UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

Diseño de un Controlador para el Manejo de Tráfico sobre Redes de Cómputo

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

MAESTRO EN INGENIERÍA

INGENIERIA ELECTRICA - CONTROL

P R E S E N T A

WILLIAM SANCHEZ CONSTANTINO

DIRIGIDA POR:

DR. HECTOR BENITEZ PEREZ

Ciudad Universitaria

Octubre - 2009

JURADO ASIGNADO

Presidente: DR. ESPINOSA PEREZ GERARDO RENE

Secretario: DR. ALVAREZ ICAZA LONGORIA LUIS AGUSTIN

Vocal: DR. BENITEZ PEREZ HECTOR

1^{er} Suplente: DR. ARTEGA PEREZ MARCO

2^{do} Suplente: DR. TANG XU YU

Lugar donde se realizó la tesis:

INSTITUTO DE INVESTIGACIONES EN MATEMATICAS APLICADAS Y EN SISTEMAS

TUTOR DE TESIS:

DR. HECTOR BENITEZ PEREZ

FIRMA

Este trabajo se desarrolló en el Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas de la Universidad Nacional Autónoma de México (UNAM) bajo la tutoria del Dr. Héctor Benítez Pérez. Se contó con el apoyo de una beca para estudios de maestría del Consejo Nacional de Ciencia y Tecnología (CONACYT).

Redacción y edición de tesis

con LaTeX

A mis padres, Arturo Sánchez Gómez e Ilsa Constantino Molina por su apoyo en todo momento, mis hermanos Arturo y Sandra Luz por ser un motivo de superación a fin de despertar en ellos la motivación, a mis abuelos paternos Don Silvestre y Doña Petrona, y en memoria de mis abuelos maternos Don Miguel y Doña Lucefina.

Por todo el ayer les dedico con cariño mi mañana

Mi familia

Se ha vuelto espantosamente obvio que nuestra tecnología ha excedido a nuestra humanidad.

Albert Einstein

De pocas partidas he aprendido tanto como de la mayoría de mis derrotas.

GM J. R. Capablanca

El que me encomienda a mi propia satisfacción, me encomienda a lo que no puedo obtener.

William Shakespeare

Agradecimientos

Deseo agradecer al único Dios vivo, porque hasta hoy he estado entendiendo lo escrito por Pablo en Romanos 1:20-21 *"Porque las cosas invisibles de él, su eterna potencia y divinidad, se echan de ver desde la creación del mundo, siendo entendidas por las cosas que son hechas; de modo que son inexcusables, Porque habiendo conocido a Dios, no le glorificaron como a Dios, ni dieron gracias; antes se desvanecieron en sus discursos, y el necio corazón de ellos fué entenebrecido."*.

Agradezco a mi tutor el Dr. Héctor Benítez Pérez por la confianza brindada desde la propuesta del proyecto, su desarrollo y la conclusión de este trabajo. Juntamente a los doctores quienes fueron mis profesores en la maestría. A todos ellos gracias por su apoyo, consejos y por compartir sus conocimientos en cada una de sus enseñanzas.

Agradecer a mis compañeros en las clases de maestría y en el grupo de trabajo del Dr. Héctor, porque siempre de una u otra forma me apoyaron, me brindaron su amistad y aportaron comentarios a este trabajo.

A todas las personas que me brindaron su apoyo y que no deseo obviar a nadie, ni uno solo de ellos, por lo que les doy este espacio.

A toda mi familia por el apoyo mostrado cuando estaba con ellos.

Agradecer al CONACYT por la beca otorgada para el desarrollo de este trabajo, por la importancia que tiene este apoyo para quienes venimos de provincia.

¡Gracias a todos ustedes he logrado alcanzar esta importante meta!

Índice general

LISTA DE FIGURAS

LISTA DE TABLAS

Resumen

En este trabajo se propone un controlador difuso para el manejo de congestión en un router con el modelo de una red TCP/AQM con retardos variables, esto debido a que se supone que el retardo es asociado a la ruta elegida por la tabla de ruteo, puesto que el ruteo es dinámico entonces el retardo debe ser variable. Así, se estudia la influencia del ruteo en el control de congestión.

1. INTRODUCCION

El objetivo en este capítulo es dar una introducción sobre las redes de cómputo, su importancia y la evolución que han tenido hasta llegar a la gran red de redes: "El Internet". Sin embargo, se expone la problemática presentada ante esta evolución a una red universal donde apararece el *Protocolo de Control de Transmisión (TCP)* y un hardware llamado *router o ruteador*. Se presenta el funcionamiento del TCP y el problema de pérdida de paquetes que se presenta en internet, y el papel fundamental que juega el ruteador para la solución de éste problema. Este capítulo esta basado en [Forouzan, 2000] y [Comer, 2000].

1.1. Internet

En los 60's ARPA (Advanced Research Projects Agency) del Departamento de Defensa de los Estados Unidos comenzó una asociación con universidades de los Estados Unidos y otros organismos de investigación para el desarrollo de nuevas tecnologías de comunicación de datos. En 1969 comenzó a funcionar una versión experimental de ARPANET con cuatro nodos. Los protocolos iniciales de esta red eran lentos y solían sufrir de frecuentes problemas. En 1974, Vinton G. Cerf y Robert E. Khan propusieron un nuevo núcleo de protocolos base para

los siguientes desarrollos del Protocolo de Internet(IP) y del Protocolo de Control de Transmisión(TCP). Entre 1980-1983 se dá la migración de los hosts de ARPANET al nuevo conjunto de protocolos. El conglomerado de redes de investigación, académicas y gubernamentales, combinado con el núcleo de la red ARPANET, fue el principio de lo que llegaría a ser conocido como Internet.

Las redes de cómputo han crecido exponencialmente. Hace dos décadas eran pocos los que tenían acceso a una red. Hoy, la comunicación por computadora se ha vuelto una parte esencial de nuestra infraestructura. El crecimiento continuo de Internet es uno de los fenómenos más interesantes de la conectividad. Se ha convertido en un sistema de comunicación en producción que llega a millones de personas de todos los continentes.

El Internet ha sido acogida a lo largo de todo el mundo. El éxito y popularidad de Internet radica en su filosofía de funcionamiento basada en el concepto de paquetes y en la forma de transmitirlos. Casi ninguna red transfiere continuamente una cantidad arbitraria de datos. En cambio, el sistema de red divide los datos en pequeños bloques llamados paquetes. Esto permite un acceso justo y rápido de todas las computadoras que comparten un recurso. Los paquetes llevan tanto la información a trasmitir como información adicional sobre éstos (información de la fuente y el destino, la longitud de paquete y el tipo de datos). Este concepto de paquete le permite a la red almacenar los paquetes por horas, si es necesario, y entonces recuperarlos y aún saber hacia dónde enviarlos. De esta forma, Internet almacena un gran número de paquetes antes de hacer uso de una conexión (por ejemplo una línea telefónica internacional) permitiendo dividir el costo de transmisión entre muchos usuarios. Más aún, si una conexión se pierde entonces los

paquetes pueden ser almacenados y transmitidos cuando la conexión se restablezca.

