“Enrutamiento Multicast Utilizando Optimización Multiobjetivo”

por
Jorge Crichigno

Orientador:
D.Sc. Ing. Benjamín Barán

Proyecto Final de Tesis

Ingeniería Electrónica
Facultad de Ciencias y Tecnología
Universidad Católica “Nuestra Señora de la Asunción”

Asunción – Paraguay
Junio de 2004

Este trabajo está dedicado a mis padres y mis hermanos. Espero poder retribuirles en alguna medida su esfuerzo, su apoyo, su comprensión y sobre todo el amor que he recibido de ellos siempre.

Agradecimientos

Al profesor Benjamín Barán, excelente profesional y por sobre todo, excelente persona. Sin su constante apoyo y orientación este proyecto no hubiera llegado a buen puerto. Su excepcional capacidad de superación frente a los problemas inyecta vida y dinamismo al quehacer diario de las personas que tenemos la suerte de formar parte de su grupo de investigación.

A la Lic. Blanca Troche de Trevisán, Directora del Centro Nacional de Computación, por ser la persona que posibilita que en dicha institución puedan ser llevados a cabo trabajos de investigación de esta naturaleza.

A los compañeros del Centro Nacional de Computación, Augusto Hermosilla, M.Sc. Ing. Jorge Lévera, Pedro Gardel, Lic. Osvaldo Gómez, M.Sc. Ing. Christian Von Lucken, M.Sc. Ing. Diana Benítez, Lic. Carlos Enciso, Lic. Francisco Talavera y Lic. Joel Prieto, por el constante apoyo que me han brindado.

Finalmente, una mención especial a todos mis profesores, compañeros y amigos de la Facultad de Ingeniería, con quienes he compartido estos años de formación.

Índice General

Índice de Figuras

Índice de Tablas

Lista de Acrónimos

AS	Autonomous System
EA	Evolutionary Algorithm
FEC	Forwarding Equivalent Class
GA	Genetic Algorithm
ICT ’04	International Conference on Telecommunications 2004
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISP	Internet Service Provider
KMB	Algoritmo propuesto por Kow, Markowsky y Berman
KPP	Algoritmo propuesto por Kompella, Pasquale y Polyzos
LSP	Label Switch Path
LSR	Label Switch Routers
MMA1	Multiobjective Multicast Algorithm 1
MMA2	Multiobjective Multicast Algorithm 2
MOEA	Multiobjective Evolutionary Algorithm
MOP	Multiobjective Optimization Problem
MPLS	MultiProtocol Label Switching
NSF	National Science Foundation
NTT	Nippon Telephone and Telegraph
OSPF	Open Shortest Path First
P1	Problema de Prueba 1
P2	Problema de Prueba 2
QoS	Quality of Service
RSVP-TE	Resource Reservation Protocol – Traffic Engineering
SK	Algoritmo propuesto por Seok, Lee, Choi y Kim
SOP	Single Objective Problem
SPEA	Strength Pareto Evolutionary Algorithm
SPT	Shortest Path Tree
TCP	Transmission Control Protocol
TE	Traffic Engineering
VoD	Video on-demand

1 Multicast y Redes de Computadoras

1.1 Introducción

Internet tiene sus raíces en la red ARPANET, una red experimental de datos financiada por los Estados Unidos en los comienzos de la década de los 60. El principal objetivo del gobierno de los Estados unidos era la creación de una red robusta con capacidad de mantenerse funcionando aún cuando parte de la misma esté sujeta a fallas. Con esta idea en mente, ARPANET fue construida basándose en un modelo de datagramas, en el cual cada paquete de datos es enviado en forma independiente a su destino. La red de datagramas, además de estar basada en una idea simple, tiene la habilidad de adaptarse automáticamente a cambios en la topología [TAN03].

Por muchos años, Internet ha sido usada por investigadores para el intercambio de información. El acceso remoto, la transferencia de archivos y el correo electrónico (e-mail) eran las aplicaciones más populares, y para estas aplicaciones el modelo de datagramas funciona bien. Sin embargo, el suceso de Internet ha creado nuevas aplicaciones con diferentes requerimientos de calidad de servicio (QoS) distintos a aquellos para los cuales Internet fue creada, como ser un requerimiento de ancho de banda mínimo garantizado para que ciertas aplicaciones funcionen bien. El modelo de datagramas –en el cual está basado Internet– tiene muy poca capacidad de manejo de tráfico de datos dentro de la red y por lo tanto no puede garantizar una capacidad –ancho de banda– mínima en bps a los usuarios. Cuando una aplicación intenta alcanzar un sitio Web o hacer una llamada a través de Internet, algunas partes de la red pueden estar tan congestionadas que los paquetes no pueden llegar a destino y simplemente son desechados. Por lo tanto, las aplicaciones no pueden funcionar en forma adecuada [WAN01].

Otro problema similar en las redes de computadoras es aquel que se da con el retardo que sufren los paquetes hasta alcanzar al destino. Por ejemplo, cuando aquellas aplicaciones como transmisiones de radio o TV tienen un retardo que sobrepasa ciertos niveles, estas carecen de utilidad. Entonces, para que las redes de computadoras actuales sean capaces de manejar tráfico de datos considerando parámetros de QoS, no solo deben ser capaces de garantizar una cantidad de ancho de banda mínimo a las aplicaciones, sino que deben proveer el servicio teniendo en cuenta el retardo a los nodos destinos [WAN01].

Históricamente, las redes basadas en el modelo de datagramas han ofrecido un servicio conocido como servicio del mejor esfuerzo (*best-effort*). *Best-effort* simplemente representa el servicio más simple que una red puede proveer; no hay ninguna forma de asignación de recursos en la red. Todos los paquetes son tratados de igual forma, sin niveles de servicio, requerimientos, reservas o garantías. Cuando un enlace está congestionado, los paquetes simplemente son descartados. Los objetivos principales de este modelo fueron la escalabilidad y el mantenimiento de la conexión bajo fallas. Los algoritmos encargados de seleccionar un camino –enrutar– entre dos nodos de la red simplemente seleccionan el camino más corto considerando una métrica simple como número de saltos o retardo. Claramente este enfoque no es adecuado para soportar asignación de recursos como ser una capacidad mínima garantizada. Por ejemplo, para llevar a cabo una reserva de recursos, es necesario encontrar un camino desde el origen al destino con cierta cantidad de ancho de banda a lo largo del camino. Pero el protocolo utilizado en las redes de datagramas –*Internet Protocol* (IP)– no cuenta con dicha información. Por lo tanto, el uso de dichos algoritmos puede conllevar a una alta tasa de rechazos de solicitudes de conexión y a una mala distribución del tráfico sobre la red. Esto es, mucho tráfico fluyendo por aquellos enlaces que conforman los caminos más cortos, creando cuellos de botella, mientras otros enlaces son apenas utilizados [WAN01].

Resulta claro que la optimización de recursos requiere capacidades adicionales a aquellas proveídas por el modelo de datagramas. Para obtener una mejor utilización de la red, es necesario tener un control explícito sobre los caminos que los datos deben atravesar. Entonces, el flujo total de datos sobre la red puede ser acomodado de forma a minimizar los recursos utilizados, maximizar la probabilidad de enrutamiento de toda la demanda de tráfico y minimizar el costo que ello implica, considerando los parámetros de QoS que cada demanda de tráfico solicita [STA01].

Una manera muy utilizada a la fecha de seleccionar explícitamente los caminos es a través de MPLS (*MultiProtocol Label Switching*). Una red MPLS consiste en un conjunto de nodos llamados *Label Switch Routers* (LSR), que tienen la capacidad de enrutar paquetes utilizando etiquetas, las cuales son adheridas a cada paquete. Las etiquetas definen un flujo de paquetes entre dos nodos de la red, o, en el caso de una transmisión punto a multipunto, entre el nodo origen y el conjunto de nodos destinos. Por cada flujo, denominado *Forwarding Equivalent Class* (FEC), un camino específico (*Label Switch Path* - LSP) es definido a través de la red. La Figura 1 muestra un dominio MPLS, el cual puede representar un sistema autónomo (*Autonomous System* - AS) administrado por alguna entidad privada, como ser un proveedor de servicios de Internet (*Internet Service Provider* - ISP) [STA01].

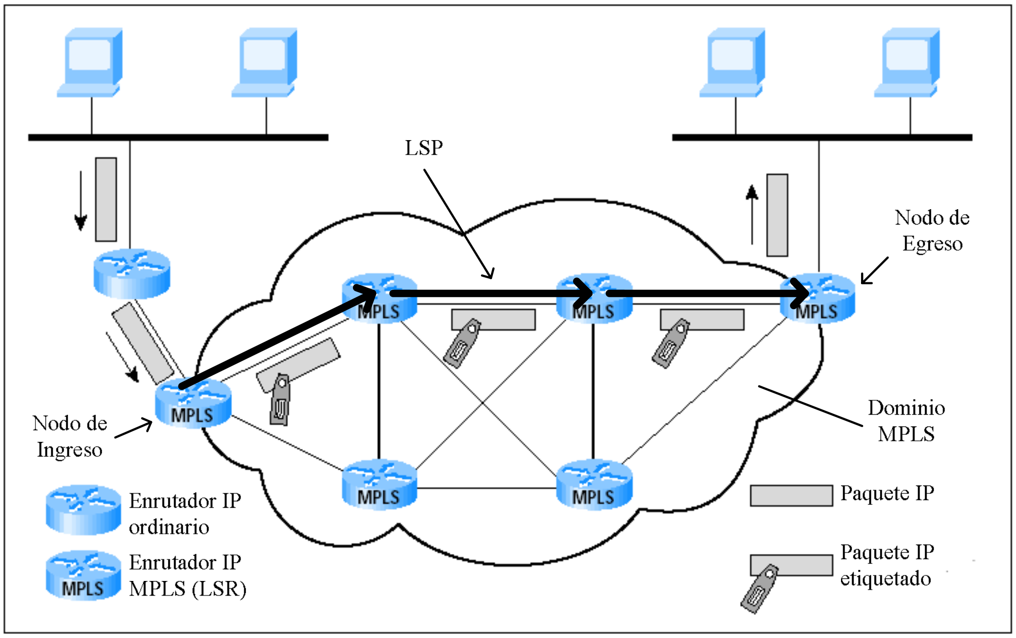


Figura 1.1. Dominio MPLS.

1.2 Ingeniería de Tráfico

La selección dinámica de los caminos para las demandas de tráfico en una red de computadoras, considerando el balanceo de carga de forma a evitar el congestionamiento de los enlaces, atendiendo los parámetros de QoS y optimizando los recursos de la red, es conocida como ingeniería de tráfico [STA01].

Debido al creciente número de aplicaciones punto a multipunto surgidas en Internet, en las cuales un origen debe transmitir a varios destinos –multicast–, como video bajo demanda (*Video on-demand* - VoD), tele conferencias, transmisiones de radio y TV, educación a distancia, se ha incrementado el interés en algoritmos de enrutamiento explícito punto a multipunto para ingeniería de tráfico. Esto se debe principalmente a que la pila de protocolos tradicionalmente usada en las redes de datagramas –TCP/IP– no puede hacer ingeniería de tráfico. Estos inconvenientes pueden ser superados utilizando MPLS y RSVP-TE (*Resource Reservation Protocol – Traffic Engineering*) [OOM02]. RSVP-TE es un protocolo de señalización utilizado para la reserva de recursos –ancho de banda– a lo largo de un camino que conecta un nodo ingreso y otro egreso, lo cual permite garantizar servicios con QoS. De esta forma, la ingeniería de tráfico multicast en una red MPLS puede llevarse a cabo en dos pasos: 1- hallando los caminos desde el nodo origen a los nodos destinos utilizando un algoritmo apropiado para tal efecto; 2- reservando los recursos a lo largo de los caminos hallados en el paso 1, utilizando RSVP-TE [DON02]. Claramente, la tarea más difícil se desarrolla en el paso 1. Los caminos deben ser elegidos de forma a acomodar todo el tráfico a un costo mínimo en términos de recursos, considerando parámetros de QoS como retardo máximo de extremo a extremo y retardo medio, y balanceando la carga de forma a evitar congestionamientos en la red. De este hecho se deriva que la ingeniería de tráfico multicast debe considerar más de una métrica, y por lo tanto, el problema de enrutamiento multicast en redes de computadora puede ser formulado como un Problema de Optimización Multiobjetivo (*Multiobjective Optimization Problem* – MOP) [COE96].

Motivado por el escaso trabajo publicado mundialmente en ingeniería de tráfico multicast y la necesidad creciente por parte de los administradores de AS de contar con herramientas informáticas eficaces para dicha tarea, esta tesis propone dos nuevos Algoritmos de Enrutamiento Multicast Multiobjetivo, denominados *Multiobjective Multicast Algorithm* 1 y 2 (MMA1 y MMA2) basados en un Algoritmo Evolutivo de Optimización Multiobjetivo (*Multiobjective Evolutionary Algorithm* - MOEA) llamado *Strength Pareto Evolutionary Algorithm* (SPEA) [ZIT99]. Dado un nodo origen que desea transmitir una cierta cantidad de flujo de información a un conjunto de nodos destinos, los algoritmos propuestos en este trabajo hallan el camino a cada uno de los receptores optimizando varias métricas. De esta forma, el algoritmo puede ser utilizado en redes MPLS en forma conjunta con RSVP-TE.

Habiendo introducido conceptos esenciales utilizados a lo largo del presente trabajo, el Capítulo 1 continúa de la siguiente manera: en la Sección 1.3 se da la definición formal de multicast, unicast y broadcast, y se ejemplifica la diferencia entre cada una de estas formas de comunicación. En la Sección 1.4 se definen las métricas de optimización consideradas en este trabajo. Posteriormente, en la Sección 1.5 se presentan los trabajos relacionados a problemas de enrutamiento multicast en redes de computadoras. Por último, en la Sección 1.6 se da una breve introducción al enfoque multiobjetivo considerado en este trabajo para la resolución del problema de enrutamiento multicast en redes de computadoras.

1.3 Multicast

La tarea de seleccionar una ruta entre dos nodos de una red es uno de los aspectos más importantes en las redes de computadoras. Esta tarea es llevada a cabo por los algoritmos de enrutamientos, los cuales se encargan de seleccionar la ruta más adecuada.

Los dos modos tradicionales de comunicación, *unicast* y *broadcast*, realmente son formas degeneradas de una tercera, llamada *multicast*. A continuación se define cada una de ellas [KOM93a].

- Unicast: Transmisión de datos desde una fuente a un destino.
- Broadcast: Transmisión de datos desde una fuente a todos los posibles destinos.
- Multicast: Transmisión de datos desde una fuente a un conjunto de destinos.

Una transmisión multicast puede ser implementada utilizando unicast o broadcast. Sin embargo, estas soluciones no son eficientes. Enviar datos a todos los nodos de la red para alcanzar a un subconjunto puede producir un desperdicio grande de recursos. Esta situación es mostrada en la Figura 1.2, donde el nodo 0 desea transmitir a los nodos 2, 3 y 6, pero también transmite a otros nodos que no desean la información, como el nodo 4.

Realizar una secuencia de unicast también puede ser indeseable. El principal motivo nuevamente es el desperdicio de recursos utilizados. Por ejemplo, los caminos 0 a 2 y 0 a 3 comparten el enlace que conecta los nodos 0 y 1. En este caso, el mismo dato atravesaría el enlace dos veces. La Figura 1.3 muestra dicha situación.

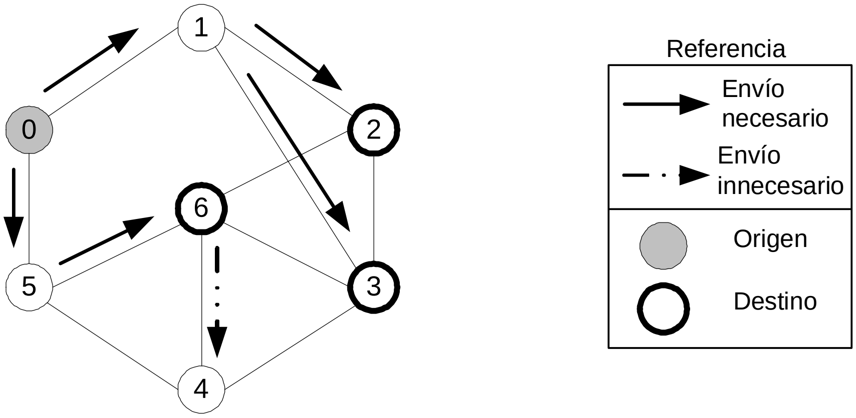


Figura 1.2. Ejemplo de utilización de broadcast para realizar multicast.

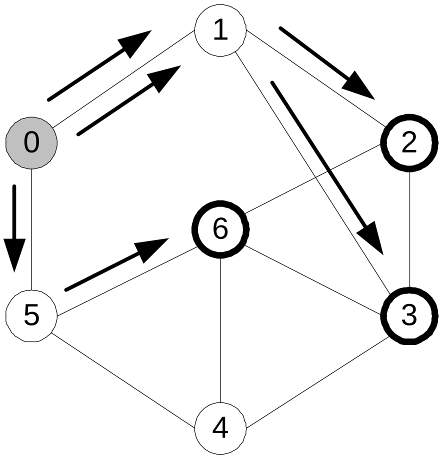


Figura 1.3. Una secuencia de conexiones unicast para realizar multicast.

La estructura apropiada para realizar multicast es el árbol. Dicha estructura consta de un nodo fuente desde el cual cada uno de los nodos destinos puede ser alcanzado, a través de un solo camino. Es decir, consiste en un grafo sin ciclos [KOM93a]. De esta forma, el árbol multicast evita el desperdicio de recursos similares a los citados anteriormente. La Figura 1.4 muestra un árbol que puede ser utilizado para la distribución de datos desde el nodo 0 a los nodos 2, 3 y 6. Note que los datos atraviesan los enlaces que pertenecen al árbol una sola vez. Además, no se producen envíos innecesarios de datos.

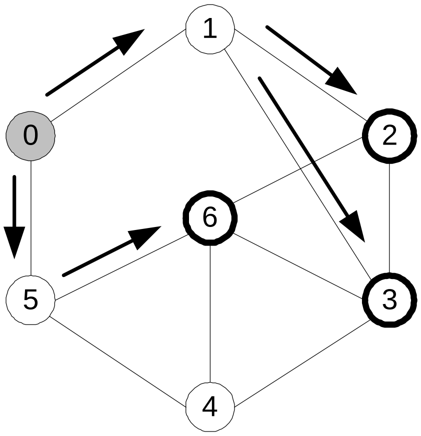


Figura 1.4. Un árbol multicast con origen en 0 y destinos en 2, 3 y 6

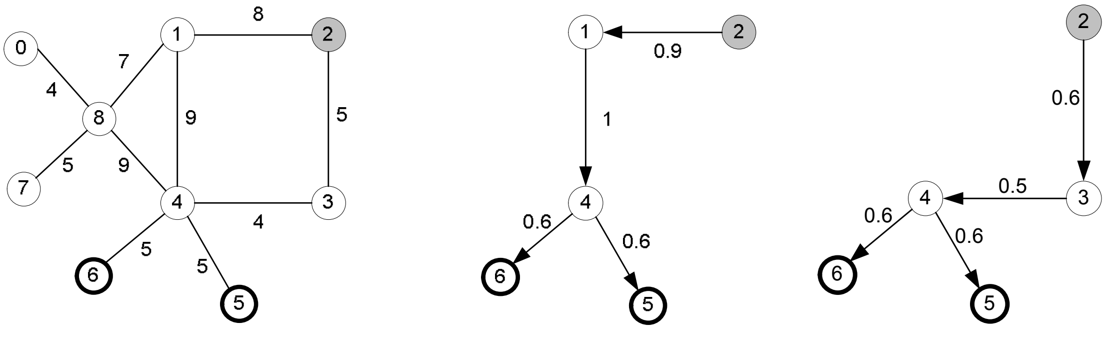
1.4 Métricas de Optimización Multicast

La construcción del árbol multicast más adecuado, con raíz en el nodo origen y alcanzando cada uno de los nodos destinos a través de un camino es llevada a cabo por el algoritmo de enrutamiento multicast. Este puede optimizar distintas métricas. Además, la aplicación para la cual se construye el árbol puede tener diversos requerimientos de QoS como retardo máximo de extremo a extremo limitado, retardo medio acotado o ancho de banda mínimo requerido para llevar a cabo la transmisión. En general, los algoritmos de enrutamiento multicast actuales deben considerar más de una métrica al construir un árbol. A continuación se presentan las distintas métricas que deben ser consideradas en la ingeniería de tráfico multicast.

1.4.1 Utilización Máxima de los Enlaces

Uno de los objetivos principales de la ingeniería de tráfico multicast es encontrar los caminos con la capacidad adecuada entre el nodo origen y el conjunto de nodos destinos, evitando el congestionamiento en la red y balanceando la carga en la misma. En la práctica, el enfoque tradicional utilizado para lograr dichos objetivos es el enrutamiento a través de los caminos con menor utilización de los enlaces, donde la utilización de un enlace se define como el tráfico que fluye a través de él dividido su capacidad, y la utilización máxima de un camino (α_p) como la utilización de su enlace más utilizado [SEO02]. En estados en los que la red está sobrecargada, el enrutamiento a través de dichos enlaces evita la congestión de los mismos, reduce la pérdida de paquetes y el retardo total [DON04]. Además, disminuye la probabilidad de rechazos de posteriores solicitudes de tráfico [SEO02]. Dado que los algoritmos de enrutamiento multicast deben conectar el nodo origen con cada uno de los destinos, es deseado que el árbol multicast esté compuesto por aquellos enlaces de menor utilización. Esto es, si la utilización máxima de los enlaces de un árbol es definido como la utilización de su enlace más utilizado o utilización máxima de sus enlaces (α_T), se desea hallar el árbol que minimice α_T [SEO02]. A continuación se presenta un ejemplo en el cual se demuestra la utilidad del enrutamiento considerando la métrica α_T.

Ejemplo 1.1. Considere la Figura 1.5 (a), donde cada enlace tiene una capacidad de 10 Mbps y un tráfico actual asignado, en Mbps, dado en la Figura. Suponga que el nodo 2 desea hacer una transmisión de 1 Mbps a los nodos 5 y 6, y que el árbol hallado para dicha transmisión es el mostrado en la Figura 1.5 (b). En dicha Figura también se muestra la utilización de cada enlace luego de haber sido enrutada la citada demanda multicast. Si bajo estas circunstancias es generada una nueva solicitud multicast de 2 Mbps con origen en el nodo 2 y destinos en los nodos 0 y 7, dicha solicitud sería bloqueada por falta de recursos. Esta situación podría haber sido evitada si la transmisión de datos del primer grupo multicast se hiciera a través del árbol con menor α_T. Dicho árbol es mostrado en la Figura 1.5 (c).



(a) **(b)** **(c)**

Figura 1.5. (a) Red de 9 nodos. Cada enlace tiene una capacidad de 10 Mbps y un tráfico actual fluyendo a través de él, en Mbps, dado en la misma Figura. (b) Árbol con origen en 2, conjunto destino {5, 6} y demanda de tráfico de 1 Mbps. La utilización de cada enlace es mostrada en la misma Figura. Para este árbol, $\alpha_T = 0.9$. (c) Árbol alternativo. Para esta opción, $\alpha_T = 0.6$.

En esencia, el Ejemplo 1 es un problema de asignación de flujo que puede ser mejor optimizado si todas las demandas multicast son conocidas a priori. Debido al dinamismo con que son generadas las demandas de tráfico en las redes de computadoras reales, es imposible conocer todas ellas de antemano. Entonces, es razonable hacer optimizaciones locales teniendo en cuenta la utilización máxima de los enlaces del árbol en el momento en que cada grupo multicast solicita su demanda de tráfico, de forma a balancear el tráfico total sobre la red de la mejor manera posible.

Si bien el enrutamiento a través de los enlaces de menor utilización es útil para el balanceo de carga y la reducción de la congestión, puede desperdiciar recursos debido a la longitud de las rutas en término de saltos (i.e. suma de ancho de banda asignado en cada enlace a lo largo del camino) [SEO02]. Esta situación es mostrada en el Ejemplo 1.2.

Ejemplo 1.2. La Figura 1.6 muestra la red de la *National Science Foundation* (NSF) [BAR00]. Los números corresponden a la utilización de los enlaces (i.e. el enlace que conecta los nodos 0 y 2 tiene una utilización de 0.25). Suponga que el nodo 0 desea transmitir un flujo de datos al nodo 2. El camino de menor utilización de los enlaces es mostrado en negrita. Note que $\alpha_p = 0.2$. Por otra parte, el camino directo desde 0 a 2 tiene una utilización máxima $\alpha_p = 0.25$, apenas superior a la alternativa mostrada en la Figura, pero un consumo de ancho de banda mucho menor.

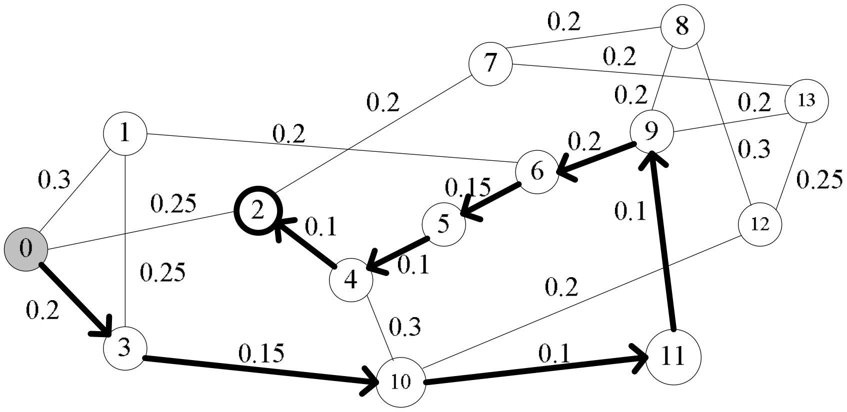


Figura 1.6. Red de la NSF, con la utilización de los enlaces como métrica. El camino de menor utilización de los enlaces, con $\alpha_p = 0.2$, es mostrado en la Figura. Note que el camino directo 0-2 tiene $\alpha_p = 0.25$, pero un costo en términos de recursos utilizados mucho menor.

1.4.2 Costo del Árbol Multicast

Del Ejemplo 1.2 se deriva que la utilización de los recursos también debe ser considerada como una métrica de optimización al construir un árbol multicast. La forma general de optimizar dicha métrica es minimizando el costo del árbol. Dado un costo positivo a cada enlace de una red, el costo del árbol C_T está dado por la suma de los costos de los enlaces que lo componen. El costo puede tener diferentes significados: puede representar alguna cantidad monetaria, como también ser proporcional al consumo total de ancho de banda del grupo multicast. Cualquiera sea el significado, aquellas soluciones de menor costo siempre son deseables. Por ejemplo, en una red cuyos enlaces tienen costo unitario, la minimización del costo del árbol multicast conduce al árbol con menor número de enlaces, y por lo tanto, al de menor consumo de ancho de banda.

Si bien la minimización del costo del árbol optimiza una métrica muy importante asociada a los recursos de la red (i.e. ancho de banda consumido), aplicaciones como video bajo demanda, transmisiones de audio y/o video, tele conferencias y aprendizaje a distancia pueden necesitar árboles de bajo retardo medio y/o bajo retardo máximo de extremo a extremo. La Figura 1.7 muestra el árbol de mínimo costo para el grupo multicast conformado por el nodo 5 como origen y el conjunto destino {0, 4, 9, 10, 13}. El costo de cada enlace está dado en la misma Figura.

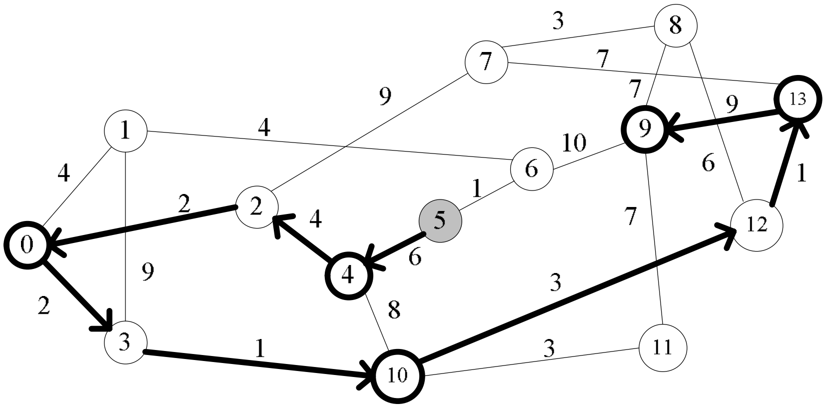


Figura 1.7. Árbol multicast de costo mínimo. C_T = 28.

Aunque la solución mostrada es óptima cuando se considera la métrica de costo, puede no ser de utilidad para aquellas aplicaciones sensibles al retardo, como las citadas arriba. Por ejemplo, para alcanzar al nodo 9, los datos deben pasar por 8 enlaces (7 nodos intermedios). De esta forma, los datos podrían llegar con valores de retardo más elevados a los que la aplicación puede soportar. Por lo tanto, las métricas asociadas al retardo también deben ser consideradas.

1.4.3 Retardo Medio y Retardo Máximo de Extremo a Extremo

La forma tradicional a optimizar las métricas de retardo (*delay*) medio (D_A) y retardo máximo de extremo a extremo (D_M) es enrutando a través del camino de menor retardo. La extensión natural al caso multicast es la transmisión a través del árbol compuesto por los caminos de menor delay desde el nodo fuente a los nodos destinos. Dicho árbol es conocido como el árbol del camino más corto (*Shortest Path Tree* - SPT). La Figura 1.8 ilustra dicha solución para el mismo grupo multicast mostrado en la Figura 1.7. El par de números sobre cada enlace corresponde al retardo y al costo del mismo. D_A está dado por la suma de los retardos de los caminos a cada uno de los nodos destinos dividido el número de nodos destinos. Note que, con estos valores de retardo, el árbol de costo mínimo de la Figura 1.7 tiene D_M=71 y D_A=41.8, mientras que el SPT tiene D_M = 23 y D_A=16.8. El retardo máximo y el retardo medio del árbol de costo mínimo son tan elevados que dicha solución podría no ser de utilidad para aquellas aplicaciones sensibles a estas métricas. En contrapartida, el costo del árbol de la Figura 1.7 es de apenas 28, mientras que el costo del SPT es 40. Claramente, existe una relación de compromiso entre estas métricas.

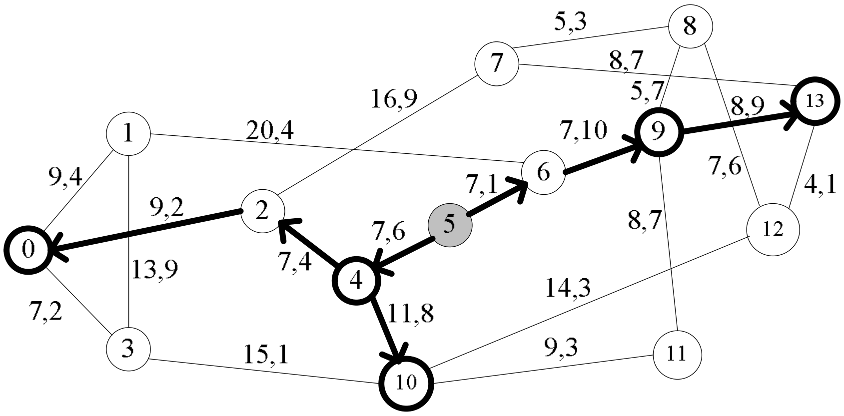


Figura 1.8. Árbol multicast óptimo cuando solo D_A es considerado. C_T = 40, D_A = 16.8 y D_M = 23. Cada enlace tiene asociado un retardo y un costo, dado en ese orden.

1.5 Trabajos Relacionados

Como se ha mencionado en la Introducción, el enfoque típico utilizado en Internet para enrutar paquetes es el enrutamiento a través del camino más corto, considerando la métrica de retardo o número de saltos. La mayor parte de los algoritmos utilizados en las redes de computadoras son variantes del algoritmo de Dijkstra [DIJ59]. Este algoritmo encuentra el camino de menor costo, en términos de retardo o saltos, desde un nodo fuente a un nodo destino. Al aplicarlo a multicast, el algoritmo produce un árbol cuyos caminos desde el nodo origen a los nodos destinos son óptimos en términos de la métrica utilizada. El estándar más utilizado en Internet es el *Open Shortest Path First* (OSPF) [STA00].

El problema de hallar el árbol de menor costo es conocido como Problema del árbol de Steiner, y es sabido que la complejidad de dicho problema es NP-completa [KOM93a]. Esencialmente, la dificultad de hallar el árbol de costo mínimo se debe a la existencia de un número de árboles que crece exponencialmente con el número de nodos de la red, y la solución óptima solo puede ser obtenida por búsqueda exhaustiva [KOM93a]. Debido al elevado tiempo de cómputo que consume hallar la solución óptima, varios algoritmos heurísticos han sido propuestos para la resolución de dicho problema. Un estudio exhaustivo de varias propuestas publicadas hasta la fecha puede ser encontrado en [HWA92]. Uno de los más famosos algoritmos fue propuesto por Kou, Markowsky y Berman (algoritmo KMB) [KOU81]. Dada una red de computadoras modelada como un grafo G y un grupo multicast, KMB transforma el grafo G a otro G’. G’ consiste en un grafo completamente conexo, en el que cada enlace representa el camino de menor costo en G entre los dos nodos involucrados. El número de nodos de G’ es igual al tamaño del grupo multicast. Utilizando G’, el algoritmo construye el árbol de expansión de menor costo. Luego, el árbol en la red original es obtenido transformando el árbol de expansión en G’ a los caminos en G que ellos representan.

Takahashi et al. [TAK80] propusieron un algoritmo heurístico para hallar el árbol de costo mínimo que iterativamente va añadiendo un destino multicast al árbol, el cual al inicio consta únicamente del nodo fuente. En cada iteración, el algoritmo encuentra el nodo destino más cercano, en términos de costo, que aún no es parte del árbol, y añade a este el camino a dicho nodo. Esto se repite hasta que todos los nodos destinos del grupo multicast son parte del árbol. La gran mayoría de los algoritmos determinísticos de árboles de costo mínimo publicados hasta la fecha se basan en esta idea [HOU99].

