de los nodos que los produjeron, se presenta la siguiente forma de fusionar estos dos parámetros para su posterior inserción al filtro:

Para identificar cuáles nodos son los que insertan información en el filtro de Bloom se utilizará su *ID*. Se asume que el *ID* de los nodos empieza en 0 y llega hasta 99 (para el ejemplo de 100 nodos). Por lo tanto el *ID* de cada nodo se presentará como un par de dígitos, por ejemplo:

Nodo 0 → 0 0
Nodo 1 → 0 1
Nodo 2 → 0 2
.
.
.
Nodo 99 → 9 9

Se fijarán límites para los datos que los nodos insertarán en el filtro de Bloom, por ejemplo, los datos pueden tener un valor de 0 a 99 (que pudieran por ejemplo representar un valor sensado de temperatura, presión, nivel de humedad, etc.). Y estos datos se representan también con dos dígitos: 00, 01, 02, . . . , 98, 99.

Cuando un nodo desee modificar el filtro de Bloom para insertar el dato sensado por él, lo que hará será fusionar su *ID* con el dato mismo. El resultado de ésta fusión será el dato o secuencia de dígitos que se introduce a las funciones *hash* para poder obtener su representación dentro del filtro de Bloom. Por ejemplo, si el nodo fuente 38 quiere insertar el dato con valor 33, la secuencia de dígitos a introducir en las funciones *hash* sería:

Nodo: 38
Dato a insertar: 33
Secuencia a insertar en las funciones *hash* del filtro de Bloom: 3 8 3 3

De esta manera, la secuencia de números a insertar en las funciones *hash* contendrá el *ID* del nodo y el dato sensado. Dentro de esta secuencia se distinguirá que los 2 primeros dígitos representan el *ID* del nodo y los últimos 2 dígitos representan el dato sensado, dando un total de 4 dígitos.

El número de dígitos puede aumentar si el número de nodos aumenta o el rango de los datos sensados también se incrementa. Por ejemplo, se podrían ocupar 4 dígitos para el *ID* del nodo (Nodo 1265 por ejemplo) y 3 dígitos para el dato (Dato 121 por ejemplo), con lo que se obtendría una secuencia de 7 dígitos (1 2 6 5 1 2 1), la cual se introduciría a las funciones *hash* del filtro de Bloom.

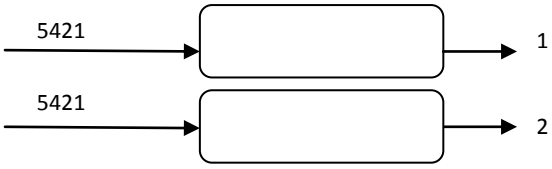
A continuación se presenta un ejemplo completo de la inserción de datos en el filtro de Bloom por 3 nodos:

Nodo → 54 Dato → 21
Nodo → 23 Dato → 52
Nodo → 99 Dato → 43

Por lo tanto, las secuencias de dígitos a insertar en las funciones del filtro de Bloom son: 5421, 2352 y 9943.

En este ejemplo suponemos que el filtro de Bloom es de tamaño igual a 6 y tenemos 2 funciones *hash*. Estas dimensiones son obviamente muy pequeñas para aplicaciones reales, pero sólo es para mostrar la manera en la que se lleva a cabo el proceso.

Insertando información del nodo 54:

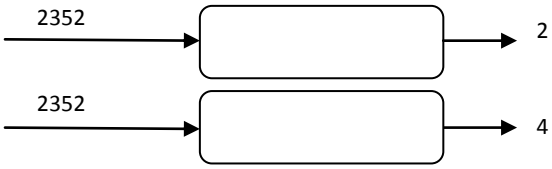


Se coloca 1 en las posiciones 1 y 2

Filtro de Bloom:

1	1	0	0	0	0

Insertando información del nodo 23:



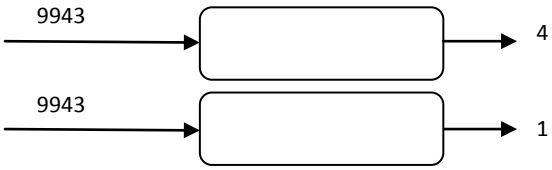
Se coloca 1 en las posiciones 2 y 4

(Si ya está puesto en 1 no se modifica)

Filtro de Bloom:

1	1	0	1	0	0

Insertando información del nodo 99:



Se coloca 1 en las posiciones 4 y 1

(Si ya está puesto en 1 no se modifica)

1	1	0	1	0	0

Y de esta manera se han introducido al filtro de Bloom los datos generados por los nodos así como el *ID* del nodo que aporta el dato. En el ejemplo anterior se observa que cuando la salida de una función *hash* es una posición del filtro que ya se encuentra puesta en 1, ésta no se modifica pues así es la forma en que funcionan los filtros de Bloom.

3.2 Envío de datos hacia el *sink*.

Este proceso se define como el regreso hacia el *sink* de los filtros de Bloom que contienen los datos insertados por los nodos durante la propagación del mensaje *interés*, proceso que se describió en secciones anteriores. A continuación se describen y detallan las partes que conforman el envío de datos hacia el *sink*.

3.2.1 Regreso de los filtros de Bloom.

Cuando se termina de propagar el mensaje *interés* en la red, ya se han recolectado todos los datos que los nodos insertaron, ahora esa información se encuentra en los filtros de Bloom. Los filtros que contienen la información completa de todos los nodos por los que pasaron son aquellos que se encuentran al final de las rutas de regreso de datos que se fueron formando.

Las rutas de regreso de datos pueden verse como un árbol. Es decir, una estructura de nodos conectados entre sí que imita la forma de un árbol. Dentro de esta estructura, los nodos pueden tener o no tener nodos hijos conectados a él. Si cierto nodo tiene hijos conectados a él, se dice que es un nodo padre. El único nodo que no tiene nodo padre se denomina nodo raíz, en nuestro ejemplo nos referimos al nodo *sink*. Los nodos que no tienen nodos hijos se les da el nombre de nodos hoja, para nuestro caso estos nodos son los que se encuentran al final de las rutas de regreso de datos.

Al irse propagando el mensaje *interés* en la red también se van formando las rutas de regreso de datos, creando un árbol. Los nodos que tienen los filtros con la información que interesa son aquellos que son hojas del árbol. La Figura 3.4 ilustra un ejemplo de cuáles serían los nodos que contiene los filtros de Bloom que nos interesan:

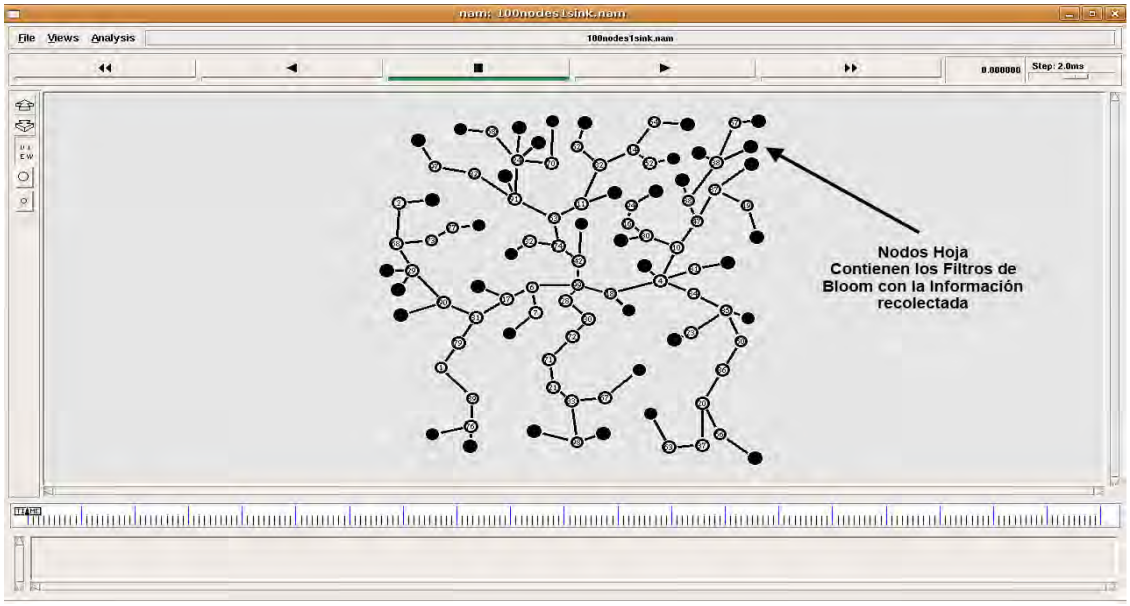


Figura 3.4 Nodos contenedores de los filtros de Bloom con la información recolectada.

En la Figura 3.4 se encuentran sombreados aquellos nodos que, después de terminada la *inundación (flooding)* inicial del mensaje *interés*, contienen los filtros de Bloom con la información completa recopilada durante la propagación del mensaje. Se observa que estos nodos no deben ser obligatoriamente nodos que se encuentren en la periferia de la red, si no que, como ya se explicó anteriormente, son aquellos nodos que se encuentran el final de las rutas de envío de datos.

Los nodos que se consideran finales en las rutas de regreso de datos, mostrados en la figura anterior, son los encargados de iniciar la transmisión de los filtros de Bloom que contienen la información hacia el nodo *sink*. La transmisión se realizará a través de las rutas que se establecieron en el momento de la propagación del mensaje *interés*. Después de este proceso, el nodo *sink* recibirá tantos filtros de Bloom como nodos vecinos tenga, pues cada uno de sus vecinos representa una ruta diferente de envío de datos.

Agregación de filtros.

Una vez que la propagación del mensaje *interés* ha terminado y por lo tanto se tienen ya establecidas las rutas de regreso de datos hacia el *sink*, los nodos que se ven como hojas del árbol de rutas inician la transmisión de sus filtros de Bloom hacia el *sink*. Cuando algún nodo recibe 2 o más filtros de sus nodos vecinos, éste tiene que hacer la agregación de los filtros para de esta manera convertirlos en uno solo. Este filtro resultante contendrá la información de todos los filtros individuales que el nodo recibió, y se podrá seguir transmitiendo hacia el *sink*. Así se realiza solamente una sola transmisión por cada nodo de la red y no tantas transmisiones como filtros de Bloom le lleguen a cada nodo que tenga 2 o más hijos.