Sin embargo, de la misma forma en que las personas que no hablan un mismo idioma tienen dificultades para comunicarse, las redes que utilizaban diferentes especificaciones e implementaciones tenían dificultades para intercambiar información. Para enfrentar el problema de incompatibilidad de redes, la Organización Internacional para la Estandarización (OSI) investigó modelos de conexión, desarrolló un modelo de red que ayuda a los fabricantes a crear redes que sean compatibles con otras, el modelo OSI.

1.2. Modelo OSI y TCP/IP

El modelo OSI esta constituido por 7 capas que definen las funciones de los protocolos de comunicaciones (Fig. 1.1). Las capas, comúnmente llamadas pilas, se comunican solamente con la capa superior o inferior de la pila, es decir, que los datos de salida pasan hacia abajo por cada una de las capas y los datos de entrada hacen lo mismo en sentido contrario.

La ventaja de esta arquitectura es que, al aislar las funciones de comunicación de la red en capas, minimizamos el impacto de cambios tecnológicos en el juego de protocolos, es decir, podemos añadir nuevas aplicaciones sin cambios en la red física y también podemos añadir nuevo hardware a la red sin tener que reescribir el software de aplicación.

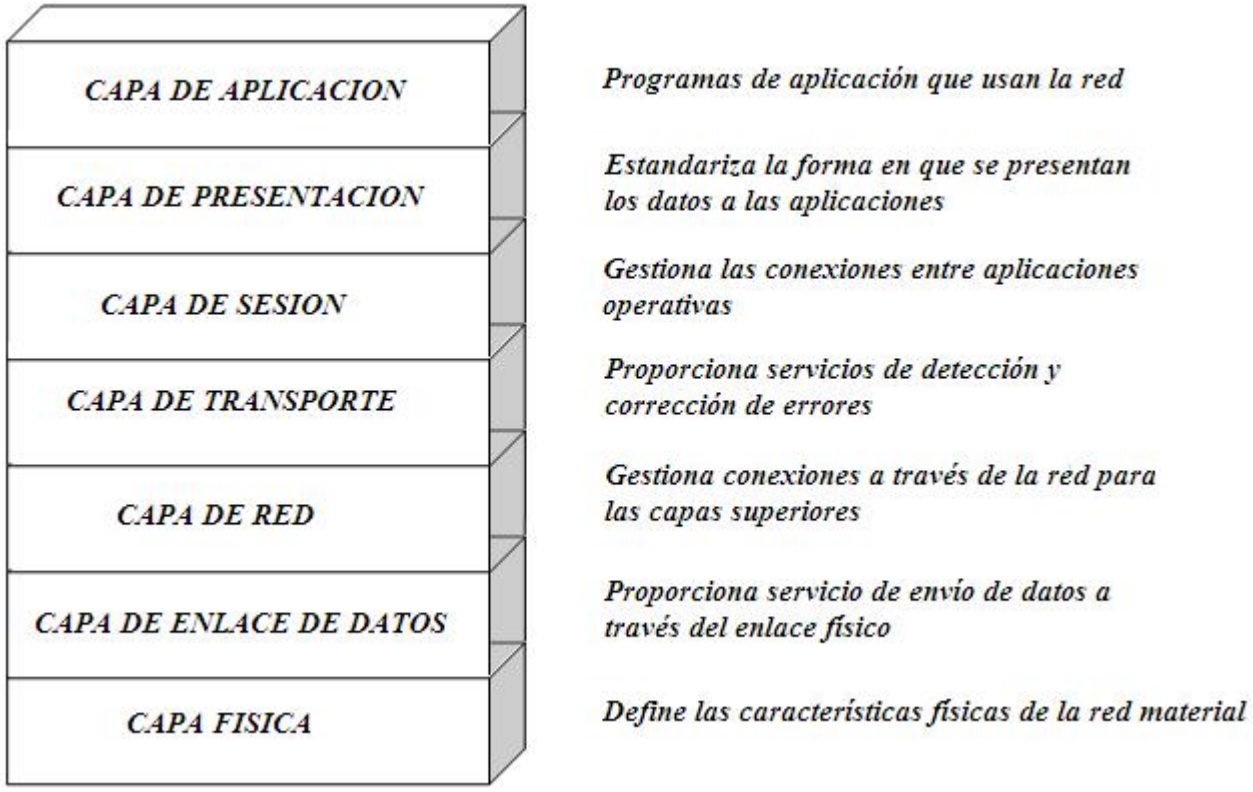


Fig. 1.1: Modelo OSI

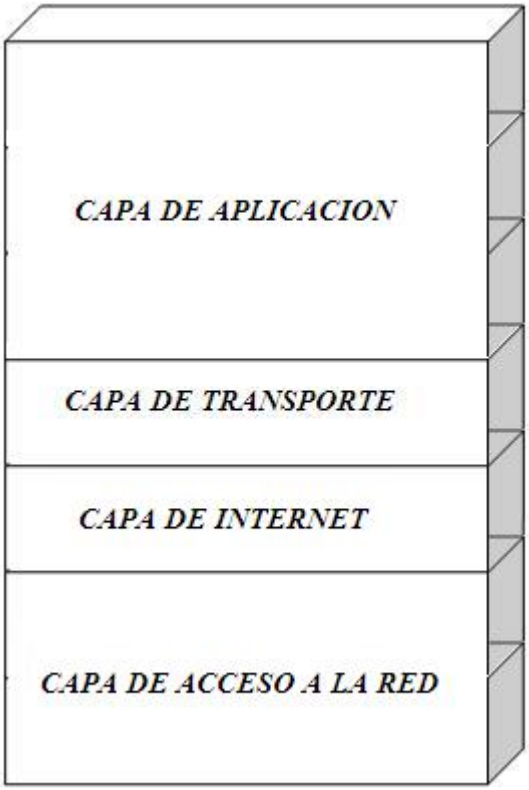


Fig. 1.2: Modelo TCP/IP

TCP/IP se diferencia del modelo OSI en que sólo tiene cuatro capas: una capa de enlace, una capa de red, una capa de transporte y una capa de aplicación, como resultado de la agrupación de diversas capas en una sola o bien por no usar alguna de las capas propuestas en dicho modelo de referencia (Fig. 1.2).

Así, por ejemplo, la capa de presentación desaparece pues las funciones a definir en ellas se incluyen en las propias aplicaciones. Lo mismo sucede con la capa de sesión, cuyas funciones son incorporadas a la capa de transporte en los protocolos TCP/IP. Finalmente la capa de enlace de datos no suele usarse en dicho

paquete de protocolos.

El conjunto TCP/IP está diseñado para enrutar y tiene un grado muy elevado de fiabilidad, es adecuado para redes grandes y medianas, así como en redes empresariales. Se utiliza a nivel mundial para conectarse a Internet y a los servidores web. Es compatible con las herramientas estándar para analizar el funcionamiento de la red.

Un inconveniente de TCP/IP es que es más difícil de configurar y de mantener que NetBEUI o IPX/SPX; además es algo más lento en redes con un volumen de tráfico medio bajo. Sin embargo, puede ser más rápido en redes con un volumen de tráfico grande donde haya que enrutar un gran número de tramas.

El conjunto TCP/IP se utiliza tanto en redes empresariales como por ejemplo en campus universitarios o en complejos empresariales, en donde utilizan muchos enrutadores y conexiones a mainframe o a ordenadores UNIX, como así también en redes pequeñas o domésticas, y hasta en teléfonos móviles y en domótica.