En [HWA00], Hwang et al. presentaron un algoritmo genético simple que minimiza el costo del árbol multicast. Dichos autores propusieron representar los cromosomas como cadenas de enteros, en el cual cada elemento de la cadena es usado para indexar una tabla de rutas de un nodo destino. La longitud de los cromosomas es igual al número de receptores multicast. La tabla de rutas para un nodo destino consiste en un conjunto de R posibles rutas que conectan el nodo origen y el destino, donde R es un parámetro del algoritmo. Cada elemento de la cadena (gen) puede tomar un valor entero entre 1 y R, el cual representa (indexa) una ruta dada de la tabla de rutas al nodo destino correspondiente. El fitness de un individuo es inversamente proporcional al costo del árbol que él representa. Los autores también citan la forma de extender la propuesta a situaciones que requieran restricciones como retardo de extremo a extremo acotado y/o ancho de banda mínimo requerido para la transmisión de datos.

Debido al creciente número de aplicaciones sensibles al retardo, los algoritmos que solo optimizan la función de costo pueden proveer soluciones que violan las restricciones de retardo máximo impuestas por estas aplicaciones. Para considerar la métrica de retardo, algunos algoritmos añaden una restricción de retardo máximo al problema del árbol de costo mínimo. Este problema fue formulado por primera vez por Kompella, Pasquale y Polyzos [KOM93b]. Dichos autores propusieron un algoritmo heurístico denominado KPP basado en el algoritmo KMB. KPP extiende el alcance de KMB a problemas con restricciones de retardo, teniendo en cuenta el retardo a cada nodo destino en el momento de formar la red completamente conexa. El retardo máximo permitido es determinado a priori. La importancia de este trabajo radica en la formulación, por primera vez, de un problema de enrutamiento multicast en redes de computadoras en el cual más de una métrica independiente debe ser considerada.

Para resolver el mismo problema, nuevas propuestas basadas en algoritmos genéticos fueron publicadas en los últimos años [RAV98, XIA99, ZHE01, ARA02]. El primero de ellos fue propuesto por Ravikumar et al. [RAV98], quienes presentaron un algoritmo genético simple en el cual un cromosoma consiste en el conjunto de enlaces que conforman el árbol que él representa. El fitness de cada individuo es directamente proporcional a la calidad del individuo, siendo un buen individuo aquel árbol de bajo costo que cumple con la restricción de retardo máximo impuesto a priori. El operador de cruzamiento consiste en los siguientes pasos: 1- hallar los enlaces comunes de ambos padres, 2- crear sub-árboles a partir de los enlaces comunes y 3- generar un hijo a través de la reconexión de los diferentes sub-árboles. La reconexión se lleva a cabo según el siguiente criterio: en caso que al menos uno de los padres cumpla con la restricción de retardo, la reconexión se hace a través del camino de menor costo entre los sub-árboles. En caso contrario, los sub-árboles se conectan a través del camino de menor retardo. El operador de mutación consiste en eliminar del árbol un enlace elegido al azar, y reconectar los sub-árboles de la misma forma que lo hace el operador de cruzamiento. Este trabajo fue mejorado consecutivamente por Zhengying et al. [ZHE01] y por Araujo et al. [ARA02].

La principal desventaja con los enfoques basados en restricciones de retardo máximo es la imposición de un valor máximo de retardo dado a priori, lo cual puede descartar buenas soluciones de muy bajo costo, con retardo máximo solo apenas superior a la restricción dada a priori.

Con el creciente despliegue de MPLS y los protocolos de señalización como RSVP-TE, los cuales en conjunto permiten el encaminamiento explícito y la reserva de recursos a través de los caminos, se ha venido incrementando el interés en nuevos algoritmos y esquemas alternativos de balanceo de carga y optimización de recursos. Para llevar a cabo dicha tarea, el objetivo adoptado en recientes publicaciones [SEO02, DON04] es la minimización de la utilización máxima de los enlaces de la red (α). Para resolver el problema dinámico de ingeniería de tráfico multicast, en el cual las solicitudes de tráfico llegan una después de otra, Seok et al. [SEO02] propusieron un algoritmo heurístico basado en la extensión del algoritmo de Dijkstra [DIJ59] que minimiza α_T. Aunque esta propuesta es útil para minimizar la utilización máxima de los enlaces de la red y reducir el congestionamiento sobre la misma, el ancho de banda consumido puede ser desperdiciado debido a posibles rutas largas en términos de saltos. Por lo tanto, los autores restringen la longitud de las rutas a un valor máximo dado a priori, de forma a hallar un árbol multicast en el cual el número de saltos a los nodos destinos esté acotado.

De esta manera, los autores intentan optimizar, además de la utilización máxima de los enlaces, el consumo de ancho de banda. El algoritmo propuesto en este trabajo puede ser dividido en dos pasos: 1- modificación del grafo original a uno en el cual el número de saltos desde el nodo origen a cualquier otro nodo de la red esté acotado por un valor dado a priori; 2- encontrar el árbol multicast que minimice la utilización máxima de los enlaces, utilizando para ello el algoritmo del camino de menor utilización máxima de los enlaces. Esta propuesta demostró que el tráfico multicast puede ser mejor balanceado –y por lo tanto la probabilidad de rechazos de solicitudes por falta de recursos puede ser disminuida– cuando la métrica de enrutamiento utilizada es la utilización de los enlaces. La formulación del problema claramente demuestra la característica multiobjetivo de los problemas de enrutamiento multicast, pues existe más de un objetivo a optimizar.

En [DON04], Donoso et al. propusieron un esquema de ingeniería de tráfico multicast multi-árbol. Con este enfoque, el flujo de datos desde el nodo fuente a los nodos destinos es enviado a través de distintos árboles, de forma a balancear la carga sobre la red. El esquema no solamente tiene en cuenta la utilización máxima de los enlaces, sino también el número total de saltos, el consumo total de ancho de banda y el retardo total de extremo a extremo. El método propuesto minimiza una función objetivo compuesta por la suma ponderada de las cuatro métricas citadas. Debido a la naturaleza NP-completa del esquema propuesto, los autores también propusieron un algoritmo heurístico para la resolución del problema. Este consiste de dos etapas: 1- obtener un grafo modificado. En esta etapa, todos los caminos posibles desde el nodo fuente a cada uno de los nodos destinos son hallados. Luego, para cada uno de los nodos destinos, para cada camino al nodo destino, para cada nodo en dicho camino, un valor de distancia basado en el número de saltos, ancho de banda consumido y retardo es calculado; 2- para cada destino multicast, hallar los caminos requeridos para la transmisión de flujo en el grafo modificado. Es importante notar que este trabajo ya considera el estudio de algoritmos evolutivos como trabajo futuro, pues la característica multiobjetivo del problema y las relaciones de compromiso entre los distintos objetivos son resaltados en las pruebas de simulación hechas por los autores.

Dada la naturaleza multiobjetivo del problema de enrutamiento multicast en redes de computadoras, la relación de compromiso entre las distintas métricas a optimizar (presentadas en la Sección 1.4) y la imperiosa necesidad de optimizar cada una de ellas en forma simultánea, este trabajo propone formular el problema como uno puramente multiobjetivo. Por tal motivo, la siguiente Sección presenta una introducción al enfoque multiobjetivo utilizado a lo largo del libro.

1.6 Enfoque Multiobjetivo

La optimización simultánea de varios objetivos en los problemas de ingeniería de tráfico multicast, como la utilización máxima de los enlaces, el costo del árbol multicast, el retardo de máximo y el retardo medio conlleva a soluciones en las que los objetivos presentan conflictos entre si; es decir, la mejora en uno conduce a un deterioro en el otro. Aunque la mayoría de los problemas de enrutamiento en redes de computadoras involucran este tipo de situaciones, las propuestas computacionales presentadas hasta la fecha se limitan a convertir el problema de objetivos múltiples en uno en que existe un solo objetivo. Esta reducción es debida a los modelos matemáticos empleados y puede realizarse de varias maneras. Por ejemplo, priorizando uno de los objetivos y utilizando los restantes como restricciones, o generando un objetivo compuesto, otorgando pesos a cada uno de ellos, de forma a optimizar la suma ponderada de los mismos. De todos modos, ninguna de estas reducciones refleja fielmente al problema y, por tanto, tampoco otorga soluciones completamente satisfactorias.

En problemas de optimización multiobjetivo en redes de computadoras, resulta evidente la existencia de múltiples soluciones y la imposibilidad de decidir cuál de ellas es mejor si se consideran todos los objetivos al mismo tiempo. Se dice que las soluciones de un problema con objetivos múltiples son óptimas cuando ninguna otra solución, en todo el espacio de búsqueda, es superior a ellas cuando se tienen en cuenta *todos* los objetivos al *mismo* tiempo, i.e. ningún objetivo puede mejorarse sin degradar algún otro objetivo. Esto ha sido mostrado en los ejemplos de la Sección 1.4, donde se ha mostrado que α_T, C_T, D_A y D_M pueden estar en conflicto entre si.

Al conjunto de estas soluciones óptimas se conoce como soluciones Pareto óptimas. Su nombre les fue dado en honor al ingeniero y economista Wilfredo Pareto, quien fue el primero en definir un nuevo criterio de optimalidad para los problemas en los que existen múltiples objetivos a optimizar, y existen conflictos al realizar la optimización simultánea de los mismos [PAR96]. A partir de este concepto se establece, como requisito para afirmar que una situación es mejor que otra, el que en ella no se empeore nada, pero se mejore a alguno. En caso contrario, según Pareto [PAR96], para decidir se requiere un juicio no objetivo de valor y la ciencia no puede guiarnos.

1.7 Resumen del Capítulo

Habiendo presentado una introducción al problema de enrutamiento multicast en redes de computadoras, el presente trabajo continúa de la siguiente manera. En el Capítulo 2 se da la definición formal de un Problema de Optimización Multiobjetivo y se definen términos relevantes referentes a dichos problemas. Luego, en el Capítulo 3 se presenta la formulación matemática del problema de enrutamiento multicast en redes de computadoras como un MOP, y se definen las funciones objetivos. Posteriormente, en el Capítulo 4 se presentan los dos algoritmos propuestos para la resolución del problema, MMA 1 y MMA 2. Los resultados experimentales son mostrados en el Capítulo 5. Por último, en el Capítulo 6 se presentan las conclusiones y los trabajos futuros. El Apéndice A presenta los trabajos publicados en la *IEEE 11th International Conference on Telecommunications* (*ICT '2004*), agosto – 2004, Brasil, mientras que el Apéndice B muestra el trabajo publicado en la *IEEE 13th International Conference on Computer Communications and Networks* (*ICCCN '2004*), octubre - 2004, USA.

2 Optimización con Objetivos Múltiples

2.1 Introducción

Este trabajo considera el problema de enrutamiento multicast en redes de computadoras como un Problema de Optimización Multiobjetivo (MOP), en el cual las siguientes funciones objetivos son optimizadas: 1- utilización máxima de los enlaces del árbol (α_T), 2- costo del árbol (C_T), 3- retardo máximo de extremo a extremo (D_M) y 4- retardo medio a los nodos destinos (D_A). Por esta razón, este Capítulo trata temas referentes a MOPs y define términos relacionados a este tipo de problemas que serán utilizados a lo largo del libro.

El Capítulo está organizado de la siguiente manera: en la Sección 2.2 se da la definición general de un problema multiobjetivo. La Sección 2.3 está dedicada a la discusión de las fases de resolución de un MOP y la forma en que ellas se entrelazan. En la Sección 2.4 se da la técnica tradicional utilizada en la actualidad para la resolución de problemas de enrutamiento multicast con más de un objetivo a optimizar, indicando brevemente sus desventajas potenciales. Posteriormente, en la Sección 2.5 se propone a los algoritmos evolutivos como herramientas que poseen cualidades deseables para la resolución de MOPs. Por último, se agrega una breve cronología de los acontecimientos más importantes en el desarrollo de la optimización multiobjetivo utilizando algoritmos evolutivos, enfocando además las áreas donde todavía existen cuestiones abiertas.

2.2 Problemas de Optimización Multiobjetivo

En el área de optimización multiobjetivo, debido a la naturaleza aún incipiente del ámbito de investigación, no existe una notación estándar, y no ha sido sino hasta hace muy poco tiempo atrás que los investigadores han empezado a preocuparse de definir con claridad estos aspectos. Sin embargo, en gran parte de los trabajos consultados se percibe aún bastante confusión al respecto. Por lo tanto, es esencial establecer una notación clara antes de iniciar la discusión. A continuación se da la definición formal de un problema de optimización multiobjetivo.

Definición 1: Problema de Optimización Multiobjetivo (*Multiobjective Optimization Problem:* MOP). Un MOP general incluye un conjunto de h parámetros (variables de decisión), un conjunto de k funciones objetivo, y un conjunto de m restricciones. Las funciones objetivo y las restricciones son funciones de las variables de decisión. Luego, el MOP puede expresarse como:

$$\begin{array}{ll} \textit{Optimizar} & \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots , f_k(\boldsymbol{x})) \\ \textit{sujeto a} & \boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \ldots , e_m(\boldsymbol{x})) \geq \boldsymbol{0} \\ \textit{donde} & \boldsymbol{x} = (x_1, x_2, \ldots , x_h) \in \boldsymbol{X} \\ & \boldsymbol{y} = (y_1, y_2, \ldots , y_k) \in \boldsymbol{Y} \end{array} \qquad \textbf{(2.1)}$$

siendo $\boldsymbol{x}$ el vector de decisión e $\boldsymbol{y}$ el vector objetivo. El espacio de decisión se denota por $\boldsymbol{X}$, y el espacio objetivo por $\boldsymbol{Y}$. Optimizar, dependiendo del problema, puede significar igualmente, minimizar o maximizar. El conjunto de restricciones $\boldsymbol{e(x)} \geq \boldsymbol{0}$ determina el conjunto de soluciones factibles $\boldsymbol{X}_f \subset \boldsymbol{X}$ y su correspondiente conjunto de vectores objetivo factibles $\boldsymbol{Y}_{\boldsymbol{f}} \subset \boldsymbol{Y}$.

Definición 2: Conjunto de soluciones factibles. El conjunto de soluciones factibles $\boldsymbol{X}_f$ se define como el conjunto de vectores de decisión $\boldsymbol{x}$ que satisface $\boldsymbol{e}(\boldsymbol{x})$:

$$\boldsymbol{X}_f = \{\boldsymbol{x} \in \boldsymbol{X} \mid \boldsymbol{e}(\boldsymbol{x}) \geq \boldsymbol{0}\} \qquad \textbf{(2.2)}$$

La imagen de $\boldsymbol{X}_f$, es decir, la región factible del espacio objetivo, se denota por

$$\boldsymbol{Y}_f = \boldsymbol{f}(\boldsymbol{X}_f) = \bigcup_{x \in X_f} \{\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})\} \qquad \textbf{(2.3)}$$

De estas definiciones se tiene que cada solución del MOP en cuestión consiste de una *h*-tupla $\boldsymbol{x}=(x_1, x_2, \ldots , x_h)$, que conduce a un vector objetivo $\boldsymbol{y} = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots , f_k(\boldsymbol{x}))$, donde cada $\boldsymbol{x}$ debe cumplir con el conjunto de restricciones $\boldsymbol{e}(\boldsymbol{x}) \geq \boldsymbol{0}$. El problema de optimización consiste en hallar la $\boldsymbol{x}$ que tenga el “mejor valor” de $\boldsymbol{f}(\boldsymbol{x})$. En general, no existe un único “mejor valor”, sino un conjunto de soluciones. Entre éstas, ninguna se puede considerar mejor que las demás si se tienen en cuenta todos los objetivos simultáneamente. De este hecho se deriva que pueden existir –y generalmente existen– conflictos entre los diferentes objetivos que componen el problema. Por ende, al tratar con MOPs se precisa de un nuevo concepto de “óptimo”.

En la optimización de un solo objetivo el conjunto de variables de decisión factibles está completamente ordenado mediante una función objetivo f. Es decir, dadas dos soluciones $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}_f$, se cumple una sola de las siguientes proposiciones: $f(\boldsymbol{u}) > f(\boldsymbol{v})$, $f(\boldsymbol{u}) = f(\boldsymbol{v})$ o $f(\boldsymbol{v}) > f(\boldsymbol{u})$. El objetivo consiste en hallar la solución (o soluciones) que tengan los valores óptimos (máximos o mínimos) de f. Cuando se trata de varios objetivos, sin embargo, la situación cambia. $\boldsymbol{X}_f$, en general, no está totalmente ordenada por los objetivos. El orden que se da generalmente es parcial (i.e. pueden existir dos vectores de decisión $\boldsymbol{u}$ y $\boldsymbol{v}$ tal que $\boldsymbol{f}(\boldsymbol{u})$ no puede considerarse mejor que $\boldsymbol{f}(\boldsymbol{v})$ ni $\boldsymbol{f}(\boldsymbol{v})$ puede considerarse mejor que $\boldsymbol{f}(\boldsymbol{u})$ ni $\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{v})$). Para expresar esta situación matemáticamente, las relaciones =, ≤ y ≥ se deben extender. Esto se puede realizar de la siguiente manera:

Definición 3: Extensión de las relaciones =, ≤ y ≥ a MOPs. Dados 2 vectores de decisión $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}$,

$$\begin{aligned}
\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{v}) \quad & \text{si y solo si} \quad \forall \boldsymbol{i} \in \{1, 2, \dots, \mathrm{k}\}\colon\ f_i(\boldsymbol{u}) = f_i(\boldsymbol{v}) \\
\boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v}) \quad & \text{si y solo si} \quad \forall \boldsymbol{i} \in \{1, 2, \dots, \mathrm{k}\}\colon\ f_i(\boldsymbol{u}) \leq f_i(\boldsymbol{v}) \\
\boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v}) \quad & \text{si y solo si} \quad \boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{u}) \neq \boldsymbol{f}(\boldsymbol{v})
\end{aligned} \tag{2.4}$$

Las relaciones ≥ y **>** se definen de manera similar. Es claro que dos vectores $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}$ solo pueden cumplir con una de tres condiciones que se expresan con los siguientes símbolos y términos:

Definición 4: Dominancia Pareto en un contexto de minimización. Para dos vectores de decisión $\boldsymbol{u}$ y $\boldsymbol{v}$,

$$\begin{aligned}
\boldsymbol{u} \succ \boldsymbol{v}\ (\boldsymbol{u} \text{ domina a } \boldsymbol{v}) \quad & \text{si y solo si} \quad \boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v}) \\
\boldsymbol{v} \succ \boldsymbol{u}\ (\boldsymbol{v} \text{ domina a } \boldsymbol{u}) \quad & \text{si y solo si} \quad \boldsymbol{f}(\boldsymbol{v}) < \boldsymbol{f}(\boldsymbol{u}) \\
\boldsymbol{u} \sim \boldsymbol{v}\ (\boldsymbol{u} \text{ y } \boldsymbol{v} \text{ no son comparables}) \quad & \text{si y solo si} \quad \boldsymbol{f}(\boldsymbol{u}) \nsucc \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{v}) \nsucc \boldsymbol{f}(\boldsymbol{u})
\end{aligned} \tag{2.5}$$

Definición 5: Dominancia Pareto en un contexto de maximización. Para dos vectores de decisión $\boldsymbol{u}$ y $\boldsymbol{v}$,

$$\begin{array}{lll} \boldsymbol{u} \succ \boldsymbol{v} \text{ (} \boldsymbol{u} \text{ domina a } \boldsymbol{v}\text{)} & \text{si y solo si} & \boldsymbol{f}(\boldsymbol{u}) > \boldsymbol{f}(\boldsymbol{v}) \\ \boldsymbol{v} \succ \boldsymbol{u} \text{ (} \boldsymbol{v} \text{ domina a } \boldsymbol{u}\text{)} & \text{si y solo si} & \boldsymbol{f}(\boldsymbol{v}) > \boldsymbol{f}(\boldsymbol{u}) \\ \boldsymbol{u} \sim \boldsymbol{v} \text{ (} \boldsymbol{u} \text{ y } \boldsymbol{v} \text{ no son comparables)} & \text{si y solo si} & \boldsymbol{f}(\boldsymbol{u}) \nsucc \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{v}) \nsucc \boldsymbol{f}(\boldsymbol{u}) \end{array} \qquad \textbf{(2.6)}$$

Alternativamente, $\boldsymbol{u} \rhd \boldsymbol{v}$ denota que $\boldsymbol{u}$ domina o es igual a $\boldsymbol{v}$.

De aquí en adelante, ya no será necesario diferenciar el tipo de optimización a realizar (minimización o maximización), al punto que un objetivo puede ser maximizado, mientras que otro puede ser minimizado.

Definido el concepto de dominancia Pareto, puede ser introducir el criterio de optimalidad Pareto de la siguiente manera:

Definición 6: Optimalidad Pareto. Dado un vector de decisión $\boldsymbol{x} \in \boldsymbol{X}_f$, se dice que $\boldsymbol{x}$ es no dominado respecto a un conjunto $\boldsymbol{V} \subseteq \boldsymbol{X}_f$ si y solo si

$$\forall\ \boldsymbol{v} \in \boldsymbol{V} : (\boldsymbol{x} \succ \boldsymbol{v} \vee \boldsymbol{x} \sim \boldsymbol{v}) \qquad \textbf{(2.7)}$$

En caso que $\boldsymbol{x}$ sea no dominado respecto a todo el conjunto $\boldsymbol{X}_f$, y solo en ese caso, se dice que $\boldsymbol{x}$ es una **solución Pareto óptima**. Por lo tanto, el conjunto Pareto óptimo $\boldsymbol{X}_{true}$ puede ser definido formalmente a continuación.

Definición 7: Conjunto Pareto óptimo. Dado el conjunto de vectores de decisión factibles $\boldsymbol{X}_f$, se denomina conjunto Pareto óptimo $\boldsymbol{X}_{true}$ al conjunto de vectores de decisión no dominados de $\boldsymbol{X}_f$, es decir:

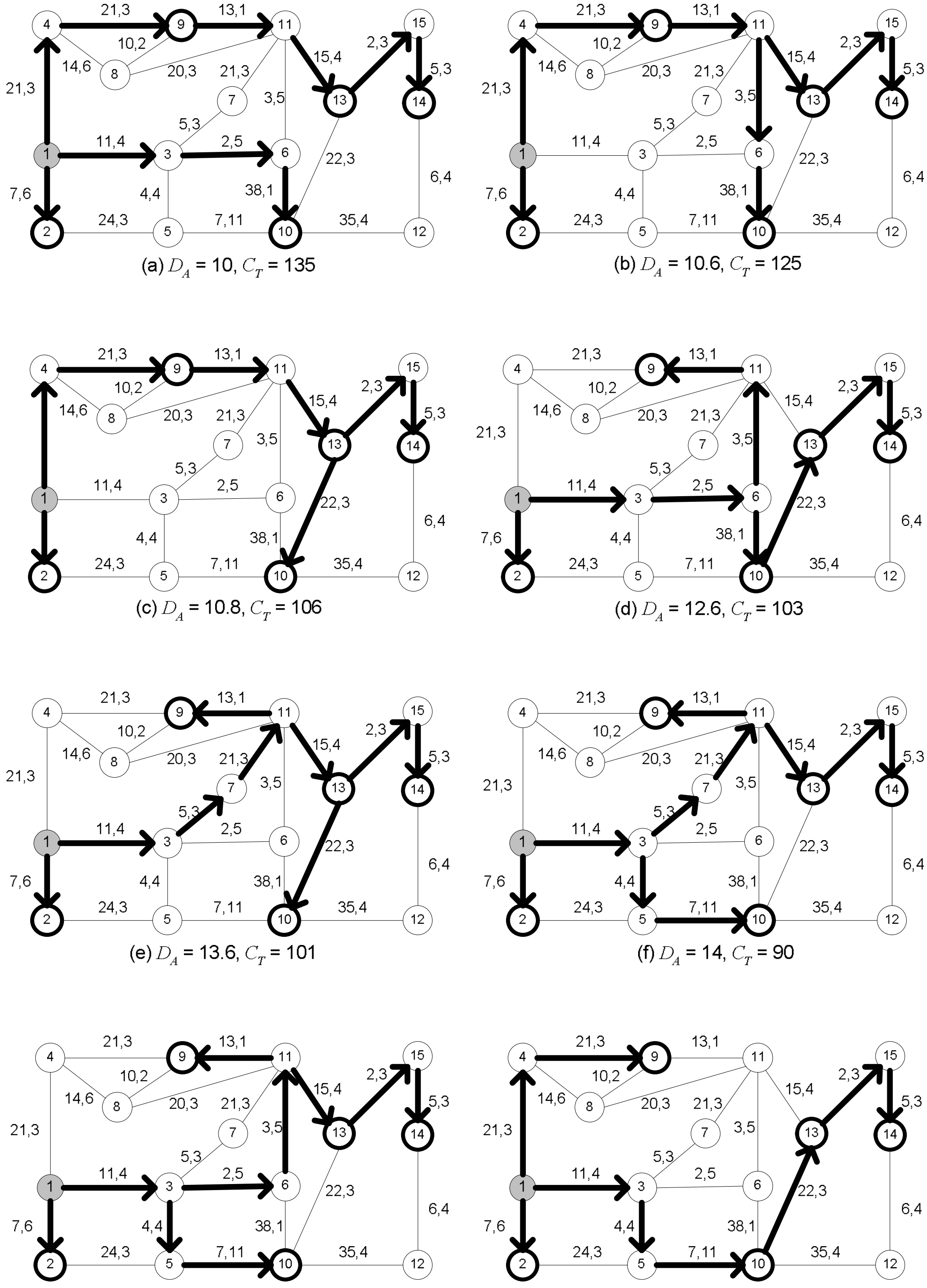
$$\boldsymbol{X}_{true} = \{\, \boldsymbol{x} \in \boldsymbol{X}_f \mid \boldsymbol{x} \text{ es no-dominado con respecto a } \boldsymbol{X}_f \,\} \qquad \textbf{(2.8)}$$

El correspondiente conjunto de vectores objetivos $\boldsymbol{Y}_{true} = \boldsymbol{f}(\boldsymbol{X}_{true})$ constituye el **frente Pareto óptimo.**

De forma a ilustrar los conceptos definidos arriba, considere el siguiente Ejemplo.

Ejemplo 2.1. La Figura 2.1 muestra una red de computadoras, donde los números sobre cada enlace de la red constituyen el costo del enlace y el retardo del mismo [ARA02]. Dado el grupo multicast conformado por el nodo 1 como fuente y el conjunto de nodos destino {2, 9, 10, 13, 14}, se desea construir un árbol minimizando las funciones objetivos de costo C_T y retardo medio D_A, donde la primera viene dada por la suma del costo de los enlaces que componen el árbol y la segunda por la suma de los retardos a los nodos destinos dividido el número de receptores multicast. Las Figuras 2.1 (a) a (h) muestran 8 posibles soluciones para el grupo multicast dado. Note que, descartando la solución (h), ninguna solución puede ser considera mejor que otra cuando ambas métricas, retardo medio y costo del árbol, son consideradas simultáneamente. Es decir, las solucione (a) a (g) son no dominadas. Por el contrario, la solución (h), con D_A = 16.2 y C_T = 100, es dominada por la solución (f), cuyo vector objetivo viene dado por D_A = 14 y C_T = 90. De hecho, por búsqueda exhaustiva, puede comprobarse que las soluciones (a) a (f) constituyen el Frente Pareto óptimo del problema dado [ARA02].

La Figura 2.2 muestra el frente Pareto del mismo problema. Los vectores objetivos de las soluciones Pareto óptimas son mostrados con puntos negros. Por ejemplo, G es el vector objetivo de la solución Pareto óptima mostrada en la Figura 2.1 (g). El vector objetivo H, mostrado en gris, corresponde a la solución dominada de la Figura 2.1 (h). Note que cada vector objetivo del Frente Pareto óptimo define una región del espacio objetivo que se extiende hasta el infinito. Una solución dada es dominada si el vector objetivo correspondiente está dentro de una de estas regiones. Note que el vector objetivo H está ubicado dentro de la región definida por el vector objetivo F, y por lo tanto, la solución correspondiente es dominada por esta.



(a) $D_A = 10$, $C_T = 135$

(b) $D_A = 10.6$, $C_T = 125$

(c) $D_A = 10.8$, $C_T = 106$

(d) $D_A = 12.6$, $C_T = 103$

(e) $D_A = 13.6$, $C_T = 101$

(f) $D_A = 14$, $C_T = 90$

(g) $D_A = 16.4$, $C_T = 69$

(h) Solución dominada por (f). $D_A = 16.2$, $C_T = 100$

Figura 2.1. (a) a (g) muestran las soluciones Pareto óptimas del Ejemplo 2.1. (h) es una solución dominada por (f).

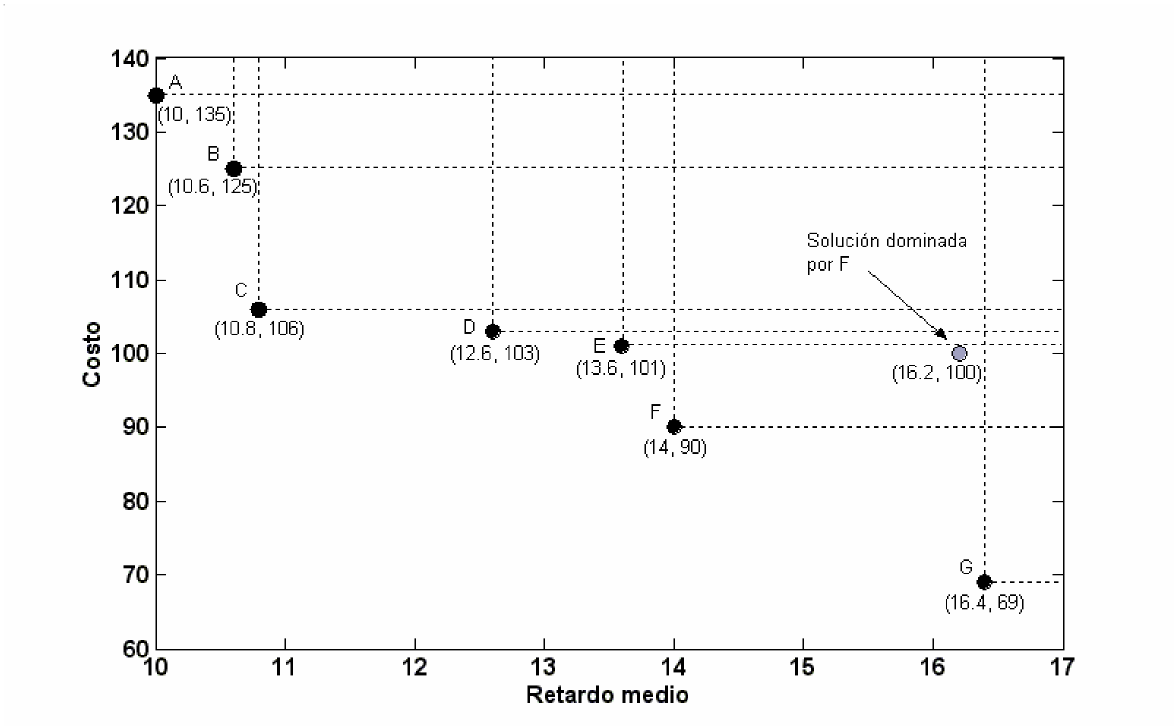


Figura 2.2. Frente Pareto del Ejemplo 2.1. Los vectores objetivos correspondientes a las soluciones Pareto óptimas son mostrados con puntos negros, mientras el vector objetivo de la solución dominada es mostrado en gris.

La Figura 2.2 también remarca las diferencias entre problemas con objetivos múltiples y aquellos mono-objetivo: en el primer caso no hay una única solución sino un conjunto de soluciones de compromiso entre las funciones de costo y retardo medio. Ninguna solución del frente Pareto se puede definir como “mejor” que las demás, a menos que se incluya alguna otra información que determine que un objetivo es más importante que otro, o que se fije un peso relativo entre los objetivos.

2.3 Búsqueda y Toma de Decisiones

Al resolver un MOP, se pueden distinguir 2 niveles de dificultad en el problema: la búsqueda y la toma de decisiones [HOR97]. El primer aspecto se refiere al proceso de optimización, durante el cual se explora el espacio de soluciones factibles buscando las soluciones Pareto óptimas. En esta fase, como ocurre en la optimización de objetivo único, el espacio de búsqueda puede ser lo suficientemente grande y complejo como para impedir el uso de técnicas de optimización que den resultados exactos [STE86]. El segundo aspecto es equivalente a seleccionar un conjunto o una única solución de compromiso del conjunto Pareto óptimo ya definido. Este paso generalmente tiene que ver con cuestiones inherentes a las condiciones del problema y los recursos disponibles.

Dependiendo de la manera en que se combinan ambas fases (búsqueda y toma de decisiones) los métodos de optimización multiobjetivo pueden clasificarse en tres categorías diferentes [COH78, HWA79]:

a) Métodos de toma de decisión previa a la búsqueda (decidir, luego buscar): los objetivos del MOP se combinan en un único objetivo que implícitamente incluye información de preferencia obtenida del responsable de la toma de decisiones. Estos son métodos de toma de decisión *a priori*. Gran parte de las técnicas tradicionales para resolución de MOPs utilizan esta estrategia [COH78, STE86].
b) Métodos de búsqueda previa a la toma de decisión (buscar, luego decidir): se realiza la optimización sin incluir información de preferencia. El responsable de la toma de decisiones escoge del conjunto de soluciones obtenidas (que idealmente son todas del conjunto Pareto óptimo). A éstos se conoce como métodos de toma de decisión *a posteriori*.
c) Métodos de toma de decisión durante la búsqueda (buscar mientras se decide): éstos permiten que quien toma las decisiones pueda establecer preferencias durante un proceso de optimización interactivo. Luego de llevarse a cabo cada paso del proceso de optimización se presenta al responsable de decisiones los conflictos existentes y se le permite especificar sus preferencias o compromisos. De esta manera se desarrolla una búsqueda guiada. Estos métodos reciben el nombre de *progresivos* o *interactivos*.

La combinación de objetivos múltiples en un único objetivo a optimizar posee la ventaja de que a la función resultante se puede aplicar cualquiera de los métodos de optimización de un único objetivo ya disponibles en la literatura [COH78, STE86], sin mayores modificaciones. Sin embargo, tal combinación requiere conocimiento del dominio del problema que no siempre está disponible. Por ejemplo, en los problemas de diseño de diversas áreas de ingeniería, se realiza la búsqueda justamente para tener un mejor conocimiento del espacio de búsqueda del problema y las soluciones alternativas. Al realizar la búsqueda antes de la toma de decisión no se requiere este conocimiento, pero se impide que quien toma la decisión exprese previamente sus preferencias, lo que probablemente conduce a una reducción de la complejidad del espacio de búsqueda. Un problema con las técnicas de toma de decisión interactivas o a posteriori es la dificultad de presentar al responsable de la toma de decisiones las diferencias entre las soluciones de un MOP de muchas dimensiones. Aún así, la integración de la búsqueda y la toma de decisiones que proponen las técnicas interactivas parece una alternativa promisoria para aprovechar las ventajas de los dos primeros tipos de técnicas.