En el capítulo donde se habló sobre la teoría a cerca de los filtros de Bloom, se mencionó que una de las propiedades que éstos tienen es que pueden ser sumados simplemente aplicando la operación *"OR"* entre ellos. Al realizar esta operación se obtendrá un filtro que contiene toda la información de los filtros de los que se deriva. Entonces, para esta aplicación, basta con que cada uno de los nodos que reciben más de un filtro de Bloom los sume. Después de realizar la suma de filtros se podría entonces realizar la transmisión de solamente un filtro y no de todos los que lleguen por separado al nodo. Las Figuras 3.5 y 3.6 muestran un ejemplo de este proceso:

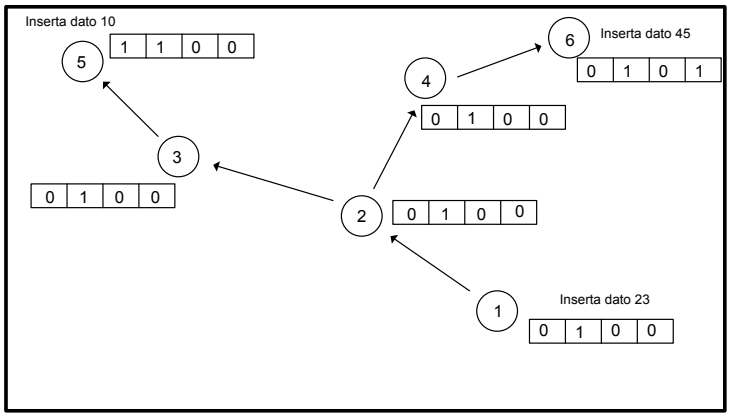


Figura 3.5 Inserción de datos en el filtro y su propagación.

En la Figura 3.5 se muestran los datos que van siendo insertados por los nodos fuente, en este ejemplo el nodo 1 inserta el dato 23, el nodo 5 inserta el dato 10 y el nodo 6 inserta el dato 45. En este caso los nodos que iniciarán el regreso de datos hacia el *sink* son los nodos 5 y 6. Puede observarse que en el filtro de Bloom que mantiene el nodo 5 se encuentra el dato 23 y el dato 10, mientras que en el filtro de Bloom que tiene el nodo 6 se encuentra el dato 23 y el dato 45. Lo que se pretende es que cuando el nodo 5 y el nodo 6 envíen los filtros de regreso hacia el *sink*, el nodo 2 sea el encargado de realizar la agregación de filtros. De esta manera al nodo 1 llegaría un solo filtro de Bloom que contiene los datos 23, 10 y 45. Esto se muestra a continuación.

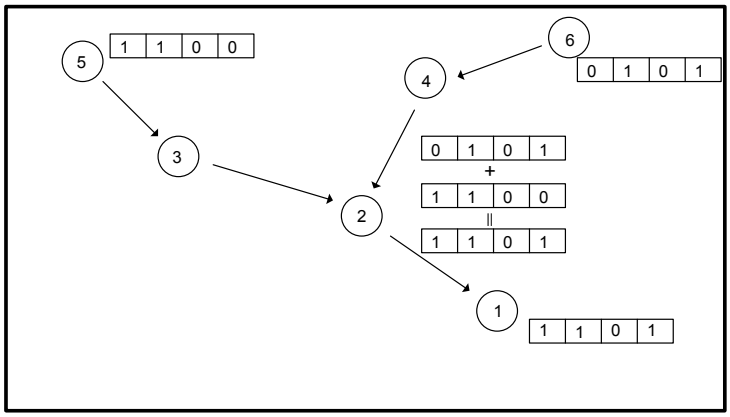


Figura 3.6 Agregación de filtros durante el regreso de datos.

En la figura anterior se observa que tanto el nodo 5 como el nodo 6 realizan el regreso de datos hacia el *sink*. Al llegar al nodo 2, éste realiza la agregación de filtros sumándolos para de esta manera obtener un solo filtro, el cual transmite al nodo 1. Este filtro resultante contiene la información de de los filtros de los que se deriva. Este proceso se repetirá en la red cuantas veces sea necesario hasta que los filtros de Bloom lleguen al *sink*.

3.2.2 Procesamiento de los datos.

El nodo *sink* recibirá de regreso tantos filtros de Bloom como nodos vecinos tenga. Para facilitar la interpretación de ellos y la obtención de la información contenida, el *sink* realizará también el proceso de agregación de filtros, para de esta manera obtener solamente uno que contenga toda la información recolectada en la red.

Una vez que el *sink* tiene solamente un filtro de Bloom, realizará el procesamiento de datos. Éste consiste en extraer la información contenida en el filtro mediante la técnica explicada anteriormente cuando se habló sobre la teoría de los filtros de Bloom y la consulta de elementos existentes en ellos. Este proceso consiste en comprobar cuáles datos se encuentran contenidos en el filtro, haciendo pasar todas las posibles combinaciones por las funciones *hash* correspondientes para ver si el elemento en cuestión está en el filtro o no. Es decir, para el ejemplo con el que se ha estado trabajando, se probara desde la cifra 1 (que representa al nodo 0 insertando el dato 1) hasta la cifra 9999 (que representa al nodo 99 insertando el dato 99).

El proceso anteriormente mencionado podría parecer pesado por la cantidad de elementos que se tienen que probar, pero solamente se realizará una vez, cuando el *sink* recibe la información de la red. Los beneficios que se obtienen con la implementación de este modelo basado en filtros de Bloom para la recolección de información en una red de sensores inalámbricos son importantes y realmente ayudan a aumentar el tiempo de vida de las baterías de los nodos de la red. Esto se logra al ahorrar en transmisiones dentro de la misma, cosa que se analiza más adelante en el capítulo de pruebas y resultados.

Una vez que se ha llevado a cabo el proceso de procesamiento de los filtros en el nodo *sink*, es posible reconstruir el estado de la red, con la información obtenida de cuáles fueron los nodos que insertaron datos y cuáles fueron esos datos.

3.3 Conclusiones.

En este capítulo se propone un modelo de recolección de información en redes de sensores inalámbricos que hace uso de filtros de Bloom. Como ya se analizó anteriormente, éstos pueden traer enormes beneficios al implementarlos en el problema planteado debido a sus características y forma de funcionamiento.

Este capítulo presenta la idea mediante la cual se implementa el filtro de Bloom en el proceso de recolección de datos en la red. Se describe cómo es que se lleva a cabo el proceso, desde la inserción de datos en el filtro hasta el procesamiento de los mismos. Mediante la idea expresada en este trabajo, se propone optimizar el proceso de recolección de datos mediante la disminución en el número de transmisiones necesarias para ello. Esto traerá grandes beneficios en cuanto a ahorro en el consumo de energía se refiere, factor que es de suma importancia en este tipo de redes.

Capítulo 4 Modelo de Simulación.

Para realizar pruebas del modelo propuesto que hace uso de los filtros de Bloom, se utilizó el simulador *NS-2* para construir el modelo de pruebas correspondiente y poder de esta manera observar los resultados. A continuación se presenta una breve introducción al simulador *NS-2*, en donde se mencionan sus características principales y su forma de funcionamiento, para posteriormente pasar a la explicación de la forma en la que fue creado el modelo sobre dicho simulador.

4.1 *NS-2* (*Newtork Simulator Version 2*).

NS-2 es un simulador de redes de eventos discretos de código abierto que fue creado en 1989, como se describe en [20]. Desde su creación hasta nuestros días, *NS-2* ha estado en constante evolución, desarrollo y mejora, ya que ha incursionado en diferentes áreas de aplicación, como lo son la industria, el gobierno y la academia, entre otras. Hoy en día *NS-2* posee capacidades de simulación para un gran número de componentes de red, ya sea en redes cableadas o inalámbricas, y es uno de los simuladores de redes más conocidos y utilizados.

Para poder visualizar las simulaciones se hace uso de la herramienta de visualización *NAM* (*Network Animator*), la cual está basada en *Tcl/Tk* y permite realizar la visualización de los nodos que conforman la topología que se haya creado así como el intercambio de paquetes que se lleva a cabo en la simulación. Esta herramienta toma los datos necesarios para mostrar la simulación de un archivo de trazas, el cual es elaborado y arrojado como salida por el simulador *NS-2* [21].

4.1.1 Conceptos básicos en las simulaciones.

Tanto la topología que se desee simular así como los protocolos y características que describan su funcionamiento se deben escribir en un *script Tcl*, el cual puede ser escrito en cualquier editor de textos, como se explica en [22]. A continuación se presentan las características que debe de contener un *script* para simular una topología simple.

4.1.2 La topología.

El primer aspecto que se debe de tener claro al momento de pensar en la realización de una simulación es la topología que se va a implementar, es decir, la forma de la red o la manera en que los nodos se encuentran conectados entre sí y cómo es que se comunican.

4.1.3 Creación de nodos.

Los nodos son objetos que contienen los siguientes componentes [22]:

- Una dirección o identificador (id_) el cual se incrementa de 1 en 1 conforme más nodos van siendo creados.
- Una lista de vecinos (neighbor_).

- Una lista de agentes (agent_).
- Un identificador de tipo de nodo (nodetype_).
- Un módulo de enrutamiento.

La creación de nodos inalámbricos necesita la definición de algunas características antes de su creación como por ejemplo el modelo de energía utilizado, tipo de protocolo de enrutamiento, entre otros [21]. La definición de estos y otros parámetros se muestran en el ejemplo de la Figura 4.1.

```
# ======================================================================================================
# Define options
# ======================================================================================================
set opt(chan)           Channel/WirelessChannel          ;# Tipo de canal
set opt(prop)           Propagation/TwoRayGround         ;# Modelo de propagación
set opt(netif)          Phy/WirelessPhy                  ;# Tipo de interface de red
set opt(mac)            Mac/802_11                       ;# Tipo MAC
set opt(ifq)            Queue/DropTail/PriQueue          ;# Tipo de cola
set opt(ll)             LL                               ;# Tipo de capa de enlace
set opt(ant)            Antenna/OmniAntenna              ;# Tipo de antena
set opt(filters)        GradientFilter                   ;# Filtro utilizado
set opt(x)              2000                             ;# Tamaño en X del escenario
set opt(y)              2000                             ;# Tamaño en Y del escenario
set opt(ifqlen)         100                              ;# Número máximo de paquetes en cola
set opt(nn)             100                              ;# Número de nodos en la topología
set opt(sndr)           1                                ;# Número de nodos fuente
set opt(rcvr)           1                                ;# Número de nodos sink
set opt(stop)           15                               ;# Tiempo de simulación
set opt(tr)             "100nodes1sink1source.tr"        ;# archivo .tr
set opt(nam)            "100nodes1sink1source.nam"       ;# archivo .nam
set opt(adhocRouting)   Directed_Diffusion               ;# Protocolo de enrutamiento
# ======================================================================================================
```

Figura 4.1 Definición de los parámetros de los nodos en la red.

4.1.4 Enlaces.

Cuando la simulación que se desea crear cuenta con nodos alámbricos es necesario especificar los enlaces entre los nodos, como se presenta en [21], lo cual se realiza de la siguiente manera:

$ns duplex-link $n0 $n3 1Mb 100ms DropTail

La instrucción anterior es un ejemplo de la creación de un enlace entre dos nodos, en este caso nodo 0 y nodo 3. Se puede observar que en la instrucción también se define el tipo de enlace (*duplex*), el ancho de banda (1Mbps), el retardo (10 ms) y el tipo de cola que se utiliza (*DropTail*). En caso de que se esté trabajando con nodos inalámbricos, no es necesaria la definición de enlaces debido a que los nodos comparten el medio de transmisión que es el aire.