1.2.1. TCP en Internet

En la capa de transporte, el Internet actual es dominado por flujos que utilizan el Protocolo de Control de Transmición (TCP). Desde 1981, cuando TCP fue introducido por primera vez [Postel, 1981], el Internet ha experimentado cambios importantes.

El TCP pertenece al nivel de transporte, siendo el encargado de dividir el mensaje original en paquetes de menor tamaño, y por lo tanto, mucho más manejables.

Los paquetes serán dirigidos a través del protocolo IP de forma individual. El protocolo TCP se encarga además de añadir cierta información necesaria a cada uno de los paquetes. Esta información se añade al inicio de los datos que componen el paquete en forma de cabecera.

Cuando la información se divide en paquetes para ser enviados, el orden en que éstos lleguen a su destino no tiene que ser el correcto. Cada uno de ellos puede llegar en cualquier momento y con cualquier orden, e incluso puede que algunos no lleguen a su destino o lleguen con información errónea. Para evitar todos estos problemas el TCP numera los paquetes antes de ser enviados, de manera que sea posible volver a unirlos en el orden adecuado. Esto permite también solicitar de nuevo el envío de los paquetes individuales que no hayan llegado o que contengan errores, sin que sea necesario volver a enviar el mensaje completo.

Al iniciar, TCP envía un paquete SYN (sincronizador) al host destino. Esto es un pre-requisito para enviar los datos reales. Una contestación del host destino confirma que esta disponible e incluye información de cómo realizar la conexión.

El paquete SYN es respondido con un paquete acknowledgement (ACK - reconocimiento). Los primeros paquetes SYN y ACK contienen información que los dos host utilizaran para numerar los paquetes. Cada paquete en una comunicación TCP es numerado para propósitos de seguimiento. Los hosts se ponen de acuerdo sobre con qué número comenzar a numerar los paquetes. Este número es conocido como número de serie. Una vez que el paquete ACK es recibido, el emisor envía un paquete adicional indicando que ha recibido la contestación y ha sincronizado a un específico numero de serie. En este punto TCP comienza a enviar paquetes

de datos.

Para cada paquete enviado, TCP observa el número de serie e inicia un contador. Si un ACK para ese número de serie no es recibido antes de que el contador expire, los paquetes serán reenviados.

El control de flujo es alcanzado regulando el número de paquetes enviados. Una vez que una ráfaga de paquetes es transmitida, ningún otro paquete es enviado hasta que llegue el paquete ACK. El receptor puede señalarle al emisor si incrementar o decrementar el tamaño de la ventana de paquetes.

1.3. Congestión

Cuando se transmiten o envían datos a través de Internet, un emisor particular puede perder parte de esos datos. Tales pérdidas ocurren en muchos casos debido a un suceso llamado congestión. Desde mediados de los años noventas se han buscado soluciones cada vez más eficientes al problema de congestión. Este análisis, gracias a la configuración punto a punto puede reducirse al estudio de tres elementos fundamentales: la fuente, el destino y el router [Saltzer et al., 1981]. Los dos primeros componentes conforman las entidades que intercambian datos. Sin conocer el estado de la red, el emisor envía información al receptor. Esta transmisión de información emisor-receptor se lleva a cabo mediante el uso del protocolo de comunicación TCP/IP. Mientras que “el router” representa un elemento fundamental para la existencia de Internet, ya que es el ente encargado de evitar que

ocurran un gran número de congestiones y permite la conexión entre diferentes redes.

La congestión ocurre cuando en un router los paquetes que entran llegan con una tasa mayor a la que el router tiene para cambiarlos o remitirlos a una conexión saliente. La congestión significa que el router tiene a su lado muchos paquetes en la cola, las cuales se almacenan en el espacio del almacenador intermediario o buffer, y es ahí donde necesariamente comienzan a caer los paquetes. Por lo tanto, las caídas de los paquetes son una forma de notificación implícita de la congestion que se está formando.

En Internet los paquetes pueden pasar por routers que esten siendo utilizados por múltiples usuarios a la vez, por lo que muchas veces tienen que esperar en la cola (buffer) del servidor antes de ser atendidos. Esto se conoce como líneas de espera. La idea general de una línea de espera es la siguiente: a lo largo del tiempo se producen llegadas de clientes a la cola de un sistema desde una determinada fuente, demandando un servicio, en general las tasas de llegada y de servicio no se conocen con certeza, sino que son de naturaleza aleatoria, es decir, los tiempos de llegada y de servicio deben describirse a través de distribuciones de probabilidad. Una vez que los clientes están esperando a ser atendidos, los servidores del sistema seleccionan miembros de la cola según una regla predefinida denominada disciplina de la cola. Cuando un cliente seleccionado termina de ser atendido, este abandona el sistema.

El principal problema es que, al ser eventos de naturaleza aleatoria, no se sabe con exactitud cuantos clientes y en que momento llegarán a solicitar un servicio.

Si se tiene demasiada capacidad de servicio para operar el sistema esto podría implicar la subutilización de los recursos, y por ende costos excesivos innecesarios. Sin embargo, el no contar con suficiente capacidad de servicio con lleva a grandes problemas tales como tiempos de espera largos, retardos excesivos e incluso pérdida de clientes.

Debido a esto, la pérdida de información en Internet es muy común, pues cada uno de los usuarios conectados a la red, transmiten paquetes de datos sin saber el estado que guarda la red, por lo que en muchas ocasiones los servidores son saturados, y en consecuencia, los paquetes que pasen por esos servidores serán desechados. Para solucionar el problema de la pérdida de la información, el receptor envía una señal de confirmación al emisor a manera de retroalimentación, por cada paquete que recibe satisfactoriamente, con lo cual el emisor puede detectar (después de un retardo de tiempo) qué paquetes se han perdido y por consecuencia debe retransmitir. El tiempo promedio que tarda un paquete en llegar a su destino y en avisar al emisor que fue recibido correctamente, se conoce como Tiempo de Viaje Redondo (RTT). De manera general, la información perdida es retransmitida desde el emisor sin que muchas veces el usuario se de cuenta de ello (Fig. 1.3).

1.4. Calidad de Servicio

Se define la calidad de servicio (CdS o QoS) como la capacidad que tiene un sistema de asegurar, con un grado de fiabilidad preestablecido, que se cumplan los

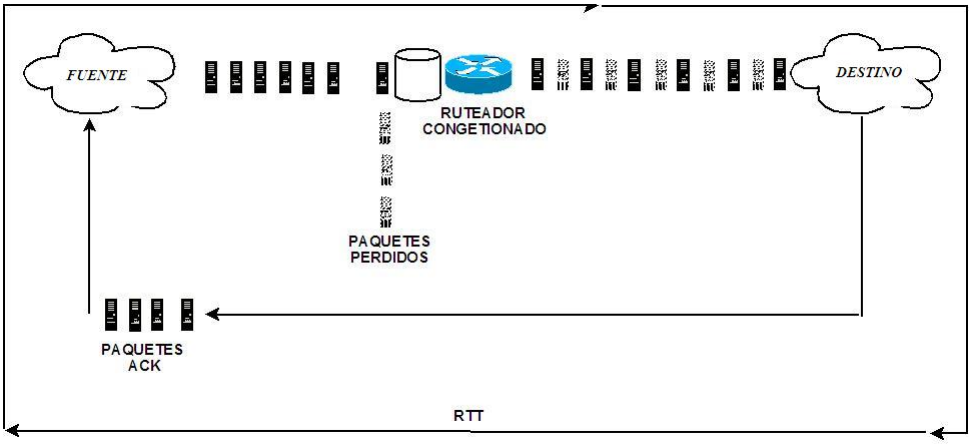


Fig. 1.3: Pérdida de Paquetes y RTT

requisitos de tráfico para un flujo de información dado.