En particular, este trabajo se enfocado en técnicas que han demostrado ser capaces de explorar espacios de búsquedas prácticamente ilimitados y altamente complejos y generar aproximaciones lo más fieles posibles al conjunto Pareto óptimo.

2.4 El Método de Suma con Pesos

Si bien el método tradicional de enrutamiento en redes de computadoras ha sido a través de esquemas que optimizan una sola métrica, recientemente se han propuestos nuevos esquemas y algoritmos que consideran mas de una métrica a optimizar [SEO02, DON04]. El método utilizado es el de la suma ponderada con pesos [COH78]. Este método combina las diferentes funciones objetivo en una sola función *F*, de la siguiente manera:

Optimizar $$F = \sum_{j=1}^{k} w_j f_j(\boldsymbol{x}) \qquad \textbf{(2.9)}$$

donde $\boldsymbol{x} \in \boldsymbol{X_f}$, y w_j es el peso usado para ponderar la j-ésima función objetivo.

Usualmente, y sin pérdida de generalidad, se escogen pesos fraccionales y diferentes de cero, de manera que se cumpla $\sum_{j=1}^{k} w_j = 1,\ w_j > 0$ [DON04].

El procedimiento es sencillo: se escoge una combinación de pesos y se optimiza la función *F* para obtener una solución óptima. Otras soluciones surgen a partir de optimizaciones realizadas sobre una combinación diferente de pesos [DON04].

En el presente contexto, optimizar implica maximizar todas las funciones objetivo, o minimizarlas. Esto es, o se maximiza o se minimiza todas las funciones $f_j(\boldsymbol{x})$; no se admite la maximización de algunos objetivos y la minimización de otros.

En [DEB99] se demuestra que, si se utiliza un algoritmo de optimización que obtiene resultados exactos siempre y los pesos escogidos son siempre positivos, el método genera soluciones que siempre pertenecen al conjunto Pareto óptimo. La interpretación de este método es la siguiente. Alterar el vector de pesos y optimizar la ecuación implica encontrar un hiperplano (una línea para el caso en que se tengan dos objetivos) con una orientación fija en el espacio de la función. La solución óptima es el punto donde un hiperplano con esta orientación tiene una tangente común con el espacio de búsqueda factible. De aquí deducimos que este método no puede usarse para encontrar soluciones Pareto óptimas en problemas de optimización multicriterio que tienen un frente Pareto óptimo cóncavo. Para una discusión más detallada de esta característica se refiere al lector a [DEB99].

Aunque no se adecuan a la naturaleza del problema, la ventaja principal del método de suma con pesos es que permite la utilización de algoritmos desarrollados para la resolución de problemas de optimización mono-objetivos (*Single Objective Problems* - *SOPs*) bien conocidos y de probada eficacia, aún cuando se trate de problemas reales, de gran tamaño. Para problemas de gran escala, muy pocas técnicas de optimización multiobjetivo reales se han presentado [HOR97], a diferencia de técnicas de optimización de un solo objetivo que han existido desde hace bastante tiempo y han sido probadas en diferentes situaciones. Sin embargo, este método cuenta con las siguientes limitaciones [DEB99]:

a) El algoritmo de optimización se debe aplicar varias veces para encontrar las soluciones Pareto óptimas múltiples. Como cada corrida es independiente de las demás, generalmente no se obtiene un efecto sinergético. Por tanto, delinear el frente Pareto óptimo resulta computacionalmente muy caro.
b) Requiere conocimiento previo del problema a resolver y es sensible a los pesos utilizados.
c) Algunos algoritmos son sensibles a la forma del frente Pareto (problemas con curvas cóncavas).

d) La variación entre las diferentes soluciones encontradas depende de la eficiencia del optimizador de un solo objetivo. Podría darse el caso de encontrar siempre la misma solución o soluciones muy parecidas, en corridas múltiples.
e) Como los optimizadores de un solo objetivo no son eficientes en búsquedas de universos discretos [DEB99], tampoco serán eficientes para optimizaciones multiobjetivo en espacios discretos.

Investigaciones publicadas en [DEB99] han demostrado que todas las dificultades arriba mencionadas pueden ser superadas con la utilización de algoritmos evolutivos. Las características de estos algoritmos hacen posible que:

a) se manejen espacios de búsqueda de casi cualquier tamaño y características;
b) debido a su paralelismo inherente, se puedan generar múltiples soluciones en una sola corrida, permitiendo un efecto sinergético.

Con la afirmación previa, se introduce el criterio que se utilizará en el presente trabajo para lidiar con el problema de enrutamiento multicast multiobjetivo en redes de computadoras. Es decir, se empleará algoritmos evolutivos. Es adecuado, entonces, dedicar la siguiente Sección a la presentación de estos algoritmos.

2.5 Algoritmos Evolutivos en Optimización Multiobjetivo

El término algoritmo evolutivo (*Evolutionary Algorithm* - EA) se refiere a técnicas de búsqueda y optimización inspiradas en el modelo de la evolución propuesto por Charles Darwin [DAR85], luego de sus viajes exploratorios.

En la naturaleza los individuos se caracterizan mediante cadenas de material genético que se denominan cromosomas. En los cromosomas se halla codificada toda la información relativa a un individuo y a sus tendencias. El cromosoma es una cadena de símbolos llamados genes. Cada individuo posee un nivel de adaptación al medio que lo dota de mayor capacidad de sobrevivencia y generación de descendencia. Tal nivel de adaptabilidad está ligado a las características que están codificadas en sus cromosomas. Como el material genético puede transmitirse de padres a hijos al ocurrir el apareamiento, los hijos resultantes poseen cadenas de cromosomas parecidas a las de sus padres y combinan las características de los mismos. Por lo tanto, si padres con buenas características se cruzan, posiblemente generarán hijos igualmente buenos o incluso mejores [GOL89].

Para resolver un problema de búsqueda u optimización utilizando algoritmos evolutivos y los conceptos sugeridos, primero se representa como individuos de una población finita a un número dado de posibles soluciones del problema. A este proceso se denomina codificación. En la codificación de un individuo debe estar presente toda la información relevante al mismo y que se considera influye en la optimización o búsqueda. Como se citó anteriormente, el cromosoma es una cadena símbolos conocidos como genes. El cromosoma representa una solución al problema de optimización.

Para cada cromosoma es determinado su nivel de aptitud o adaptabilidad (fitness), dependiendo de la calidad de la solución que representa. Posteriormente los individuos existentes generan a otros individuos mediante los operadores genéticos como selección, cruzamiento y mutación. El operador de selección elige los padres que se cruzarán. La probabilidad de que un individuo sea escogido como padre y/o que sobreviva hasta la siguiente generación está ligada a su adaptabilidad: a mayor adaptabilidad, mayor probabilidad de sobrevivencia y de tener descendientes, de la misma forma que ocurre en los procesos naturales. Luego de escogerse los padres, se procede a la recombinación o cruzamiento de los mismos para obtener a la nueva generación. De esta manera, en cada nueva generación se tienen buenas probabilidades de que la población se componga de mejores individuos, ya que los hijos heredarán las características buenas de sus padres, y al combinarlas podrán ser aún mejores. Por otro lado, durante la recombinación pueden ocurrir alteraciones (mutaciones) en la información genética de un individuo. Si tales alteraciones se producen para bien, originarán un individuo bueno con alta adaptabilidad y la alteración se transmitirá a los nuevos individuos; si el cambio no es benéfico, el individuo alterado tendrá una adaptabilidad baja y poca o ninguna descendencia, con lo que la alteración prácticamente morirá con él. De esta manera, luego del curso de varias generaciones, la población habrá evolucionado hacia individuos con nivel de adaptabilidad elevado, es decir, representarán buenas soluciones al problema propuesto [GOL89].

Los operadores descritos reciben el nombre de operadores de búsqueda u operadores genéticos. La reproducción enfoca la atención en los individuos con alta adaptabilidad, y de esta manera **explota** la información disponible sobre la adaptación del individuo al medio ambiente. La recombinación y la mutación perturban de alguna manera a los individuos y proveen así de heurísticas para la **exploración** del espacio de búsqueda. Por ello se dice que los EAs utilizan conceptos de explotación y exploración. A pesar de ser simplistas desde el punto de vista de la biología, estos algoritmos son suficientemente complejos como para proveer mecanismos de búsqueda robustos y que se adaptan a gran variedad de problemas [GOL89].

2.5.1 Algoritmos Genéticos

Los algoritmos genéticos se utilizan en varios ámbitos, principalmente para búsquedas y optimizaciones. En la práctica se implementa el algoritmo escogiendo una codificación para las posibles soluciones del problema. La codificación se realiza mediante cadenas de bits, números o caracteres para representar a los cromosomas. Luego, las operaciones de cruzamiento y mutación se aplican de manera muy sencilla mediante funciones de manipulación de vectores. A pesar de que se han realizado numerosas investigaciones [HEI00] sobre codificación usando cadenas de longitud variable y aún otras estructuras, gran parte del trabajo con algoritmos genéticos se enfoca en cadenas de bits de longitud fija. Justamente el hecho de utilizar cadenas de longitud fija y la necesidad que existe de codificar las posibles soluciones son las dos características cruciales que diferencian a los algoritmos genéticos de la programación genética, que no posee representación de longitud fija y que normalmente no utiliza una codificación del problema y sus soluciones.

El ciclo de trabajo de un GA (también conocido como ciclo generacional) es generalmente el siguiente: calcular la adaptabilidad de los individuos en la población, crear una nueva población mediante selección, cruzamiento y mutación, y finalmente descartar la población vieja y seguir iterando utilizando la población recién creada. A cada iteración de este ciclo se la conoce como una generación. Cabe observar que no hay una razón teórica para que este sea el modelo de implementación. De hecho este comportamiento puntual no se observa como tal en la naturaleza. Sin embargo el modelo sigue siendo válido y conveniente [GOL89].

La primera generación (generación 0) de este proceso, opera sobre una población de individuos generados al azar. A partir de allí, los operadores genéticos se aplican para mejorar a la población. La Figura 2.3 presenta el pseudocódigo del algoritmo genético simple.

```
Inicio
    $t = 0$                                          // generación 0
    InicializarPoblacion P(t)                        // inicializar la población de individuos al azar
    Evaluar P (t)                                    // Hallar el fitness de todos los individuos
    Hacer{
            P(t+1) = SeleccionarPadres P(t)          // seleccionar población para generar descendientes
            Cruzar P(t+1)                            // recombinar los “genes” de padres seleccionados
            Mutar P(t+1)                             // perturbar la población generada estocásticamente
            Adaptabilidad P(t+1)                     // calcular adaptabilidad de la población recién creada
            $t = t + 1$                              // incrementar el contador de generaciones
    }Mientras no se cumpla el criterio de parada
Fin
```

Figura 2.3. Pseudocódigo del algoritmo genético simple.

El operador de selección, según se ha apuntado, simula el proceso de selección natural donde el más fuerte tiene mayor capacidad de supervivencia. En el GA la capacidad de supervivencia de un individuo está ligada a su valor de adaptabilidad. Este operador se aplica a cada iteración sobre una población de individuos de tamaño constante, con el objetivo de seleccionar individuos prometedores para generar la nueva población. Entre los individuos seleccionados pueden hallarse dos o más individuos idénticos. Esto se debe a que los individuos con baja adaptabilidad tienen poca probabilidad de ser elegidos, mientras que los de buena adaptabilidad son seleccionados con mayor frecuencia. El operador de selección puede aplicarse de diversas maneras. A continuación se describen dos implementaciones muy conocidas: selección por torneo y selección por ruleta. En la selección por torneos [GOL89] se escoge al azar un grupo de individuos y gana el torneo aquel con mejor adaptabilidad. La cantidad de individuos que se escogen para la competencia se fija de antemano y permanece constante en la implementación tradicional. Esta forma de selección refleja de manera más adecuada el proceso natural de selección. Por otro parte, la implementación de selección por ruleta es fácilmente comprensible imaginando una ruleta en la que el número de partes en que se divide la misma es igual a la cantidad de individuos de la población, siendo el tamaño de cada parte proporcional a la adaptabilidad de cada individuo. Es de esperar que al hacer girar la ruleta varias veces, se obtendrá mayor cantidad de individuos con alta adaptabilidad. Aunque pueden existir otras formas de realizar la selección, las dos anteriormente mencionadas son las más comunes en las implementaciones publicadas.

Una vez seleccionados los individuos, se aplica a cada par de ellos el operador de cruzamiento. La operación de cruzamiento también puede ser realizada de maneras diferentes y aquí solo se discutirá una de ellas: el cruzamiento de dos puntos. En el mismo, se escogen aleatoriamente dos puntos de cruce. El primer punto de cruce define el punto (gen) a partir del cual se hará el intercambio genético, y el segundo define la longitud del segmento (número de genes) a intercambiar. La Figura 2.4 ilustra esta operación, en la cual los cromosomas consisten de cadenas de enteros de longitud 6.

Figura 2.4. Operación de cruzamiento de dos puntos. El primer punto de cruce es 2 mientras que la longitud del segmento a intercambiar es 3.

El operador de cruzamiento representa una forma de búsqueda local, en las inmediaciones del espacio de búsqueda que rodea a los padres. Por su parte, el proceso de mutación es básicamente una búsqueda aleatoria. Se selecciona aleatoriamente una posición específica dentro de cromosoma del individuo a mutar, para luego cambiar el valor contenido en dicha posición. Dado que en la naturaleza la probabilidad de que ocurra una mutación es pequeña, los GAs tratan de representar lo mismo asignando un valor de ocurrencia muy bajo al operador de mutación [GOL89].

2.5.2 Algoritmos Evolutivos y Optimización Multiobjetivo

Los EAs resultan interesantes pues están especialmente dotados para lidiar con las dificultades propuestas por los MOPs. Esto se debe a que pueden devolver un conjunto entero de soluciones luego de una corrida simple y no presentan otras limitaciones propias de las técnicas tradicionales. Incluso, algunos investigadores han sugerido que los EAs se comportarían mejor que otras técnicas de búsqueda ciega [FON95a, FON95b]. Esta afirmación todavía requiere de una demostración fehaciente y debería considerarse a la luz de los teoremas de “no hay almuerzo gratis” (“*no free lunch*”) relacionados con la optimización, y que indican que si un método se comporta “mejor” que otros en un conjunto de problemas, tendrá un desempeño “peor” en otro conjunto de problemas [WOL97]. De todos modos, la realidad es que hoy día existen pocas alternativas válidas en cuanto a otros posibles métodos de solución [HOR97]. De hecho, las numerosas publicaciones recientes sobre resolución de MOPs usando EAs (que pueden hallarse compiladas en [COE00]), parecen considerar este hecho, y han dado lugar al nacimiento de todo un nuevo campo de investigación: los MOEAs, algoritmos evolutivos aplicados a optimización multiobjetivo.

Ya en 1967, Rosemberg sugirió, sin hacer simulaciones, un método de búsqueda genética para aplicaciones químicas con diversos objetivos [ROS67]. Sin embargo, las primeras implementaciones prácticas fueron sugeridas recién en 1984 [SHA84]. Posterior a esto no se realizaron estudios significativos por casi una década, a excepción de un procedimiento de ordenación basado en no-dominancia, expuesto por Goldberg [GOL89]. Nuevas implementaciones de MOEAs surgieron entre 1991 y 1994 [KUR91, HAJ92, FON93, HOR94, SRI94]. Posteriormente, estas sugerencias –y variaciones de las mismas– se implementaron para la resolución de diferentes MOPs [PAR98]. Recientemente, algunos investigadores se han puesto la tarea de tratar tópicos específicos de la búsqueda y optimización multiobjetivo con algoritmos evolutivos, como son: convergencia al frente Pareto [VAN98a, VAN98b, RUD98] y aplicación de elitismo [PAR98, OBA98]. También se han publicado resúmenes analíticos generales, comparaciones y compendios [FON95b, HOR97, VAN98a, DEB99, COE99].

2.6 Resumen del Capítulo

En el presente Capítulo se ha dado la definición formal de un problema de optimización multiobjetivo y se han definido términos y conceptos que serán utilizados en los posteriores Capítulos. Luego, se ha propuesto a los MOEAs como herramientas que poseen cualidades deseables para la resolución de MOPs. Habiendo introducido estos conceptos, en el Capítulo 3 se formula el problema de enrutamiento multicast como un MOP. Posteriormente, en el Capítulo 4 se proponen dos MOEAs para la resolución de dicho problema.

3 Formulación Matemática

3.1 Introducción

Según se introdujo en el Capítulo anterior, la optimización multiobjetivo puede aplicarse a un número muy grande de problemas. En particular, en este trabajo, se considera el problema de enrutamiento multicast en redes de computadoras.

Uno de los objetivos principales del problema de enrutamiento multicast en redes de computadoras es la minimización del retardo máximo. Cuando hay más de un nodo destino que debe recibir el flujo de datos desde el nodo fuente, surgen otros objetivos como ser el retardo medio a los nodos destinos y el costo total del árbol multicast. Además, como se vio en el Capítulo 1, la minimización de la utilización máxima de los enlaces balancea la carga sobre la red, minimizando de esta manera el congestionamiento, reduciendo el retardo total y minimizando la pérdida total de paquetes [DON04]. Este trabajo considera el Problema de Enrutamiento Multicast como un MOP, en el cual debe ser construido un árbol multicast para un grupo dado minimizando 1- la utilización máxima de los enlaces del árbol, 2- el costo del árbol, 3- el retardo máximo de extremo a extremo, o simplemente retardo máximo y 4- el retardo medio.

El presente Capítulo está organizado de la siguiente manera: en la Sección 3.2 se da el modelo matemático del problema de enrutamiento multicast en redes de computadoras. Posteriormente, de forma a aclarar el significado de las notaciones matemáticas, la Sección 3.3 presenta un ejemplo de un problema de enrutamiento multicast multiobjetivo.

3.2 Modelo Matemático

La red es modelada como un grafo dirigido $G = (V, E)$, donde V es el conjunto de vértices y E el conjunto de arcos. Los vértices del grafo representan nodos de la red, y los arcos representan los enlaces entre los nodos. Sea:

- $(i,j) \in E$: enlace entre los nodos i y j; $i, j \in V$.

- $z_{ij} \in \Re^+$: capacidad del enlace (i,j).
- $c_{ij} \in \Re^+$: costo del enlace (i,j).
- $d_{ij} \in \Re^+$: retardo (delay) del enlace (i,j).
- $t_{ij} \in \Re^+$: tráfico actual fluyendo a través del enlace (i,j).
- $s \in V$: nodo origen del grupo multicast.
- $N \subseteq V\text{-}\{s\}$: conjunto de nodos destinos del grupo multicast.
- $\phi \in \Re^+$: demanda de tráfico del grupo multicast, en bps.
- $T(s,N)$: árbol multicast con origen en s y conjunto de nodos destinos N.
- $p_T(s, n) \subseteq T(s,N)$: camino que conecta el nodo fuente s y el nodo destino $n \in N$.
- $d(p_T(s, n))$: retardo del camino $p_T(s, N)$, dado por la suma de los retardos de los enlaces que conforman el camino. Esto es,

$$d\left(p_T(s,n)\right) = \sum_{(i,j)\in p_T(s,n)} d_{ij} \qquad \textbf{(3.1)}$$

Usando las notaciones definidas arriba, el problema de enrutamiento multicast puede ser definido como un MOP que trata de hallar el árbol multicast $T(s,N)$ minimizando las siguientes funciones objetivos:

1- Utilización máxima de los enlaces del árbol:

$$\alpha_T = \underset{(i,j)\in T}{Max} \left\{ \frac{\phi + t_{ij}}{z_{ij}} \right\} \qquad \textbf{(3.2)}$$

2- Costo del árbol:

$$C_T = \phi \sum_{(i,j)\in T} c_{ij} \qquad \textbf{(3.3)}$$

3- Retardo máximo de extremo a extremo:

$$D_M = \underset{n\in N}{Max} \left\{ d\left(p_T(s,n)\right) \right\} \qquad \textbf{(3.4)}$$

4- Retardo medio:

$$D_A = \frac{1}{|N|} \sum_{n \in N} d\left(p_T(s,n)\right) \tag{3.5}$$

sujeto a restricciones de capacidad en los enlaces:

$$\phi + t_{ij} \leq z_{ij} \;\; ; \; \forall (i,\, j) \in T(s, N) \tag{3.6}$$

La Ecuación 3.2 da la utilización máxima de los enlaces del árbol. La minimización de esta métrica implica que son preferidos aquellos árboles con la menor utilización máxima de sus enlaces, de forma a balancear la carga sobre la red [SEO02, DON04].

La Ecuación 3.3 da el costo total del árbol multicast. Formulado de esta manera, esta métrica puede tener más de un significado. Si el costo representa alguna cantidad monetaria para el proveedor de servicio, entonces sería deseable proveer un servicio multicast a un costo mínimo. Por ejemplo, el costo de cada enlace puede tener un significado monetario a pagar por bit de información por unidad de tiempo. En este caso, la función objetivo dada por la Ecuación 3.3 da el costo total a pagar por bit de información por unidad de tiempo para llevar a cabo la transmisión multicast. Por otra parte, si el costo de cada enlace tiene un valor unitario, el costo total es el consumo total de ancho de banda del grupo. En cualquiera de los casos, la minimización de esta métrica es siempre deseable. Es claro que formular el problema con enlaces cuyos costos pueden tener valores arbitrarios positivos provee generalidad al problema.

La Ecuación 3.4 define el retardo máximo de extremo a extremo. Debido al creciente número de aplicaciones multicast sensibles a esta métrica, es de suma importancia que los árboles multicast sean construidos minimizando esta función objetivo. A diferencia de los trabajos publicados hasta la fecha [KOM93, RAV98, XIA99, ZHE01, ARA02], en los cuales el retardo máximo es considerado como una restricción, en el presente trabajo se formula explícitamente como una función objetivo a minimizar. Las ventajas de este enfoque serán resaltadas en el Ejemplo 3.1.

La Ecuación 3.5 define el retardo medio a los nodos destinos. Es importante notar que, si bien las Ecuaciones 3.3 y 3.4 definen funciones objetivos relacionadas, tienen distintos significados.

Por último, la Ecuación 3.6 restringe el tráfico máximo en cada enlace a la capacidad del mismo.

3.3 Problema de Ejemplo

Para ilustrar la formulación propuesta, se presenta un ejemplo de enrutamiento multicast. Considere la red de computadoras mostrada en la Figura 3.1 [BAR00], en donde los números sobre cada enlace (i, j) corresponden a sus respectivos valores de retardo d_{ij}, costo c_{ij} y tráfico t_{ij}. El retardo está dado en milisegundos mientras que el tráfico a través de los enlaces en Mbps. Cada enlace tiene una capacidad de 1.5 Mbps. En el estado actual de la red, se genera una solicitud multicast en la que el nodo s = 5 desea transmitir un flujo de información ϕ = 0.2 Mbps a los nodos N = {0, 4, 9, 10, 13}.

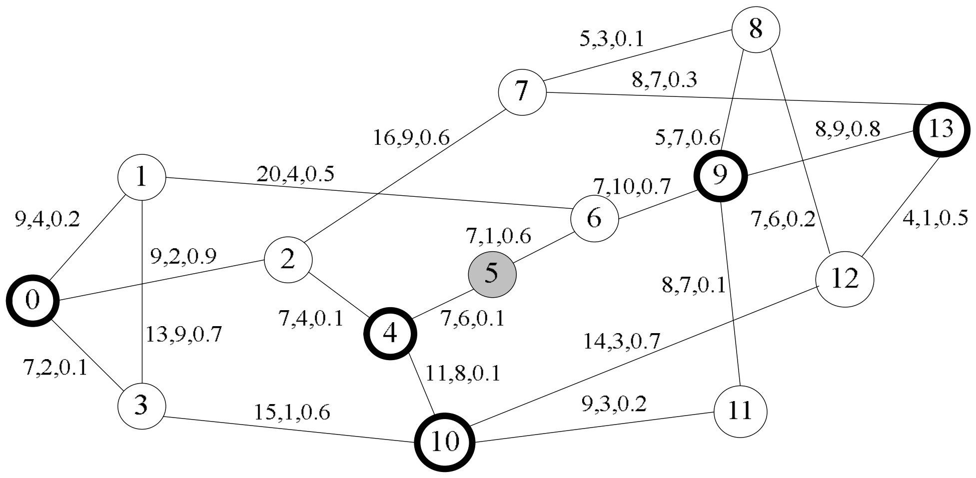
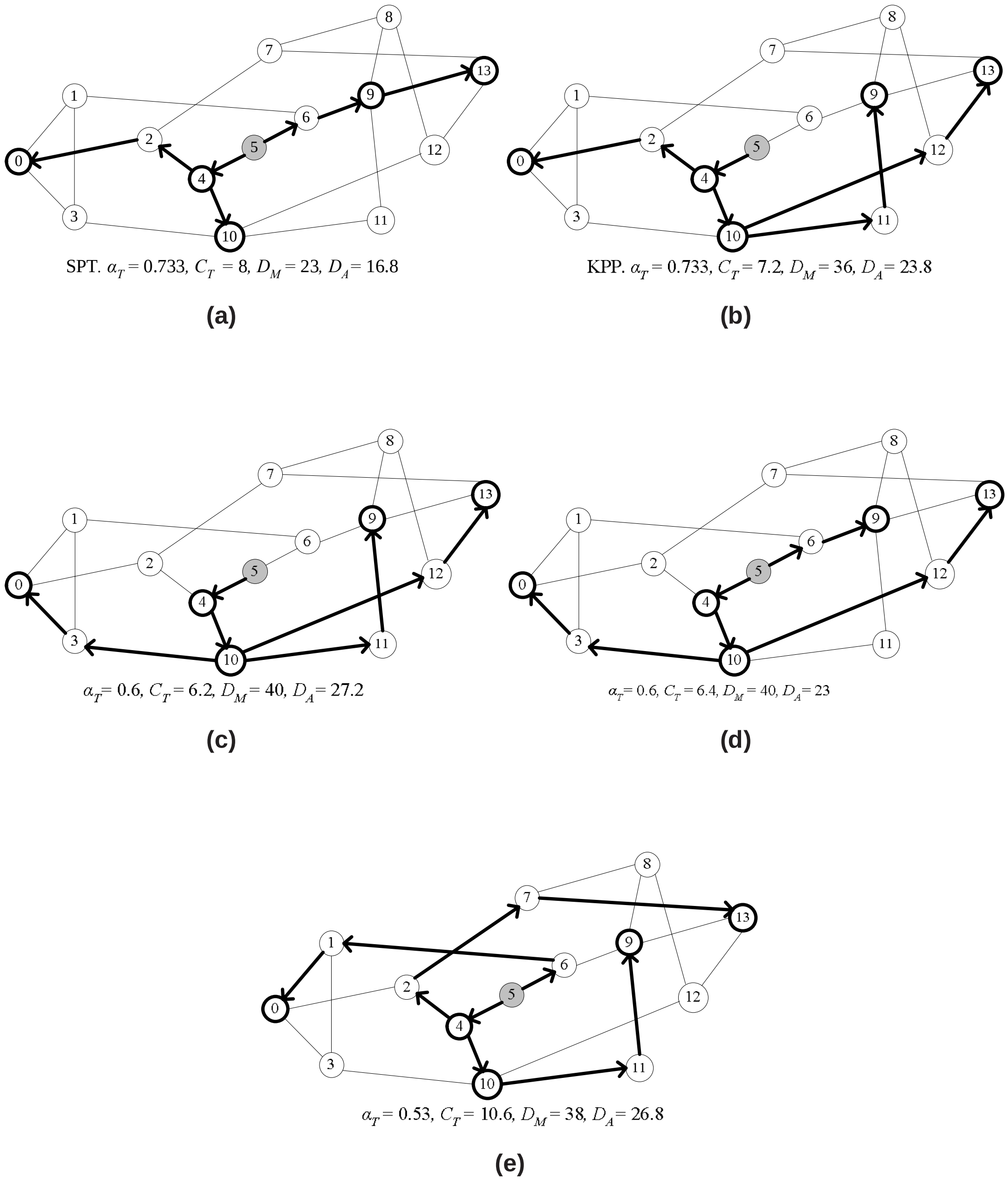


Figura 3.1. Red de la NSF. Cada enlace (i,j) tiene asignado sus correspondientes valores de retardo d_{ij}, costo c_{ij} y tráfico t_{ij} a través de él. Los enlaces tienen una capacidad de 1.5 Mbps. En el estado actual, se genera una solicitud multicast dada por s = 5, ϕ = 0.2 Mbps y N = {0, 4, 9, 10, 13}.

La forma tradicional de enrutar la demanda solicitada es a través del árbol del camino más corto. Si la aplicación es sensible al retardo máximo, entonces, para cada nodo destino, el camino desde el origen hasta él debe tener un retardo menor que el valor máximo permitido. Una alternativa es entonces la construcción del árbol multicast constituido por los caminos más cortos (SPT), en términos de retardo, a los nodos destinos. Esta solución es mostrada en la Figura 3.2 (a).

Dado que SPT no considera ninguna otra métrica, el costo y la utilización de los enlaces pueden tener valores altos. Esto significa que el árbol podría estar formado por enlaces muy utilizados y que el costo podría ser muy elevado. Por lo tanto, se podrían hallar otras soluciones que, sacrificando el retardo, tengan un menor costo y una utilización de los enlaces mas baja. Se podría usar un algoritmo basado en restricciones como KPP [KOM93a, KOM93b], de forma a optimizar el costo del árbol. KPP y otros algoritmos basados en restricciones [RAV98, XIA99, ZHE01, ARA02] requieren un valor de retardo máximo definido a priori. Entonces, suponga que el grupo multicast desea un árbol con un valor D_M menor a 40 ms. En dicho caso, KPP hallaría la solución mostrada en la Figura 3.2 (b). Note que el costo disminuyó a 7.2, mientras que el retardo máximo de extremo a extremo aumentó a 36 y el medio a 23.8. A pesar de ello, D_M es inferior a la restricción dada, como era de esperar.

El problema de los algoritmos basados en restricciones es la necesidad del valor de retardo máximo dado a priori, que puede descartar soluciones de muy bajo costo con valores D_M apenas superiores al máximo permitido. La Figura 3.2 (c) muestra un árbol con estas características. Esta solución tiene un costo de apenas 6.2. Note que dicha solución no puede ser hallada por KPP ni por ningún otro algoritmo basado en restricciones, dado que su retardo máximo de extremo a extremo es de 40 ms. En la práctica, esta solución puede ser una muy buena alternativa, dado que el límite superior de retardo máximo dado a priori es apenas inferior. Una alternativa similar es mostrada en la Figura 3.2 (d). Esta solución, a pesar de tener un costo un poco mayor a la anterior, tiene un retardo medio mucho menor. Por lo tanto, también podría ser una buena opción si el retardo medio es considerado. Por último, la Figura 3.2 (e) muestra un árbol cuya utilización máxima de los enlaces es menor que la α_T de las anteriores soluciones.



SPT. $\alpha_T = 0.733$, $C_T = 8$, $D_M = 23$, $D_A = 16.8$

(a)

KPP. $\alpha_T = 0.733$, $C_T = 7.2$, $D_M = 36$, $D_A = 23.8$

(b)

$\alpha_T = 0.6$, $C_T = 6.2$, $D_M = 40$, $D_A = 27.2$

(c)

$\alpha_T = 0.6$, $C_T = 6.4$, $D_M = 40$, $D_A = 23$

(d)

$\alpha_T = 0.53$, $C_T = 10.6$, $D_M = 38$, $D_A = 26.8$

(e)

Figura 3.2. Cinco soluciones alternativas para el grupo multicast de la Figura 3.1.

Note que las cinco alternativas son soluciones no dominadas. De hecho, por búsqueda exhaustiva, puede comprobarse que estas 5 soluciones forman parte del conjunto Pareto del problema. Por lo tanto, son soluciones óptimas en un sentido multiobjetivo. Esto puede verse con mayor claridad en las Figuras 3.3 a 3.6, donde se muestra el frente Pareto del problema, que consta de 16 vectores objetivos no dominados.

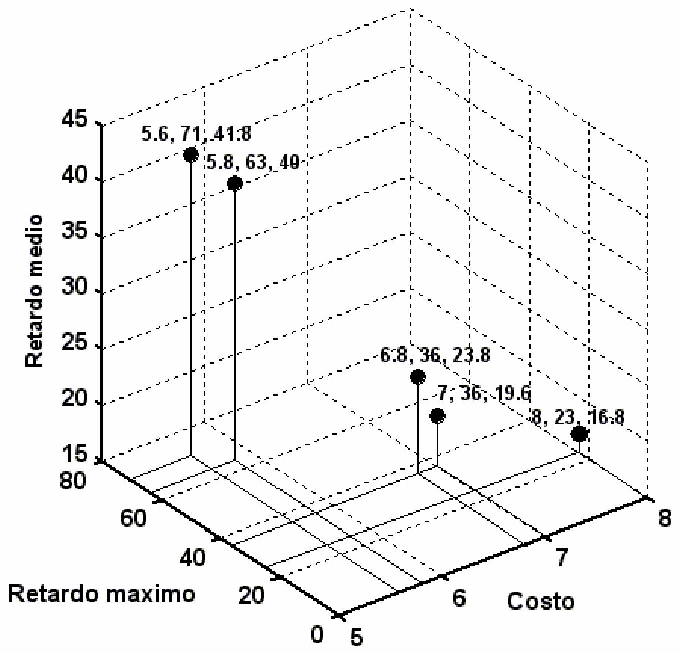


Figura 3.3. Vectores objetivos del frente Pareto con α_T = 0.73.

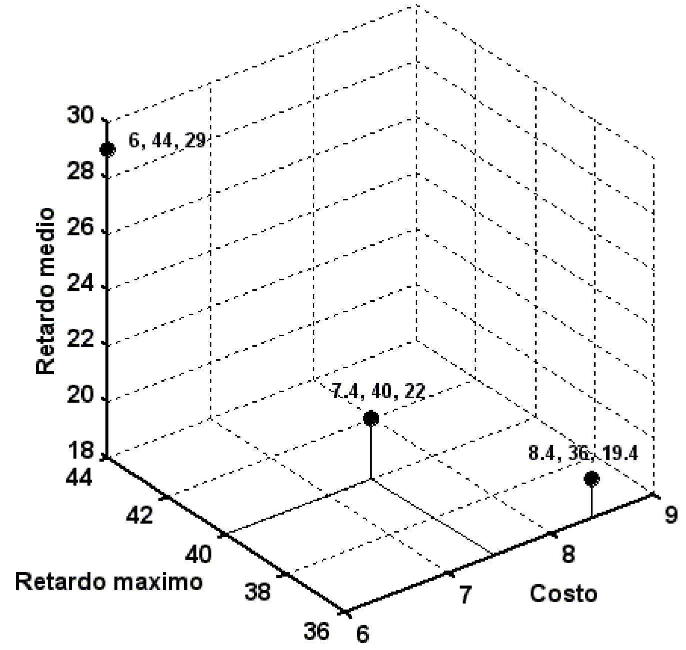


Figura 3.4. Vectores objetivos del frente Pareto con α_T = 0.66.