4.1.5 Transmisión de datos.

La transmisión de datos en una simulación de *NS-2* se lleva a cabo entre agentes, como se describe en [21], de un agente a otro. Los agentes más utilizados son *TCP* y *UDP*. La forma en la que se crean los agentes es la siguiente:

set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

En las instrucciones anteriores se presenta el ejemplo de la creación de un agente *UDP*, al cual se le asigna el nombre de udp0 y enseguida, en la segunda línea, se le agrega este agente a un nodo, en este caso el nodo 0. Este agente se encargará de enviar tráfico, para ello se necesita también crear un generador de tráfico, lo cual se realiza de la siguiente manera [21]:

set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0

En las líneas anteriores se crea un generador de tráfico *CBR*, al que se le asigna el nombre de cbr0 y enseguida se agrega al agente udp0 creado anteriormente. Con estos pasos ya se tiene listo el nodo que envía datos con su respectivo agente y generador de tráfico. El siguiente paso es crear también un agente que sea el encargado de recibir ese tráfico que será enviado, para esto es necesario crear un agente nulo, las siguientes instrucciones lo realizan [21]:

set null0 [new Agent/Null]
$ns attach-agent $n3 $null0

El agente nulo se le agrega a un nodo, en este caso al nodo 3 y una vez que se tienen creados y configurados los dos agentes, se conectan entre sí por medio de la siguiente instrucción:

$ns connect $udp0 $null0

Es posible indicar el tiempo en el que se desea iniciar y detener el envío de tráfico con el uso de las dos instrucciones siguientes:

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"

4.2 Elaboración del modelo de simulación en *NS-2*.

A continuación se describe la forma en la que fue construido el modelo así como todas sus características y forma de funcionamiento.

Se elaboró un modelo que consta de un escenario con 100 nodos inalámbricos estáticos. Estos 100 nodos están distribuidos aleatoriamente en todo el escenario, el cual tiene dimensiones de 2000 m. de ancho por 2000 m. de largo. Existe un nodo *sink*, el cual es el encargado de realizar la consulta a la red. El número de nodos fuente es inicialmente 1, pero este número se varió a lo largo de las pruebas para obtener los resultados necesarios y hacer las comparaciones correspondientes en el rendimiento de la red conforme se aumenta el número de nodos fuente. La Figura 4.2 muestra el escenario descrito anteriormente.

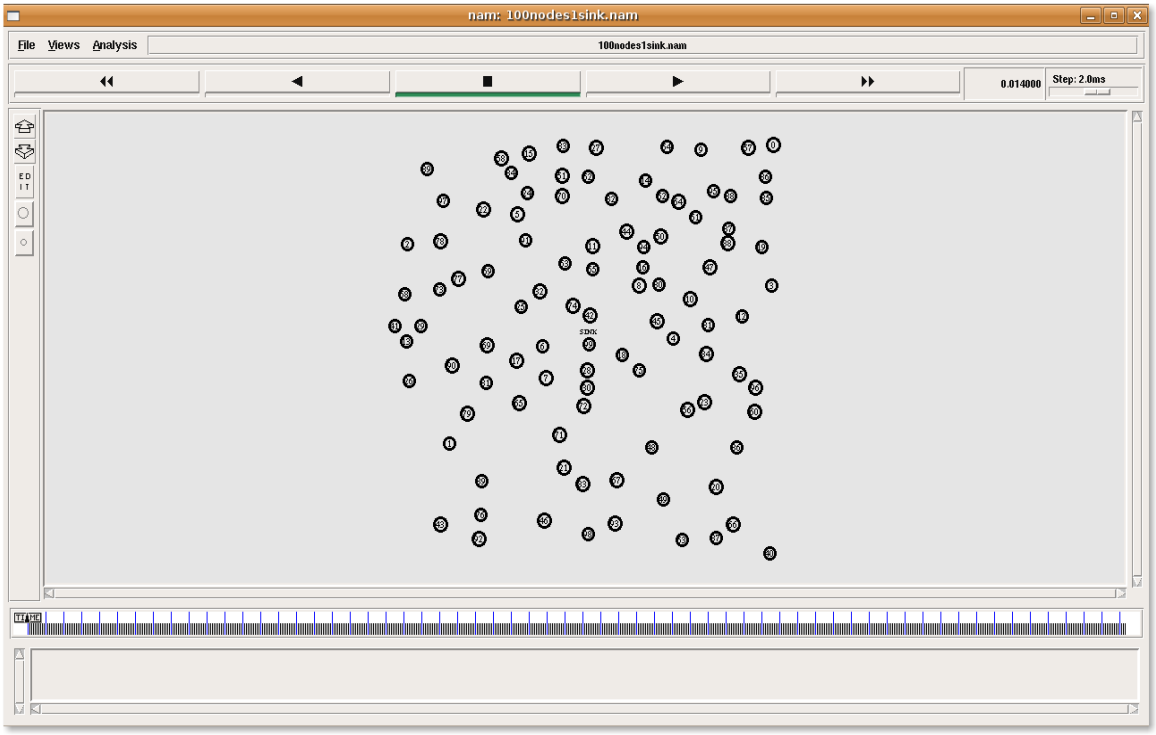


Figura 4.2 Escenario utilizado para las pruebas.

4.2.1 Implementación del filtro de Bloom.

Uno de los objetivos de este trabajo de investigación es reducir y optimizar, además del número de transmisiones, el número de *bytes* transmitidos en una red de sensores inalámbricos durante el proceso de recolección de la información. Para esto es necesario primero calcular el tamaño que debe de tener el filtro de Bloom que se le agregará a los paquetes transmitidos. Posteriormente se tomará en cuenta para calcular el tamaño completo del paquete en las pruebas que se hagan. En el capítulo 2 se realizó un análisis de los filtros de Bloom y la probabilidad de falsos positivos, de ese análisis se desprende lo siguiente:

Tomando el valor óptimo de funciones *hash*, la probabilidad de falsos positivos puede ser reescrita y acotada:

$$\frac{m}{n} \geq \frac{1}{\ln 2}$$

Donde m es el tamaño en *bits* del filtro de Bloom y n es el número de elementos a insertar en el filtro. Esto significa que para mantener una probabilidad de falsos positivos fija, la longitud del filtro de Bloom debe crecer linealmente con el número de elementos a introducir. El número requerido de *bits* (m), dado n y una probabilidad de falsos positivos p, y asumiendo el valor óptimo de k, es [14]:

$$m = -\frac{n \ln p}{(\ln 2)^2}$$

Los ejemplos que se realizaron fueron con 1, 5, 15 y 50 nodos que insertan datos al filtro, es por eso que se muestra la siguiente tabla en donde se resume el tamaño necesario del filtro Bloom para ese número de nodos. Se tomaron en cuenta varias probabilidades de falsos positivos y los valores fueron calculados con la fórmula anteriormente analizada.

Pfp / Fuentes	0.5	0.4	0.3	0.2	0.1	0.01	0.001	0.0001
1	2	2	3	4	5	10	15	20
5	8	10	13	17	24	48	72	96
15	22	29	38	1	72	144	216	288
50	73	96	126	168	240	480	719	959

En la tabla anterior se puede observar que si se desea una probabilidad de falsos positivos menor, el tamaño del filtro de Bloom será mayor.

Programación del filtro de Bloom.

Para poder trabajar con la idea presentada en este trabajo, es necesario modificar el código del simulador *NS-2* para de esta manera incluir el filtro de Bloom en los paquetes que son transmitidos a través de la red y así poder realizar las pruebas correspondientes. En todas las simulaciones que se realizan en *NS-2*, los paquetes que son transmitidos tienen cierto encabezado el cual contiene algunos campos que son comunes para todos los paquetes, sin importar de qué tipo sean. Otros campos son exclusivos para el tipo de paquete que se esté trabajando, dependiendo del protocolo que se implemente. Para este trabajo de investigación se tomó como base el protocolo *Directed Diffusion* para la propagación inicial de la consulta dentro de la red de sensores inalámbricos. Es por este motivo que se modificó el encabezado de los paquetes que utiliza este protocolo para agregarle un nuevo campo que represente el filtro de Bloom. Dentro de la estructura que define al encabezado del paquete, se agrega una variable que representa al filtro de Bloom, quedando definida de la manera mostrada en la Figura 4.3.

```
struct hdr_diff {
        int32_t         last_hop;
        int32_t         nexto_hop;
        int8_t          version;
        int8_t          msg_type;
        int16_t         data_len;
        int32_t         pkt_num;
        int32_t         rdm_id;
        int32_t         bloom[30];      //arreglo que representa el filtro de Bloom
        int16_t         num_attr;
        int16_t         src_port;
};
```

Figura 4.3 Estructura del encabezado del paquete con el filtro de Bloom agregado.

Como se puede observar en la figura anterior, la variable utilizada para representar el filtro de Bloom es un arreglo de 30 enteros de 32 *bits*. De esta manera se tienen 960 *bits* disponibles para el

filtro y la respectiva inserción de elementos en él. Como se mostró en la tabla que se presenta en el análisis del tamaño del filtro de Bloom, esta cantidad de *bits* es suficiente para almacenar la información insertada por 50 nodos fuente hasta para una probabilidad de falsos positivos de 0.0001. Lo cual es más que suficiente para las pruebas realizadas en este trabajo de investigación, pues como se muestra en las gráficas presentadas en el capítulo de pruebas y resultados, se utilizaron probabilidades de falsos positivos de 0.1 y 0.01 en todas las pruebas que se reportan, ya que con esa cifra se logra un buen funcionamiento del filtro.

4.2.2 Funcionamiento del modelo.

En las siguientes secciones se hace una descripción de la forma en que funciona el modelo que fue elaborado para realizar las pruebas en este trabajo de investigación

4.2.2.1 Inundación (*Flooding*) inicial.

Como todos y cada uno de los nodos que componen la red reciben el mensaje *interés* propagado en forma de *inundación (flooding)*, cada uno de ellos se encarga de establecer como ruta de regreso de datos el nodo del cual le llegó por primera ocasión el mensaje inicial transmitido por el *sink*. Es posible que si un nodo cuenta con varios vecinos dentro de su rango de cobertura, éste pueda recibir varios mensajes. La forma de funcionamiento de la *inundación (flooding)* es que si un nodo recibe varios mensajes iguales, éste solo transmite el primer mensaje recibido, descartando los demás. De esta manera se evitan mensajes duplicados y ciclos en la transmisión de los mensajes. Esta característica de la *inundación (flooding)* se aprovecha para hacer que cada nodo tome como ruta de regreso de datos a aquel nodo del cual le llegó el primer mensaje en la propagación del *interés*.