Parámetros de Calidad de Servicio

- **El retardo:** En TCP, cuanto mayor es el retardo, más grande se hace y llega a no haber servicio. En UDP, el aumento de retardo hace que llegue a ser imposible la comunicación.
- **Variación del retardo de transmisión (jitter):** Es la fluctuación del retardo de tránsito entre extremos. TCP hace que si aumenta mucho la variación de retardo, las estimaciones se hagan conservadoras y disminuya mucho el rendimiento. En UDP puede llegar incluso a distorsionarse la señal en el destino.
- **Ancho de banda:** Es la máxima velocidad de transferencia de datos entre extremos de la red.

- **Fiabilidad:** Es la tasa media de error de la red. TCP corrige este problema con retransmisiones. En UDP, como no hay retransmisiones, la señal llega distorsionada (ya que UDP se encarga de transmisiones en tiempo real).

Entonces, el desarrollo de nuevas estrategias para evitar la congestión juega un papel muy importante en tanto incrementa la demanda para el desempeño en aplicaciones en Internet. Tales aplicaciones incluye voz sobre IP (VoIP), clases de servicio (CoS) y video.

1.4.1. Objetivos

El objetivo de éste trabajo es diseñar un controlador difuso para un router considerando los retardos de tiempo ocasionados por el efecto de ruteo en una red bajo el esquema AQM (Active Queue Management). Se controlará la conducta de la cola en presencia de retardos variables y más aún, ante cambios en la elección de las rutas.

1.4.2. Metas

Históricamente, el propósito primario del ruteo IP ha sido mantener la conectividad en presencia de cambios de topología y fallas en la red. El ruteo IP típicamente elige la ruta más corta al destino, basado en simples métricas como por el número de saltos o distancia. Mientras que la simplicidad de este enfoque ha hecho al ruteo IP altamente escalable, ha habido un gran deseo por mejorar la fiabilidad y el desempeño del Internet mediante el uso de técnicas de ruteo que son más sencibles a la congestión [Yilmaz and Matta, 2002].

En la siguiente investigación se pretende establecer un algortimo de gestión activa de cola (AQM-Active Queue Management), donde se involucra el retardo variable acumulativo en la ruta hacia el receptor; se establece la ruta donde el retardo sea mínimo y se fija esa ruta para el envío de información, logrando así una justa utilización de los enlaces, solucionar el problema de congestión en internet y minimizar el retardo en la red.

1.4.3. Alcances

Proponemos diseñar un controlador difuso para resolver el problema de congestión en los routers tomando el retardo en la ruta mínima, establecida mediante el algoritmo de Dijkstra, por ejemplo. El protocolo de ruteo elige, mediante éste algoritmo, las diferentes rutas por las que se enviarán los paquetes, lo cual lleva a retardos variables. El retardo se obtiene mediante algún método como el mencionado en el capítulo 4 para establecer los costes en los enlaces y se incorpora al modelo propuesto en [Misra V. and D., 2000] con la suposición de que el retardo en la cola es mucho menor que el retardo de propagación, lo cual nos lleva a un modelo simplificado [Melchor and Castillo, 2007]. El controlador difuso presenta robustez y muchas otras ventajas como se verá en el siguiente capítulo.

2. ANTECEDENTES

Recientemente ha habido un interés en el análisis del comportamiento dinámico del control de congestión en Internet; en este capítulo se presentan los trabajos revisados en la literatura donde abordan el problema de congestión de Internet mediante la teoría de control proponiendo protocolos de control de congestión, enfocándose sobre redes TCP. Se presenta la estrategia AQM que coopera con el control de congestión de TCP en las fuentes, para compartir equitativamente los recursos de la red, trabajando solamente con un elemento de la red, "el router". Así también se presenta la teoría sobre el control difuso, que se basa en conceptos de teoría difusa, en reglas del tipo si-entonces y en métodos de inferencia difusa; también se proporciona en este capítulo información sobre los algoritmos de ruteo.

2.1. Revisión Bibliográfica

Con el decremento de precio de los equipos, incremento de la velocidad, mayor amplitud de uso, el sin número de software y aplicaciones, y una infraestructura bien establecida, las redes tienen mayor aplicación en la industria para

aplicaciones de control [Kaplan, 2001]. Las aplicaciones de control pueden utilizar el Internet para telecontrol a larga distancia sin invertir en infraestructura.

En el campo de control de congestion en la red [Low et al., 2002], [Srikant, 2004], una línea de investigación de mucho interés es la dinámica de los protocolos de control de congestión. La estabilidad es decisiva para asegurar que el punto de operación del sistema es de hecho el punto de equilibrio previsto, con la eficiencia y equidad deseada.

Gracias a la configuración punto a punto el estudio de la congestión puede reducirse al estudio de tres elementos fundamentales: "fuente, destino y router" [Saltzer et al., 1981].

Como se mencionó anteriormente, los mecanismos utilizados en los routers es el manejo activo de colas (AQM) que gerencia el tamaño de la cola mediante la eliminación de paquetes [Hassan and Jain, 2004]. La ventaja de los algoritmos AQM es que su implementación solamente involucra a un elemento de la red, el router, no siendo necesario cambiar los demás componentes para su utilización.

Muchos esquemas AQM han sido estudiados en trabajos recientes [Long C. and Yang, 2005]. En la Tabla 2.1 se presentan estos esquemas.

Así, AQM coopera con el control de congestión TCP en las fuentes, para limitar las pérdidas de paquetes y compartir equitativamente los recursos de la red. Anticipa la congestión en vez de esperar hasta que la cola del router se desborde [Hassan and Jain, 2004]. La detección aleatoria anticipada (RED) [Floyd and

Jacobson, 1993], es un algoritmo AQM que ha sido ampliamente utilizado en la industria. Desecha paquetes con una probabilidad que es una función del promedio del tamaño de la cola, siempre que el promedio del tamaño de la cola medido exceda un umbral prestablecido.

Jacobson [V., 1998], propuso mecanismos de control de flujo anticipando que solo los routers manejaban la información suficiente para controlar la distribución de la capacidad de la red entre todas las conexiones debido a que en ellos convergen los flujos y pueden observar la red de forma completa.

Como una variación de TCP, anuncio de ventana generalizada (GWA-generalized window advertising) [Gerla et al., 2002] usa una técnica "ventana receptora" para controlar la salida del remitente TCP, solo que éste router GWA-TCP utiliza un encolamiento por espera costoso (expensive per-flow queuing), y requiere un largo bufer para alcanzar alta utilización de la red [Gerla et al., 2000].

Recientemente se han desarrollado trabajos significantes en el entendimiento teórico del control de congestión en redes. Comenzando en 1998 por Kelly [Kelly et al., 1998] que modela la medida de congestión en las fuentes y plantea el control de flujo de la red como un problema de optimización, y muestra que el problema de control de la tasa de transmisión puede ser resuelto de manera decentralizada [Low and Lapsey, 1999].