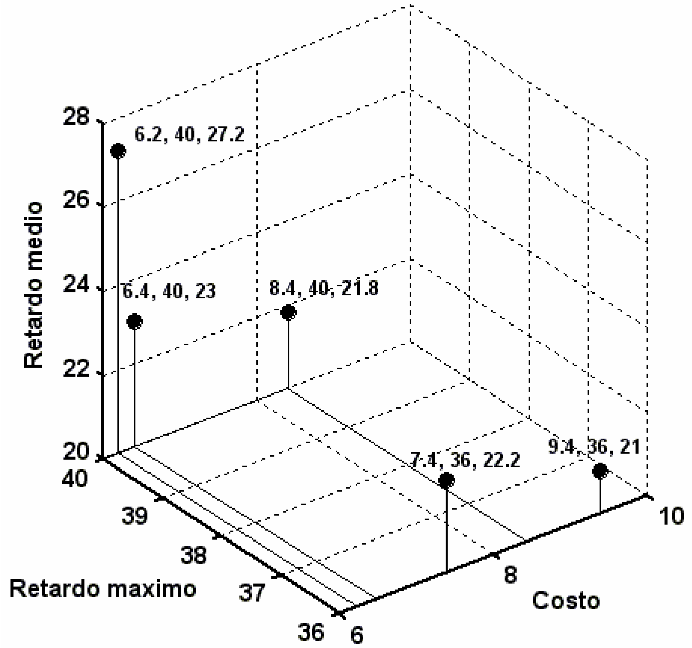


Figura 3.5. Vectores objetivos del frente Pareto con $\alpha_T = 0.6$.

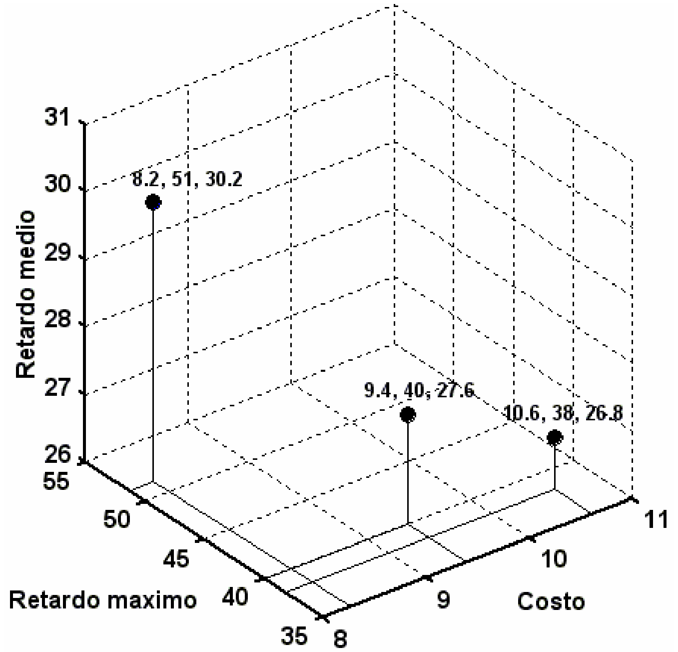


Figura 3.6. Vectores objetivos del frente Pareto con $\alpha_T = 0.53$.

La Figura 3.3 muestra aquella parte del frente Pareto con la función objetivo α_T =0.73. De forma similar, las Figuras 3.4, 3.5 y 3.6 muestran las partes del frente Pareto con la función objetivo α_T = 0.66, α_T = 0.6 y α_T = 0.53 respectivamente. Note la relación de compromiso entre las distintas funciones objetivos. Por ejemplo, de la Figura 3.5 se puede derivar que, si se desea una solución con α_T = 0.6, existen 5 soluciones óptimas. Entre ellas, algunas con D_M = 40 y otras D_M = 36. Si se opta por una solución con D_M = 36, se puede elegir entre una solución con C_T = 7.1 y D_A = 22.2, y otra con C_T = 9.4 y D_A = 21.

Claramente un conjunto de soluciones óptimas como este puede ser de gran utilidad en la práctica, pues la elección de la mejor opción puede ser tomada a posteriori y según las necesidades de cada caso. De la formulación matemática se puede concluir que esta gran variedad de soluciones óptimas solo puede ser obtenida si las cuatro funciones objetivos son tratadas en forma independiente. Es decir, no forman una función objetivo escalar combinada a través de una combinación lineal, ni tampoco son tratadas como simples restricciones. De esta forma, el enfoque multiobjetivo elude el problema de definir los pesos relativos para cada objetivo o hacer restricciones a priori, hallando todo un conjunto de soluciones óptimas.

Dado que el problema del árbol multicast de costo mínimo en una red de computadoras es un problema NP-completo [KOM93a], el problema formulado en esta sección también lo es. Esencialmente, la intratabilidad del problema se debe a la existencia de un número de árboles que crece exponencialmente con el número de nodos de la red, y las soluciones óptimas solo pueden hallarse por búsqueda exhaustiva [KOM93a]. Evidentemente, dicha técnica no tiene practicidad alguna, por lo que se necesitan métodos alternativos que no exijan la exploración completa del espacio de búsqueda. En general, estos métodos no tienen por objetivo hallar todas las soluciones del conjunto Pareto óptimo, sino simplemente encontrar una aproximación lo suficientemente buena.

La complejidad del espacio de búsqueda, su tamaño y las últimas propuestas presentadas de algoritmos genéticos multiobjetivo [ZIT99], hacen de estos últimos candidatos ideales para ser aplicados en este ámbito. Debido a que estos algoritmos no han sido ampliamente estudiados en aplicaciones de redes de computadoras, el presente trabajo es un aporte original y útil en el estudio de MOEAs.

La ventaja principal de los MOEAs es su paralelismo inherente, pues son capaces de mantener todo un conjunto de soluciones (al mantener una población) y por tanto pueden obtener varios miembros del conjunto Pareto óptimo en una única corrida. Además, no requieren mayor información sobre el espacio de búsqueda; basta con especificar los objetivos y la manera de calcularlos. Tampoco presentan dificultades relacionadas con la complejidad del espacio de búsqueda, es decir, son bastante generales y robustos [ZIT99].

3.4 Resumen del Capítulo

En el presente Capítulo se ha formulado el problema de enrutamiento multicast como un Problema de Optimización Multiobjetivo, donde las funciones objetivos son formuladas explícitamente y deben ser optimizadas simultánea e independientemente. El siguiente Capítulo presenta dos algoritmos evolutivos para la resolución de este problema. Dichos algoritmos están basados en el *Strength Pareto Evolutionary Algorithm* (SPEA), un MOEA caracterizado por su buen funcionamiento en la resolución de problemas de optimización multiobjetivo de complejidad similar al presentado en esta sección [ZIT99, BAR01, SOT02, BAR03].

4 Algoritmos Propuestos

4.1 Introducción

Los MOEAs son herramientas algorítmicas desarrolladas recientemente para la resolución de MOPs. Como hemos apuntado en capítulos previos, su popularidad se debe principalmente a que:

a) resultan promisorios para la integración de los aspectos de búsqueda y de toma de decisiones que plantean los MOPs;
b) pueden realizar búsquedas aún en espacios ilimitados y altamente complejos;
c) tienen la habilidad de mantener toda una población de soluciones;
d) existen pocas alternativas –aparte de ellos– para el tratamiento adecuado de MOPs.

La característica principal de un MOEA es que optimiza simultáneamente un conjunto múltiple de objetivos.

En [VAN99] se puede hallar una lista prolija de reportes de utilización de MOEAs para la resolución de MOPs complejos. A partir de este trabajo es posible identificar las propuestas que han marcado más fuertemente el desarrollo del área y que pueden denominarse en la categoría de "más usualmente imitados". En particular, hoy día, se ha identificado al *Strength Pareto Evolutionary Algorithm* (SPEA) como uno de los algoritmos más promisorios debido a su capacidad de obtener buenos resultados bajo pruebas exhaustivas con problemas de prueba cuidadosamente seleccionados [SRI94, ZIT99]. SPEA usa una mezcla de técnicas previamente usadas y otras recién establecidas para encontrar soluciones Pareto óptimas diferentes y en paralelo. Las características que comparte con otros algoritmos previamente propuestos son:

a) guarda las soluciones no dominadas que ha encontrado hasta el momento en una población externa. De esta manera implementa elitismo, que consiste en seleccionar las mejores soluciones de la generación actual y copiarlas en la población de la siguiente generación;

b) usa el concepto de dominancia Pareto para la asignación de adaptabilidad a los individuos;

c) realiza *clustering* [MOR80] para reducir el número de soluciones no dominadas que tiene almacenadas, sin destruir las características del frente Pareto delineado hasta el momento.

Por otro lado, SPEA presenta, de manera exclusiva, los siguientes conceptos:

a) combina las tres características arriba mencionadas en un único algoritmo (hasta ahora ellas siempre habían sido propuestas aisladas unas de otras);
b) la adaptabilidad de un individuo se determina a partir de las soluciones que se encuentran en la población de individuos no dominados solamente; de este modo, no es relevante que los miembros de la población general se dominen unos a otros;
c) todas las soluciones que forman parte del conjunto externo de soluciones no dominadas participan en la selección;
d) se sugiere un nuevo método para inducir la formación de regiones dentro de la población y así preservar la diversidad genética; este método está basado en el concepto de dominancia Pareto.

Además de los estudios comparativos citados arriba [SRI94, ZIT99], SPEA también ha demostrado su buen funcionamiento en la resolución de problemas de ingeniería como el diseño topológico de redes de computadoras [BAR03], la programación de bombeo de agua en sistemas de suministro de agua [SOT02] y la compensación reactiva de potencia [BAR01]. Por tal motivo, para resolver el problema de enrutamiento multicast en redes de computadoras, en el presente trabajo se proponen dos algoritmos evolutivos multiobjetivos, denominados MMA1 y MMA2 (*Multiobjective Multicast Algorithm*) basados en el SPEA.

Habiendo presentado las principales características del SPEA, el presente Capítulo continúa de la siguiente manera: en la Sección 4.2 se explica en detalle el algoritmo MMA1. Dado que ambos algoritmos se basan en el SPEA y siguen el mismo ciclo generacional, la mayor parte de las explicaciones también son válidas para MMA2. Luego, en la Sección 4.3 se presenta MMA2 y se explica su funcionamiento y las diferencias con MMA1.

4.2 Algoritmo MMA1

Debido a que el algoritmo propuesto está basado en el SPEA, el primer punto a considerar es la representación de los cromosomas. La codificación utilizada está basada en el trabajo de [HWA00], y es explicada a continuación.

4.2.1 Representación de los Cromosomas

Dado un grupo multicast con origen en un nodo s y conjunto de nodos destinos $N=\{n_1, n_2, \ldots, n_{|N|}\}$, donde $|\cdot|$ denota cardinalidad, una tabla de rutas para cada destino $n_i \in N$ es construida. Cada tabla está compuesta por las R rutas de menor retardo, las R de menor costo y las R de menor utilización máxima de los enlaces, donde el retardo d_p, el costo c_p y la utilización α_p de una ruta p se definen a continuación:

Retardo d_p:

$$d_p = \sum_{(i,j)\in p} d_{ij} \tag{4.1}$$

Costo c_p:

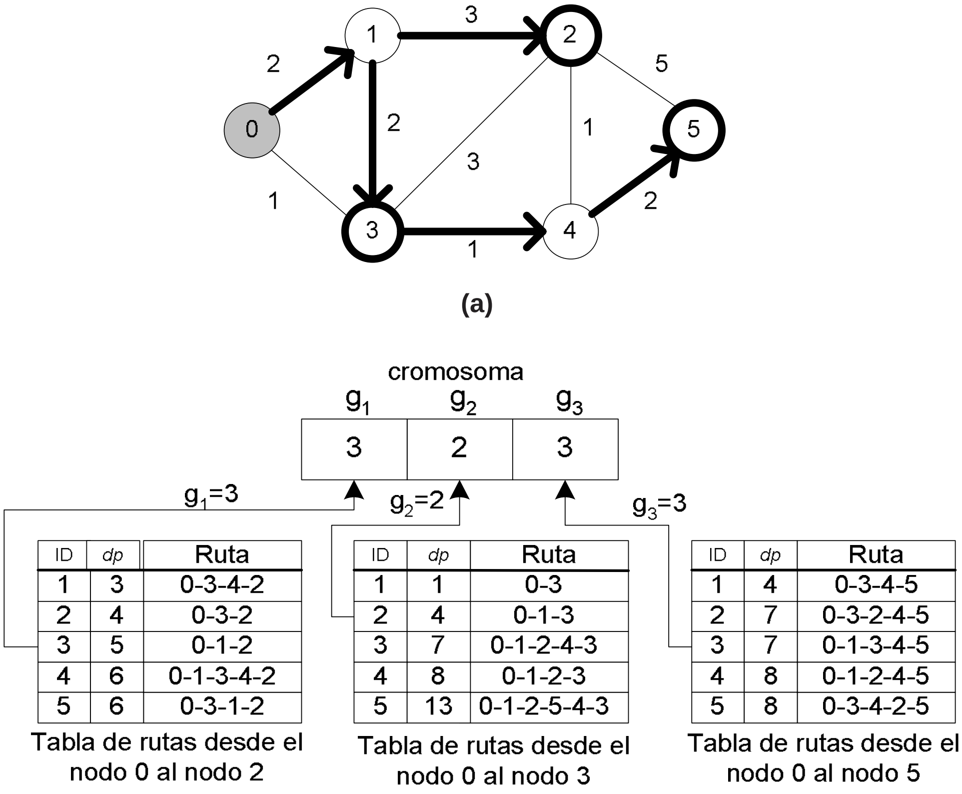
$$c_p = \sum_{(i,j)\in p} c_{ij} \tag{4.2}$$

Utilización α_p:

$$\alpha_p = \underset{(i,j)\in p}{Max}\left\{\frac{t_{ij}}{z_{ij}}\right\} \tag{4.3}$$

Para hallar las R rutas de menor retardo, las R de menor costo y las R de menor utilización de los enlaces, es utilizado el algoritmo de Yen [YEN71]. Dicho algoritmo iterativamente encuentra las R rutas más cortas, empezando por la de menor longitud.

Para hallar las R rutas de menor retardo, la longitud de una ruta viene dada por la Ecuación 4.1. De forma similar, para hallar las R rutas de menor costo y las R de menor utilización, la longitud de una ruta está dada por la Ecuación 4.2 y 4.3, respectivamente. Entonces, un cromosoma es representado por una cadena de enteros de longitud $|N|$, donde cada elemento (gen) $g_i \in \{1, 2, \ldots, 3R\}$, representa la ruta desde el nodo s hasta el nodo destino n_i. La relación entre un cromosoma, los genes y las tablas de rutas es mostrada en la Figura 4.1. El cromosoma en (b) representa el árbol en (a). Por simplicidad, en este ejemplo solo la métrica de retardo d_{ij} es considerada, y la tabla de rutas está compuesta por los R = 5 caminos de menor retardo. Cada tabla cuenta con 3 columnas: el identificador de la ruta (ID), el retardo de la ruta (dp) y la ruta.



(a)

ID	*dp*	Ruta
1	3	0-3-4-2
2	4	0-3-2
3	5	0-1-2
4	6	0-1-3-4-2
5	6	0-3-1-2

Tabla de rutas desde el nodo 0 al nodo 2

ID	*dp*	Ruta
1	1	0-3
2	4	0-1-3
3	7	0-1-2-4-3
4	8	0-1-2-3
5	13	0-1-2-5-4-3

Tabla de rutas desde el nodo 0 al nodo 3

ID	*dp*	Ruta
1	4	0-3-4-5
2	7	0-3-2-4-5
3	7	0-1-3-4-5
4	8	0-1-2-4-5
5	8	0-3-4-2-5

Tabla de rutas desde el nodo 0 al nodo 5

(b)

Figura 4.1. (a) Una red de 6 nodos con su respectiva métrica de retardo d_{ij} sobre cada enlace (i, j). (b) Relación entre un cromosoma, los genes y las tablas de rutas.

4.2.2 Descripción del Algoritmo

Habiendo presentado la representación de los cromosomas, a continuación se explica el algoritmo propuesto. Siguiendo el esquema del SPEA [ZIT99], el algoritmo mantiene una población evolutiva P y un conjunto externo de soluciones Pareto P_{nd} halladas hasta la generación actual. Empezando con una población inicial aleatoria, los individuos evolucionan hasta soluciones óptimas, las cuales son guardadas en P_{nd}. La Figura 4.2 muestra el pseudocódigo del algoritmo propuesto. Para enfatizar el hecho que P y P_{nd} son conjuntos que van evolucionando a lo largo de las generaciones, estas aparecen como funciones de t, el contador de generaciones.

```
Inicio
    Leer grupo multicast y demanda
    t=0;
    Construir tablas;
    Inicializar $P$(t) y $P_{nd}$(t);
    Hacer{
            Descartar individuos iguales en $P$(t);
            Evaluar individuos en $P$(t);
            Marcar individuos no dominados en $P$(t);
            Actualizar conjunto Pareto $P_{nd}$(t);
            Calcular adaptabilidad en $P$(t) y $P_{nd}$(t);
            $P$(t+1) = Seleccionar de $P$(t) $\cup$ $P_{nd}$(t);
            $P$(t+1) = Cruzamiento $P$(t+1);
            $P$(t+1) = Mutación $P$(t+1);
            t = t+1;
}Mientras no se cumpla el criterio de parada
```

Figura 4.2. Pseudocódigo del algoritmo MMA1.

La **inicialización** de la población evolutiva P consiste en generar $|P|$ cromosomas en forma aleatoria. Con la representación propuesta, la inicialización simplemente consiste en crear vectores de enteros de longitud $|N|$, donde cada elemento puede tomar un valor entre 1 y $3R$. La población P_{nd} es inicializada con aquellos individuos no dominados de P. Luego, al iniciar el ciclo principal del algoritmo, la función **Descartar individuos iguales en *P*** compara cada individuo de la población con los demás individuos de P, y reemplaza los cromosomas duplicados por nuevos generados aleatoriamente. Este procedimiento es utilizado con el fin de evitar que la operación de cruzamiento entre dos individuos idénticos produzco otro igual, lo cual podría reducir la habilidad de búsqueda del algoritmo [HWA00]. Entonces, la población P consiste en individuos distintos unos de otros. Luego, estos son evaluados por la función **Evaluar *P*** utilizando las ecuaciones 3.2, 3.3, 3.4 y 3.5.

Dado que el SPEA mantiene una población externa de soluciones no dominadas P_{nd} que es actualizada en cada generación, el algoritmo debe hallar aquellas soluciones no dominadas de P candidatas a ser incluidas en P_{nd}. El pseudocódigo mostrado en la Figura 4.3, denominado **Marcar individuos no dominados en P**, es usado para dicha tarea. El procedimiento consiste en recorrer la población de individuos P y comparar cada uno de ellos con todos los demás. Si alguno domina al individuo que se está examinando, tal individuo recibe una marca de dominado. Los individuos que no recibieron tal marca al final del procedimiento conforman el conjunto de soluciones no dominadas de la población P.

```
Inicio
    Desde $i$ = 1 hasta |$P$| - 1
        Si el individuo $i$ es no dominado
            Desde $j$ = $i$ + 1 hasta |$P$|
                Si  el individuo $j$ domina al individuo $i$
                    Marcar al individuo $i$ como dominado
                    $j$ = |$P$|
                Fin Si
                Si  el individuo $i$ domina al individuo $j$
                    Marcar al individuo $j$ como dominado
                Fin Si
            Fin Desde
        Fin Si
    Fin Desde
Fin
```

Figura 4.3. Pseudocódigo del procedimiento Marcar individuos no dominados en P.

Una vez halladas las soluciones no dominadas en P, cada una de estas es comparada con los individuos de P_{nd}, de forma a **actualizar el conjunto Pareto P_{nd}**. Si un individuo no dominado de P es no dominado respecto a P_{nd}, entonces es copiado a este conjunto. Además, todos los individuos P_{nd} dominados por dicho individuo son removidos del conjunto externo P_{nd}.

Una de las particularidades del *SPEA* es su forma de asignar la adaptabilidad a cada individuo. El proceso se realiza de modo a lograr que los individuos dominantes y más representativos tengan mejores valores de adaptabilidad. A la vez, el proceso está pensado para inducir a mantener la diversidad genética y obtener soluciones distribuidas en todo el frente Pareto. Para ello, el procedimiento de asignación de adaptabilidad se realiza en dos fases: 1- a cada individuo $i \in P_{nd}$ se le asigna un valor real $q_i \in [0,1)$, llamado la fuerza o *strength*[1] de dicha solución. q_i es proporcional al número de individuos $j \in P$ para los cuales $i \rhd j$ (es decir, es una medida de la cantidad de individuos dominados por i) dividido el tamaño de la población evolutiva P:

$$q_i = \frac{\left| j \mid j \in P \wedge i \rhd j \right|}{|P|} \tag{4.4}$$

2- La fuerza de un individuo $j \in P$ se calcula como la suma de las fuerzas de todas las soluciones externas no dominadas $i \in P_{nd}$ que dominan a j. Se suma 1 a ese total para asegurar que los miembros de P_{nd} tengan mejor fuerza que los miembros de P. Así, la fuerza $q_j \in [1,N)$ del individuo $j \in P$ se expresa de la siguiente manera:

$$q_j = 1 + \sum_{i \in P_{nd}, i \rhd j} q_i \tag{4.5}$$

La adaptabilidad puede ser entonces calculada como la inversa de la fuerza. Debido al operador de selección utilizado en este trabajo, torneo binario, no es necesario el cálculo de dichos valores, pues simplemente son utilizados los valores de fuerza de los individuos.

Para aclarar el efecto de este mecanismo de asignación de fuerza, considere la Figura 4.4. Dicha gráfica corresponde al espacio objetivo de un problema con dos funciones objetivos a minimizar, una población de 6 individuos y un conjunto externo de soluciones no dominadas de tamaño 4. Gráficamente, el espacio objetivo, el cual es dominado por los cuatro vectores de decisión correspondientes a las 4 soluciones no dominadas representadas por cruces, está dividido en cuatro regiones. Los vectores objetivos tienen asociado sus respectivos valores de fuerza, calculados según las ecuaciones 4.4 y 4.5. Cada subconjunto de P_{nd} define una nueva región dominada por los individuos que conforman el subconjunto. Por ejemplo, la esquina superior derecha, sombreada con mayor intensidad, es dominada por el subconjunto conformado por los cuatro vectores no dominados. De forma similar, la región en la cual está ubicado el vector objetivo con fuerza 14/7 está dominada por los vectores con fuerza 4/7 y 3/7. Note

---

[1] Este término se tomó de [ZIT99], donde fue introducido en el contexto de sistemas clasificadores. Se refiere

que, a mayor número de soluciones Pareto dominando una región, mayor la intensidad del sombreado de dicha región. El objetivo de esta forma de asignación de fuerza es distribuir los individuos en toda la superficie tal que:

a) las regiones dominadas por unos pocos individuos $i \in P_{nd}$ (aquellas sombreadas con menor intensidad) contengan más individuos (tengan menor fuerza o mayor adaptabilidad, de forma a aumentar la probabilidad de ser seleccionados) que aquellas regiones dominadas por muchos individuos $i \in P_{nd}$ (aquellas sombreadas más intensamente), y
b) una región contenga tantos individuos como aquellas regiones dominadas por la misma cantidad de individuos $i \in P_{nd}$.

Este mecanismo intuitivamente refleja la idea de preferir individuos ubicados en las cercanías del frente Pareto y al mismo tiempo, distribuirlos en toda la superficie factible. El primer aspecto puede ser fácilmente notado en la Figura: individuos localizados en las regiones mas claras tienen valores más bajos de fuerza (mayor adaptabilidad). Para notar el segundo aspecto, considere la región superior izquierda y la inferior derecha. Ambas son dominadas por un solo individuo almacenado en P_{nd}. Sin embargo, el individuo ubicado en la región inferior derecha tiene menor fuerza (mayor adaptabilidad) que los dos individuos ubicados en la región superior derecha. De esta forma, la probabilidad de ser seleccionado es mayor en aquellas regiones con menos individuos.

Habiendo calculado la fuerza de cada individuo, el operador de **selección** es aplicado en cada generación sobre el conjunto $P \cup P_{nd}$, con el objeto de seleccionar buenos individuos que generarán la siguiente población P. En el presente trabajo se propone el uso del torneo binario [GOLD89] como operador de selección, tal como fue explicado en la Sección 2.5.1. Este operador selecciona pares de individuos (con reemplazo), de los cuales aquel individuo de menor fuerza gana el torneo y es seleccionado para formar parte de la población de la siguiente generación. El número de individuos seleccionados es igual a $|P|$, el tamaño de la población evolutiva. Luego, sobre cada par de individuos seleccionados es aplicado el operador de **cruzamiento** de dos puntos [GOLD89, HWA00]

---

a una cantidad que resume la utilidad de una regla. Aquí refleja la utilidad de una solución no dominada.

como se introdujo en la Sección 2.5.1. Los genes de los nuevos individuos son **mutados** (cambiados) aleatoriamente con una probabilidad P_{mut} [GOL89].

Por último, el algoritmo se detiene cuando se alcanza uno de los siguientes **criterios de parada**: 1- se ha alcanzado un número de generaciones máximo N_{MAX}; 2- no se han encontrado nuevas soluciones óptimas en las últimas N_{SUC} generaciones.

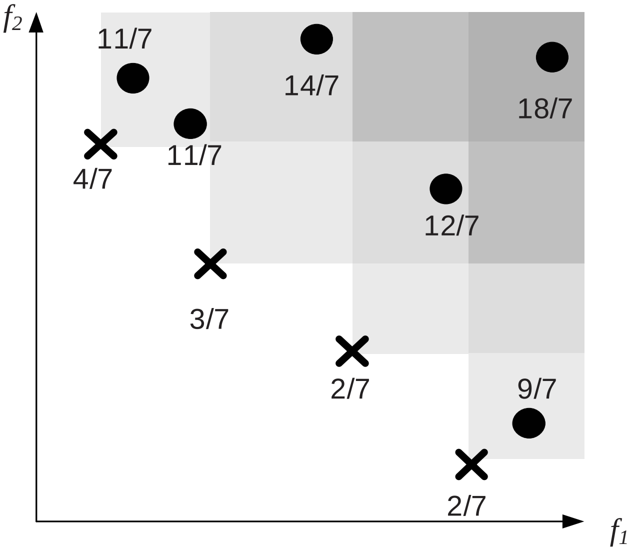


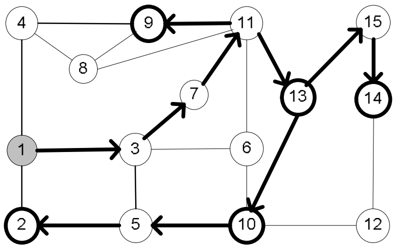
Figura 4.4. Mecanismo de asignación de *strength* usado por el SPEA.

4.3 Algoritmo MMA2

El segundo algoritmo propuesto para la resolución del problema de enrutamiento multicast multiobjetivo, MMA2, también está basado en el SPEA. La principal diferencia con MMA1 es la representación de los cromosomas. De este hecho se deriva que los procedimientos de inicialización de la población, el cruzamiento y la mutación también son diferentes. Los demás procedimientos, asignación de fuerza y adaptabilidad, actualización del conjunto externo P_{nd} y cálculo de las funciones objetivos son los mismos. Por lo tanto, a continuación se presentan la codificación utilizada en MMA2 y los procedimientos exclusivos de este algoritmo.

4.3.1 Representación de los Cromosomas

Una representación alternativa de cromosomas puede ser utilizada si cada individuo es representado directamente con un árbol, sin la utilización de tablas como en MMA 1. En este caso, un árbol multicast es representado como un conjunto de sus enlaces [RAV98, ARA02]. La Figura 4.5 muestra un árbol multicast para un grupo dado y su representación utilizando el esquema propuesto.



(a)

{ (1, 3), (3, 7), (7, 11), (11, 9),
(11, 13), (13, 10), (10, 5),
(5, 2), (13, 15), (15, 14) }

(b)

Figura 4.5. (a) Árbol multicast con origen $s = 1$ y conjunto destino $N = \{2, 9, 10, 13, 14\}$. (b) Representación como un conjunto de sus enlaces.

4.3.2 Descripción del Algoritmo

Dado que MMA 2 también está basado en el SPEA, tiene un esquema similar al mostrado en la Figura 4.2, excepto que no necesita del procedimiento "Construir tablas". Como en MMA 1, MMA2 requiere de la inicialización de una población evolutiva *P* en forma aleatoria. MMA 2 propone un algoritmo de construcción de árboles basado en el algoritmo de Prim [HOR86]. Empezando con el nodo fuente *s*, el algoritmo expande el árbol eligiendo un nuevo enlace en cada iteración, añadiendo de esta forma un nuevo nodo al árbol. La elección del enlace se hace aleatoriamente. El algoritmo se detiene cuando todos los nodos destinos multicast son añadidos al árbol. De esta forma, el algoritmo, denominado PrimRST (*Prim Random Steiner Tree*), **inicializa la población** evolutiva *P*. Como en MMA 1, la población P_{nd} es inicializada con aquellos individuos no dominados pertenecientes a la primera generación de *P*. El pseudocódigo del procedimiento PrimRST es mostrado en la Figura 4.6.

```
PrimRST($G(V,E)$, $s$, $N$)                      //recibe la red $G(V, E)$ y el grupo $s$, $N$.
$T_p = \{\ \}$;                                  //Inicialización del árbol a crear (vacío)
$V_c = \{s\}$;                                   //Nodos conectados a $T_p$
$A = \{(s, j) \in E \mid j \in V\}$;             //Enlaces candidatos a ser añadidos a $T_p$
Mientras ($N \cup \{s\} \not\subset V_c$)
    Elegir aleatoriamente un enlace $(i, j) \in A$; $i \in V_c$;
    $A = A - \{(i, j)\}$;
    Si $j \notin V_c$ entonces                   // conectar $j$ al árbol $T_p$
        $T_p = T_p \cup \{(i, j)\}$
        $V_c = V_c \cup \{j\}$;
        $A = A \cup \{(j, w) \in E \mid w \notin V_c\}$;
    Fin si
Fin Mientras
Borrar los enlaces no utilizados de $T_p$
Retornar $T_p$
```

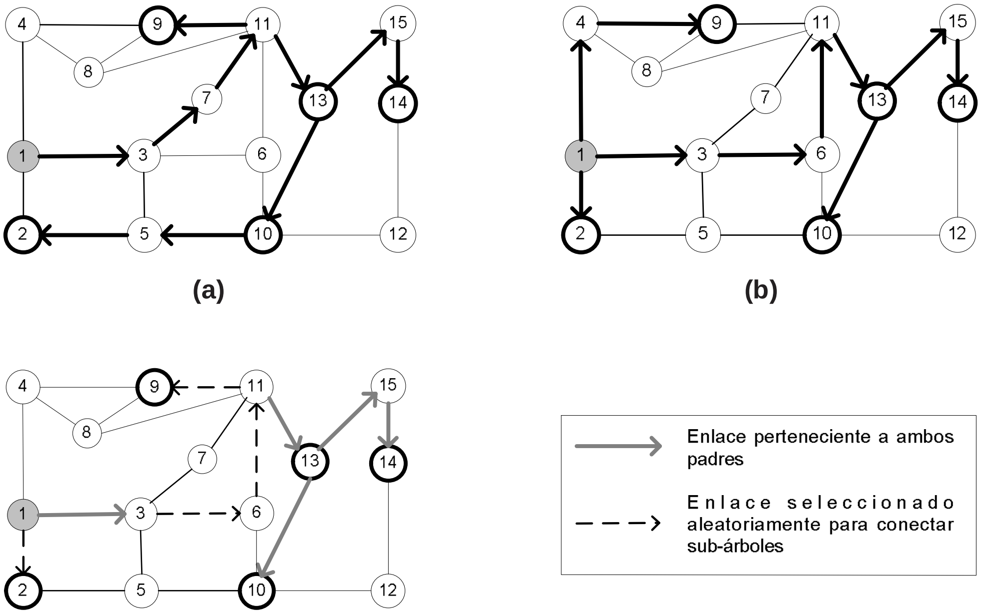
Figura 4.6. Procedimiento utilizado en MMA 2 para la construcción aleatoria de un árbol T_p.

Dada la diferencia en la representación de los individuos, los operadores de cruzamiento y mutación difieren de aquellos implementados en MMA1. En MMA 2, la operación de **cruzamiento** consiste de dos pasos:

1- identificar los enlaces comunes de ambos padres y copiarlos al individuo hijo. De acuerdo al procedimiento de selección explicado en MMA2, los individuos con menor fuerza tienen mayor probabilidad de ser seleccionados. Entonces, los enlaces de los padres posiblemente sean "buenos". Sin embargo, considerar solo estos enlaces puede conducir a un individuo hijo que consiste en sub-árboles separados, y por lo tanto algunos nuevos enlaces deben ser añadidos;

2- conectar los sub-árboles hasta formar un árbol multicast. En esta etapa, los sub-árboles son conectados de forma aleatoria, y cada sub-árbol tiene asignado un nodo raíz. El algoritmo de interconexión añade en cada iteración un enlace cuyo nodo origen forma parte de un sub-árbol. Dos sub-árboles son conectados cuando el enlace elegido tiene como nodo destino la raíz de uno de los sub-árboles – T_1 – y como origen un nodo perteneciente al otro sub-árbol – T_2 –. La raíz del nuevo sub-árbol (compuesto por los dos sub-árboles) es el nodo raíz de T_2. El algoritmo para cuando todo el grupo multicast es interconectado. Luego, los enlaces no utilizados en la interconexión del grupo son borrados del árbol.

La Figura 4.7 ilustra un ejemplo de cruzamiento entre dos individuos seleccionados.



(a) (b)

(c)

Figura 4.7. (a), (b) Individuos seleccionados, sobre los cuales se realiza la operación de cruzamiento. (c) Resultado del operador de cruzamiento.