Después de que la *inundación (flooding)* inicial del *interés* se termina, cada uno de los nodos de la red tiene establecido una ruta de regreso de los datos, es decir, otro nodo al cual le enviará la información que reciba al momento de realizar el regreso de datos hacia el nodo *sink*. De esta manera, se puede decir que queda formado un árbol dentro de la red, donde cada uno de los nodos está conectado a alguna rama del mismo. La Figura 4.4 ilustra cómo es que queda formado este árbol para el regreso de datos hacia el *sink*. Es importante mencionar que las líneas mostradas en la Figura 4.4 no representan enlaces alámbricos, sino que son únicamente para ilustrar las rutas de envío de datos.

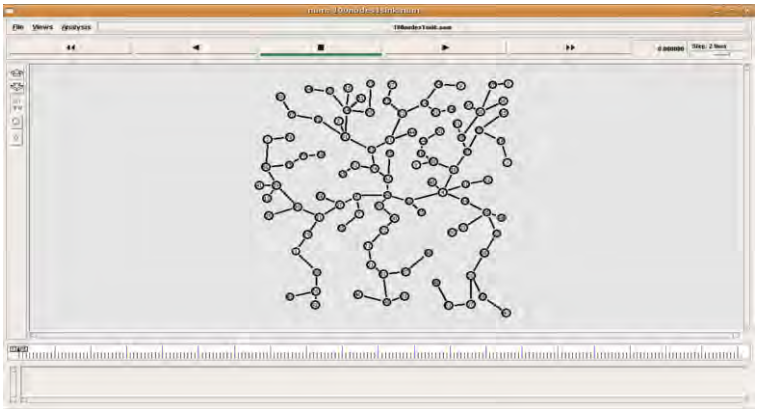


Figura 4.4 Representación de las rutas de regreso de datos hacia el *sink*.

También es importante recordar que después de la finalización de la transmisión del mensaje *interés* en forma de *inundación (flooding)*, cada uno de los nodos de la periferia (nodos hojas en el árbol) mantiene un filtro de Bloom. Este filtro contiene toda la información de los nodos recolectada a lo largo de la rama del árbol a la que pertenece. Como ya se explicó anteriormente en el capítulo 3, esta información fue recolectada durante el proceso de transmisión del mensaje *interés*.

4.2.2.2 Regreso de datos al *sink*.

Como ya se mencionó anteriormente, la información recolectada y almacenada en los filtros se encuentra en los nodos finales de las rutas de regreso de datos, los cuales se había mencionado que podían verse como hojas del árbol de regreso de datos que se forma. Estos nodos son los encargados de iniciar el regreso de los filtros de Bloom hacia en nodo *sink* para el posterior procesamiento de los datos. El regreso de los datos hacia el nodo *sink* se lleva a cabo a través de las rutas creadas durante la propagación del mensaje *interés*.

Para la realización de este proceso se elaboró un programa en *matlab*, el cual toma como entrada los filtros de Bloom que se encuentran en los nodos finales de las rutas de regreso de datos. Estos filtros contienen la información que los nodos fuente insertaron durante el proceso de propagación del mensaje *interés*. Una vez teniendo estos filtros de Bloom y conociendo las rutas de regreso de la información en la red, formadas anteriormente, el programa mencionado simula el proceso de regreso de los datos hacia el *sink*. Se toma en cuenta todo lo necesario para eliminar información duplicada y hacer agregación de filtros. Una vez realizado este proceso, el programa también muestra los filtros de Bloom resultantes que llegan al nodo *sink*, para su posterior interpretación y obtención de la información de la red. La Figura 4.5 muestra la interface del programa elaborado.

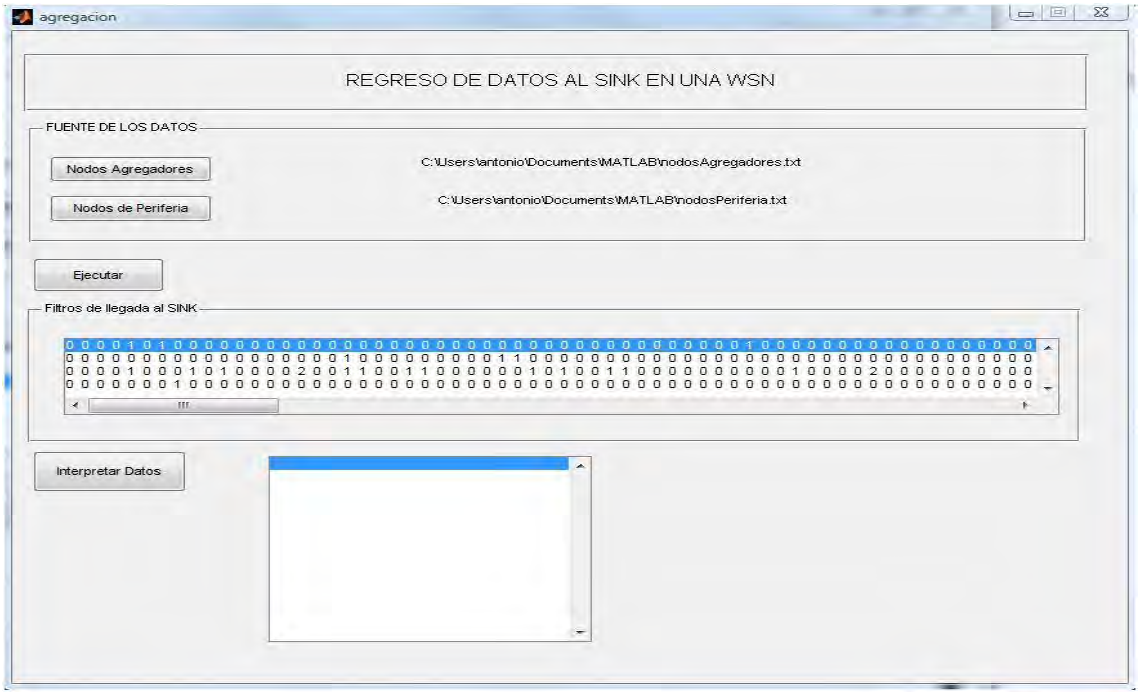


Figura 4.5 Programa para simular el regreso de datos al *sink* y la obtención de resultados.

Como se observa en la interface del programa elaborado, éste también solicita como entrada aquellos nodos que sean nodos agregadores de filtros, es decir, aquellos nodos que reciben filtros provenientes de más de uno de sus vecinos, para de esta manera poder llevar a cabo la eliminación de información duplicada y la agregación de filtros de Bloom.

4.2.2.3 Interpretación de los filtros de Bloom.

El proceso de interpretación de los datos existentes en un filtro de Bloom fue explicado en secciones anteriores. El programa elaborado para llevar a cabo el regreso de datos hacia el nodo *sink* también realiza este proceso de interpretación. Para dicho propósito cuenta con un botón en la parte inferior, el cual al presionarlo, mostrará en la caja de texto la información recibida en el *sink* por medio de los filtros de Bloom. También presentará una gráfica que muestra la red de sensores inalámbricos del modelo con el que se está trabajando y, en aquellos nodos que insertaron datos al filtro, muestra el dato insertado en la posición del nodo. En la Figura 4.6 se muestra un ejemplo de la interpretación de datos en el programa en cuestión, además se presenta la gráfica resultante para este caso.

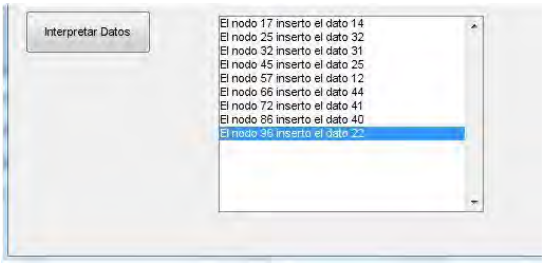


Figura 4.6 Interpretación de datos.

Como se observa en la figura anterior, el programa presenta los datos introducidos por los nodos fuente, así como el número de nodo que lo insertó. También muestra la gráfica de la red con los datos recién obtenidos, esto se muestra en la Figura 4.7.

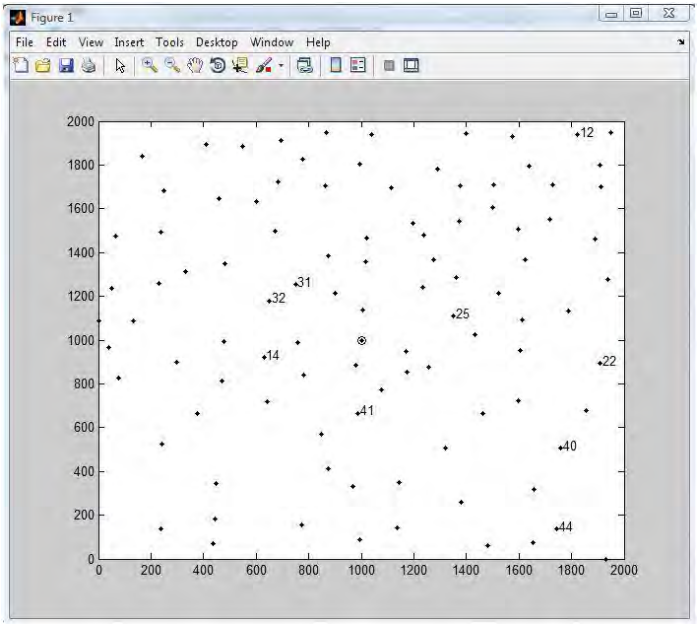


Figura 4.7 Recreación de la red mostrando los datos recolectados.

En la figura anterior se muestra la red completa, en donde podemos observar que al centro se encuentra el nodo *sink* (representado por un círculo). Además se puede apreciar el dato que cada uno de los nodos fuente insertó en los filtros de Bloom de la manera explicada en secciones anteriores.

4.3 Conclusiones.

Sin lugar a dudas, *NS-2* se ha convertido en el simulador de redes de código abierto más utilizado y en uno de los simuladores de redes más usados en general. La razón principal por la que se eligió el simulador *NS-2* es porque sus características se adaptan a las necesidades que se presentan en la elaboración del modelo que se utiliza en este trabajo de investigación. Además, al ser libre y de código abierto permite la creación o modificación de módulos que puedan ser necesarios. Por último, es un simulador que se encuentra en constante desarrollo e investigación, lo que permite la futura adición de nuevas características y funciones.

Capítulo 5 Pruebas y Resultados.

Para comparar el funcionamiento del método de recolección de datos por medio de filtros de Bloom en una red de sensores inalámbricos con los otros protocolos analizados en los capítulos anteriores se utilizó un escenario con las siguientes características:

- Escenario de 2000 por 2000 metros.
- 100 nodos sensores estáticos.
- Posiciones aleatorias de los nodos.
- 1 nodo *sink* localizado en el centro del escenario.
- Número de nodos fuente variable.