Otro de los logros en el área de control de congestión es el desarrollo de modelos matemáticos para el control de flujo [Kelly, 2001], [Hollot et al., 2002], [Jacobsson et al., 2006], [Deb and Srikant, 2006]. En [Tang et al., 2007] se presenta la importancia de tener modelos exactos para análisis y diseño.

Las actividades en el area de Sistemas Interconectados se pueden clasificar en:

- Control de redes
- Control sobre redes
- Sistemas muti-agentes

En Control en Redes se deben distribuir pequeños, incontables, pero frecuentes paquetes entre un relativamente gran conjunto de nodos para cumplir con los requerimientos de tiempo crítico. Un elemento clave que distingue al control en red de datos es la capacidad de soportar aplicaciones de tiempo real y tiempo crítico [li Lian et al., 1999].

Las métricas del rendimiento de sistemas en red que impactan los requerimientos de sistemas de control incluyen retardo en el acceso, tiempo de transmisión, tiempo de respuesta, retardo del mensaje, colisiones de mensajes (porcentaje de colisión), transferencia de mensajes (porcentaje de paquetes descartados), tamaño del paquete, utilización de la red y límites del determinismo. Para sistemas de control, las redes de control candidatas deben cumplir dos criterios principales: retardo de tiempo acotado y transmisión garantizada; esto es, un mensaje debe ser transmitido exitosamente dentro de un retardo de tiempo acotado.

En Sistemas de Control en Red (NCS, Networked Control Systems), los mensajes transmitidos sin éxito o con retardos largos de un sensor a un actuador, por ejemplo, puede deteriorar el desempeño del sistema o hacerlo inestable. Muchos

protocolos han sido propuestos para cumplir estos requerimientos para sistemas de control (Ethernet, Token bus, Token ring, CAN).

El cambio en la arquitectura de comunicación de punto-punto a bus común, introduce formas diferentes de incertidumbre del retardo entre sensor, actuador y controladores. Estos retardos de tiempo vienen del tiempo de distribución del medio de comunicación así como el tiempo extra requerido para codificar las señales físicas y el proceso de la comunicación. El retardo puede ser constante, acotado o aún aleatorio, dependiendo del protocolo de red adoptado y el hardware elegido.

Es ampliamente conocido que los retardos degradan el rendimiento de un sistema de control. Los retardos en la red pueden no afectar de forma significativa un sistema de control en lazo abierto tal como un sistema on-off en plantas industriales. Sin embargo, la configuración de control en lazo abierto puede no ser apropiada y adecuada para sistemas de control de alta sensibilidad al tiempo tales como telerobótica y telecirujía. Estas aplicaciones requieren retroalimentación de datos enviados a través de la red para corregir el error de salida. Las metodologías existentes para retardos constantes pueden no ser directamente convenientes para controlar un sistema sobre la red puesto que los retardos de la red usualmente son variables, especialmente en el Internet. Por lo tanto, para manejar los retardos de la red en un sistema de control en lazo cerrado sobre una red, se requiere una metodología avanzada.

2.1.1. Configuración NCS

Hay dos configuraciones NCS en general:

- *Estructura Directa*. Está compuesta por un controlador y un sistema remoto que contiene una planta física, sensores y actuadores. El controlador y la planta están físicamente localizados en diferentes lugares y están directamente enlazados mediante una red de datos para el funcionamiento de control remoto en lazo cerrado (Fig. 2.1).

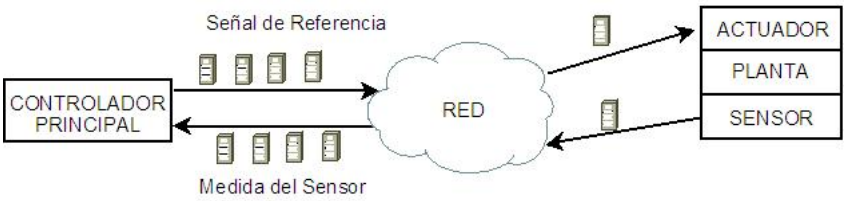


Fig. 2.1: NCS en Estructura Directa

La señal de control es encapsulada en un paquete y enviada a la planta por la red. Entonces, la planta regresa la salida del sistema al controlador poniendo la medida del sensor en un paquete también. En una implementación práctica, multiples controladores pueden ser implementados en un solo hardware para manejar múltiples NCS en estructura directa.

- *Estructura Jerárquica*. La base de la estructura jerárquica consiste de un controlador principal y un sistema remoto en lazo cerrado (Fig. 2.2).

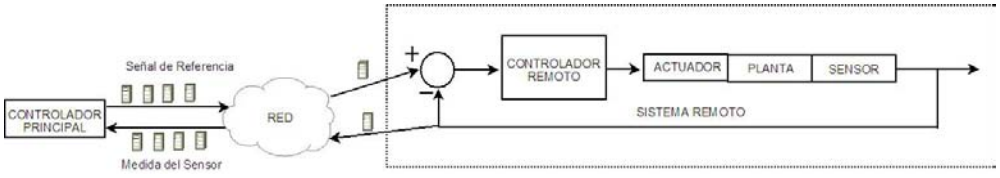


Fig. 2.2: NCS en Estructura Jerárquica

Periódicamente el controlador principal calcula y envía la señal de referencia en un paquete por la red al sistema remoto. Entonces el sistema remoto procesa la señal de referencia para el control local del sistema en lazo cerrado y regresa la medida del sensor al controlador principal para el control en lazo cerrado en red. El lazo de control en red usualmente tiene un mayor período de muestreo que el lazo de control local puesto que el controlador remoto debe satisfacer la señal de referencia antes de procesar la nueva señal de referencia que llega. Similar a la estructura directa, el controlador principal puede ser implementado para manejar multiples lazos de control en red para muchos sistemas remotos.

El uso de una u otra estructura depende de los requerimientos de la aplicación y preferencias del diseñador.

Entonces un NCS que opera sobre una red, la transferencia de datos entre controlador y el sistema remoto inducirá un retardo de red en adición al retardo del procesamiento del controlador. En la Fig. 2.3 se muestran los retardos en la red en el lazo de control, donde r es la señal de referencia, u es la señal de control, y es la señal de salida, k es el índice de tiempo y T es el período de muestreo.

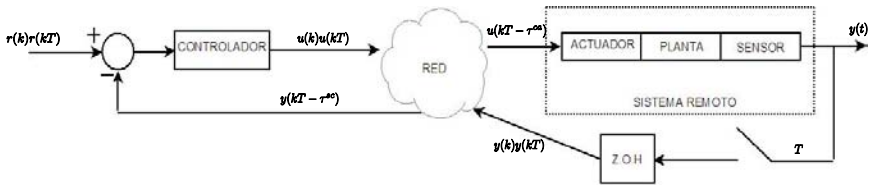


Fig. 2.3: Configuración General de NCS y Retardos en la Red

La Fig. 2.4 muestra el diagrama de tiempo correspondiente de los retardos de propagación en la red.