Por último, es necesario apuntar que el operador de mutación no fue considerado en MMA 2, pero si el procedimiento que descarta individuos iguales.

4.4 Resumen del Capítulo

En el presente Capítulo se han propuesto dos algoritmos evolutivos basados en el SPEA para la resolución del problema de enrutamiento multicast en redes de computadoras. Habiendo explicado en detalle cada uno de ellos, el siguiente Capítulo presenta los resultados experimentales de las simulaciones y compara ambos algoritmos, MMA1 y MMA2, con otros algoritmos publicados en conferencias internacionales arbitradas.

5 Resultados Experimentales

5.1 Introducción

En el presente Capítulo se presentan los experimentos y las simulaciones realizadas con el objeto de evaluar los algoritmos propuestos.

En la Sección 5.2 se presentan los resultados experimentales de MMA1 y MMA2 en problemas de prueba estáticos, comparando ambos algoritmos entre si. En estos problemas, se conoce el frente Pareto óptimo. Por lo tanto, se evalúan los algoritmos considerando como métricas de desempeño la cantidad de soluciones óptimas encontradas y el tiempo de cómputo. Posteriormente, en la Sección 5.3, MMA1 y MMA2 son evaluados utilizando problemas de prueba dinámicos, en los cuales se simula el comportamiento de una red de computadoras donde las solicitudes multicast llegan a la red una tras otra.

5.2 Problemas de Prueba Estáticos

5.2.1 Problema de Prueba 1

El primer Problema de Prueba fue tomado de [ZHE01, ARA02]. Sea G (V, E) la red mostrada en la Figura 5.1, que consiste de 15 nodos y 44 enlaces no dirigidos. Cada enlace (i, j) tiene asociado un costo c_{ij} y un retardo d_{ij} en ms, en este orden. Dado el grupo conformado por s = 1 y N = {2, 9, 10, 13, 14}, se desea hallar el árbol multicast de costo mínimo C_T, sujeto a D_M < 26 ms. En este trabajo extendemos la formulación del problema a uno multiobjetivo, donde los objetivos a optimizar son: 1-C_T, 2-D_M y 3-D_A. C_T se calcula como la suma de los costos de los enlaces (Ecuación 3.3, sin multiplicar por ϕ, dado que este problema no considera dicha variable). De esta forma, la restricción de retardo ha sido formulada explícitamente como una nueva función objetivo. Note que este es un problema similar al Ejemplo 2.1, pero con una función objetivo más: retardo máximo de extremo a extremo. A pesar de la nueva función objetivo, el conjunto Pareto óptimo está compuesto por las mismas siete soluciones mostradas en la Figura 2.1. Este conjunto fue hallado por búsqueda exhaustiva en un tiempo de corrida de aproximadamente 20 minutos [CRI04b]. El frente Pareto correspondiente es mostrado en la Figura 5.2.

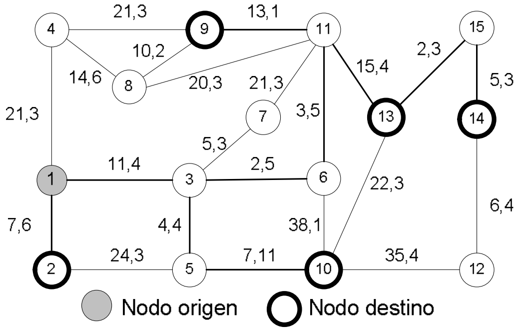


Figura 5.1. Problema de Prueba 1 [ZHE01, ARA02].

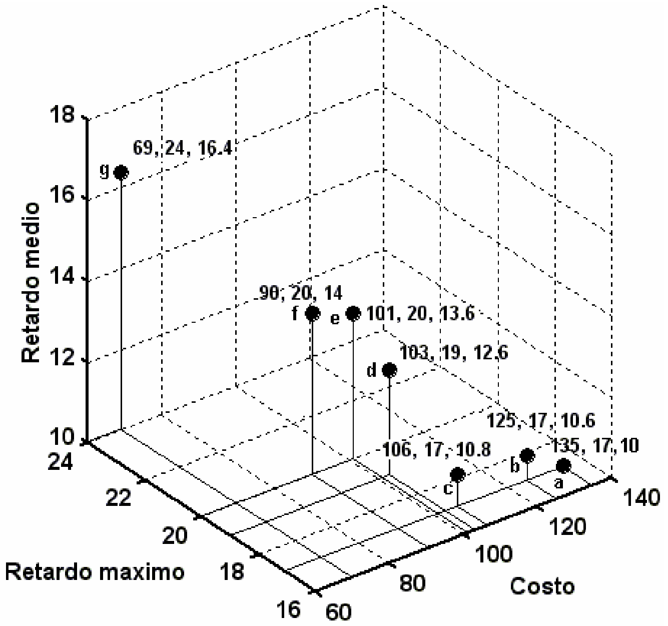


Figura 5.2. Frente Pareto óptimo del Problema de Prueba 1 [ZHE01, ARA02].

Para este Problema de Prueba, 100 corridas fueron realizadas. Los algoritmos pararon al ocurrir uno de los siguientes eventos: 1- el tiempo de corrida alcanzó los 150 ms, o 2- se han encontrado las 7 soluciones óptimas. El tamaño de la población fue de 40 individuos para MMA1 y 25 individuos para MMA2. La probabilidad de mutación de MMA1 fue de 0.4.

MMA2 halló el conjunto Pareto en las 100 corridas, en un tiempo medio de 22.7 ms. La corrida de mayor duración fue de 50 ms y la desviación estándar de 11.4 ms. Por su parte, MMA1 no pudo hallar el conjunto Pareto en todas las corridas. En algunas, solo obtuvo parte de dicho conjunto. La Tabla 5.1 muestra la frecuencia de soluciones Pareto halladas por MMA1. En el peor de los casos (8 corridas), 3 soluciones óptimas fueron halladas, mientras que en el mejor (47 corridas), todas las soluciones del conjunto Pareto fueron obtenidas.

Tabla 5.1. Frecuencia de soluciones halladas por MMA1

Soluciones halladas	3	4	5	6	7
Nº corridas	8	22	17	6	47

Los resultados experimentales mostraron que MMA1 halló las soluciones a, b y c en el 100 % de las corridas, y las soluciones a, b, c y d en el 92 % de las corridas. Las soluciones e, f, y g (juntas) solo fueron obtenidas en aquellas corridas donde todas las soluciones óptimas fueron halladas. Es decir, en el 47 % de los casos.

De la Figura 5.2, note que los vectores objetivos del frente Pareto están dispersos unos de otros, dificultando la obtención del conjunto Pareto óptimo. Sin embargo, MMA2 pudo hallar dicho frente en todas las corridas. Esto demuestra una mayor capacidad de exploración de las regiones del espacio por parte de este algoritmo.

A pesar de que solo MMA2 obtuvo el conjunto Pareto en todas las corridas, los resultados de MMA1 aún son satisfactorios, dado que en casi el 50 % de las corridas obtuvo dicho conjunto. Proveer esta amplia variedad de soluciones es de gran importancia, pues, a diferencia de los algoritmos mono-objetivos, la solución más adecuada puede ser elegida para cada ocasión. Como ejemplo, considere un problema en el cual se desee una solución con D_M < 20 ms (a priori). En el mejor de los casos, un algoritmo basado en restricciones obtendría la solución d. Sin embargo, este enfoque no permite ver al tomador de decisiones la existencia de las soluciones e y f, las cuales tienen un retardo máximo de 20 ms, solo 1 ms por arriba de la restricción. Con un enfoque multiobjetivo, dichas soluciones serían proveídas y la solución más adecuada podría ser tomada a posteriori.

5.2.2 Problema de Prueba 2

El segundo Problema de Prueba fue formulado en la Sección 3.3. El conjunto Pareto óptimo, hallado por búsqueda exhaustiva en un tiempo de corrida de aproximadamente 3 horas [CRI04a], está compuesto por las 16 soluciones de la Tabla 5.2. El frente Pareto correspondiente es mostrado en las Figuras 3.3 a 3.6.

Tabla 5.2. Conjunto Pareto óptimo del Problema de Prueba 2 [CRI04a]

	Vector objetivo	Árbol
a	0.73, 8, 23, 16.8	(5,4),(4,2),(2,0),(4,10),(5,6),(6,9),(9,13)
b	0.73, 7, 36, 19.6	(5,4),(4,10),(10,12),(12,13),(4,2),(2,0),(5,6),(6,9)
c	0.73, 6.8, 36, 23.8	(5,4),(4,2),(2,0),(4,10),(10,11),(11,9),(10,12),(12,13)
d	0.6, 6.4, 40, 23	(5,4),(4,10),(10,12),(12,13),(10,3),(3,0),(6,9)
e	0.66, 8.4, 36, 19.4	(5,4),(4,10),(5,6),(6,1),(1,0),(6,9),(9,13)
f	0.6, 9.4, 36, 21	(5,4),(4,10),(5,6),(6,1),(1,0),(6,9),(9,8),(8,12),(12,13)
g	0.6, 7.4, 36, 22.2	(5,4),(4,10),(10,12),(12,13),(5,6),(6,9),(6,1),(1,0)
h	0.53, 10.6, 38, 26.8	(5,4),(4,10),(10,11),(11,9),(4,2),(2,7),(7,13),(5,6),(6,1),(1,0)
i	0.53, 9.4, 40, 27.6	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(4,2),(2,7),(7,13)
j	0.6, 8.4, 40, 21.8	(5,4),(4,10),(10,3),(3,0),(5,6),(6,9),(9,8),(8,12),(12,13)
k	0.66, 7.4, 40, 20.2	(5,4),(4,10),(10,3),(3,0),(5,6),(6,9),(9,13)
l	0.6, 6.2, 40, 27.2	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(10,12),(12,13)
m	0.53, 8.2, 51, 30.2	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(9,8),(8,12),(12,13)
n	0.66, 6, 44, 29	(5,4),(4,10),(10,3),(3,0),(10,12),(12,13),(13,9)
o	0.73, 5.8, 63, 40	(5,4),(4,2),(2,0),(0,3),(3,10),(10,11),(11,9),(10,12),(12,13)
p	0.73, 5.6, 71, 41.8	(5,4),(4,2),(2,0),(0,3),(3,10),(10,12),(12,13),(13,9)

Para este problema, 100 corridas fueron realizas. Los parámetros de MMA1 fueron: $|P| = 40$, $R = 30$ y $P_{mut} = 0.4$. El tamaño de la población de MMA2 fue de 25 individuos. Las corridas pararon al alcanzar 100 ms.

Las métricas de comparación de este problema son: número máximo (S_{max}), mínimo (S_{min}), promedio (S_a) y desviación estándar (σ_s) de soluciones Pareto óptimas obtenidas en las corridas. Los resultados son presentados en la Tabla 5.3.

Tabla 5.3. Resultados del Problema 2

	MMA2	MMA1
S_{max}	16	15
S_{min}	14	14
S_a	15.82	14.13
σ_s	0.41	0.1

Ambos algoritmos tuvieron un buen rendimiento es este problema de prueba, encontrando al menos 14 soluciones óptimas (87,5 % del conjunto Pareto). MMA1 obtuvo 15 soluciones óptimas en 2 corridas y 14 en 98 oportunidades. Por su parte, MMA2 encontró el conjunto Pareto en 83 ocasiones, 15 soluciones óptimas en 16 corridas y 14 en 1 oportunidad. Es claro que MMA2 tuvo un mejor rendimiento que MMA1.

Una particularidad de MMA1 en estas corridas fue que halló 14 soluciones Pareto óptimas en el 98 % de los casos. Las dos soluciones no obtenidas en estas corridas fueron los árboles (o) y (p). De forma a clarificar las razones por las cuales no halló estas soluciones en menos de 100 ms, la Figura 5.3 muestra el frente Pareto del problema en un espacio objetivo donde se ha suprimido la métrica α_T, de manera a graficarlo en un sistema coordenado considerando las funciones objetivos C_T, D_M y D_A como coordenadas. Sin embargo, es necesario aclarar que cada vector objetivo está constituido por la n-tupla dada por α_T, C_T, D_M y D_A. Los vectores objetivo (o) y (p) están situados en una "esquina" del espacio objetivo. Es de suponer entonces que la principal dificultad de MMA1 se debe a su capacidad de exploración de aquellas regiones del espacio solución donde podrían encontrarse buenas soluciones.

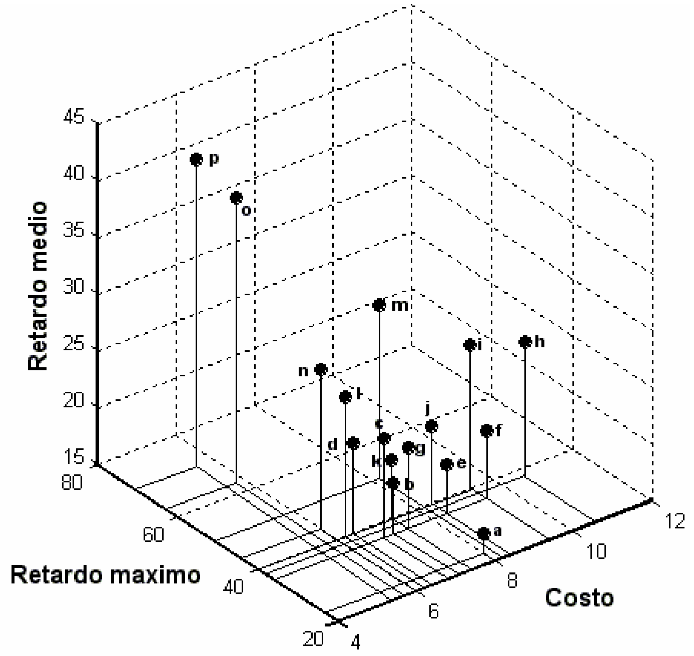


Figura 5.3. Frente Pareto óptimo del Problema de Prueba 2 [CRI04a].

Dado que ambos algoritmos están basados en el SPEA, y MMA2 no ha tenido el problema citado arriba, es importante apuntar la diferencia entre ambos algoritmos, de forma a entender el motivo de la disparidad de rendimiento. Como se explicó en el Capítulo 4, la principal diferencia entre MMA1 y MMA2 es la representación cromosómica. De esto se deriva que el cruzamiento tampoco es el mismo. Por lo tanto, se infiere que la codificación utilizada en MMA2 da a este algoritmo una mayor capacidad de exploración de las regiones del espacio solución. Esta afirmación también fue notada en los problemas dinámicos de prueba presentados en la siguiente sección.

5.3 Problemas de Prueba Dinámicos

MMA1 y MMA2 también fueron evaluados en situaciones dinámicas, utilizando 4 topologías de red: NTT [BAR00], NSF [CRI04a] y 2 topologías obtenidas de [SPR02]. En dicho trabajo, se publican 5 topologías de AS. En este trabajo, las dos redes de mayor tamaño fueron utilizadas. Para estas pruebas, $\forall(i,j)\in E$, c_{ij} = 1. De esta forma, C_T (ver Ecuación 3.4) calcula el consumo total de ancho de banda. Para cada topología, cuatro distintas situaciones dinámicas (corridas), en las cuales las solicitudes multicast llegan una después de otra, fueron simuladas. La primera corrida consiste en solicitudes multicast en las que el tamaño de los grupos varía entre 3 y el 35 % del número total de nodos de la red. En la segunda corrida, el tamaño de los grupos varía entre 35% y 70% del número de nodos de la red. La tercera corrida es similar a la primera y la cuarta a la segunda, con la diferencia que en estas dos corridas d_{ij} = 1, $\forall(i, j)\in E$. De esta forma, los algoritmos propuestos se evalúan en modos denso y esparzo [SEO02], considerando también la situación particular en la que se desea optimizar el número de saltos en vez del retardo.

Para cada caso, un número de solicitudes multicast fue creado aleatoriamente. Cada solicitud consiste en un nodo origen s, un conjunto de nodos destinos N y una demanda de tráfico ϕ. La duración de cada solicitud de tráfico tuvo una distribución exponencial con promedio de 60 segundos. El instante de llegada de cada solicitud fue aleatoriamente distribuida entre 0 y 1800 segundos.

Dado que MMA1 y MMA2 proveen más de una solución óptima para cada grupo, el árbol de menor α_T fue elegido. En caso de existir más de un árbol con α_T mínimo, aquel árbol con menor C_T fue seleccionado. Es decir, el de menor consumo de ancho de banda.

Los algoritmos propuestos en este trabajo fueron comparados entre si, y con el algoritmo propuesto por Seok et al. [SEO02], denominado algoritmo SK. Este algoritmo minimiza α_T y C_T, intentando además minimizar el número de saltos a través de una restricción de número de saltos máximo dada a priori. SK es un algoritmo que consiste de dos etapas: 1- modificación del grafo original *G* a uno *G'* en el cual el número de saltos desde el nodo origen a cualquier otro nodo de la red está acotado por un valor H dado a priori. Utilizando H, cada ruta desde el nodo origen tiene como máximo H_{min} + H saltos, donde H_{min} es el mínimo número de saltos desde el nodo origen al nodo destino más alejado en término de saltos. Es decir, todos los nodos destinos son alcanzables desde el nodo fuente en H_{min} saltos. H es el número adicional de saltos que es sumado a H_{min}; 2- encontrar el árbol multicast en *G'* que minimice la utilización máxima de los enlaces del árbol, utilizando para ello el algoritmo del camino de menor utilización máxima de los enlaces. En esta etapa, también es aplicada una heurística para minimizar el consumo de ancho de banda del árbol multicast. Para más detalles del algoritmo, se deriva al lector a [SEO02].

Dado que SK tiene asociado el parámetro H definido a priori, en cada corrida se ha variado dicho parámetro, y las distintas soluciones proveídas de esta manera se han comparado con MMA1 y MMA2.

De forma a evaluar cuantitativamente los resultados obtenidos en estas simulaciones dinámicas, se han considerado las métricas de dominancia definidas a continuación:

ND_{A1-A2}:	Número de árboles construidos por el algoritmo 1 (A_1) que dominan a los correspondientes árboles construidos por el algoritmo 2 (A_2).
ND_{A2-A1}:	Número de árboles construidos por el algoritmo 2 (A_2) que dominan a los correspondientes árboles construidos por el algoritmo 1 (A_1).
I_{A1-A2}:	Número de árboles construidos por el algoritmo 1 (A_1) indiferentes (no pueden considerarse mejor ni peor) que los correspondientes árboles construidos por el algoritmo 2 (A_2).

Además, con el objeto de mostrar cualitativamente los resultados, se han graficado los valores normalizados a MMA2 de la utilización máxima de los enlaces de la red, el consumo total de ancho de banda y el retardo total, de las simulaciones hechas sobre la red NTT. Los valores normalizados se definen a continuación:

Utilización máxima de los enlaces de la red normalizada a MMA2:

$$\alpha_N^W = \frac{\alpha_W - \alpha_{MMA2}}{\alpha_{MMA2}} \quad \textbf{(5.1)}$$

Consumo total de ancho de banda normalizado a MMA2:

$$B_N^W = \frac{B_W - B_{MMA2}}{B_{MMA2}} \quad \textbf{(5.2)}$$

Retardo total normalizado a MMA2:

$$D_N^W = \frac{D_W - D_{MMA2}}{D_{MMA2}} \quad \textbf{(5.3)}$$

Donde

w:	algoritmo a comparar con MMA2.
α_w:	utilización máxima de los enlaces de la red utilizando w.
α_{MMA2}:	utilización máxima de los enlaces de la red utilizando MMA2.
B_w:	consumo total de ancho de banda de la red utilizando w. Esto es, la carga total sobre la red.
B_{MMA2}:	consume total de ancho de banda utilizando MMA2.
D_w:	retardo total usando w, dado por la suma del retardo total de los grupos multicast en la red. El retardo total de un grupo está dado por la suma de los retardos de extremo a extremo a cada uno de los nodos destinos.
D_{MMA2}:	retardo total usando MMA2.

A continuación se resumen los resultados de las simulaciones sobre cada una de las diferentes redes, enfatizando con gráficas las simulaciones hechas sobre la NTT, por ser una red de prueba bien conocida y representativa [BAR00, CRI04a].

5.3.1 Simulaciones usando la red NTT

La Figura 5.4 muestra la red de la NTT [BAR00]. Sobre cada enlace se muestra la métrica de retardo d_{ij}, $\forall(i, j)\in E$. La capacidad de los enlaces es de 6 Mbps.

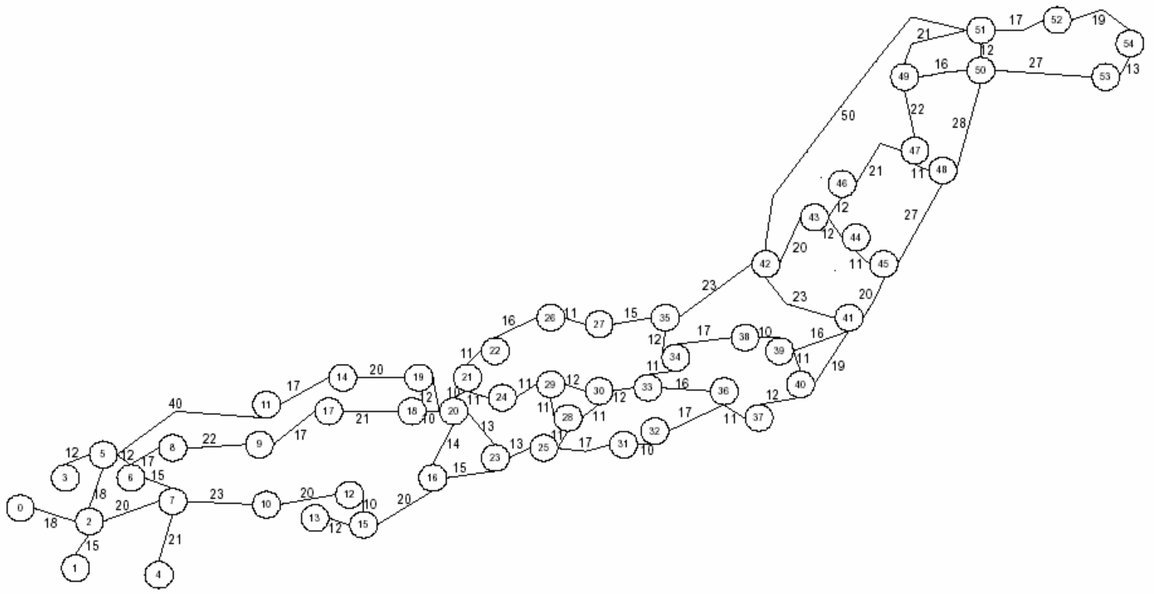


Figura 5.4. Red de la NTT con el retardo sobre cada enlace.

Para cada una de las cuatro corridas, 400 solicitudes multicast fueron generadas, cada una con una demanda $\phi = 600$ Kbps. El parámetro R de MMA1 fue 30. La Tabla 5.4 da el tamaño de la población y el número de generaciones de MMA1 y MMA2 utilizados en las simulaciones, y el rango del tamaño de los grupos multicast. Para las corridas en modo esparzo, el tamaño de los grupos varía entre 3 y 20, mientras que en modo denso entre 20 y 40.

Tabla 5.4. Parámetros de MMA1 y MMA2, y rango del tamaño de los grupos multicast

		$\lvert P \rvert$	Generaciones	Tamaño grupo
Corrida 1	MMA1	50	500	[3, 20]
	MMA2	40	60	
Corrida 2	MMA1	60	500	[20, 40]
	MMA2	50	80	
Corrida 3	MMA1	50	500	[3, 20]
	MMA2	40	75	
Corrida 4	MMA1	50	500	[20, 40]
	MMA2	40	100	

5.3.1.1 Corrida 1

En esta corrida, SK (H = 0) solo pudo enrutar 397 solicitudes, mientras que MMA1, MMA2, SK (H = 3) y SK (H = 6) enrutaron todas las solicitudes. La Tabla 5.5 muestra las relaciones de dominancia entre los algoritmos comparados.

Tabla 5.5. Resultados de la corrida 1

A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	51	89	260
MMA1 - SK (H=0)	141	8	251
MMA1 - SK (H=3)	239	5	156
MMA1 - SK (H=6)	303	6	91
MMA2 - SK (H=0)	101	0	299
MMA2 - SK (H=3)	216	2	182
MMA2 - SK (H=6)	315	1	84

De la Tabla 5.5 se puede concluir que el algoritmo de mejor rendimiento, considerando la métrica de dominancia, fue MMA2, dado que construyó mejores árboles que MMA1 y SK. De las 400 solicitudes multicast, 89 árboles de MMA2 dominaron a los correspondientes árboles de MMA1, mientras que MMA1 construyó 51 árboles mejores que los de MMA2. Además, en el caso más favorable al algoritmo SK, cuando H = 0, 101 árboles construidos por MMA2 dominaron a los respectivos árboles de SK (H = 0). De forma similar, MMA1 construyó 141 árboles mejores que SK (H = 0). Note que si se considera SK (H = 6), la diferencia a favor de MMA1 y MMA2 fue aún mayor. Ambos algoritmos dominaron a SK (H = 6) en más de 300 casos. De esta forma, más del 75 % de los árboles multicast proveídos por MMA1 y MMA2 tuvieron al menos una función objetivo menor, y las demás funciones objetivo menor o igual que las funciones objetivo de los árboles proveído por SK (H = 6).

Los tiempos medio de cómputo de un árbol multicast de SK fueron: 15 ms (H = 0), 25 ms (H = 3) y 60 ms (H = 6). Por su parte, los tiempos de MMA1 y MMA2 fueron 325 ms y 465 ms, respectivamente. Es necesario aclarar que, si bien los tiempos de MMA1 y MMA2 son mayores, estos algoritmos proveen un conjunto amplio de soluciones, del cual puede ser elegida la solución más adecuada para cada ocasión. En estas pruebas dinámicas simplemente se toma una de las soluciones óptimas, de forma a poder comparar ambos algoritmos con uno mono-objetivo como SK.

Las Figuras 5.5 a 5.8 muestran los valores normalizados de la utilización máxima de los enlaces de la red, dada por la Ecuación 5.1. Note que la utilización máxima de los enlaces de SK (H = 0) fue en casi todo momento superior a la utilización máxima de los enlaces de MMA2. En gran parte del tiempo, $\alpha_N^{SK(H=0)}$ toma valores entre 0.2 y 0.6, indicando que SK (H = 0) produjo una utilización máxima al menos 20% mayor que la utilización máxima de MMA2. Con H = 3 y H = 6, SK mejoró esta métrica respecto a SK (H=0). Aún así, $\alpha_N^{SK(H=3)}$ y $\alpha_N^{SK(H=6)}$ son positivos en gran parte del tiempo. $\alpha_N^{SK(H=3)}$ toma valores entre -0,3 y 0.5, mientras que $\alpha_N^{SK(H=6)}$ lo hace entre -0,35 y 0.5. Por su parte, MMA1 produjo una utilización máxima de los enlaces normalizada con valores entre -0,5 y 0,35, mostrando que minimizó esta métrica en forma similar a MMA2.

La Figura 5.9 muestra el consumo total de ancho de banda normalizado a MMA2, dado por la Ecuación 5.2. Note que todas las curvas tienen valores positivos, indicando que MMA2 consumió una menor cantidad de ancho de banda que SK y MMA1. Al aumentar H, SK aumentó el consumo de ancho de banda. Con H = 0 y H = 3, SK utilizó aproximadamente entre 20 y 45% más ancho de banda que MMA2, mientras que con H = 6 la utilización de recursos fue aún mayor. Note que, en t ≈ 1600 segundos, la cantidad de ancho de banda utilizada con H = 3 y H = 6 alcanzó el pico máximo y $B_N^{SK(H=3)} \approx B_N^{SK(H=6)}$ ≈ 0.7. Por su parte, la curva de consumo de MMA1 también es mayor que 0 en casi todo momento. A pesar de ello, claramente MMA1 tuvo un consumo menor que SK.

La Figura 5.10 muestra el retardo total normalizado a MMA2, dado por la Ecuación 5.3. D_N^{MMA1} toma valores entre -0.1 y 0.1, indicando que, por momentos, el retardo total de MMA2 fue menor al retardo de MMA1 y viceversa. Por otra parte, SK produjo retardos notoriamente superiores a los de MMA2 y MMA1. La mayor parte del tiempo $D_N^{SK(K=0)}$ toma valores positivos entre 0 y 0.15. Al aumentar H, SK incrementó el retardo total, y la diferencia entre los algoritmos propuestos en este trabajo y SK fue aún mayor. Por ejemplo, cuando H = 6, el retardo de SK alcanzó picos muy elevados, aproximándose el retardo total a aun valor 60 % mayor que el retardo de MMA2 en t ≈ 300, t ≈ 750 y t ≈ 1100.

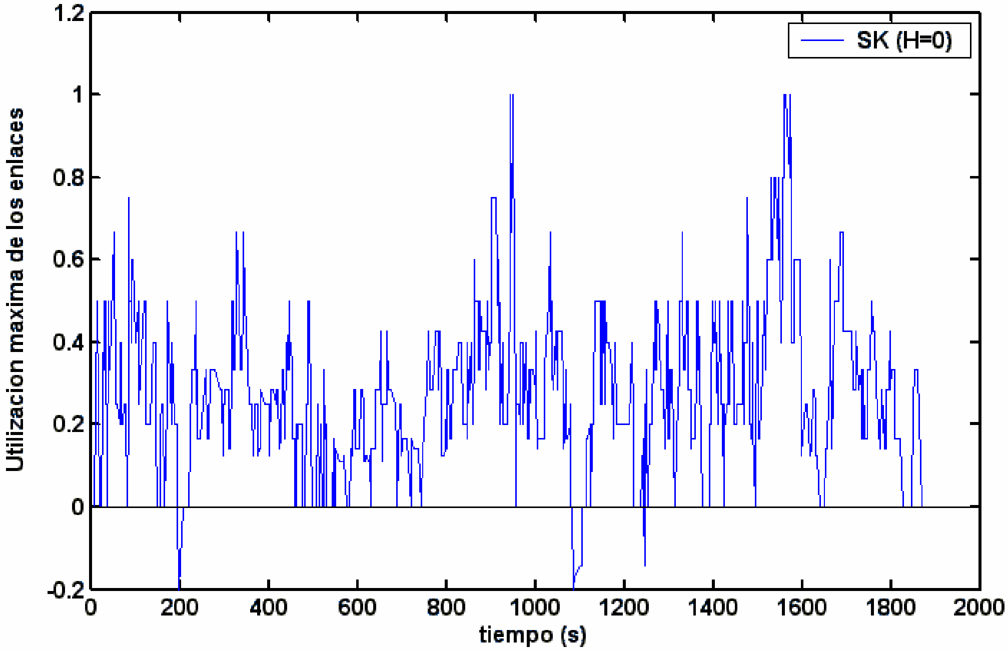


Figura 5.5. $\alpha_N^{SK(H=0)}$, corrida 1.

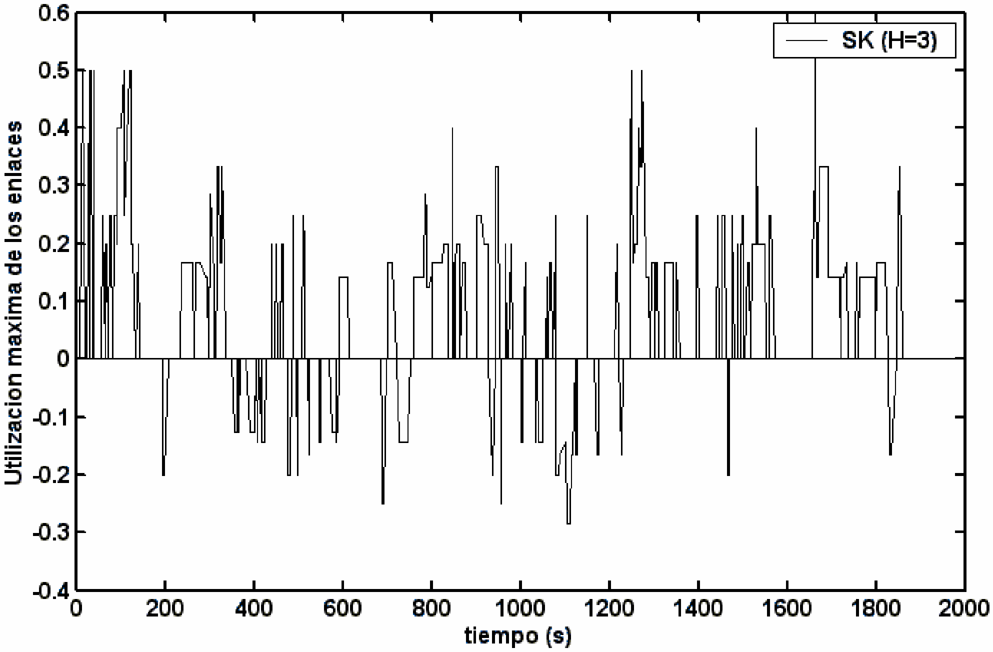


Figura 5.6. $\alpha_N^{SK(H=3)}$, corrida 1.

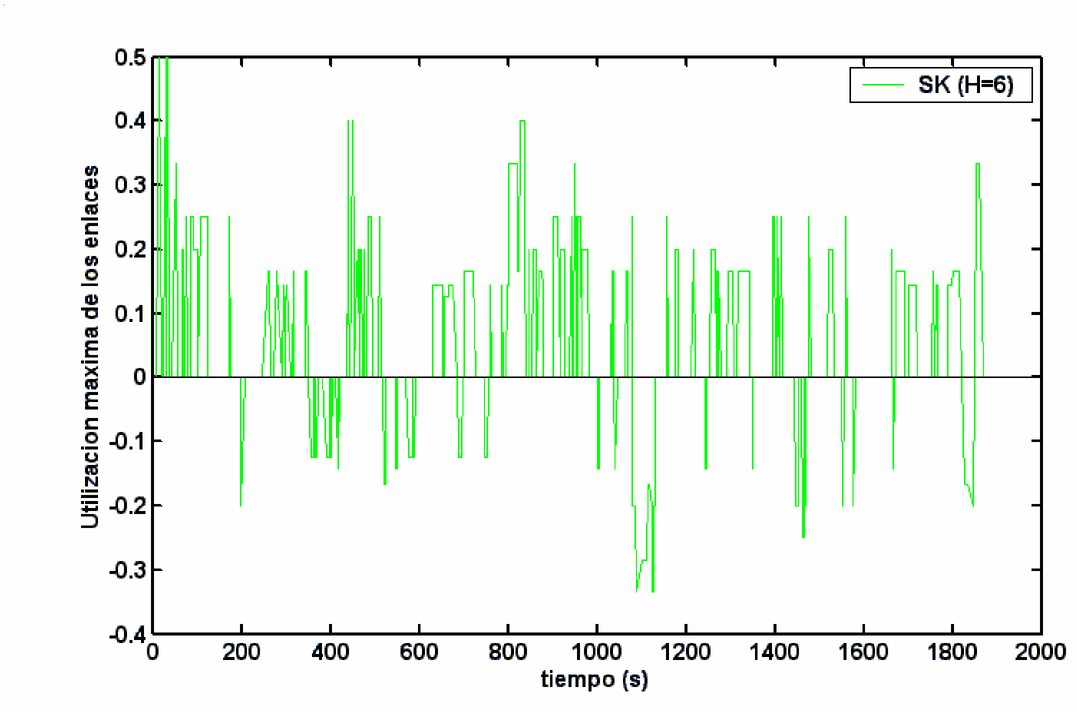


Figura 5.7. $\alpha_N^{SK(H=6)}$, corrida 1.