El escenario utilizado para las pruebas se muestra en la siguiente figura:

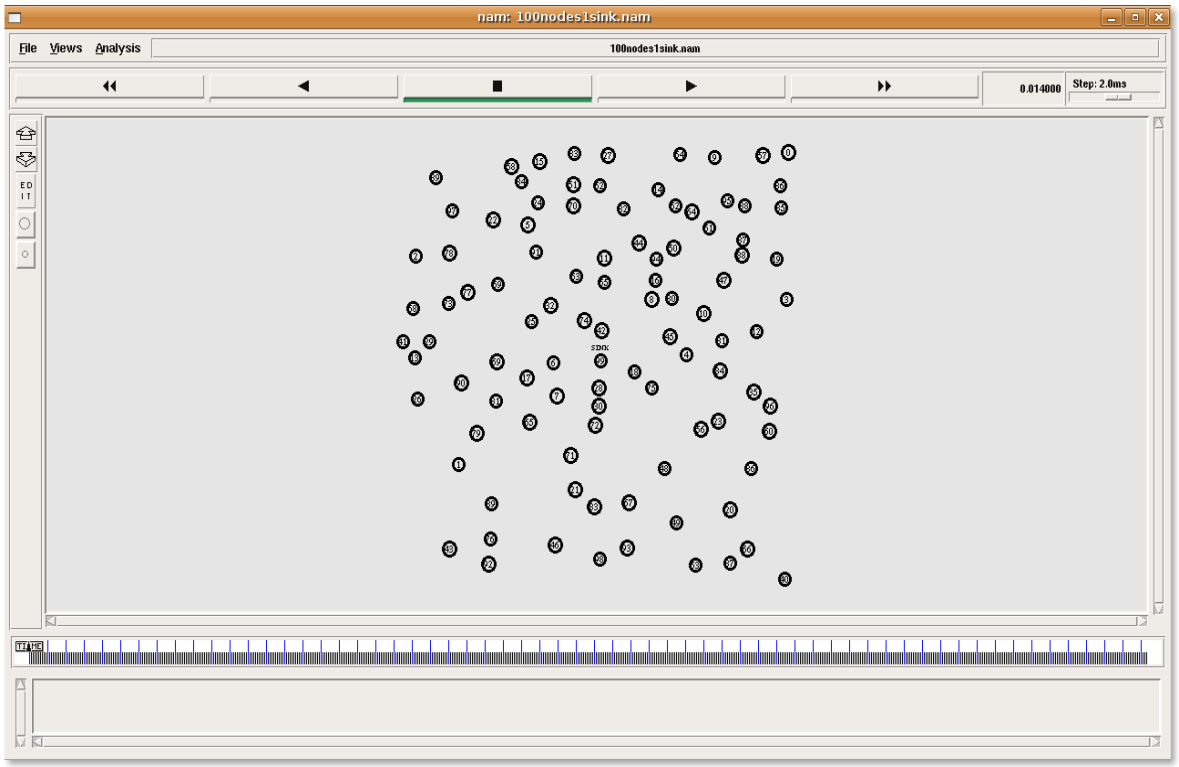


Figura 5.1 Escenario de pruebas.

Se realizaron pruebas con 1, 2, 3,. . ., 50 nodos fuente para los 4 modelos de recolección de datos mencionados:

- *Directed Diffusion*.
- *Cluster Head*.

- *Cluster Head* con concatenación.
- Filtros de Bloom.

Los aspectos que se estudiaron fueron:

- Número de transmisiones realizadas.
- Número de *bytes* transmitidos.
- Energía consumida.
- Tiempo de vida de baterías usando dicho modelo.

5.1 Número de transmisiones realizadas.

Tomando como escenario el presentado anteriormente, se ejecutaron sobre él los 4 modelos de recolección de datos analizados. Así fue posible obtener estadísticas a cerca del número de transmisiones realizadas en cada uno de ellos cuando el nodo *sink* realiza una consulta a la red y un número variable de nodos fuente responden a la petición. En la gráfica de la Figura 5.2 se muestra el número de transmisiones realizadas en cada caso, el tamaño del paquete transmitido es de 52 *bytes* ya que ese es el tamaño que tienen los paquetes en el simulador *NS-2*.

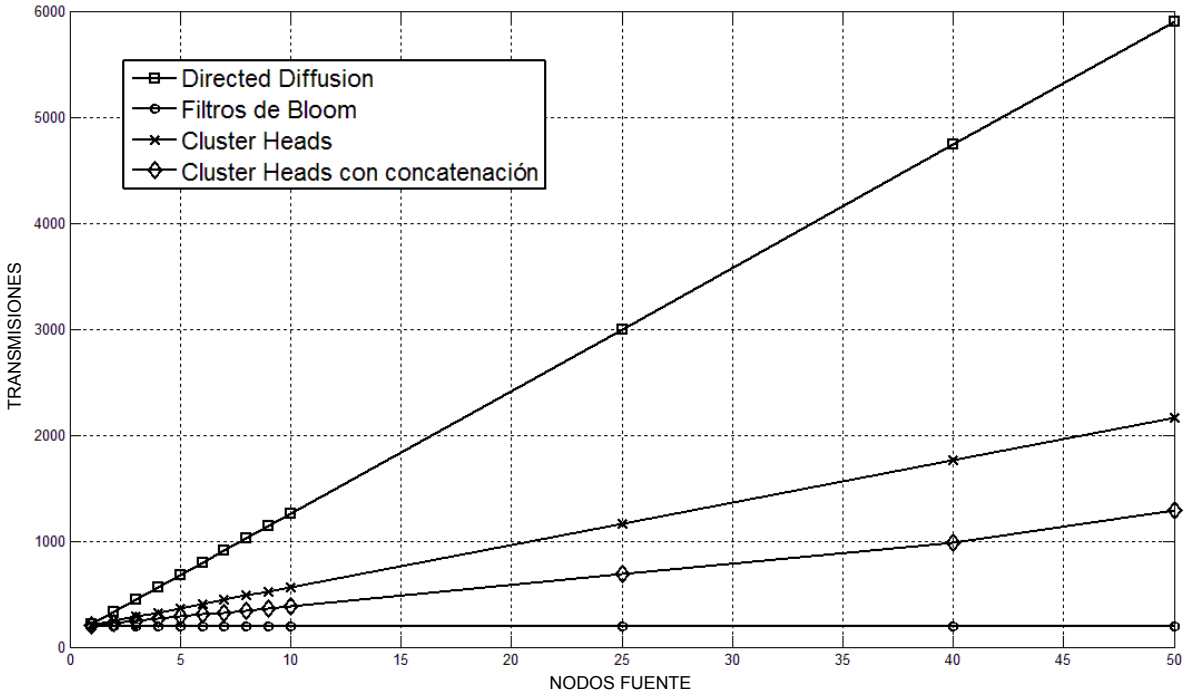


Figura 5.2 Gráfica de las transmisiones generadas en la red con 100 nodos durante la recolección de datos.

En la gráfica de la Figura 5.2 se puede observar que cuando se utiliza el modelo de filtros de Bloom para la recolección de datos dentro de la *WSN*, el número de transmisiones realizadas permanece constante sin importar que el número de nodos fuente aumente. Esto constituye una disminución significativa en el número de transmisiones realizadas en comparación con los otros modelos de recolección de la información dentro de la red. En la gráfica de la figura se observa que tanto para los modelos de *Cluster Heads* como para *Directed Diffusion* el hecho de que haya un mayor número de nodos fuente dentro de la *WSN* sí repercute en el número de transmisiones realizadas.

En *Directed Diffusion* podemos observar el peor caso, pues el número de paquetes transmitidos crece considerablemente conforme se aumenta el número de nodos fuente. Para los modelos de *Cluster Heads* el número de transmisiones también se incrementa al aumentar el número de nodos fuente, pero no en demasía, por lo que se coloca por debajo de *Directed Diffusion*.

Es importante hacer notar cómo es que la curva que representa el número de transmisiones realizadas usando *Directed Diffusion* crece en una mayor proporción en comparación con las curvas de los otros 3 modelos. Basándose en lo anterior, se puede deducir que *Directed Diffusion* es el método que más perjudicado se ve cuando se aumenta el número de nodos fuente. Si se compara esa curva con la que representa al método de filtros de Bloom, se puede observar la gran diferencia que hay entre una y otra en cuanto al número de transmisiones realizadas se refiere. La gráfica de la figura anterior es bastante significativa pues se puede apreciar el ahorro en transmisiones que se tiene dentro de la *WSN* al ejecutar el modelo basado en filtros de Bloom en comparación con los otros modelos analizados.

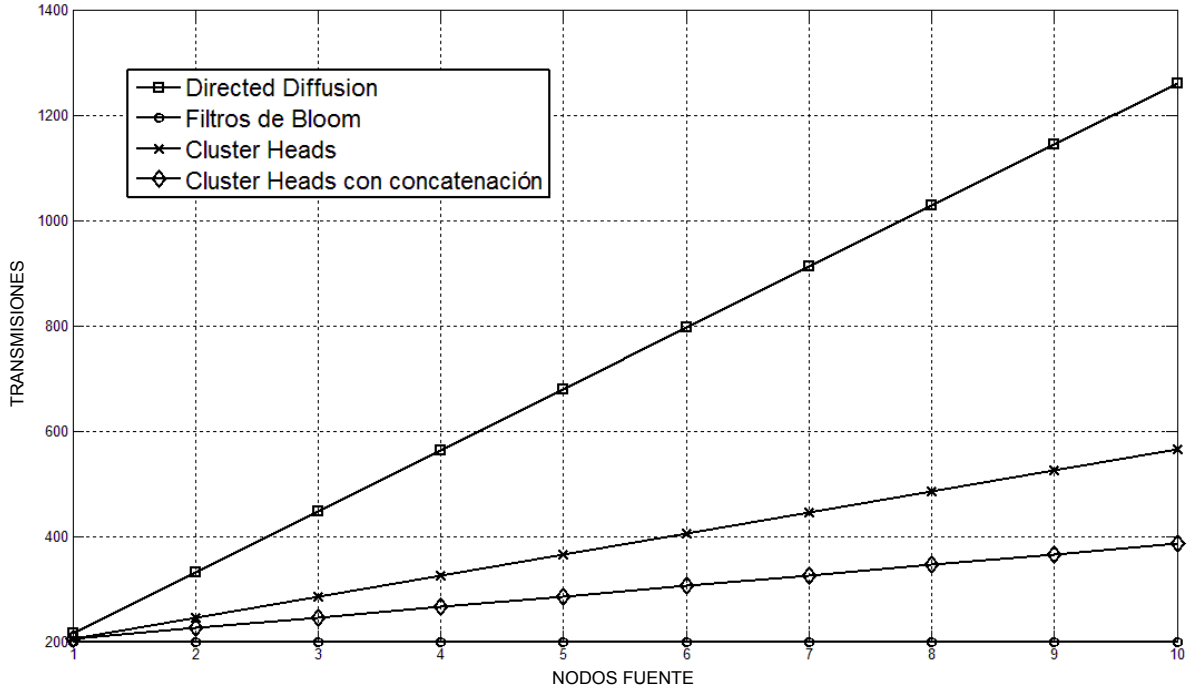


Figura 5.3 Gráfica de las transmisiones generadas en la red por hasta 10 nodos durante la recolección de datos.