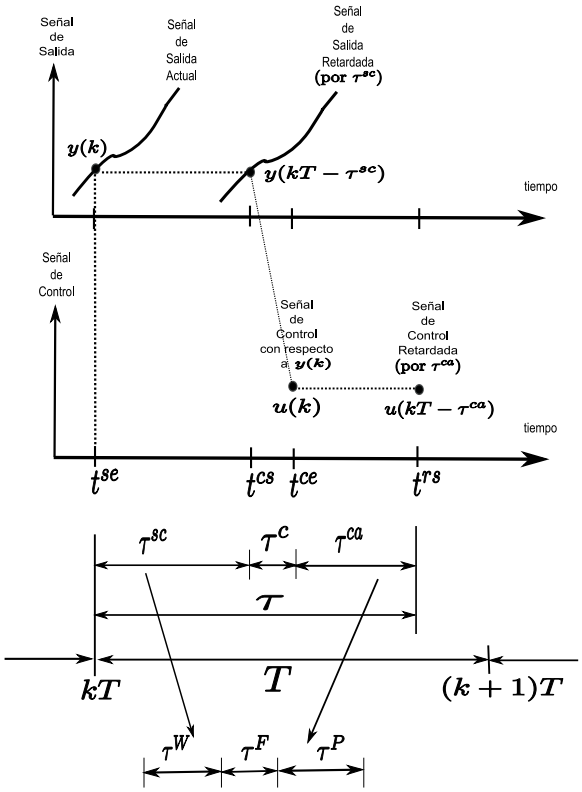


Fig. 2.4: Diagrama de Tiempo del Retardo de Propagación en la Red

Los retardos de la red en un NCS se categorizan de la dirección de transferencias de datos como el retardo del sensor-al-controlador τ^{sc} y el retardo del controlador-al-actuador τ^{ca}. El retardo se calcula:

$$\tau^{sc} = \tau^{cs} - \tau^{se}, \tau^{ca} = \tau^{rs} - \tau^{ce}$$

donde τ^{se} es el instante de tiempo en que el sistema remoto encapsula la medición en un paquete para enviarlo, τ^{cs} es el instante de tiempo en que el controlador comienza a procesar la medición en el paquete entregado, τ^{ce} es el instante de tiempo en que el controlador principal encapsula la señal de control en un paquete para ser enviado, y τ^{rs} es el instante de tiempo en que el sistema remoto comienza a procesar la señal de control.

De hecho, ambos retardos en la red pueden ser más largos o más cortos que el período de muestreo T. El retardo de procesamiento del controlador τ^c y ambos retardos pueden ser agrupados como el retardo de control para facilitar el análisis. El retardo τ^c usualmente es pequeño comparado al de la red y puede ser despreciado.

Los retardos τ^{sc} y τ^{ca} se componen de las siguientes partes [F.L. Lian and Tilbury, 2001]:

- *Retardo de Espera (τ^W)*. Es el tiempo que debe esperar una fuente (controlador principal o sistema remoto) en cola y ser despachado.
- *Retardo de empaquetamiento (τ^F)*. Es el tiempo que le lleva a la fuente colocar un paquete en la red.
- *Retardo de Propagación (τ^P)*. Es el retardo que presenta el paquete en viajar a través de un medio físico. El retardo de propagación depende de la velocidad de la transmisión de la señal y la distancia entre la fuente y destino.

Estas tres partes del retardo son retardos fundamentales que ocurren en una red de área local. Cuando los datos de control o sensado viajan a través de la red,

puede haber retardos adicionales tal como el retardo de cola en un switch o un router, y el retardo de propagación entre los nodos de la red. Los retardos τ^{sc} y τ^{ca} también dependen de otros factores tales como máximo ancho de banda de las especificaciones del protocolo y el tamaño del paquete.

Los protocolos de red de capas superiores tal como TCP requieren retransmisión si ocurre un error en un paquete, un router descarta el paquete, como se mencionó en la sección anterior. Esto es una compensación para un NCS. Aunque algunas señales de control o sensadas se pierden debido a la transmisión en la red, algunas NCS operan aceptablemente. En éste caso la retransmisión puede no ser deseable porque la NCS puede ser severamente afectada por la extensión del retardo como resultado de la retransmisión.

Características del Retardo

Las características del retardo en NCS básicamente dependen del tipo de red utilizado, descritos a continuación.

Red de Servicio Cíclico. En los protocolos de red de área local con servicio cíclico tal como IEE8024, SAE token bus, PROFIBUS, IEEE802.5, SAE token ring, MILSTD-1553B y FIP, las señales de control y sensadas son transmitidas en un orden cíclico con comportamiento determinístico. Así, los retardos son periódicos y pueden ser modelados simplemente como una función periódica tal como $\tau_k^{sc} = \tau_{k+1}^{sc}$ y $\tau_k^{ca} = \tau_{k+1}^{ca}$, donde τ_k^{sc} y τ_k^{ca} son los retardos de sensor-a-actuador y controlador-a-actuador en el periódo de muestreo k [Halevi and Ray, 1988]. Los

modelos trabajan perfectamente en el caso ideal. En la práctica, NCS puede experimentar variaciones pequeñas en retardos periódicos debido a muchas razones. Por ejemplo, la discrepancia en los generadores de reloj en el controlador y un sistema remoto provoca variaciones del retardo.

Red de Acceso Aleatorio. Las redes de acceso aleatorio tales como CAN y Ethernet implican más retardos inciertos. Las partes significativas de los retardos aleatorios en la red son el retardo de espera debido a la cola y las colisiones en la red. Cuando un NCS opera a través de la red, muchos factores pueden incrementar la aleatoriedad en los retardos de la red tal como el retardo de cola en un router y el retardo de propagación de diferentes rutas. Además, un servicio cíclico conectado a un acceso aleatorio también da lugar a retardos aleatorios.

En el área de redes, los retardos aleatorios han sido modelados mediante formulaciones basadas en probabilidad y características de la fuente y destino. La gama de enfoques va desde simples como procesos de Poisson a enfoques más sofisticados como Cadenas de Markov [S. Shakkottai and Anvekar, 2001], modelo de flujo de fluidos [Filipiak, 1988], modelo ARMA (Li), etc. Estas técnicas se han traido a las formulaciones de NCS en varios estudios, pero pueden tener que ser modificados o reformulados para metodologías de control en redes específicas.

Las investigaciones en NCS son diferentes a las de los sistemas con retardo tradicional, donde los retardos se asumen constantes o acotados. De la variabilidad del retardo inducido en la red, los NCS pueden ser sistemas variantes en el tiempo que hace al análisis y diseño más complicado. Modelos de NCS con retardos son presentados en [Lian et al., 2001].

En este trabajo nos concentraremos en el Control de Redes que se encarga principalmente de proveer un cierto nivel de rendimiento al flujo de datos en la red mientras se alcanza una eficiente y justa utilización de los recursos de la red. En esta área los principales problemas de interés son admisión de llamada, planeación, ruteo, control de flujo, control de energía y otros problemas de asignación de recursos [Zampieri, 2008].

En [Misra V. and D., 2000] modelan el comportamiento de TCP/AQM como un modelo de flujo de fluidos para describir el comportamiento del TCP soportando flujos AQM en routers congestionados. En [Hollot et al., 2001] proponen el algortimo PI-AQM, linealizan el modelo de [Misra V. and D., 2000] y hacen el análizis considerando un esquema de control realimentado con RED (Ramdom Early Detection), un esquema AQM muy popular actualmente, y muestran los problemas de sintonización de los parámetros de RED; también, demuestran las condiciones suficientes para estabilidad L_2 con entradas y salidas acotadas. En [Hollot et al., 2002] proponen controladores Proporcional (P) y Proporcional-Integral (PI) como estrategias AQM. En [Quet and Ozbay, 2004] tomando el trabajo linealizado en [Hollot et al., 2001], proponen controladores para la optimización de una función de costo H_∞ mejorando el desempeño de RED.