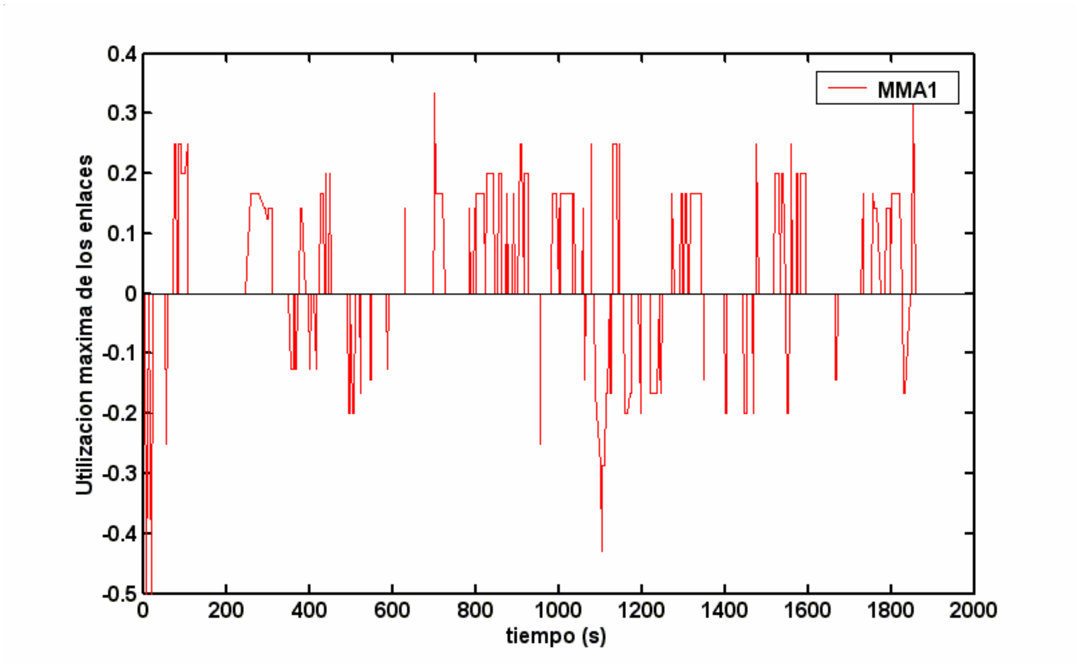


Figura 5.8. α_N^{MMA1}, corrida 1.

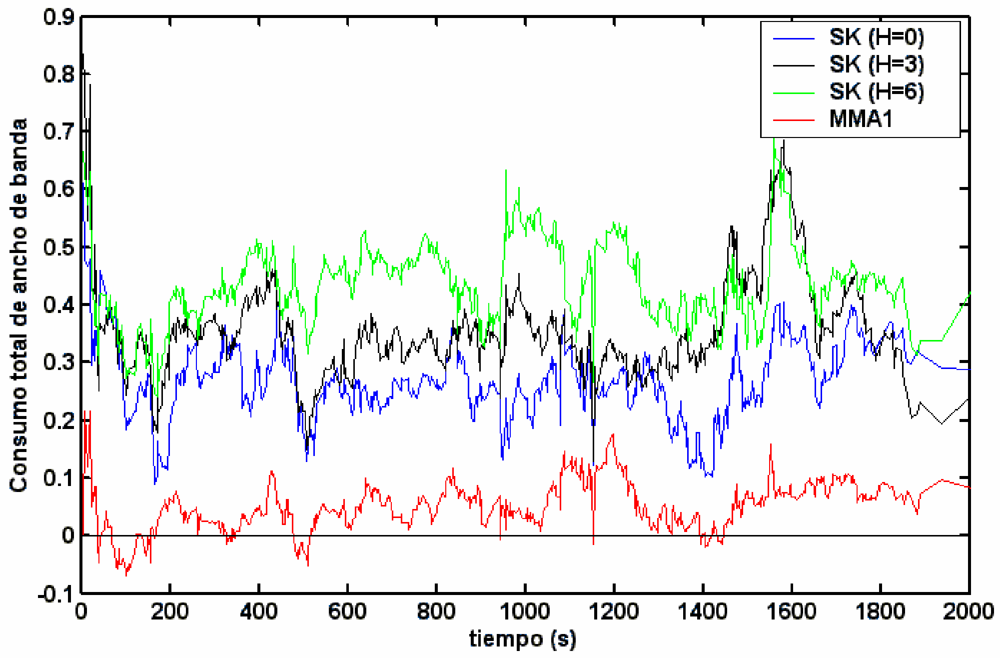


Figura 5.9. Consumo total de ancho de banda normalizado a MMA2, corrida 1.

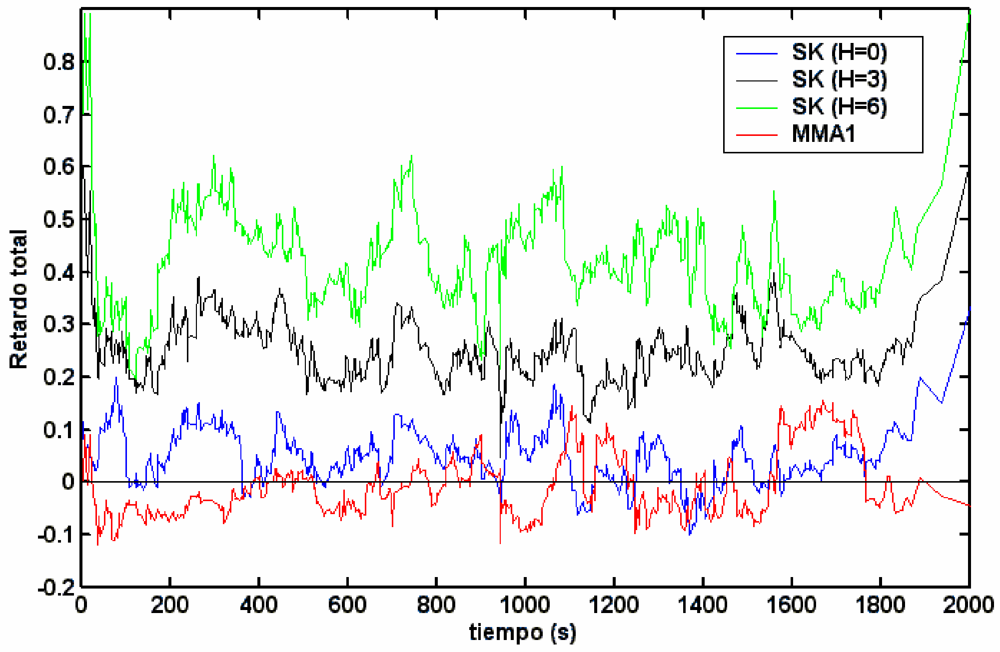


Figura 5.10. Retardo total normalizado a MMA2, corrida 1.

5.3.1.2 Corrida 2

En esta corrida, algunas solicitudes fueron rechazadas por falta de capacidad de la red. La Tabla 5.6 muestra el número de solicitudes aceptadas por cada algoritmo. Considerando el número de solicitudes aceptadas, MMA2 obtuvo el mejor resultado. Sin embargo, la diferencia entre todos los algoritmos fue mínima.

La Tabla 5.7 muestra las relaciones de dominancia entre los algoritmos comparados. Note que MMA1 construyó 26 árboles que dominaron a MMA2, mientras que 40 árboles construidos por MMA2 dominaron a los correspondientes árboles de MMA1. Comparado con SK, MMA1 generó 180 (H = 0), 231 (H = 3) y 254 (H = 6) árboles mejores que los árboles construidos por este algoritmo. Es decir, para aproximadamente el 50% de las solicitudes multicast, los árboles de MMA1 dominaron a SK. La relación de dominancia entre MMA2 y SK nuevamente favorece al algoritmo propuesto en este trabajo: 88 (H = 0), 205 (H = 3) y 251 (H = 6) árboles construidos por MMA2 fueron mejores que los árboles construidos por SK. En contrapartida, 9 (H = 0), 5 (H = 3) y 5 (H = 6) árboles de SK dominaron a los respectivos árboles de MMA2.

Tabla 5.6. S. aceptadas

	S. acept.	%
MMA1	336	84
MMA2	340	85
SK (H=0)	336	84
SK (H=3)	339	84,75
SK (H=6)	339	84,75

Tabla 5.7. Resultados de la corrida 2

A1 – A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	26	40	334
MMA1 - SK (H=0)	180	11	209
MMA1 - SK (H=3)	231	8	161
MMA1 - SK (H=6)	254	8	138
MMA2 - SK (H=0)	88	9	303
MMA2 - SK (H=3)	205	5	190
MMA2 - SK (H=6)	251	5	144

Los tiempos medio de cómputo de un árbol multicast de SK fueron: 55 ms (H = 0), 105 ms (H = 3) y 250 ms (H = 6). Por su parte, los tiempos de MMA1 y MMA2 fueron: 815 ms y 570 ms.

Las Figuras 5.11 a 5.14 muestran los valores normalizados de la utilización máxima de los enlaces. Considerando esta métrica, los tres algoritmos demostraron un rendimiento similar, y de las gráficas no se puede concluir que un algoritmo produce una utilización máxima de los enlaces menor que otro, pues en algunos intervalos de tiempo, un algoritmo gana al otro y viceversa.

La Figura 5.15 muestra el consumo total de ancho de banda normalizado a MMA2. Nuevamente en esta corrida, MMA2 obtuvo un consumo de ancho de banda menor. De forma similar, MMA1 también consumió menos ancho de banda que SK. Al aumentar H, SK aumentó el consumo y la diferencia entre los algoritmos propuestos en este trabajo y SK fue aún mayor.

La Figura 5.16 muestra el retardo total normalizado a MMA2. Como en la corrida 1, MMA1 produjo un retardo total menor que los demás algoritmos. También se puede observar que, en casi todo momento, el retardo total de MMA2 fue menor que el retardo de SK con sus diferentes parámetros de H. Por ejemplo, con H = 6, el retardo de SK fue en casi todo momento, como mínimo, 40 % mayor que el retardo de MMA2.

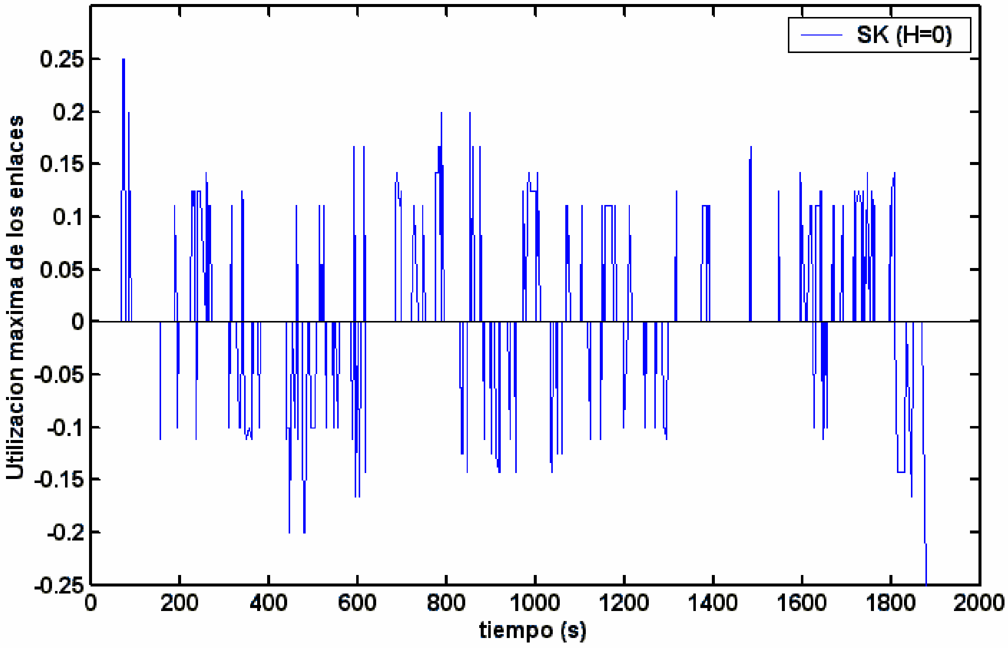


Figura 5.11. $\alpha_N^{SK(H=0)}$, corrida 2.

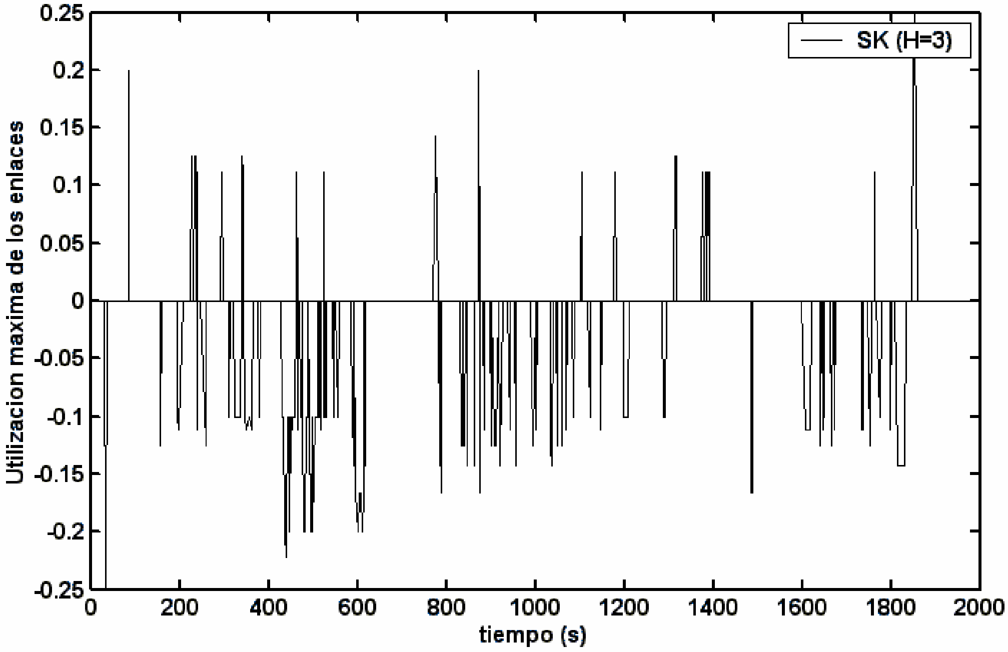


Figura 5.12. $\alpha_N^{SK(H=3)}$, corrida 2.

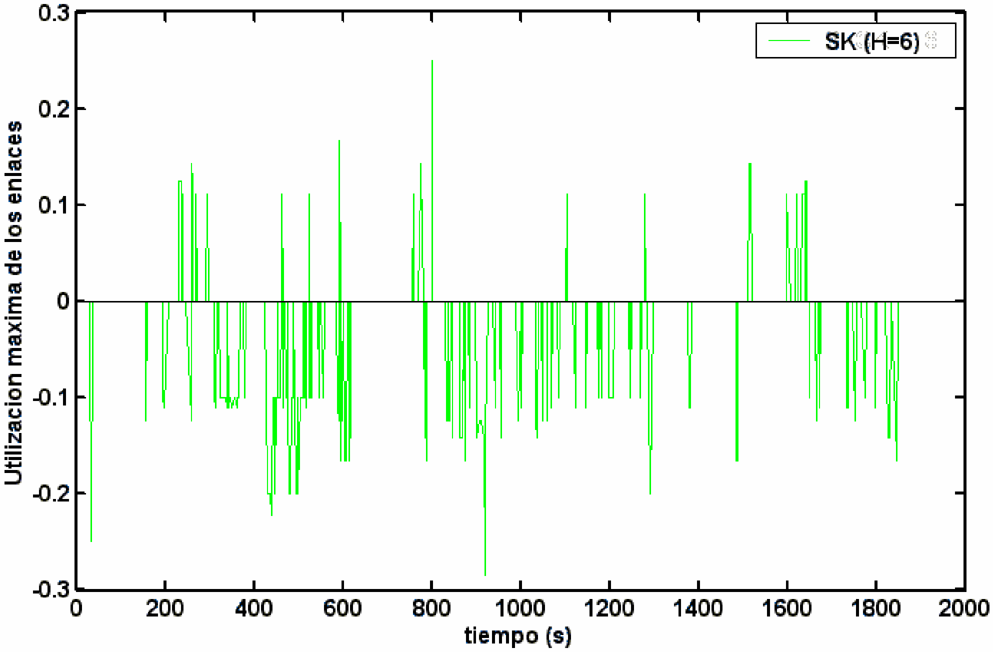


Figura 5.13. $\alpha_N^{SK(H=6)}$, corrida 2.

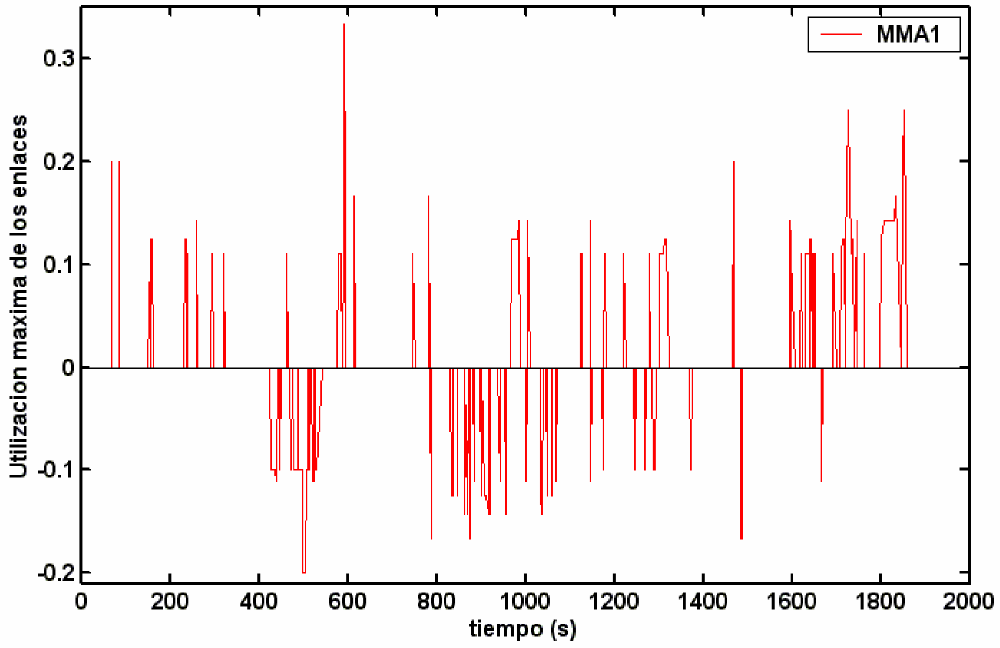


Figura 5.14. α_N^{MMA1}, corrida 2.

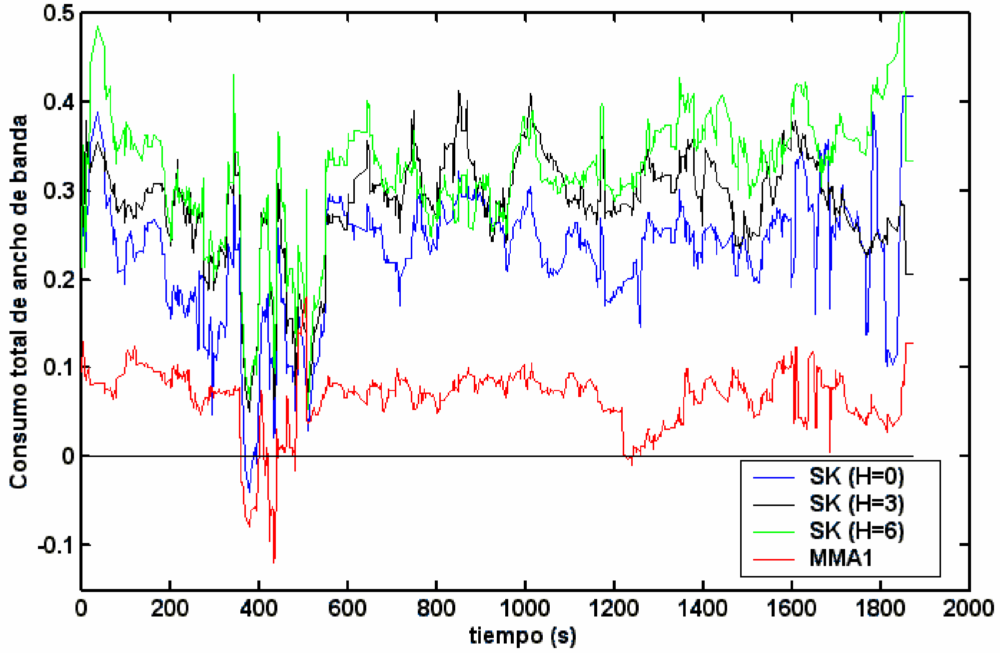


Figura 5.15. Consumo total de ancho de banda normalizado a MMA2, corrida 2.

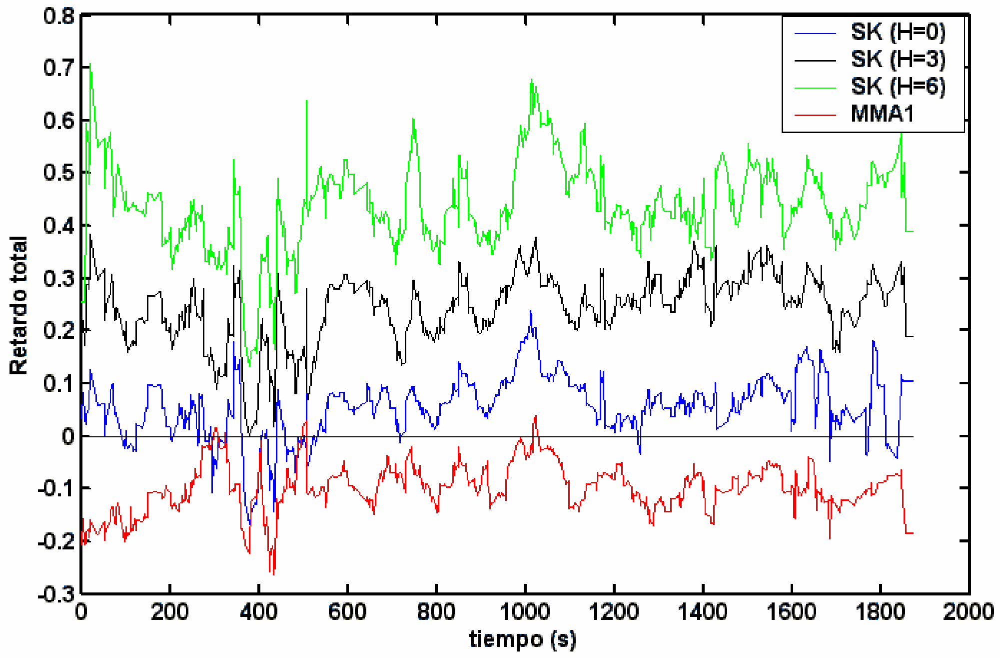


Figura 5.16. Retardo total normalizado a MMA2, corrida 2.

Las primeras dos simulaciones mostraron que los algoritmos propuestos en este trabajo han obtenido resultados satisfactorios en ambos modos: denso y esparzo. MMA2 construyó un mayor número de árboles multicast que dominaron a las correspondientes soluciones de MMA1 y SK. Si bien el número de solicitudes multicast enrutadas por los tres algoritmos fue similar, MMA2 enrutó la mayor cantidad de solicitudes en modo denso.

5.3.1.3 Corrida 3

De forma similar a la corrida 1, todas las solicitudes multicast fueron enrutadas. La Tabla 5.8 muestra las relaciones de dominancia entre los algoritmos comparados. Note que MMA2 tuvo mejor rendimiento que MMA1 y SK. MMA2 obtuvo 98 árboles que dominaron a los correspondientes árboles de MMA1, mientras que MMA1 construyó 41 árboles que dominaron a las soluciones de MMA2. A pesar de ello, el número de indiferencias fue mayor al 65 % (261 casos), lo cual implica una alta relación de compromiso entre las soluciones proveídas por ambos algoritmos. Como en la corrida 1, MMA1 y MMA2 obtuvieron mejor rendimiento que SK. 107, 234 y 286 árboles construidos por MMA1 dominaron a los correspondientes árboles generados por SK con H = 0, H = 3 y H = 6, mientras que 68, 228 y 311 soluciones de MMA2 fueron mejores que las soluciones correspondientes de SK. Es decir, en el mejor caso de SK (cuando H = 0), MMA1 produjo mejores árboles en más del 25 % de los casos (107 soluciones), mientras que MMA2 lo hizo en casi el 20 % de las solicitudes multicast (68 casos).

Tabla 5.8. Resultados de la corrida 3

A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	41	98	261
MMA1 - SK (H=0)	107	4	289
MMA1 - SK (H=3)	234	7	159
MMA1 - SK (H=6)	286	3	111
MMA2 - SK (H=0)	68	4	328
MMA2 - SK (H=3)	228	10	162
MMA2 - SK (H=6)	311	0	89

Los tiempos medio de construcción de un árbol multicast de SK fueron: 25 ms (H = 0), 29 ms (H = 3) y 57 ms (H = 6); los tiempos de MMA1 y MMA2 fueron: 325 ms y 579 ms.

La Figura 5.17 muestra la utilización máxima de los enlaces de SK (H = 0). Note que $\alpha_N^{SK(H=0)}$ es positivo en casi todo momento, indicando que MMA2 optimizó mejor dicha métrica (por extensión, de la Figura 5.20 se deriva que MMA1 también optimizó mejor esta métrica). Además, en las Figuras 5.21 y 5.22 se puede notar que también el consumo total de ancho de banda y el número total de saltos (dado que $d_{ij} = 1, \forall(i,j)\in E$) fueron mejor optimizados por MMA2 y MMA1.

Las Figuras 5.18 a 5.20 muestran que MMA1, MMA2, SK (H = 0) y SK (H = 3) produjeron una utilización máxima de los enlaces similar. La Figura 5.21 muestra MMA2 consumió una menor cantidad de ancho de banda para enrutar las solicitudes multicast. MMA1 también obtuvo un menor consumo de ancho de banda que SK. Al aumentar H, SK consumió más ancho de banda aún. Note que $B_N^{SK(H=6)}$ toma valores entre 0.3 y 0.6, indicando que SK (H = 6) consumió al menos 30 % más ancho de banda que MMA2.

La Figura 5.22 muestra el retardo total normalizado a MMA2. Como se mencionó anteriormente, $d_{ij} = 1$, $\forall(i, j)\in E$, y por lo tanto el retardo total es igual al número total de saltos. Nuevamente los retardos de MMA1 y MMA2 fueron menores que el retardo total de SK. En el caso mas favorable a SK, cuando H = 0, $D_N^{SK(H=0)}$ toma valores entre -0,1 y 0.2. Al aumentar H a 3, el número de saltos de MMA1 y MMA2 ya fue menor que el de SK en todo momento. Cuando H = 6, la diferencia fue aún mayor. En dicho caso, el número de saltos de SK fue al menos 30 % mayor que el número de saltos de MMA2, alcanzado incluso un número de saltos de casi 1.8 veces el número de saltos de MMA2 (en t ≈ 1400). El número de saltos de MMA1 tomó valores entre -0.1 y 0.2, siendo en algunos momentos mayor que el número de saltos de MMA2, y en otros momentos menor.

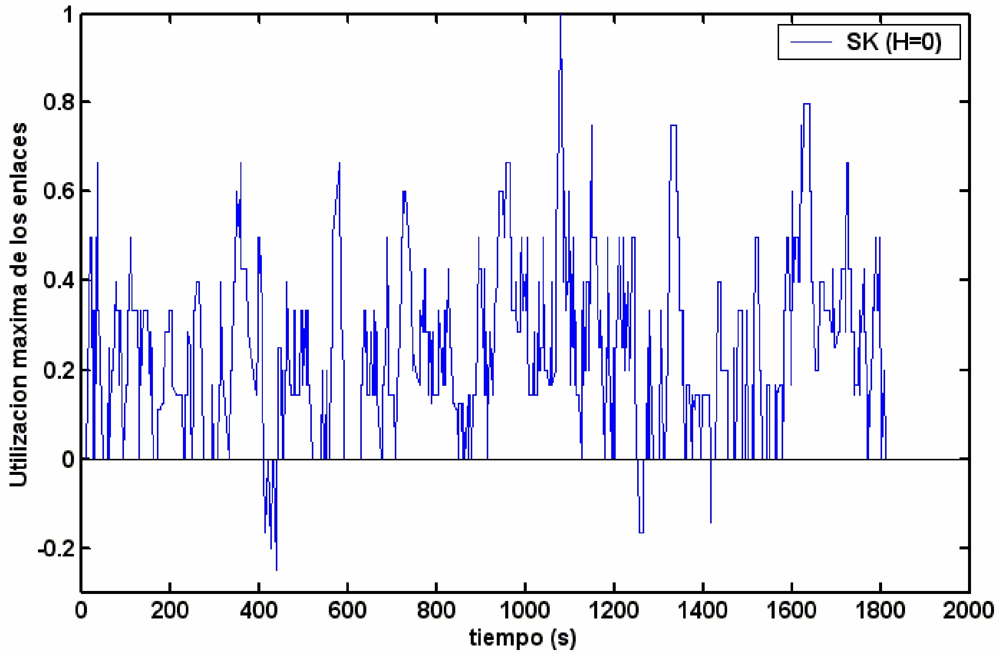


Figura 5.17. $\alpha_N^{SK(H=0)}$, corrida 3.

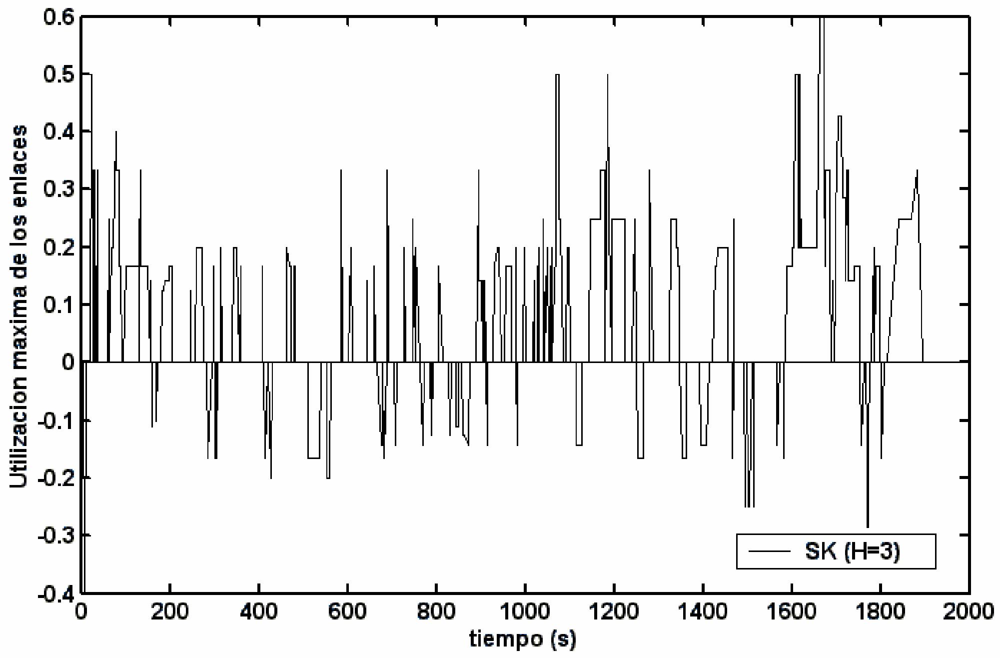


Figura 5.18. $\alpha_N^{SK(H=3)}$, corrida 3.

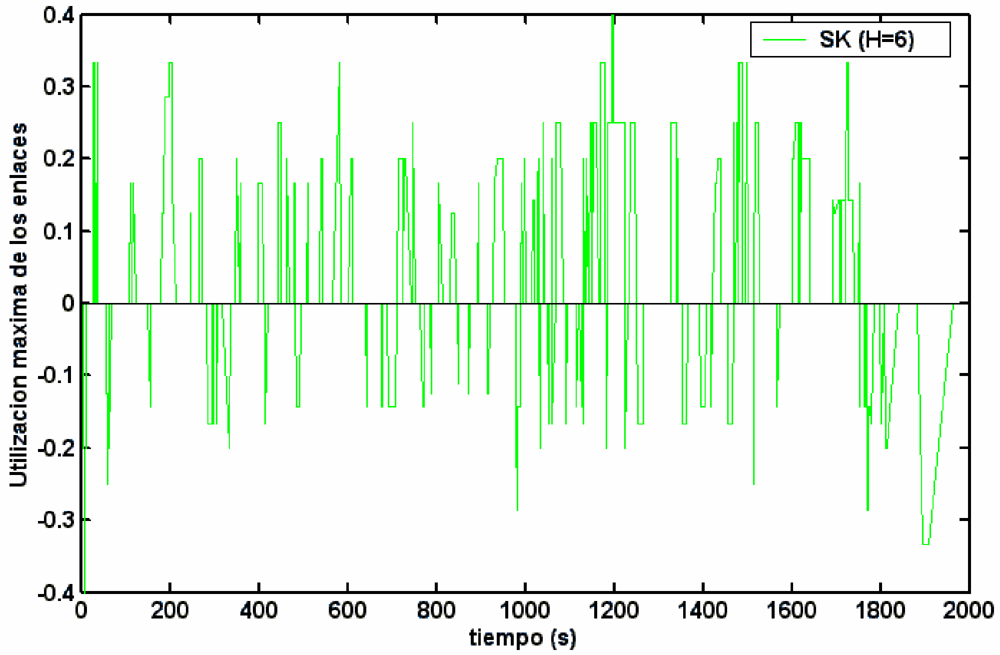


Figura 5.19. $\alpha_N^{SK(H=6)}$, corrida 3.