La gráfica de la Figura 5.3 muestra un acercamiento a la primera parte de la gráfica de la Figura 5.2, para cuando el número de nodos fuente varía de 1 a 10. En esta gráfica se puede apreciar de mejor manera el comportamiento de los modelos analizados cuando el número de nodos fuente es pequeño. Se puede observar que cuando se cuenta solamente con 1 nodo fuente, todos los modelos realizan un número similar de transmisiones para la recolección de datos dentro de la red. A partir de 2 nodos fuente en adelante, se empiezan a marcar las diferencias en el número de transmisiones entre los diferentes modelos.

5.2 Número de *bytes* transmitidos.

Para calcular el número de *bytes* transmitidos durante la recolección de información dentro de la *WSN* cuando el nodo *sink* realiza una consulta a la red, es necesario conocer el tamaño del paquete que se transmite. En el simulador *NS-2*, tanto para los modelos de C*luster Heads* como para *Directed Diffusion*, el tamaño de los paquetes transmitidos es de 52 *bytes*, tomando en cuenta el encabezado y la parte de datos. Para el caso del método que hace uso de los filtros de Bloom, el paquete contiene además un campo adicional que es el que representa al filtro de Bloom, por lo que el tamaño de este paquete es mayor.

El tamaño del filtro de Bloom depende del número de nodos fuente que haya en la *WSN*. Cuando existe un número pequeño de nodos fuente el tamaño del filtro es menor, mientras que cuando se aumenta el número de nodos fuente el tamaño del filtro debe de ser mayor. El incremento en el tamaño del filtro es para poder almacenar de buena manera la información insertada en él y mantener constante la probabilidad de falsos positivos.

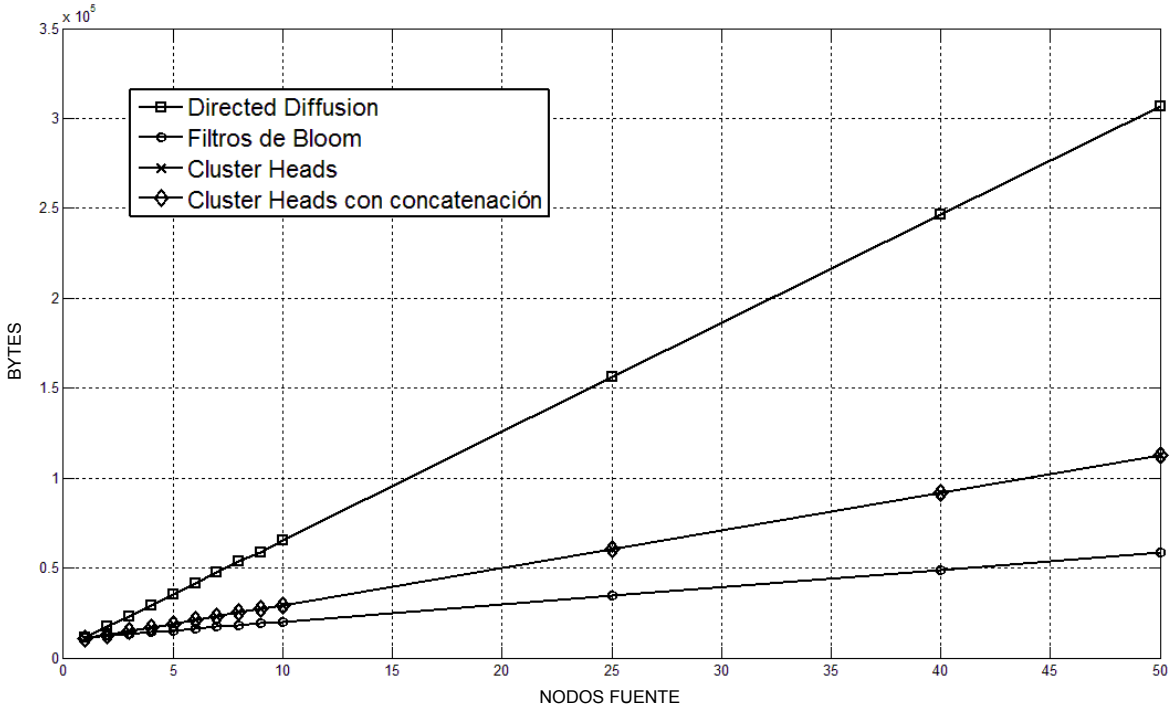


Figura 5.4 Gráfica del número de *bytes* transmitidos en la red con 100 nodos durante la recolección de datos.

En la gráfica de la Figura 5.4 se observa el número de *bytes* transmitidos dentro de la *WSN* cuando el nodo *sin*k realiza una consulta a la red y un número variable de nodos fuente responden a ella. Se puede observar que el caso en el que más *bytes* se transmiten es *Directed Diffusion*, debido a que el número de paquetes transmitidos es muy superior en comparación con los otros modelos. Los dos modelos de *Cluster Heads* realizan la transmisión del mismo número de *bytes*. Aunque en el método de *Cluster Heads* con concatenación se realice un número de transmisiones menor, los paquetes son de mayor tamaño, pues dentro de cada uno van varios paquetes con la información concatenada. Se observa también que el número de *bytes* transmitidos con el método de los filtros de Bloom va en aumento conforme hay más nodos fuente dentro de la red. Esto se debe a que, aunque el número de paquetes transmitidos permanezca constante, el tamaño de ellos depende del número de nodos fuente. Cuando existe una mayor cantidad de nodos fuente se tendrá que aumentar el tamaño del filtro de Bloom, lo que aumentará el tamaño del paquete.

Aunque en este caso todas las curvas vayan en crecimiento conforme se incrementa la cantidad de nodos fuente, se observa que el modelo que hace uso de los filtros de Bloom consigue realizar la consulta dentro de la *WSN* y la recopilación de información con una menor cantidad de *bytes* transmitidos, en comparación con los otros tres modelos analizados. Es importante mencionar que para la realización de esta prueba se definió tener una probabilidad de falsos positivos de 0.01 para el modelo que hace uso de los filtros de Bloom. Esta probabilidad es importante conocerla pues afectará en el tamaño del filtro y por lo tanto en el tamaño del paquete.

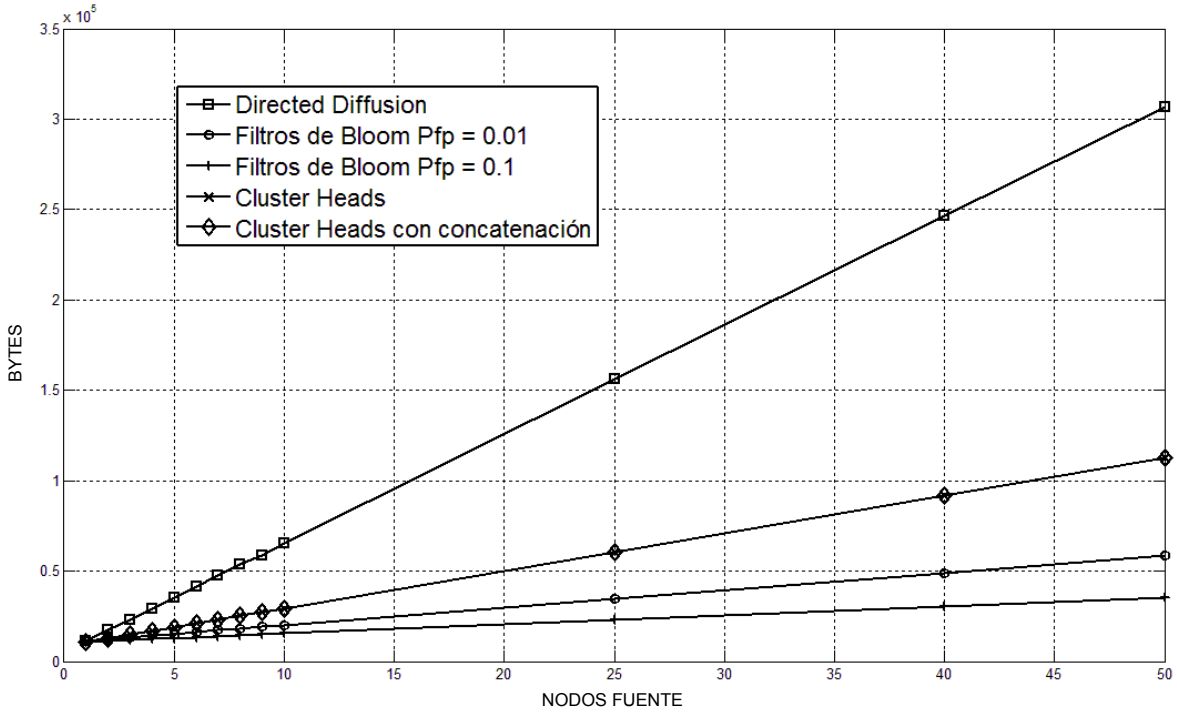


Figura 5.5 Gráfica del número de *bytes* transmitidos en la red con 100 nodos durante la recolección de datos para diferentes Pfp.

La gráfica de la Figura 5.5 también muestra el número de *bytes* transmitidos dentro de la red, pero en esta ocasión se agrega una curva la cual representa el número de *bytes* transmitidos mediante el modelo que hace uso de los filtros de Bloom usando una probabilidad de falsos positivos de 0.1.

Como se observa en la gráfica de la figura anterior, cuando se usa una probabilidad de falsos positivos mayor para el método de filtros de Bloom, el número de *bytes* transmitidos disminuye. Esto es en comparación a la cantidad de *bytes* transmitidos por el mismo método con una probabilidad de falsos positivos menor. Esto se debe a que al utilizar una probabilidad de falsos positivos mayor, el tamaño del filtro de Bloom disminuye, y por lo tanto el tamaño del paquete transmitido también disminuye, logrando de esta manera realizar la transmisión de una menor cantidad de *bytes*.

Aunque al aumentar la probabilidad de falsos positivos se logra que el tamaño del paquete disminuya y por lo tanto se transmitan una menor cantidad de *bytes*, es importante tomar en cuenta los problemas que pueden surgir al aumentar esta probabilidad. Cuando se aumenta la probabilidad de falsos positivos, se tiene una mayor probabilidad de leer información errónea del filtro de Bloom. Esto se debe a la forma de operar de este tipo de filtros, en especial a la forma en la cual la información es almacenada y recuperada dentro de ellos.

Como se pudo observar en las gráficas de las figuras anteriores, el hecho de aumentar la probabilidad de falsos positivos disminuye la cantidad de *bytes* que se transmiten dentro de la red. Pero los problemas que esto acarrea pueden ser graves, provocando datos equivocados al momento de interpretar la información recolectada por el filtro de Bloom en la red. En nuestro caso y para las pruebas realizadas se prefiere permanecer con una probabilidad de falsos positivos de 0.01 o menor, la cual brinda un buen funcionamiento del filtro de Bloom y buenos resultados en lo que a número total de *bytes* transmitidos se refiere.