En [Michiels et al., 2006] analizan la estabilidad del modelo linealizado en [Hollot et al., 2001] proponiendo un controlador Proporcional como estrategia AQM. Demuestran las condiciones de necesidad y suficiencia para estabilidad asintótica que se obtienen con herramientas del dominio de la frecuencia para sistemas con retardo.

En [Carrero, 2004] se propone una serie de reglas de ajuste de los parámetros

del PI-AQM, llamadas reglas CAM. Se ajustan las ganancias de los parámetros del controlador para mantener el tamaño de la cola en un valor deseado y minimizar oscilaciones.

En [Melchor and Castillo, 2007] se establecen condiciones necesarias y suficientes para estabilidad asintótica del sistema linealizado en lazo cerrado, encontrando el conjunto completo de controladores PI que estabilizan localmente el punto de equilibrio.

En la revisión bibliográfica no se encontraron para el modelo en [Misra V. and D., 2000], trabajos que toman al retardo en la red variable, tampoco con controladores difusos que son ampliamente utilizados en la industria, debido a que cada vez se presentan procesos más complejos y desafiantes con características no lineales, así también por su buen manejo de incertibumbre y ambigüedad. En este trabajo se abordan ambos temas pensando en las bondades que ofrece la lógica difusa.

2.2. Control Difuso

La lógica difusa (heurística) ha sido aplicada en distintas situaciones: control de procesos complejos [L.A., 1973], modelación de procesos industriales [Takagi and Sugeno, 1985], diagnóstico de fallos [Yasunobu and Miyamoto, 1985], programación matemática, procesamiento de imágenes, reconocimientos de patrones [Mamdani, 1975], etc.

Categoría	**AQMs**	**Método de diseño**	**Ventajas/Desventajas**
Uso del tamaño de la cola	RED	Heurístico; tamaño promedio de la cola; controlador pseudo proporcional	Elimina problemas de sincronización global; respuesta lenta
	ARED	Modifica p_{max} de forma adaptable basado en cola promedio.	Soluciona uno de los problemas de configuración de parámetros; "avg" resulta en un efecto dominó de retraso (*Lang domino effect*).
	SRED	Estima el número de fuentes activas sin los estados de cada una de las fuentes.	Minimza la fluctuación de la cola instantánea; poca confiabilidad en la exactitud de la estimación.
	DRED	Heurístico; EWMA del error de la cola instantánea; control integral.	Estabiliza la cola instantánea en valor deseado; problemas para elegir la ganancia del control.
	PI	Método clásico de teoría de control; error de la cola instantánea.	Mejora sensibilidad; elimina error en estado estable; desempeño local debido a linealización.
Uso de la cola y de la carga	REM	Error de la cola instantánea y proporción desigual; método de optimización.	Desacopla índice de congestión e índice de rendimiento; estabiliza la proporción de entrada alrededor de la capacidad de enlace como la cola alrededor de un objetivo pequeño; calcula peso; parámetro de sensibilidad.
	VRC	Controlador PID; método de análisis de estabilidad	Mejora desempeño de la sensibilidad; desempeño local de linealización.
Uso de la carga	Blue	Heurístico; control on-off modificado.	Respuesta rápida; retardos largos y fluctuantes en la formación de colas bajo tráfico dinámico.
	AVQ	Método de optimización; modelo *token bucket* modificado.	Respuesta rápida; retardos pequeños en colas; los retardos de las colas se incrementan con el crecimiento de la congestión.
	Yellow	Utilidades principales/secundarias; análisis de teoría de control.	Respuesta rápida; estabilidad de cola; pequeñas fluctuaciones del retardo en las colas.

Tab. 2.1: Esquemas AQM

Las ventajas de la lógica difusa son en parte sus desventajas. Dado que no está basado en un modelo, es difícil desarrollar un método que pueda probar su estabilidad. A pesar de eso, la lógica difusa puede resultar ventajosamente comparada con otras aproximaciones de control inteligente, y el control PID puede ser utilizado en conjunción con sistemas de control difuso para asegurar su estabilidad.

El propósito de un controlador es alcanzar o mantener un proceso en un estado determinado, mediante la monitorización de un conjunto de variables y la selección de las acciones de control adecuadas.

El problema del diseño de sistemas de control se suele reducir a la selección de un determinado tipo de controlador y, al ajuste de sus parámetros, de tal forma que se verifiquen ciertas especificaciones dadas para el proceso a controlar.

Un sistema de control realimentado es aquél que tiende a mantener una relación preestablecida entre la salida del sistema y y la entrada de referencia r, comparando ambas y utilizando la diferencia como parámetro de control. Los elementos del proceso a controlar constituyen el sistema controlado. El controlador es el encargado de mantener el punto de referencia de la salida ante posibles perturbaciones externas o cambios en la referencia, mediante su acción u (Figura 2.5).

Un controlador difuso es un sistema de control basado en información imprecisa, que utiliza la teoría de conjuntos y la lógica borrosa para la representación e inferencia de ese conocimiento. Se basa en la descripción lingüística de la es-

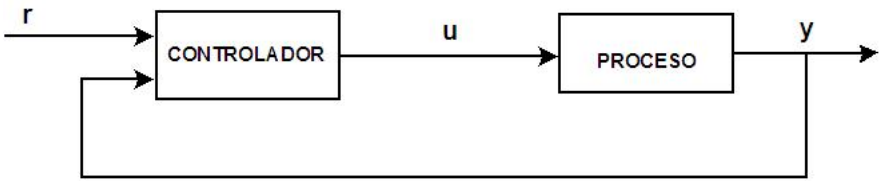


Fig. 2.5: Diagrama de un Sistema de Control

trategia de control que utilizaría un operario o experto en el control manual del proceso. Por esta razón también se denominan "Controladores Lingüísticos".

Un controlador difuso se compone internamente de una base o conjunto de reglas lingüísticas de control, que tienen como antecedentes los valores posibles de las variables de entrada, y que concluyen la acción de control a efectuar, en términos también lingüísticos.

2.2.1. Estructura Genérica de un Controlador Difuso

La configuración básica de un controlador difuso se indica en la Figura 2.6. Se distinguen los siguientes módulos básicos:

- **Difusificación**. Es la etapa inicial del controlador difuso. Transforma las variables controladas entregadas por sensores del proceso, en variables del tipo lingüística que conforman las particiones definidas en el universo de discusión, nombre con que se le conoce al rango de variación de las variables del tipo determinísticas.