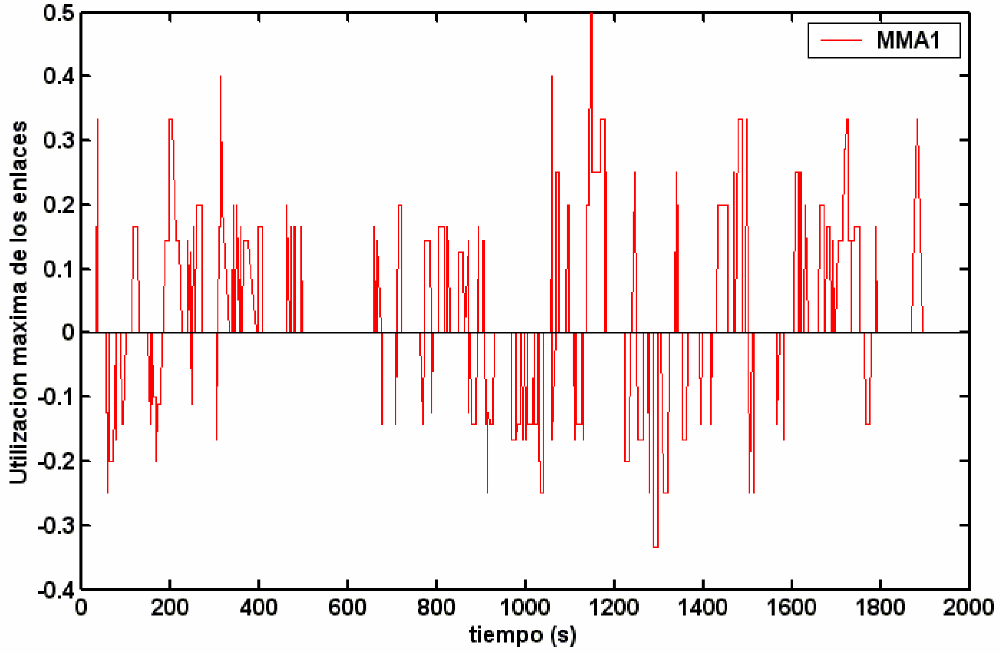


Figura 5.20. α_N^{MMA1}, corrida 3.

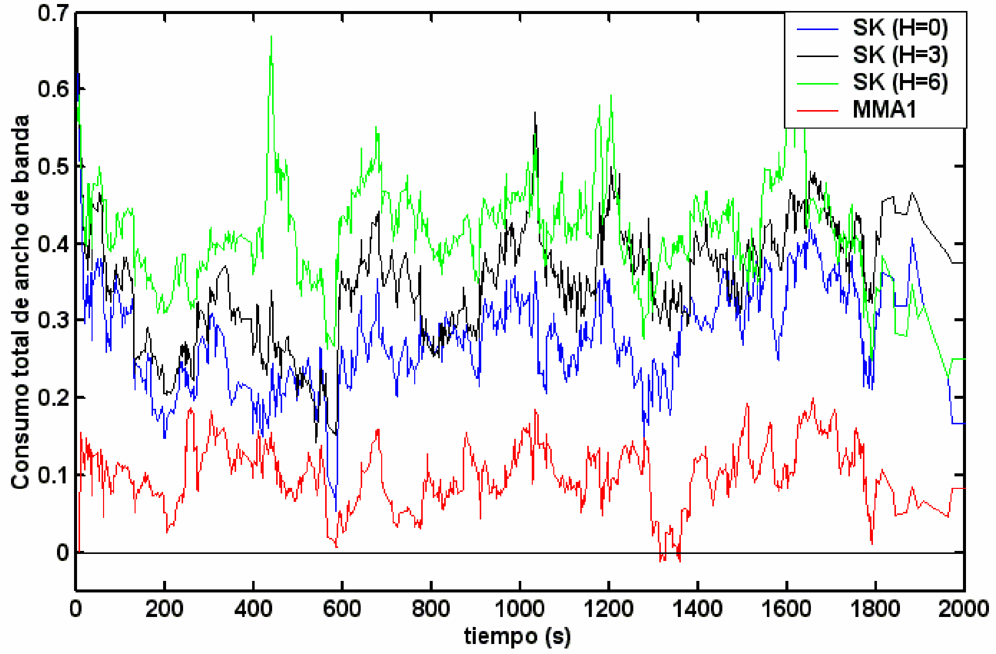


Figura 5.21. Consumo total de ancho de banda normalizado a MMA2, corrida 3.

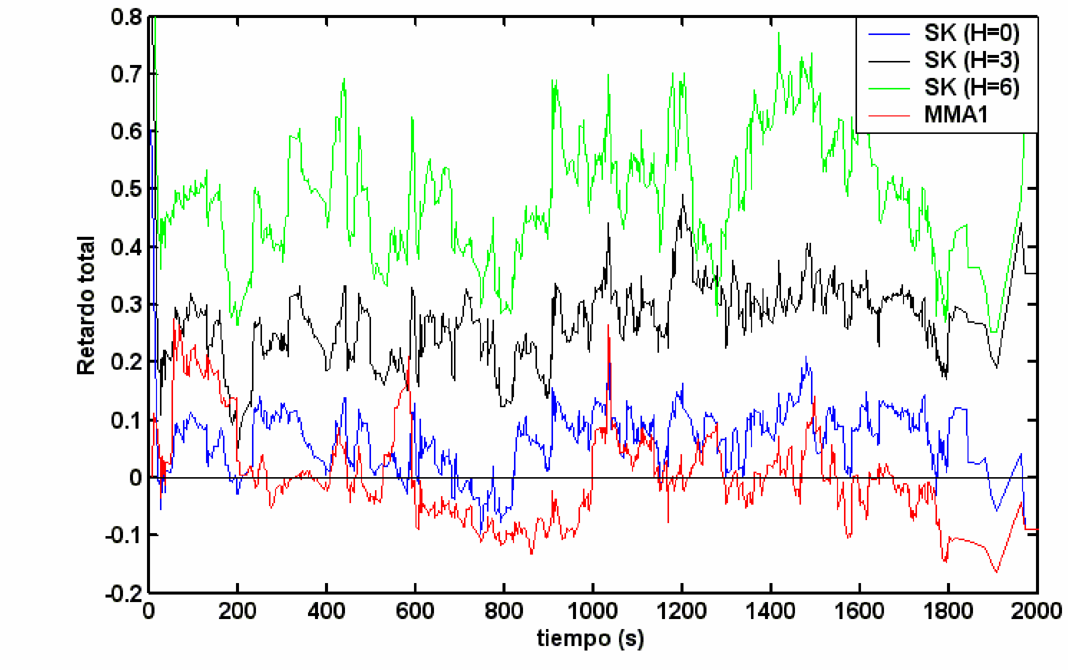


Figura 5.22. Retardo total normalizado a MMA2, corrida 3.

Nuevamente en esta corrida, MMA1 y MMA2 tuvieron mejor rendimiento que SK, considerando la métrica dominancia. Entre ambos, MMA2 obtuvo un número mayor de soluciones que dominaron a MMA1. Sin embargo, como se apuntó al inicio de esta sección, el gran número de indiferencias entre los dos algoritmos propuestos implica una relación de compromiso entre las soluciones proveídas por ellos.

5.3.1.4 Corrida 4

La Tabla 5.9 muestra el número de solicitudes aceptadas por cada algoritmo. Note que, en esta corrida, MMA1 enrutó el mayor número de solicitudes multicast, en contraposición a la corrida 2. A pesar de ello, la diferencia entre el número de solicitudes enrutadas por cada algoritmo fue mínima.

La Tabla 5.10 muestra las relaciones de dominancia entre los algoritmos comparados. Nuevamente MMA2 tuvo mejor rendimiento que MMA1 y SK. MMA2 obtuvo 73 árboles que dominaron a las correspondientes soluciones de MMA1, mientras que MMA1 construyó 22 árboles que dominaron a las soluciones de MMA2. Sin embargo, el número de indiferencias fue alto, mayor al 75% de las solicitudes. En esta corrida, MMA1 y MMA2 nuevamente obtuvieron mejores soluciones que SK. MMA1 generó mejores árboles que SK en 145 (H = 0), 205 (H = 3) y 272 (H = 6) oportunidades. Es decir, para valores de H mayores a 0, MMA1 dominó a SK en más del 50 % de los casos. De forma similar, MMA2 dominó a SK en 53 (H = 0), 165 (H = 3) y 264 (H = 6) oportunidades.

Tabla 5.9. Solicitudes aceptadas

	S. aceptadas	%
MMA1	356	89
MMA2	354	88,5
SK (H=0)	347	86,75
SK (H=3)	355	88,75
SK (H=6)	355	88,75

Tabla 5.10. Resultados de la corrida 4

A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	22	73	305
MMA1 - SK (H=0)	145	6	249
MMA1 - SK (H=3)	205	5	190
MMA1 - SK (H=6)	272	4	124
MMA2 - SK (H=0)	53	3	344
MMA2 - SK (H=3)	165	3	232
MMA2 - SK (H=6)	264	5	131

Los tiempos medio de cómputo de un árbol multicast de SK fueron 80 ms (H = 0), 140 ms (H = 3) y 290 ms (H = 6), mientras que los de MMA1 y MMA2 fueron 845 ms y 650 ms.

Las Figuras 5.23 a 5.26 muestran los valores normalizados de la utilización máxima de los enlaces. α_N^{MMA1} toma valores entre -0,2 y 0,2. Note también que en una gran cantidad de intervalos de tiempo α_N^{MMA1} vale cero. Por lo tanto, no puede concluirse que un algoritmo optimizó mejor que otro esta métrica. Por otra parte, $\alpha_N^{SK(H=3)}$ y $\alpha_N^{SK(H=6)}$ mejoran con respecto a $\alpha_N^{SK(H=0)}$, indicando que SK mejoró la utilización de los enlaces al aumentar H. La curva de $\alpha_N^{SK(H=3)}$ muestra que SK (H = 3) produjo una mejor utilización de los enlaces que MMA2. Sin embargo, existen muchos intervalos de tiempo donde $\alpha_N^{SK(H=3)}$ vale cero, indicando que ambos algoritmos, MMA2 y SK (H = 3), tuvieron la misma utilización máxima de los enlaces en esos intervalos.

La Figura 5.27 muestra el consumo total de ancho de banda normalizado. Nuevamente MMA2 tuvo el menor consumo de ancho de de banda. Note que MMA1 consumió aproximadamente 10 % más ancho de banda que MMA2 en casi todo momento, mientras que el consumo de SK estuvo entre 1.1 y 1.5 veces el consumo de

MMA2. Por otra parte, MMA1 también consumió una menor cantidad de ancho de banda que SK, pues B_N^{MMA1} en casi todo momento es menor que $B_N^{SK(H=0)}, B_N^{SK(H=3)}$ y $\alpha_N^{SK(H=6)}$.

La Figura 5.28 muestra el retardo total (número de saltos) normalizado a MMA2. MMA1 y MMA2 obtuvieron un número total de saltos parecido, aunque en una gran cantidad de intervalos de tiempo, D_N^{MMA1} es menor que cero (i.e. en t ≈ 800, el número de saltos de MMA1 es 20 % menor que el número de saltos de MMA2). Ambos algoritmos, MMA1 y MMA2, optimizaron mejor esta métrica que SK. Cuando H = 3, el número de saltos de SK fue, al menos, 10 % mayor que el número de saltos de MMA2 en todo momento, mientras que cuando H aumentó a 6, la diferencia aumentó aún más, dado que $D_N^{SK(H=6)}$ toma valores entre 0.3 y 0.8.

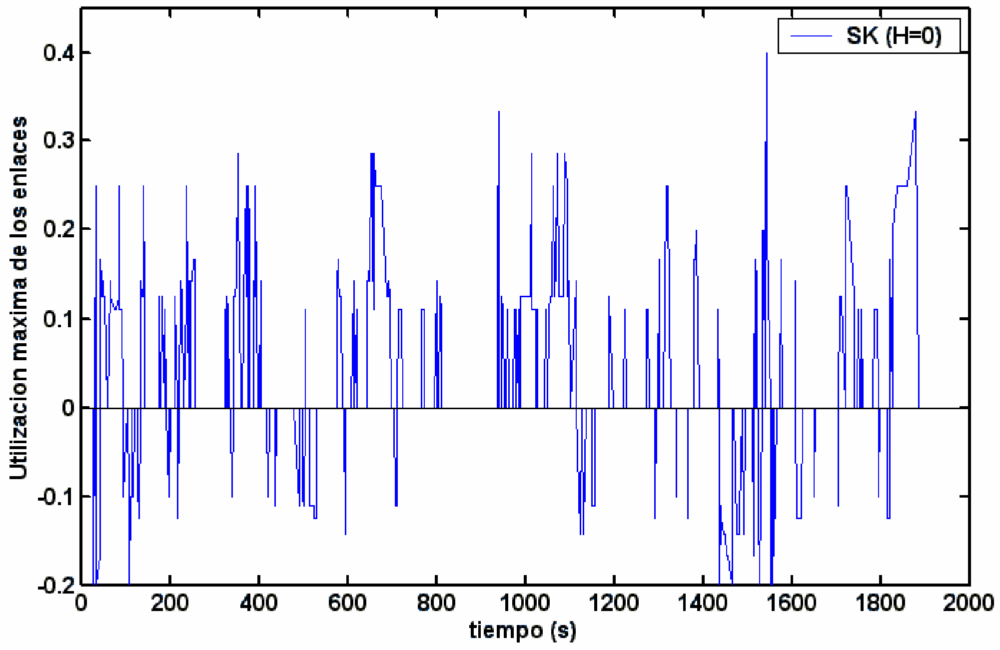


Figura 5.23. $\alpha_N^{SK(H=0)}$, corrida 4.

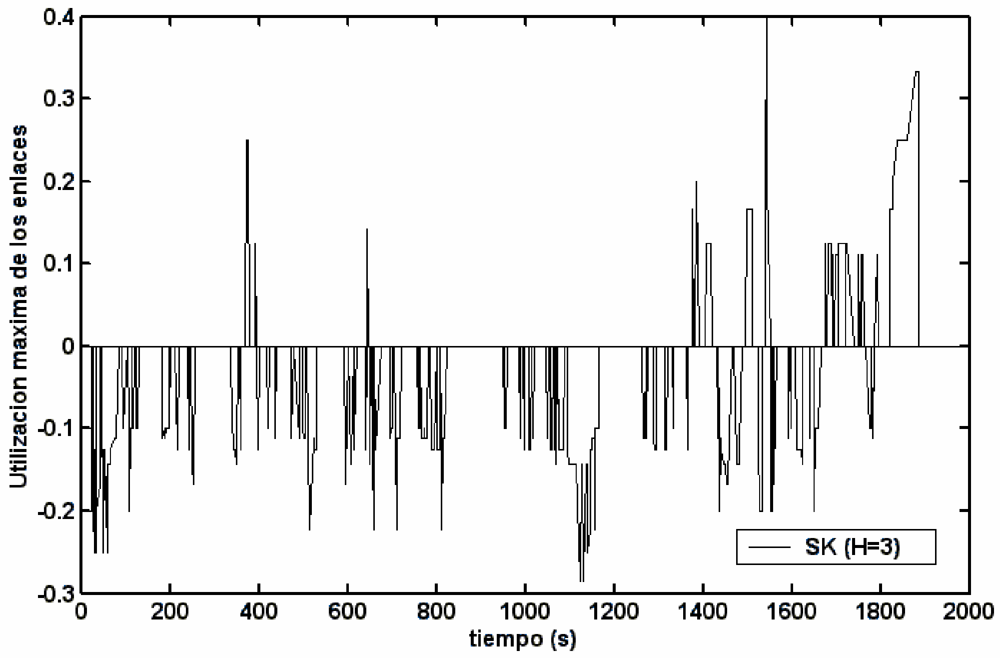


Figura 5.24. $\alpha_N^{SK(H=3)}$, corrida 4.

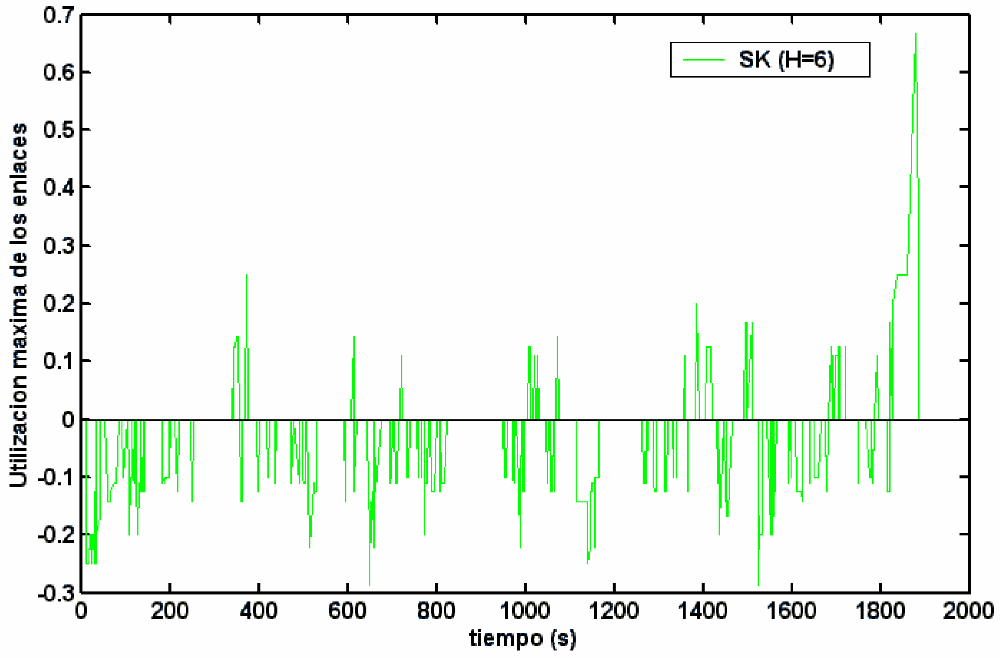


Figura 5.25. $\alpha_N^{SK(H=6)}$, corrida 4.

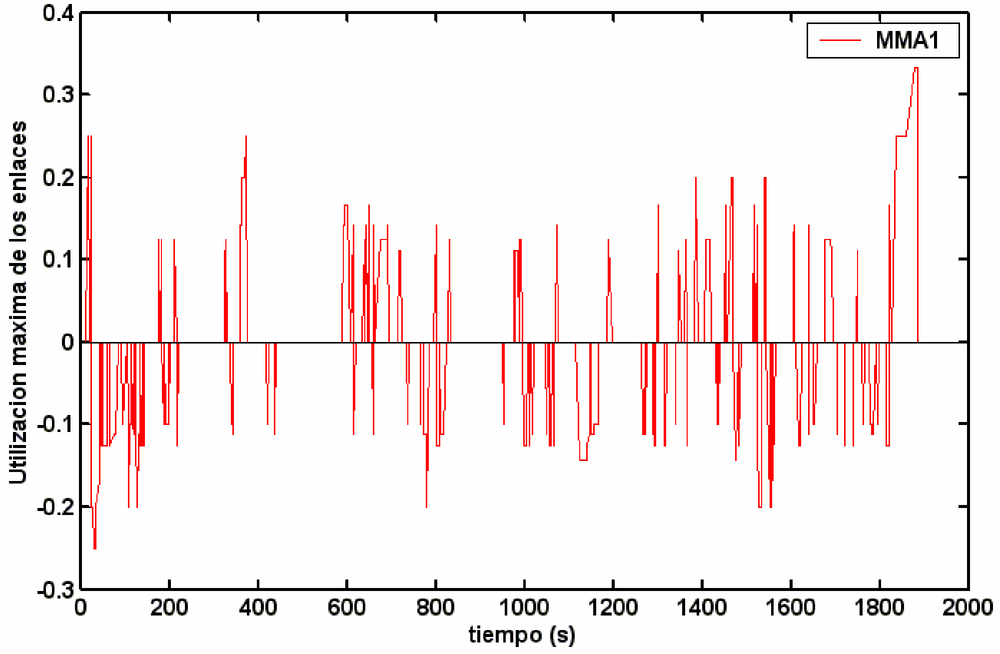


Figura 5.26. α_N^{MMA1}, corrida 4.

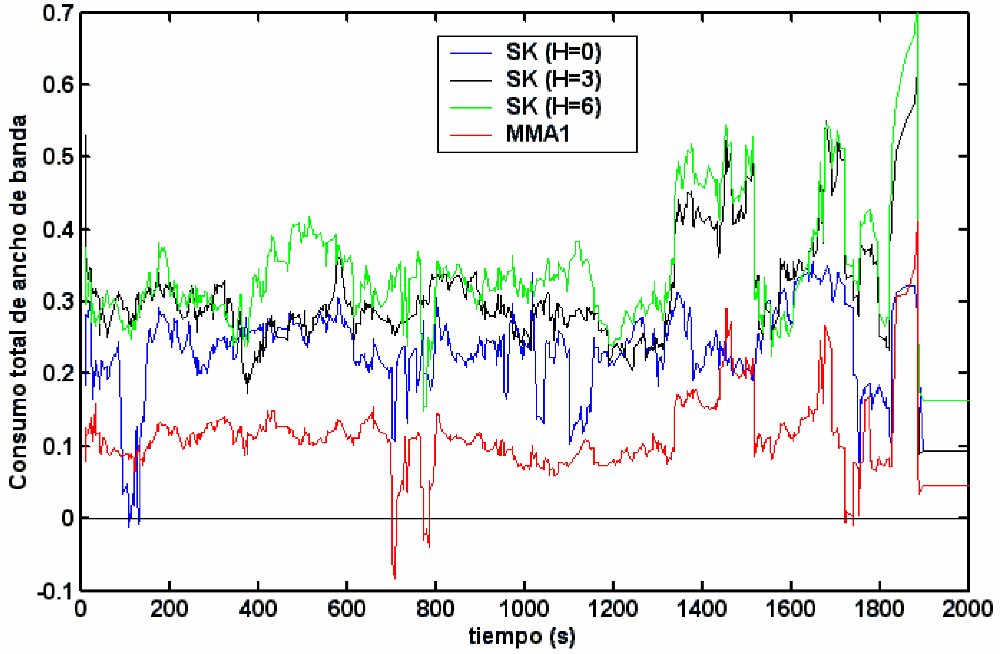


Figura 5.27. Consumo total de ancho de banda normalizado a MMA2, corrida 4.

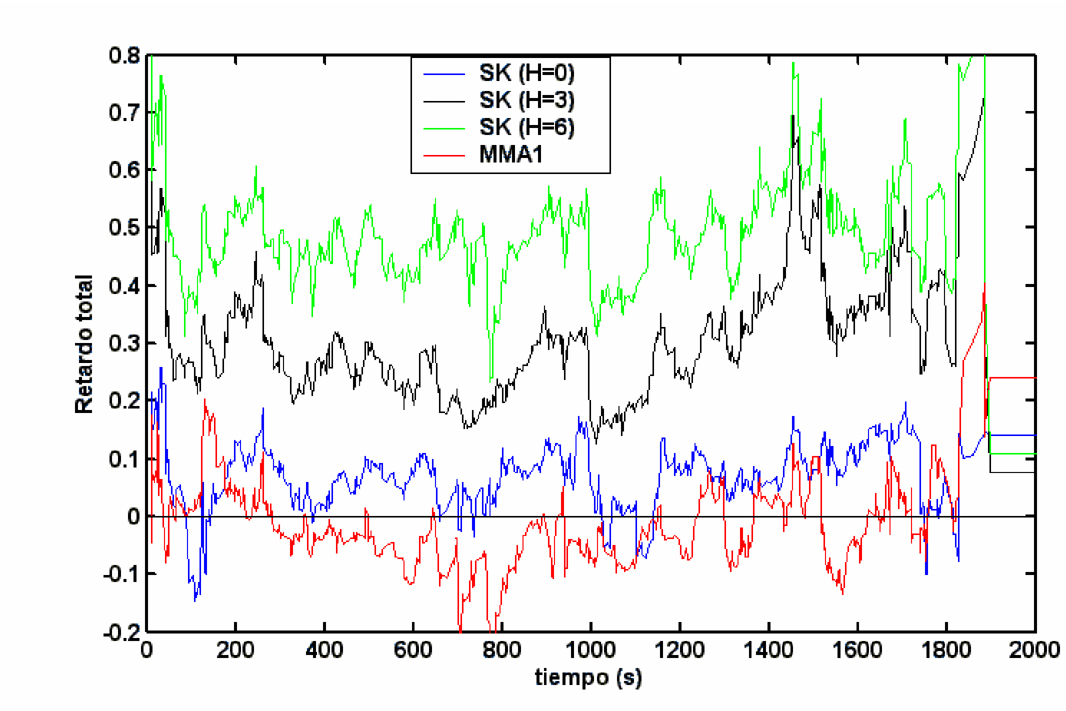


Figura 5.28. Retardo total normalizado a MMA2, corrida 4.

Claramente, MMA1 y MMA2 obtuvieron resultados satisfactorios en ambos modos, esparzo y denso, demostrando un buen comportamiento en esta red de prueba, hallando un gran número de soluciones no dominadas por el algoritmo SK, y dominando a un gran número de árboles proveídos por este algoritmo (Tablas 5.5, 5.7, 5.8 y 5.10). MMA1 construyó, en más del 50 % de los casos, mejores árboles que SK (H = 3) y SK (H = 6). Por su parte, MMA2 obtuvo mejores soluciones que SK (H = 3) y SK (H = 6) en más del 41 % de las solicitudes de la corrida 4, mientras que en las demás corridas el porcentaje fue mayor al 50 %.

Los gráficos muestran que, para las simulaciones en modo esparzo (corridas 1 y 3), MMA2 optimizó mejor que MMA1 y SK el consumo total de ancho de banda (Figuras 5.9 y 5.21), mientras que la utilización de los enlaces fue similar en ambos algoritmos propuestos y SK con H = 3 y H = 6 (Figuras 5.6, 5.7, 5.8 y 5.18, 5.19 y 5.20). Por otra parte, el retardo total de MMA1 y MMA2 también fue menor que el retardo total de SK (Figuras 5.10 y 5.22).

Para las simulaciones en modo denso (corridas 2 y 4), MMA1, MMA2 y SK obtuvieron valores similares de utilización máxima de los enlaces (Figuras 5.11, 5.12, 5.13, 5.14 y 5.23, 5.24, 5.25 y 5.26), aunque SK (H = 3) y SK (H = 6) mostraron una ventaja sobre ambos algoritmos propuestos en este trabajo. Por otra parte, MMA2 optimizó mejor que los otros algoritmos el consumo total de ancho de banda (Figuras 5.15 y 5.27), tal como ocurrió en modo esparzo. Note que este resultado está relacionado con los resultados obtenidos en los Problemas de Prueba estáticos, donde MMA2 encontraba aquellas soluciones de bajo costo y alto retardo máximo y medio, situadas en las esquinas del espacio de búsqueda.

A pesar de que el consumo de MMA1 fue mayor que el de MMA2, los resultados demostraron que este algoritmo también optimizó mejor que SK esta métrica (Figuras 5.15 y 5.27). Con respecto al retardo total, MMA1 obtuvo retardos menores que MMA2 y SK. MMA2 también optimizó mejor que SK esta métrica (Figuras 5.16 y 5.28).

5.3.2 Simulaciones Usando la Red de la NSF

La Figura 5.29 muestra la red de la NSF [BAR00] y la métrica de retardo d_{ij}, $\forall(i, j)\in E$. La capacidad de los enlaces es de 1.5 Mbps.

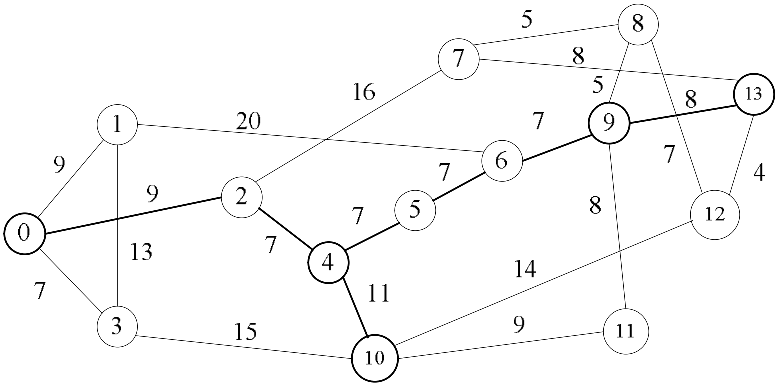


Figura 5.29. Red de la NSF, con el retardo en ms sobre cada enlace.

Para cada una de las corridas, 300 solicitudes multicast fueron generadas, cada una con una demanda ϕ = 150 Kbps. La Tabla 5.11 muestra los parámetros de MMA1 y MMA2 con los cuales se realizaron las simulaciones, el tiempo medio de cómputo de un árbol multicast (t_a) y el rango del tamaño de los grupos multicast. El tiempo medio de cómputo de SK para los distintos valores de H fue de 1 ms aproximadamente.

Tabla 5.11. Parámetros de MMA1 y MMA2, tiempo medio de construcción de un árbol multicast y rango del tamaño de los grupos multicast

		|*P*|	Generaciones	t_a	Tamaño grupo
Corrida 1	MMA1	40	400	60 ms	[3, 5]
	MMA2	30	50	45 ms	
Corrida 2	MMA1	40	400	90 ms	[6, 10]
	MMA2	30	50	40 ms	
Corrida 3	MMA1	40	400	60 ms	[3, 5]
	MMA2	30	50	50 ms	
Corrida 4	MMA1	40	400	90 ms	[6, 10]
	MMA2	30	50	45 ms	

En estas corridas, los tres algoritmos enrutaron todas las solicitudes multicast. La Tabla 5.12 resume los resultados de las corridas 1 y 2, mientras que la 5.13 resume los resultados de las corridas 3 y 4.

En todas las corridas, MMA1 y MMA2 construyeron un mayor número de árboles que dominaron a los respectivos árboles de SK. Al aumenta H, la diferencia en la métrica de dominancia fue aún mayor a favor de los algoritmos propuestos en este trabajo. En las 4 corridas, MMA1 y MMA2 construyeron mejores árboles que SK (H = 2) y SK (H = 4) en más del 50 % de las solicitudes multicast. Cuando los algoritmos propuestos en este trabajo fueron comparados con SK (H = 0), se observa un gran número de indiferencias. Esto se debe a la relación de compromiso entre los objetivos. Dado que MMA1 y MMA2 eligen una solución del frente, dicha solución puede no tener las cuatro funciones objetivo menores que la solución de SK, y viceversa, conllevando a una situación donde se presentan conflictos entre los objetivos, y creando las citadas relaciones de compromiso. De todas maneras, note que $ND_{MMA1\text{-}SK(H=0)}$ es mayor que $ND_{SK(H=0)\text{-}MMA1}$. De forma similar, $ND_{MMA2\text{-}SK(H=0)}$ también es mayor que $ND_{SK(H=0)\text{-}MMA2}$, indicando que ambos algoritmos proveyeron un mayor número de soluciones que dominaron a los correspondientes árboles de SK (H=0).

Cuando MMA1 y MMA2 son comparados, el aspecto principal a resaltar es que, en modo denso (corridas 2 y 4), MMA2 claramente tuvo un mejor comportamiento que MMA1, dado que $ND_{MMA2\text{-}MMA1}>ND_{MMA1\text{-}MMA2}$. En modo esparzo (corridas 1 y 3), MMA1 obtuvo un mayor número de dominancias en la corrida 1, mientras que en la corrida 3 la dominancia entre ambos algoritmos fue similar.

Tabla 5.12. Resultados de las corridas 1 y 2

	Corrida 1			Corrida 2		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	63	36	201	49	81	170
MMA1 - SK (H=0)	82	26	192	69	13	218
MMA1 - SK (H=2)	173	19	108	181	14	105
MMA1 - SK (H=4)	243	7	50	226	4	70
MMA2 - SK (H=0)	70	31	199	64	11	225
MMA2 - SK (H=2)	169	14	117	201	7	92
MMA2 - SK (H=4)	245	11	44	241	4	55

Tabla 5.13. Resultados de las corridas 3 y 4

	Corrida 3			Corrida 4		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	53	54	193	49	79	172
MMA1 - SK (H=0)	64	33	203	58	16	226
MMA1 - SK (H=2)	165	23	112	195	13	92
MMA1 - SK (H=4)	236	8	56	233	5	62
MMA2 - SK (H=0)	66	40	194	52	8	240
MMA2 - SK (H=2)	164	24	112	191	13	96
MMA2 - SK (H=4)	247	11	42	255	2	43

5.3.3 Simulaciones Sobre Sistemas Autónomos

Las siguientes dos redes de prueba fueron tomadas de [SPR02]. Dichas topologías representan sistemas autónomos reales. Debido a que [SPR02] solo provee la topología de la red y el retardo de sus enlaces, para las simulaciones se supone que cada enlace tiene capacidad de 1 Mbps. La Tabla 5.14 muestra el número de nodos y enlaces, el número de AS y el nombre de ambas redes.

Tabla 5.14. Topologías proveídas por [SPR02]

AS	Nodos	Enlaces
Sprintlink (US) 1239	44	166
Tiscali (Europa) 3257	49	172

5.3.3.1 Simulaciones Sobre el AS 1239

Se han generado 400 solicitudes multicast para la corrida 2 y 300 para las corridas 1, 3 y 4. La demanda de tráfico de cada solicitud fue de ϕ= 50 Kbps para la corrida 2 y ϕ=100 Kbps para las demás. En estas corridas, todos los algoritmos enrutaron la misma cantidad de solicitudes multicast. En las corridas 1, 2 y 3 fueron enrutadas todas las solicitudes, mientras que en corrida 4 fueron enrutadas 263 de las 300 solicitudes y 37 fueron rechazadas por falta de capacidad de la red. La Tabla 5.15 da los parámetros utilizados por MMA1 y MMA2, y el rango del tamaño de los grupos multicast.

Tabla 5.15. Parámetros de MMA1 y MMA2, y rango del tamaño de los grupos

		$\|P\|$	Generaciones	Tamaño grupo
Corrida 1	MMA1	40	500	[3, 16]
	MMA2	30	75	
Corrida 2	MMA1	50	500	[16, 32]
	MMA2	50	100	
Corrida 3	MMA1	40	500	[3, 16]
	MMA2	40	90	
Corrida 4	MMA1	50	500	[16, 32]
	MMA2	50	100	

Las Tablas 5.16 y 5.17 resumen los resultados de las corridas. En esta red, MMA1 y MMA2 solo fueron comparados con SK (H = 0) y SK (H = 2), debido al gran tiempo de cómputo que SK requeriría para enrutar las demandas de tráfico con valores mayores de H.