5.3 Energía consumida.

Para obtener las mediciones de la cantidad de energía consumida por nodo dentro de la red al momento de recopilar la información, se tomó como modelo el nodo sensor *IRIS*, de la compañía *Crossbow*, el cual cuenta con las siguientes características:

- Consumo en modo de recepción: 16 mA.
- Consumo en modo de transmisión: 10 mA.
- Batería: 2 baterías AA.
- Voltaje: 2.7 V – 3.3 V.

De los datos anteriores, podemos observar que el consumo en mA por nodo al realizar el proceso de transmisión y recepción por paquete es de 26 mA. Además se sabe que se cuenta con una fuente de alimentación de 2.7 V – 3.3 V, y conociendo estos datos podemos aplicar la siguiente fórmula para calcular el consumo en Watts:

$$P = V * I$$

Donde P es la potencia en Watts, V es el voltaje e I es la corriente en Amperes. Aplicando esta fórmula se tendrá la potencia consumida en Watts por cada nodo en el proceso de transmisión y recepción de cada paquete. Para hacer la conversión a Joules se hace uso de la siguiente relación:

$$1J = 1W * s.$$

De esta manera, tomando en cuenta los datos de consumo presentados anteriormente, así como la fuente alimentación con la que cuentan los nodos, se puede calcular el consumo en Joules mediante el proceso que se describió arriba. Se tomaron en cuenta también los datos del número de paquetes transmitidos en cada método de recolección de información analizado. La gráfica de la Figura 5.6 muestra el promedio en el consumo de Joules por nodo de la red para cada caso.

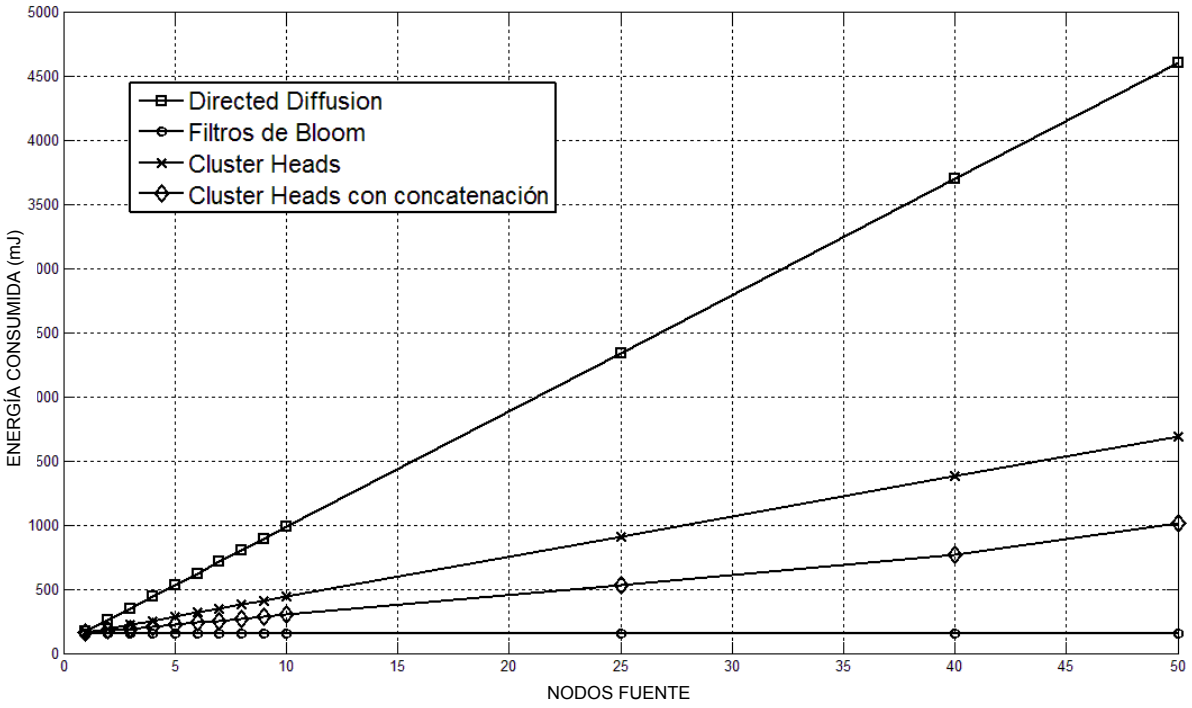


Figura 5.6 Gráfica del consumo de energía por nodo en la red durante la recolección de información.

En la gráfica de la Figura 5.6 se observa que el método que hace uso de los filtros de Bloom provoca un consumo menor de energía por cada nodo de la red comparado con los otros métodos.

Se puede apreciar que la ganancia es considerable, especialmente si se compara con *Directed Diffusion*, en donde el consumo es más elevado.

5.4 Ejemplo de consumo de energía en una *WSN*.

Una vez teniendo en cuenta todos los parámetros anteriormente analizados, a continuación se presenta un ejemplo donde se muestra el tiempo de vida de las baterías para una *WSN* particular, la cual cuenta con las siguientes características:

- 100 nodos.
- 1 nodo *sink*.
- 5 nodos fuente.
- Consultas de información cada 12 horas.

La gráfica de la Figura 5.7 muestra cómo es que se lleva a cabo en promedio el proceso de descarga de las baterías dentro del ejemplo con los parámetros anteriormente mencionados. Es importante mencionar que para este ejemplo nuevamente se tomó como modelo el nodo sensor *IRIS* y sus características de consumo de energía.

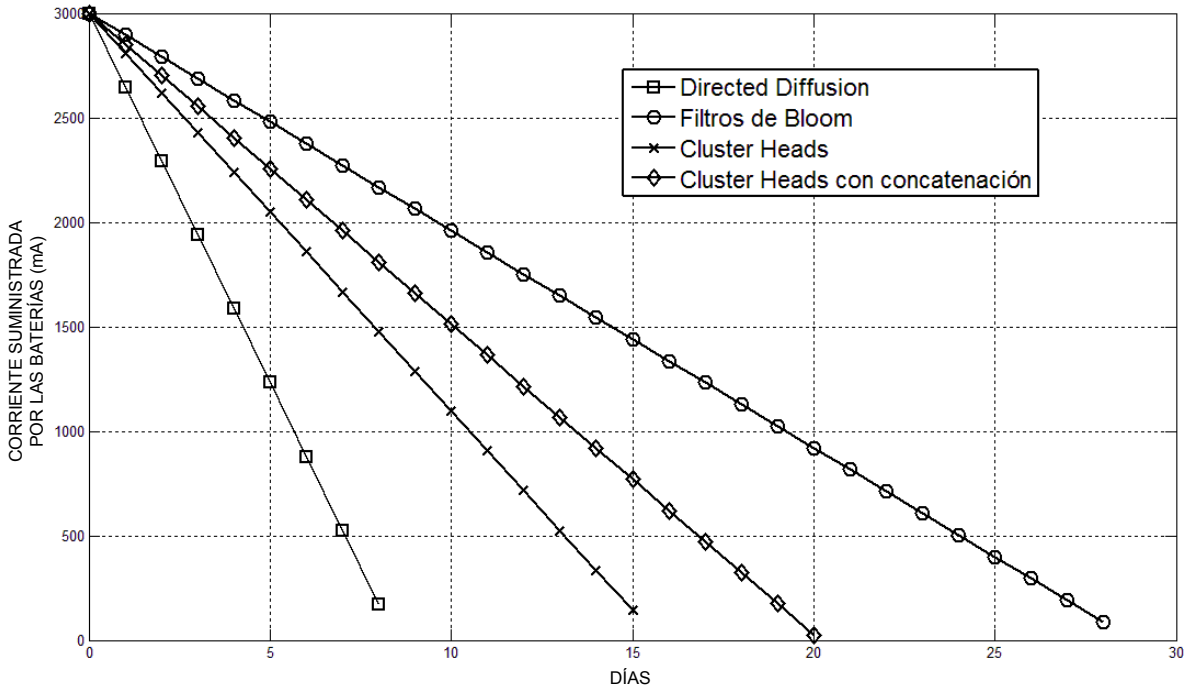


Figura 5.7 Gráfica del tiempo de descarga de las baterías en los nodos de la red.

En la gráfica de la Figura 5.7 se puede observar que el modelo que ofrece un mayor ahorro de energía y por lo tanto un mayor tiempo de vida de las baterías de los nodos dentro de la red es el método que hace uso de los filtros de Bloom. Los modelos de *Cluster Heads* quedan ubicados en la parte central de la gráfica, con un menor tiempo de vida de las baterías comparado con los filtros de Bloom, mientras que con *Directed Diffusion* se logra un tiempo de vida de las baterías considerablemente menor al de los otros modelos.

Los resultados obtenidos en la figura anterior se deben a que el método de filtros de Bloom realiza tanto un menor número de transmisiones como una menor transmisión de *bytes* dentro de la red. Debido a eso el ahorro de energía es significativo comparado con los otros modelos analizados.

5.5 Conclusiones.

A partir de los resultados obtenidos y que se muestran en las figuras, es posible mencionar que la implementación del modelo que hace uso de los filtros de Bloom logra una disminución significativa en el número de transmisiones realizadas durante el proceso de recolección de datos. Lo anterior se menciona en comparación con los otros protocolos analizados y contra los cuales se realizaron las pruebas y comparaciones en las figuras que se mostraron en este capítulo.

Como consecuencia de la disminución en el número de transmisiones, también se observa que el número de *bytes* transmitidos en total es menor, sin importar que se agregue el filtro de Bloom a los paquetes, ya que el beneficio que se obtiene es mayor. Así mismo se puede observar también que el consumo de energía en la red es menor cuando se utiliza el modelo propuesto en este trabajo. Esto se da como consecuencia del hecho de la disminución en el número de transmisiones y en el número de *bytes* transmitidos.

Capítulo 6 Conclusiones.

En este trabajo se realizó una revisión de los protocolos de enrutamiento en redes de sensores inalámbricos y su forma de funcionamiento, para después profundizar en dos de ellos: *Directed Diffusion* y *Cluster Based Routing Protocol*. Se puso especial atención en lo referente al concepto de energía dentro de este tipo de redes, ya que el objetivo principal de este trabajo está enfocado a la disminución en el consumo de energía en los nodos de la red.

Para cumplir el objetivo de reducir el consumo de energía en la red, se presentó un modelo de recolección de datos que hace uso de los filtros de Bloom para dicho proceso. El objetivo de este modelo es disminuir el número de transmisiones necesarias para llevar a cabo la recolección de datos en una red de sensores inalámbricos. Para demostrar la eficiencia del modelo propuesto se comparó con los protocolos mencionados anteriormente.

Para la realización de las pruebas se construyó un escenario en el que se analizaron los protocolos que sirvieron como punto de comparación para el modelo propuesto. Se analizó lo referente al número de transmisiones realizadas por los modelos durante el proceso de recolección de información. De esta manera también se analizó el modelo que hace uso de los filtros de Bloom, en donde los parámetros que se estuvieron variando fueron el número de nodos que reportan datos en la red así como la probabilidad de falsos positivos, concepto importante para los filtros de Bloom.

Se analizó detalladamente el proceso de recolección de información en los tres modelos de interés para este trabajo: *Directed Diffusion*, *Cluster Based Routing Protocol* y el modelo aquí propuesto. Las pruebas realizadas fueron para obtener datos a cerca del número de transmisiones para cada caso. Lo que se pretende con el modelo propuesto es disminuir el número de transmisiones dentro de la red para así lograr un menor consumo de energía por parte de los nodos que la conforman, y por lo tanto un mayor tiempo de vida de las baterías.

En los resultados obtenidos de la realización de las pruebas se puede observar que el modelo que hace uso de los filtros de Bloom, propuesto en este trabajo, consigue llevar a cabo el proceso de recolección de la información dentro de la red de sensores inalámbricos con un menor número de transmisiones. Todo lo anterior en comparación con los protocolos *Directed Diffusion* y *Cluster Based Routing Protocol*. Este resultado es importante ya que el lograr una disminución en el número de transmisiones en la red implica una reducción en el consumo de energía, y como se sabe, la energía es un recurso muy importante y crucial en este tipo de redes.

También se presentan resultados a cerca de la cantidad de *bytes* transmitidos durante el proceso de recolección de información dentro de la red. En ellos se puede observar que, a pesar de que a los paquetes que se transmiten usando el modelo propuesto en este trabajo se les agrega un campo adicional para representar el filtro de Bloom, la cantidad de *bytes* que se transmiten en total en el proceso es menor en comparación a los otros modelos presentados. Este resultado está ligado al anterior, de esta manera se puede decir que el modelo propuesto en este trabajo logra disminuir el número de transmisiones y la cantidad de información transmitida durante el proceso de recolección de datos.

Es importante mencionar que los resultados obtenidos relacionados con la disminución del número de transmisiones y cantidad de *bytes* transmitidos se logran sin importar el número de nodos fuente con los que la red cuente, ya que conforme aumenta el número de nodos fuente en la red, la diferencia entre los protocolos analizados y el modelo propuesto se hace mas grande, es decir, se obtienen mejores resultados.

Se presenta además un análisis a cerca del consumo de energía, el cual está ligado directamente con el número de *bytes* transmitidos y el número de transmisiones realizadas en el proceso de recolección de la información. De esta manera se presentan también resultados a acerca del tiempo de vida de las baterías de los nodos dentro de la red, tomando en cuenta los protocolos analizados y el modelo que se propone en este trabajo. En estos resultados se puede observar que efectivamente el consumo de energía en la red disminuye cuando se utiliza el modelo propuesto en este trabajo. Como consecuencia de los beneficios que resultan de la utilización del modelo propuesto, el tiempo de vida de las baterías de los nodos aumenta, logrando extenderlo significativamente en comparación con los otros protocolos analizados.

Lo que se logró con este trabajo es una disminución tanto en el número de transmisiones realizadas como en la cantidad de *bytes* transmitidos durante el proceso de recolección de datos en la red, lo que implica una disminución en el consumo de energía de los nodos. Debido a la forma de funcionamiento del modelo propuesto, los nodos tienen que realizar un procesamiento extra al introducir los datos al filtro de Bloom, el cual no tiene demasiadas complicaciones si se considera que las operaciones a realizar son simples. De igual forma, el nodo *sink*, al momento de obtener los datos del filtro de Bloom, realiza un proceso más largo el cual puede consumir más recursos del nodo. A pesar de esto, la disminución en el consumo de energía que se logra mediante la implementación del modelo propuesto en este trabajo es considerable y es por eso que no se mencionan demasiado las cargas de procesamiento para los nodos.

6.1 Trabajos futuros.

Las comunicaciones inalámbricas están en constante desarrollo e investigación, continuamente surgen nuevas tecnologías y se proponen nuevas aplicaciones que permiten el mejoramiento de los sistemas, protocolos y modelos existentes. Este trabajo propone una forma de implementar los filtros de Bloom para la recolección de información en las redes de sensores inalámbricos. Aquí se presenta una manera específica de la inserción de datos dentro de la estructura, sin embargo, este hecho puede ser mayormente explotado, buscando nuevas formas de implementar el filtro o aprovechar la existencia de éste dentro de los paquetes que circulan en la red para otros fines. Esto podría justificar mayormente la adición de esta estructura dentro de los paquetes, teniendo siempre en cuenta el costo que esto implica contra los beneficios que se obtienen.

De esta manera, queda a trabajos futuros una distinta implementación de los filtros de Bloom para la recolección de información en las redes de sensores inalámbricos. Buscando otras formas de inserción de datos, o variantes, se podrían obtener buenos resultados en cuanto a la disminución de transmisiones y optimización de la energía se refiere. También puede seguirse explorando la disminución en el consumo de energía mediante la optimización de otros procesos, como el procesamiento de datos en los nodos de la red.

Un aspecto igualmente importante a tomar en cuenta en trabajos futuros es la seguridad de la información transmitida durante el funcionamiento del modelo propuesto en este trabajo. Es importante poder asegurar, a través de algún mecanismo, que los datos insertados en los filtros de Bloom o lo mismos filtros no puedan ser modificados o manipulados por nodos ajenos a la red o por nodos que se hagan pasar como parte de ella.

Glosario de términos y acrónimos.

CAT	Cluster Adjacency Table.
CBR	Constant Bit Rate.
CBRP	Cluster Based Routing Protocol.
Cluster	Conjunto de componentes que interactúan entre sí.
Cluster Head	Componente o nodo central dentro de un cluster.
DD	Directed Diffusion.
Duplex	Se refiera a comunicación bidireccional, enviar y recibir mensajes de forma simultánea.
Flooding	Inundación, mecanismo de difusión de información dentro de una red, el cual consiste en enviar los mensajes hacia los nodos vecinos sucesivamente hasta que éste llegue a todos los nodos de la red .
Función hash	Método para generar claves de manera casi unívoca a un documento, archivo, elemento, etc.
Gateway	Nodo por medio del cual se lleva a cabo la comunicación hacia el exterior de un cluster.
Gradiente	Dirección creada por cada uno de los nodos de la red en Directed Diffusion cuando éstos reciben el mensaje interés. Cada nodo crea esta dirección hacia el nodo del cual recibió dicho mensaje.
Interés	Mensaje enviado por el nodo sink en Directed Diffusion para realizar alguna consulta o tarea de sensado a la red.
IP	Internet Protocol.
LAN	Local Area Network.
MAC	Media Access Control.
Mica2	Nodo sensor desarrollado por la compañía Crossbow.
Multicast	Envío de la información en una red a múltiples destinos simultáneamente.
NAM	Network Animator. Herramienta de animación para visualizar las simulaciones en NS2.
NS2	Network Simulator Version 2.
OR	Tipo de operación lógica.
P2P	Peer-to-peer. Tipo de red en donde los nodos se comportan como iguales entre sí, sin clientes ni servidores fijos.
Script	Archivo de órdenes creado para su posterior interpretación.
Sink	Nodo central o principal en una red de sensores inalámbricos, es el encargado de realizar las consultas a la red.
Sleep	Periodo en el cual un objeto no se encuentra realizando las actividades para las cuales fue creado.
Tcl	Tool Command Language. Utilizado para el desarrollo rápido de prototipos, aplicaciones “script”, interfaces gráficas y pruebas.
TCP	Transfer Control Protocol. Protocolo de comunicación orientado a conexión y fiable del nivel de transporte.
Tk	Tool Kit. La combinación con Tcl, llamada Tcl/Tk, se utiliza para la creación de interfaces gráficas.

UDP	User Datagram Protocol. Protocolo de comunicación no orientado a la conexión basado en el intercambio de datagramas.
Unicast	Envío de información desde un único emisor hacia un único receptor.
WSN	Wireless Sensor Network.

Referencias.

[1] Nitaigour P. Mahalik. *Sensor Networks and Configuration*. Springer. Germany 2007.

[2] Feng Zhao, Leonidas Guibas. *Wireless Sensor Networks*. USA 2005.

[3] Luca Mottola, *Proyecto Logical Neighborhoods*. 2008.

[4] Edgar H. Callaway. Wireless Sensor Networks: Architectures and Protocols. CRC Press 2004.

[5] Department of Primary Industries *(DPI)*. Australia. 2006.

[6] Jamal N. Al-Karai, Ahmed E. Kamal. *Routing Techniques in Wireless Sensor Networks: A Survey*. USA 2005.

[7] W. Heinzelman, J. Kulik, H. Balakrishnan. *Adaptive Protocols for Information Dissemination in Wireless Sensor Networks*. Proc. 5th ACM/IEEE Mobicom Conference (MobiCom '99), pp. 174-85. Seattle, WA, August, 1999.

[8] J. Kulik, W. R. Heinzelman, H. Balakrishnan. *Negotiation-based protocols for disseminating information in wireless sensor networks*. Wireless Networks, Volume: 8, pp. 169-185, 2002.

[9] Y. Xu, J. Heidemann, D. Estrin. *Geography-informed Energy Conservation for Ad-hoc Routing*. In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking 2001, pp. 70-84.

[10] C. Rahul, J. Rabaey. *Energy Aware Routing for Low Energy Ad Hoc Sensor Networks*. IEEE Wireless Communications and Networking Conference (WCNC), vol.1, pp. 350-355. March 17-21, 2002, Orlando, FL.

[11] K. Sohrabi, J. Pottie. *Protocols for self-organization of a wireless sensor network.* IEEE Personal Communications, Volume 7, Issue 5, pp 16-27, 2000.

[12] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin. *Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*. USA 2004.

[13] Tim Daniel Hollerung. *The Cluster-Based Routing Protocol*. Denmark 2004.

[14] Andrei Broder, Michael Mitzenmacher. *Network Applications of Bloom Filters*. USA 2004.

[15] Christian Antognini. *Bloom Filters*. Switzerland 2003.

[16] Christian Grothoff. *A Quick Introduction to Bloom Filters*. USA 2002.

[17] A. Whitaker, D. Wetherall. *Forwarding without Loops in Icarus*. In Proceedings of the Fifth IEEE Conference on Open Architectures and Network Programming (OPENARCH), pp. 63-75. Los Alamitos, CA: IEEE Computer Society, 2002.

[18] C. Estan, G. Varghese. *New Directions in Traffic Measurement and Accounting*. ACM SIGCOMM Computer Communication Review (Proceedings of the 2002 SIGCOMM Conference) 32:4 (2002), 323-336.

[19] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, y W. T. Strayer. *Hash-Based IP Traceback*. ACM SIGCOMM Computer Communication Review (Proceedings of the 2001 SIGCOMM Conference) 31:4 (2001), 3-14.

[20] Teerewat Issariyakul, Ekram Hossain. *Introduction to Network Simulator NS2*. USA 2009.

[21] M. Greis. *Tutorial for the Network Simulator ns*. http://www.isi.edu/nsnam/ns/tutorial/index.html.

[22] Kevin Fall, Kannan Varadhan. *The ns manual*. USA 2009.