En las corridas 1 y 2, MMA1 tuvo un rendimiento superior a MMA2, considerando la métrica de dominancia. Una situación inversa se dio en las corridas 3 y 4, en las que $\mathrm{ND}_{\mathrm{MMA2\text{-}MMA1}} > \mathrm{ND}_{\mathrm{MMA1\text{-}MMA2}}$. A pesar de ello, el rendimiento es considerado parejo, dado el alto número de indiferencias entre ambos algoritmos. En las corridas 1, 2 y 4, el número de árboles indiferentes fue mayor al 50 %, mientras que la corrida 4 el porcentaje fue mayor al 47 %.

La diferencia de dominancia entre los algoritmos propuestos en este trabajo y SK fue aún mayor que las diferencias en las anteriores redes de prueba. En la corrida 2, una sola solución de SK dominó a la correspondiente solución de MMA2, mientras que ninguna solución de este algoritmo dominó a alguna solución de MMA1. En contrapartida, aún en el caso más favorable de SK (cuando H = 0), MMA1 dominó a SK en casi el 90 % de los árboles, mientras que MMA2 lo hizo en el 74 % de los casos. Cifras similares fueron obtenidas en las corridas 1, 3 y 4.

Tabla 5.16. Resultado de las corridas 1 y 2

	Corrida 1			Corrida 2		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	61	30	209	77	38	285
MMA1 - SK (H=0)	210	9	81	354	0	46
MMA1 - SK (H=2)	272	2	26	387	0	13
MMA2 - SK (H=0)	163	5	132	296	0	104
MMA2 - SK (H=2)	246	1	53	360	1	39

Tabla 5.17. Resultado de las corridas 3 y 4

	Corrida 3			Corrida 4		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	48	63	189	32	43	225
MMA1 - SK (H=0)	180	8	112	192	1	107
MMA1 - SK (H=2)	272	0	28	232	0	68
MMA2 - SK (H=0)	138	9	153	132	0	168
MMA2 - SK (H=2)	269	1	30	243	0	57

El tiempo medio consumido para calcular un árbol multicast, en segundos, es mostrado en la Tabla 5.18. Note que SK (H = 0) fue mucho más rápido que MMA1 y MMA2, pero cuando H = 2, sus tiempos de cómputo ya sobrepasaron el segundo, una situación que no había ocurrido en las redes de prueba anteriores. Esto se debe a que el AS 1239 es una red con mayor número de enlaces que las anteriores redes de prueba, NSF y NTT. Sin embargo, los tiempos de cómputo de MMA1 y MMA2 todavía fueron menores al segundo.

Tabla 5.18. Tiempo medio de cómputo de un árbol multicast, en segundos

	MMA1	MMA2	SK (H = 0)	SK (H = 2)
Corrida 1	0,165	0,170	0,014	0,24
Corrida 2	0,41	0,5	0,091	1,775
Corrida 3	0,16	0,21	0,01	0,2
Corrida 4	0,43	0,62	0,1	1,64

5.3.3.2 Simulaciones Sobre el AS 3257

Se han generado 400 solicitudes multicast para cada corrida. La demanda de tráfico de cada solicitud fue de $\phi = 50$ Kbps. Nuevamente todos los algoritmos enrutaron la misma cantidad de solicitudes multicast. En las corridas 1 y 3 fueron enrutadas todas las solicitudes multicast. En la corrida 2 fueron rechazadas 3 solicitudes multicast, mientras que en la corrida 4 fue rechazada 1 solicitud. La Tabla 5.19 da los parámetros de MMA1 y MMA2, y el rango del tamaño de los grupos para cada corrida.

Tabla 5.19. Parámetros de MMA1 y MMA2, y rango del tamaño de los grupos multicast

		$\lvert P \rvert$	Generaciones	Tamaño grupo
Corrida 1	MMA1	40	500	[3, 17]
	MMA2	40	60	
Corrida 2	MMA1	50	500	[17, 35]
	MMA2	50	100	
Corrida 3	MMA1	40	500	[3, 17]
	MMA2	40	60	
Corrida 4	MMA1	50	500	[17, 35]
	MMA2	50	100	

Las Tablas 5.20 y 5.21 resumen los resultados de las corridas. MMA1 obtuvo un mayor número de soluciones que dominaron a las correspondientes soluciones de MMA2 en las corridas 1, 2 y 4. Sin embargo, como ha ocurrido en las demás redes de prueba, el número de indiferencias entre ambos algoritmos fue elevado, mayor al 50 % de las solicitudes. Nuevamente en la corrida 3, MMA2 obtuvo un mayor número de dominancias que MMA1 ($ND_{MMA2\text{-}MMA1} > ND_{MMA1\text{-}MMA2}$). Es decir, en modo esparzo y con retardo unitario, el número de árboles construidos por MMA2 que dominaron a las correspondientes soluciones de MMA1 fue mayor que la cantidad de árboles de MMA1 que dominaron a las correspondientes soluciones de MMA2.

La diferencia de dominancia entre los algoritmos propuestos en este trabajo y SK fue nuevamente considerable a favor de MMA1 y MMA2. Note que SK (H = 2) no produjo un solo árbol que haya dominado a los correspondientes árboles de MMA1 y MMA2. Además, más del 96 % de los árboles de SK (H = 2) fueron dominados por los árboles de MMA1 y MMA2 (al menos 385 casos). Cuando MMA1 y MMA2 son comparados con SK (H = 0), las cifras son similares. En más del 78 % de los casos, los árboles proveídos por ambos algoritmos dominaron a los correspondientes árboles de dicho algoritmo.

Tabla 5.20. Resultado de las corridas 1 y 2

	Corrida 1			Corrida 2		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	62	22	316	137	41	222
MMA1 - SK (H=0)	337	6	57	385	0	15
MMA1 - SK (H=2)	391	0	9	394	0	6
MMA2 - SK(H=0)	325	7	68	386	0	14
MMA2 - SK(H=2)	385	0	15	395	0	5

Tabla 5.21. Resultado de las corridas 3 y 4

	Corrida 3			Corrida 4		
A1 - A2	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}	ND_{A1-A2}	ND_{A2-A1}	I_{A1-A2}
MMA1 - MMA2	30	80	290	107	68	225
MMA1 - SK (H=0)	325	6	69	382	1	17
MMA1 - SK (H=2)	391	0	9	397	0	3
MMA2 - SK(H=0)	315	5	80	391	0	9
MMA2 - SK(H=2)	395	0	5	399	0	1

La Tabla 5.22 muestra el tiempo medio consumido para calcular un árbol multicast. Como en las simulaciones sobre AS 1239, SK (H = 0) fue más rápido que MMA1 y MMA2. Sin embargo, cuando H = 2, los tiempos de SK ya sobrepasaron el segundo, mientras que MMA1 y MMA2 tuvieron tiempos de cómputo menores a 650 ms.

Tabla 5.22. Tiempo medio de cómputo de un árbol multicast, en segundos

	MMA1	MMA2	SK (H = 0)	SK (H = 2)
Corrida 1	0,17	0,19	0,028	0,13
Corrida 2	0,42	0,53	0,045	1,05
Corrida 3	0,17	0,22	0,007	0,13
Corrida 4	0,44	0,62	0,053	1,28

5.3.4 Resumen de las Simulaciones en Ambientes Dinámicos

Las corridas sobre la NTT demostraron que la utilización de los enlaces de MMA1, MMA2, SK (H = 3) y SK (H = 6) fue similar en ambos modos, denso y esparzo (Figuras 5.6, 5.7, 5.8, 5.12, 5.13, 5.14, 5.18, 5.19, 5.20, 5.24, 5.25 y 5.26). Sin embargo, SK (H = 3) y SK (H = 6) obtuvieron una ventaja sobre MMA1 y MMA2, en modo denso y retardo unitario (Figuras 5.24, 5.25 y 5.26), mientras que SK (H = 0) produjo una pobre utilización de los enlaces en modo esparzo (Figuras 5.5 y 5.17).

El consumo total de ancho de banda fue optimizado de una mejor manera por MMA2 (Figuras 5.9, 5.15, 5.21 y 5.27). Por otra parte, a pesar de tener un consumo mayor de ancho de banda que MMA2, MMA1 también optimizó mejor que SK esta métrica.

Mientras que en modo esparzo MMA1, MMA2 y SK (H = 0) tuvieron un retardo total similar (Figuras 5.10 y 5.22), en modo denso MMA1 tuvo un retardo total inferior a los demás algoritmos (Figuras 5.16 y 5.28). En este modo, MMA2 también obtuvo un menor retardo que SK.

Las Tablas 5.23, 5.24, 5.25 y 5.26 resumen los resultados, considerando la métrica de dominancia, de las corridas 1, 2, 3 y 4 llevadas a cabo sobre las distintas topologías. Cada fila de las tablas compara dos algoritmos. La columna $ND_{A1\text{-}A2}$ es la suma de todos los árboles, en las cuatro topologías de red usadas, del algoritmo A1 que dominan a los árboles del algoritmo A2. $ND_{A2\text{-}A1}$ da la relación inversa, mientras que $I_{A1\text{-}A2}$ da el número de indiferencias entre los algoritmos. La columna solicitudes da el número total de árboles multicast, el cual es la suma de las solicitudes de las corridas analizadas en las distintas topologías. Por ejemplo, la fila 1 de la Tabla 5.23 compara las métricas de dominancia de MMA1 y MMA2 en las corridas 1. El número total de solicitudes es 1400, dado que hubo 4 corridas 1 (una sobre cada red: NTT, NSF, AS 1239 y AS 3257). Sobre las redes NTT y AS 3257 se generaron 400 solicitudes, mientras que sobre las redes NSF y AS 1239 se generaron 300 solicitudes multicast. De esta forma, la suma totaliza 1400 solicitudes multicast en modo esparzo con retardo no unitario. La Tabla muestra que MMA1 y MMA2 produjeron soluciones que dominaron al otro algoritmo en un porcentaje similar. Mientras MMA2 dominó a MMA1 en el 12,64 % de los casos, MMA1 lo hizo en casi el 17 % de ellos. Dado que la indiferencia fue mayor al 70 %, la relación de compromiso entre las soluciones proveídas por ambos algoritmos es evidente. Por otra parte, note que MMA1 dominó a SK (H = 0) en el 55 % de los casos, mientras la relación inversa fue menor al 4 %. Al aumentar H, la dominancia a favor de MMA1 fue aún mayor, alcanzando 83,6 % con H = 2. Cifras similares se dieron entre MMA2 y SK.

Tabla 5.23. Resumen de los resultados en modo esparzo y retardo no unitario (corridas 1)

	ND_{A1-A2}	%	ND_{A2-A1}	%	I_{A1-A2}	%	Solicitudes
MMA1 - MMA2	237	16,93	177	12,64	986	70,43	1400
MMA1 - SK (H=0)	770	55,00	49	3,50	581	41,50	1400
MMA1 - SK (H=2)	836	83,60	21	2,10	143	14,30	1000
MMA1 - SK (H=3)	239	59,75	5	1,25	156	39,00	400
MMA1 - SK (H=4)	243	81,00	7	2,33	50	16,67	300
MMA1 - SK (H=6)	303	75,75	6	1,50	91	22,75	400
MMA2 - SK (H=0)	659	47,07	43	3,07	698	49,86	1400
MMA2 - SK (H=2)	701	70,10	32	3,20	267	26,70	1000
MMA2 - SK (H=3)	216	54,00	2	0,50	182	45,50	400
MMA2 - SK (H=4)	245	81,67	11	3,67	44	14,67	300
MMA2 - SK (H=6)	315	78,75	1	0,25	84	21,00	400

La Tabla 5.24 muestra que en modo denso y con retardo no unitario MMA1 produjo mejores árboles que MMA2 en casi el 20 % de los casos, mientras que la relación inversa se dio en casi el 13,3 % de las solicitudes multicast. De nuevo la indiferencia entre las soluciones de ambos algoritmos fue alta, mayor al 67 %. La dominancia de MMA1 sobre SK fue mayor al 55 %, mientras que la de MMA2 sobre SK fue mayor al 50 %, en todos los casos. Además, en ningún caso, la dominancia de SK sobre MMA1 o MMA2 fue mayor al 2 %. Esto indica que ambos algoritmos han obtenido aún mejores soluciones que SK en modo denso.

Tabla 5.24. Resumen de los resultados en modo denso y retardo no unitario (corridas 2)

	ND_{A1-A2}	%	ND_{A2-A1}	%	I_{A1-A2}	%	Solicitudes
MMA1 - MMA2	289	19,27	200	13,33	1011	67,40	1500
MMA1 - SK (H=0)	988	65,87	24	1,87	488	32,53	1500
MMA1 - SK (H=2)	962	87,45	14	1,27	124	11,27	1100
MMA1 - SK (H=3)	231	57,75	8	2,00	161	40,25	400
MMA1 - SK (H=4)	226	75,33	4	1,33	70	23,33	300
MMA1 - SK (H=6)	254	63,50	8	2,00	138	34,50	400
MMA2 - SK (H=0)	834	55,60	20	1,33	646	43,07	1500
MMA2 - SK (H=2)	956	86,91	8	0,73	136	12,36	1100
MMA2 - SK (H=3)	205	51,25	5	1,25	190	47,50	400
MMA2 - SK (H=4)	241	80,33	4	1,33	55	18,33	300
MMA2 - SK (H=6)	251	62,75	5	1,25	144	36,00	400

Las Tabla 5.25 muestra que en modo esparzo y retardo unitario, MMA2 dominó a MMA1 en más del 20 % de los casos, mientras que MMA1 lo hizo en el 12,29 % de las solicitudes. Si bien la relación entre ambos fue distinta a los casos anteriores (en los cuales el porcentaje de dominancia de MMA1 fue mayor), también en esta corrida la relación de indiferencia fue alta. Por otra parte, de nuevo MMA1 y MMA2 produjeron mejores soluciones que SK en un gran porcentaje de los casos. MMA1 dominó a SK (H = 0) en más del 48 % de los casos, mientras que la dominancia de MMA2 fue mayor al 41%. Los porcentajes de dominancia a favor de MMA1 y MMA2 fueron aún mayores cuando se aumentó H.

Tabla 5.25. Resumen de los resultados en modo esparzo y retardo unitario (corridas 3)

	ND_{A1-A2}	%	ND_{A2-A1}	%	I_{A1-A2}	%	Solicitudes
MMA1 - MMA2	172	12,29	295	21,07	933	66,64	1400
MMA1 - SK (H=0)	676	48,29	51	3,64	673	48,07	1400
MMA1 - SK (H=2)	828	82,80	23	2,30	149	14,90	1000
MMA1 - SK (H=3)	237	59,25	7	1,75	159	39,75	400
MMA1 - SK (H=4)	236	78,67	8	2,67	56	18,67	300
MMA1 - SK (H=6)	286	71,50	3	0,75	111	27,75	400
MMA2 - SK (H=0)	587	41,93	58	4,14	755	53,93	1400
MMA2 - SK (H=2)	828	82,80	25	2,50	147	14,70	1000
MMA2 - SK (H=3)	228	57,00	10	2,50	162	40,50	400
MMA2 - SK (H=4)	247	82,33	11	3,67	42	14,00	300
MMA2 - SK (H=6)	311	77,75	0	0,00	89	22,25	400

Por último, la Tabla 5.26 muestra que en modo denso y retardo unitario, nuevamente MMA2 dominó a MMA1 en un mayor número de casos. Sin embargo, la relación de indiferencia, como en todas las corridas en los distintos modos, fue mayor al 60 %. También en este modo, el porcentaje de soluciones de MMA1 y MMA2 que dominaron a las correspondientes soluciones de SK, aun en el caso más favorable a este algoritmo, fue mayor al 40 %.

Tabla 5.26. Resumen de los resultados en modo denso y retardo unitario (corridas 4)

	ND_{A1-A2}	%	ND_{A2-A1}	%	I_{A1-A2}	%	Solicitudes
MMA1 - MMA2	210	15,00	263	18,79	927	66,21	1400
MMA1 - SK (H=0)	777	55,50	24	1,71	599	42,79	1400
MMA1 - SK (H=2)	824	82,40	13	1,30	163	16,30	1000
MMA1 - SK (H=3)	205	51,25	5	1,25	190	47,50	400
MMA1 - SK (H=4)	233	77,67	5	1,67	62	20,67	300
MMA1 - SK (H=6)	272	68,00	4	1,00	124	31,00	400
MMA2 - SK (H=0)	617	44,07	20	1,43	763	54,50	1400
MMA2 - SK (H=2)	833	83,30	13	1,30	154	15,40	1000
MMA2 - SK (H=3)	165	41,25	3	0,75	232	58,00	400
MMA2 - SK (H=4)	255	85,00	2	0,67	43	14,33	300
MMA2 - SK (H=6)	264	66,00	5	1,25	131	32,75	400

5.4 Resumen del Capítulo

En el presente Capítulo se han presentado los resultados experimentales de las pruebas realizadas con el objeto de evaluar el desempeño de MMA1 y MMA2. En primer lugar, ambos algoritmos han sido evaluados con pequeños problemas de prueba donde el conjunto Pareto es conocido de antemano. Las corridas demostraron el buen funcionamiento de MMA1 y MMA2 sobre ambos problemas de prueba estáticos. Sin embargo, el rendimiento de MMA2 ha sido claramente superior al de MMA1. El análisis de los resultados experimentales muestra que la representación cromosómica de MMA2 provee a este algoritmo una mayor capacidad de exploración, permitiendo obtener soluciones Pareto ubicadas en “esquinas” del espacio de búsqueda que no siempre son obtenidas por MMA1.

Posteriormente, MMA1 y MMA2 fueron comparados entre si y con SK a través de simulaciones de ambientes dinámicos, donde las solicitudes multicast llegan una tras otra. Las extensas simulaciones hechas sobre distintas topologías de redes mostraron el buen funcionamiento de los algoritmos propuesto en este trabajo. En la mayor parte de ellas, MMA1 y MMA2 obtuvieron mejores resultados que SK, tal como lo resumen las Tablas 5.23, 5.24, 5.25 y 5.26.

6 Conclusiones

Como culminación del presente trabajo, en este capítulo son presentadas las conclusiones finales y son propuestos algunos tópicos como posibles trabajos futuros.

6.1 Conclusiones Finales

El problema de enrutamiento multicast en redes de computadoras, con más de una métrica a considerar, generalmente es tratado como un problema de optimización mono-objetivo, donde las funciones objetivo son combinadas escalarmente a través de una suma ponderada. El presente trabajo formula por primera vez este problema como uno puramente multiobjetivo, donde la utilización máxima de los enlaces del árbol α_T, el costo C_T, el retardo máximo de extremo a extremo D_M y el retardo medio D_A son tratados como funciones objetivo a minimizar en forma simultánea e independiente. Desde un punto de vista teórico, esto constituye un claro aporte, pues hasta la fecha, el problema solo había sido formulado y resuelto como un SOP.

Para resolver este problema, dos nuevos algoritmos evolutivos basados en el SPEA fueron propuestos: MMA1 y MMA2. Estos algoritmos obtienen un conjunto Pareto óptimo de soluciones en una corrida. Esta característica es de gran importancia, pues la solución más adecuada puede ser elegida para cada caso particular, sin considerar restricciones a priori.

Dado que hasta la fecha no han sido publicados trabajos sobre enrutamiento multicast multiobjetivo ni *test beds* con problemas de prueba que sirvan de base comparativa y experimental, los algoritmos propuestos en este trabajo fueron evaluados con pequeños problemas de prueba estáticos formulados para la ocasión. El Problema de Prueba 1 (P1) fue tomado de trabajos anteriores y su formulación extendida a multiobjetivo, mientras que el Problema de Prueba 2 (P2) fue propuesto por primera vez para esta tesis. En ambos, el conjunto Pareto óptimo es conocido. Además, MMA1 y MMA2 fueron evaluados en ambientes dinámicos, donde las solicitudes multicast arriban una después de otra. En este caso, una solución particular del conjunto Pareto, proveído por el algoritmo evaluado, fue elegida para cada solicitud multicast. Los resultados fueron comparados con SK, un algoritmo recientemente publicado en un simposio internacional arbitrado por la IEEE.

En el Problema de Prueba 1 (P1), MMA2 convergió al conjunto Pareto en todas las corridas, mientras que en el Problema de Prueba 2 (P2) halló todas las soluciones óptimas en el 83 % de los casos. Por su parte, MMA1 obtuvo el frente Pareto en el 47 % de las corridas de P1, y en ninguna ocasión halló el conjunto óptimo de P2. Sin embargo, en todas las corridas obtuvo más del 87 % de las soluciones óptimas. El análisis de los resultados experimentales muestra que la representación cromosómica de MMA2 provee a este algoritmo una mayor capacidad de exploración, permitiendo obtener aquellas soluciones Pareto óptimas ubicadas en “esquinas” del espacio de búsqueda que no siempre son halladas por MMA1.

En situaciones dinámicas, las extensas simulaciones hechas sobre distintas topologías mostraron el buen funcionamiento de los algoritmos propuesto en este trabajo. En la mayoría de los casos, en ambos modos (denso y esparzo), los árboles proveídos por MMA1 y MMA2 dominaron a los correspondientes árboles de SK (Tablas 5.23, 5.24, 5.25 y 5.26). Es decir, tuvieron al menos una función objetivo menor, y las demás funciones objetivo menor o igual que las funciones objetivo de los árboles hallados por SK. De esta forma, no solo se obtuvieron soluciones con valores de retardo medio y máximo menores, sino también se optimizó mejor la utilización máxima de los enlaces y el costo total en términos de recursos, como ancho de banda consumido.

6.2 Trabajos Futuros

De forma a continuar con el trabajo iniciado en esta tesis, los siguientes tópicos son propuestos como trabajos futuros:

- modelado matemático de los algoritmos propuestos, de manera a hallar la complejidad de los mismos;
- evaluación comparativa de MMA1, MMA2 y nuevos algoritmos multiobjetivos basados en otros MOEAs como SPEA2, NSGA, NSGA2 y CNSGA2;

- estudio e implementación de nuevos esquemas de ingeniería de tráfico multi-árbol, donde el flujo de datos de un grupo multicast es transmitido a los nodos destinos a través de varios árboles;
- definición de un *test bed* con problemas de prueba que sirvan de base comparativa y experimental para los algoritmos desarrollados y publicados, extendiendo los paradigmas mono-objetivos a multiobjetivos, tal como se hizo con el Problema de Prueba 1.

Bibliografía

[ARA02] P. T. de Araujo, y G. M. Barbosa, “Multicast Routing with Quality of Service and Traffic Engineering Requirements in The Internet, Based On Genetic Algorithm”, *Proceedings of the 7*th *Brazilian Symposium on Neural Networks* (SBRN´02), Brasil, 2002.

[BAR03] B. Barán, S. Duarte, y D. Benítez, “Telecommunication Network Design with Parallel Multiobjective Evolutionary Algorithm”, *IFIP/ACM Latin American Networking Conference*, Bolivia, 2003.

[BAR01] B. Barán, J. Vallejos, R. Ramos, y U. Fernández, “Multiobjective Reactive Power Compensation”, *Proceedings of the IEEE Transmission and Distribution Conference and Exposition*, USA, 2001.

[BAR00] B. Barán, y R. Sosa, “A New Approach for Antnet Routing”, *IEEE International Conference on Computer Communication and Networks* (ICCCN’2000), USA, 2000.

[BEV03] R. Beverly, y K. Claffy, “Wide-Area IP Multicast Traffic Characterization”, *IEEE Network*, Vol. 17, N° 1, pág. 8 -15, 2003.

[COE00] C. Coello, “EMOO Web Page, A complete list in alphabetical order”, *Evolutionary Optimization: an International Journal on Internet*, 2000. Disponible desde http://www.jeo.org/emo.

[COE99] C. Coello, “A comprehensive survey of evolutionary-based multiobjective optimization”, *Knowledge and Information System*, 1999.

[COE96] C. Coello, “A Empirical Study of Evolutionary Techniques for multi-objective optimization in Engineering Design”, *Ph.D Thesis,* Department of Computer Science, Tulane University, New Orleans, USA, 1996.

[COH78] J. Cohon, *Multiobjective programming and planning*, Academic Press, 1978.

[CRI04a] J. Crichigno, y B. Barán, “Multiobjective Multicast Routing Algorithm”, *11th International Conference on Telecommunications (ICT’2004)*, Brasil, 2004.

[CRI04b] J. Crichigno, y B. Barán, “A Multicast Routing Algorithm Using Multiobjective Optimization”, *11th International Conference on Telecommunications (ICT’2004)*, Brasil, 2004.

[CRI04c] J. Crichigno, y B. Barán, “Multiobjective Multicast Routing Algorithm for Traffic Engineering”, *13th International Conference on Computer Communications and Networks (ICCCN’2004)*, Chicago - USA, 2004.

[DAR85] C. Darwin, *On the Origin of Species by Means of Natural Selection*, 6° Edición, Penguin Classics, 1985.

[DEB99] K. Deb, “Evolutionary algorithms for multi-criterion optimization in engineering design”, *Proceedings of the Evolutionary Algorithms in Engineering and Computer Science (EUROGEN’99)*, 1999.

[DIJ59] E. Dijkstra, “A note on two problems in connexion with graphs”, *Numerische Mathematik*, Vol. 1, pág. 269-271, 1959.

[DON04] Y. Donoso, R. Fabregat, y J. Marzo, “Multi-objective Optimization Algorithm for Multicast Routing with Traffic Engineering”, *IEEE 3rd International Conference on Networking* (ICN´2004), Guadalupe – Caribe Francés, 2004.

[DON02] Y. Donoso, R. Fabregat, J. Marzo, y E. Calle, “Extensión de los Métodos Hop-by-Hop, CR-LDP y RSVP-TE para Multicast IP sobre MPLS”, *XXVIII Conferencia Latinoamericana de Informática*, Uruguay, 2002.

[FON95a] C. Fonseca, y P. Fleming, “Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms I: A Unified Formulation”, *Technical*

Report 564, Department of Automatic Control and System Engineering, University of Sheffield, Reino Unido, 1995.

[FON95b] C. Fonseca, y P. Fleming, “An Overview of Evolutionary Algorithm in Multiobjective Optimization”, *Evolutionary Computation*, Vol. 3, N° 1, pág. 1-16, 1995.

[FON93] C. Fonseca, y P. Fleming, “Genetic Algorithm for Multiobjective Optimization: Formulation, Discussion and Generalization”, *Proceedings of the 5th International Conference on Genetic Algorithms*, pág. 416-423, San Mateo, CA, USA. University of Illinois at Urbana Champaign, Morgan Kaufmann, 1993.

[GOL89] D. Goldberg, *Genetic Algorithm is Search, Optimization & Machine Learning*, Addison Wesley, 1989.

[HAJ92] P. Hagela, y C. Yin, “Genetic Search Strategies in Multicriterion Optimal Design”, *Structural Design*, Vol. 2, pág. 99-107, 1992.

[HEI00] J. Heitkoetter, y B. Heitkoetter, “The Hitch-Hiker’s Guide to Evolutionary Computation: A List of Frequently Asked Questions (FAQ)”, USENET: comp.ai.genetic, disponible vía FTP desde tfm.mit.edu/pub/usenet/news.answers/ai-faq/genetic, 2000.

[HOR97] J. Horn, “The nature of niching: genetic algorithms and the evolution of optimal, cooperative population”, *Ph.D Thesis*, University of Illinois at Urbain Champaign, USA, 1997.

[HOR94] J. Horn, N. Lafpliotis, y D. Golberg, “A Niched Pareto Genetic Algorithm for Multiobjective Optimization”, *Proceeding of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, Vol. 1, pág. 82-84, Piscataway-USA.

[HOR86] E. Horowitz, y S. Sahni, *Fundamentos de Estructura de Dados*, Campus LTDA, 1986.

[HOU99] S. Hougardy, y H. Promel, “A 1.589 approximation algorithm for the Steiner problem in graphs”, *ACM/SIAM SODA ´99*, pág. 448-453, 1999.

[HWA00] R. Hwang, W. Do, y S. Yang, “Multicast Routing Based on Genetic Algorithms”, *Journal of Information Science and Engineering*, Vol. 16, pág. 885-901, 2000.

[HWA92] F. Hwang y D. Richards, “Steiner Tree Problems”, *Networks*, Vol. 22, N° 1, pág 55-89, 1992.

[HWA79] C. Hwang, y A. Masud, *Multiple Objective Decision Making – Methods and Applications: A State of the Art Survey*, Springer Verlag, 1979.

[KOM93a] V. Kompella, “Multicast Routing Algorithm for Multimedia Traffic”, *Ph.D. Thesis,* University of California, San Diego, USA, 1993.

[KOM93b] V. Kompella, J. Pasquale, y G. Polyzos, “Multicast routing in multimedia communication”, *IEEE/ACM Transactions on Networking*, Vol. 1 N° 3, pág. 286-291, 1993.

[KOU81] L. Kou, G. Markowsky, y L. Berman, “A Fast Algorithm for Steiner Trees”, *Acta Informatica*, Vol. 15, N° 2, pág. 141-145, 1981.

[MOR80] J. Morse, “Reducing the size of the non-dominated set: pruning by clustering”, *Computer and Operation Research* 7(1-2), 55-66, 1980.

[OBA98] S. Obayashi, S. Takahashi, y Y. Takeguchi, “Niching and elitist models for mogas”, *5th International Conference on Parallel Problem Solving from Nature (PPSN-V)*, Berlín, Alemania, pág. 260-269, 1998.

[OOM02] D. Ooms, B. Sales, W. Livens, A. Acharya, F. Griffoul, y F. Ansari, “Overview of IP Multicast in a Multi-Protocol Label”, *RFC 3353,* 2002.

[PAR98] G. Parks, y I. Miller, “Selective breeding in a multiobjective genetic algorithm”, *5th International Conference on Parallel Problem Solving from Nature (PPSN-V)*, Berlín, Alemania, pág. 250-259, 1998.

[RAV98] C. P. Ravikumar, y R. Bajpai, “Source-based delay bounded multicasting in multimedia networks”, *Computer Communications*, Vol. 21, pág. 126-132, 1998.

[ROS67] R. Rosemberg, “Simulation on Genetic Populations with Biochemical Properties”, *Ph.D Thesis*, University of Michigan, Ann Harbor, Michigan, USA, 1967.

[RUD98] G. Rudolph, “On a multiobjective evolutionary algorithm and its convergence to the Pareto set”, *IEEE International Conference on Evolutionary Computation* (*ICEC’98*), Piscataway, USA, pág. 511-516.

[SEO02] Y. Seok, Y. Lee, Y. Choi, y C. Kim, “Explicit Multicast Routing Algorithm for Constrained Traffic Engineering”, *IEEE Proceedings of 7th International Symposium on Computer and Communications* (*ISCC´02*), Italia, 2002.

[SHA84] J. Shaffer, “Multiple Objective Optimization with Vector Evaluated Genetic Algorithms”, *Ph.D Thesis*, Vanderbilt University, USA, 1984.

[SOT02] A. Sotelo, B. Barán y C. Von Lucken, “Multiobjective Evolutionary Algorithms for Pump Scheduling Optimization”, *3rd International Conference on Engineering Computational Technology ECT-2002*, Rca. Checa, 2002.

[SPR02] N. Spring, R Mahajan, y D. Wetheral, “Measuring ISP topologies with Rocketfuel”, *Proceedings of the ACM SIGCOMM´02 Conference*, 2002.

[SRI94] N. Srinivas, y K. Deb, “Multiobjective Optimization Using Non-dominated Sorting Genetic Algorithm”, *MIT Evolutionary Computation*, Vol. 2, N° 3, pág. 221-248, 1994.

[STA01] W. Stallings. "MPLS", *The Internet Protocol Journal*, Vol. 4, N° 3, 2001.

[STA00] W. Stallings, *Comunicaciones y Redes de Computadores*, Prentice Hall, 2000.

[STE86] R. Steuer, *Multiple criteria optimization: theory, computation and application*, New York, 1986.

[TAK80] H. Takahashi y A. Matsuyama, "An approximate solution for the Steiner problem in graphs", *Mathematic Japonica*, Vol. 6, pág 573-577, 1980.

[TAN03] A. Tanenbaum, *Computer Networks*, Prentice Hall, 2003.

[VAN99] D. Veldhuizen, "Multiobjective Evolutionary Algorithms: Classifications, Analysis, and New Innovations", *Ph.D Thesis*, Graduated School of Engineering of the Air Force Institute of Technology, Air University, 1999.

[VAN98a] D. Veldhuizen, y B. Lamont, "Evolutionary computation and convergence to a Pareto front", Genetic Programming 1998: *Proceedings of the 3rd Annual Conference*, San Francisco, CA-USA, pág. 22-25, Morgan Kaufmann, 1998.

[VAN98b] D. Veldhuizen, y B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis", *Technical report TR-98-03*, Department of Electrical and Computer Engineering, Graduated School of Engineering, Air Force Institute of Technology, Wright-Paterson AFB, Ohio, 1998.

[WAN01] Z. Wang, Internet QoS, *Architectures and Mechanism for Quality of Service*, Morgan Kaufmann, 2001.

[WOL97] D. Wolpert, y W. Macready, "No Free Lunch Theorems for Optimizations", *IEEE Transactions on Evolutionary Computation*, Vol. 1, Nº 1, pág. 67-82, 1997.

[XIA99] F. Xiang, L. Junzhou, W. Jieyi, y G. Guanqun, "QoS routing based on genetic algorithm", *Computer Communications*, Vol. 22, pág. 1392-1399, 1999.

[YEN71] J. Yen, “Finding the k shortest loopless path in a network”, *Management Science*, 17:712-716, 1971.

[ZHE01] W. Zhengying, S. Bingxin, y Z. Erdun, “Bandwidth-delay-constraint least-cost multicast routing based on heuristic genetic algorithm”, *Computer Communications*, Vol. 24, pág. 685-692, 2001.

[ZHU03] Y. Zhu, W. Shu, y M. Wu, “Comparison study and evaluation of overlay multicast network”, *IEEE International Conference on Multimedia and Expo (ICME2003)*, Baltimore, USA, 2003.

[ZIT99] E. Zitzler, y L. Thiele, “Multiobjective Evolutionary Algorithms: A comparative Case Study and the Strength Pareto Approach”, *IEEE Trans. Evolutionary Computation*, Vol. 3, N° 4, pág. 257-271, 1999.