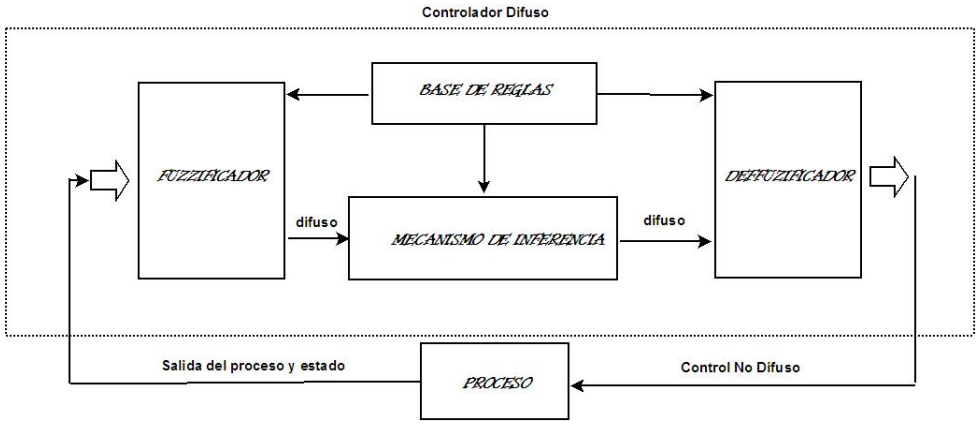


Fig. 2.6: Esquema General del Control Difuso

Como resultado de la difusificación se obtienen valores de pertenencia de los conjuntos difusos para los valores medidos de variación del universo de discusión. La transformación de un conjunto clásico a un conjunto difuso se produce mediante el uso del operador de fusificación $\mathcal{F}$ [Passino and Yurkovich, 1998].

- **Base de conocimientos**. Está formado de dos componentes básicos: *la base de datos y la base de reglas de control difuso*. Los conceptos asociados con la base de datos son usados para caracterizar las reglas de control difuso, y la base de reglas permite la toma de acciones de control. Estos conceptos se definen subjetivamente con base en, por ejemplo, operarios expertos en el proceso. Las reglas de control consisten en estructuras de la forma:

  If (condiciones) ***Then*** (acciones).

  La cláusula ***If***, un antecedente, es una condición en el dominio de aplicación; la cláusula ***Then***, una consecuencia, es una acción de control dado al

proceso bajo control.

Con un conjunto de reglas difusas, el mecanismo de inferencia difusa es capaz de derivar una acción de control para un conjunto de valores de entrada. En otras palabras, una acción de control es determinada por las entradas observadas, las cuales representan el estado del proceso a ser controlado mediante el uso de las reglas de control.

- **Mecanismo o motor de inferencia**. Procedimientos destinados a la toma de acciones de control difuso, por medio del uso de la implicancia y reglas lingüísticas de control. Se toma como patrón a seguir el conjunto de reglas de la base de conocimientos.

  Se calcula el grado a el cual cada regla "*dispara*" a una entrada difusificada considerando los conjuntos de regla y etiqueta. Se dice que una regla dispara cuando las condiciones sobre la cuál depende ocurren. Puesto que estas condiciones estan definidas mediante conjuntos difusos que tienen grados de membresía, una regla tendrá un grado de disparo o fuerza de disparo β_j. Según [Abonyi, 2003] el producto de los grados de membresía puede ser obtenido mediante:

$$\beta_j = \prod_{i=1}^{n} A_{i,j} \tag{2.1}$$

$$\tag{2.2}$$

  $A_{i,j}$ es la función de membresía sobre la entrada i usada en la regla j.

  Por supuesto, existen diferentes métodos, ver [D. Driankov and Reinfrank, 1993].

- **Desfusificación**. El objetivo es la transformación de las acciones de control de tipo difuso a acciones de control de tipo cuantitativo o determinístico, que permitan un adecuado funcionamiento del o los actuadores del sistema bajo control.

  Existen muchos métodos para desdifusificar [D. Driankov and Reinfrank, 1993], por ejemplo: método de centro de gravedad, centro de sumas, primer máximo, último máximo, media de máximos,etc. Uno muy utilizado es el del centro de gravedad o método del centroide:

$$y = \frac{\sum_{j=1}^{R} \beta_j f_j(x)}{\sum_{j=1}^{R} \beta_j} \tag{2.3}$$

  Donde:
  R son las reglas, β_j el j-ésimo conjunto y f_j la j-ésima función lineal.

2.3. Protocolos de Ruteo

Como se mencionó en el capítulo 1 para el objetivo de este trabajo necesitamos conocer el retardo en las rutas desde una fuente a un destino y establecer la ruta mínima. Esta tarea esta a cargo de los protocolos de ruteo.

El algoritmo de enrutamiento es la parte del software de la capa de red encargada de decidir la línea de salida por la que se transmitirá un paquete de entrada.

Los routers son capaces de rutear dinámicamente [Trick, 1998], es decir, son capaces de seleccionar el camino que debe seguir un paquete en el momento en el que les llega, teniendo en cuenta factores como líneas más rápidas, líneas más baratas, líneas menos saturadas, etc.

Los routers son más sofisticados que los switches, sin embargo, esto los hace más caros. A diferencia de los switches y bridges, que sólo leen la dirección MAC (Medium Acces Control), los routers analizan la información contenida en un paquete de red leyendo la dirección de red. Los routers tienen la peculiaridad de que pueden almacenar datos en unas tablas, estas tablas se conocen como **tablas de ruteo**.

En general en esta tabla se pueden encontrar tipos de rutas:

- *Rutas directas*, para redes conectadas localmente.
- *Rutas indirectas*, para redes accesibles a través de uno o más routres.

Los routers leen cada paquete y lo envían a través del camino más eficiente posible al destino apropiado, según una serie de reglas recogidas en sus tablas. Los routers se utilizan a menudo para conectar redes geográficamente separadas. El router es entonces la conexión vital entre una red y el resto de las redes. Un router también sabe cuándo mantener el tráfico de la red local dentro de ésta y cuándo conectarlo con otras LAN (*Local Area Networks*), es decir, permite filtrar los broadcasts de nivel de enlace. Un router dispondrá de una o más interfases de

red local, las que le servirán para conectar múltiples redes locales usando protocolos de nivel de red.

La función principal del nivel de Internet (nivel de Red) es hacer llegar los paquetes de una computadora a otra no importando cual sea el medio físico que utilicen ni los datos que estén transmitiendo; el enrutamiento es justamente eso.

El propósito de un algoritmo de ruteo es simple: dado un grupo de routers con enlaces conectando a los mismos, un algoritmo de ruteo es aquel que busca un "buen" camino desde el router fuente hasta el router destino. Usualmente, un buen camino es aquel que presenta el menor coste de enlace. Sin embargo en la práctica, existen muchas políticas que entran en juego en la toma de decisiones en el ruteo de paquetes y que hacen simple el concepto de algoritmos de ruteo en algoritmos complejos.

Un grafo es usado para formular problemas de ruteo. Un grafo $G = (N, E)$ donde N es un grupo de nodos, y E es una colección de arcos, en donde cada arco contiene un par de nodos contenidos en N. En el contexto de ruteo en redes de comunicaciones, los nodos en una gráfica representan los routers (que son los puntos donde se toma la decisión de la ruta de envío del paquete) y los arcos que conectan a estos nodos representan a los enlaces físicos que conectan a dichos routers [Kurose and Ross, 2005].

En la Figura 2.7 se muestra una representación abstracta de una topología de siete nodos ($x_1, x_2 \ldots$). Como podemos observar en el modelo, para cada arco o enlace se tiene un valor, dicho valor representa el costo o peso del enlace

$(c(x_1, x_2) = 3, c(x_1, x_3) = 3, c(x_2, x_4) = 1, \ldots)$. El peso del enlace puede ser el retardo en la transmisión de nodo a nodo, por ejemplo.

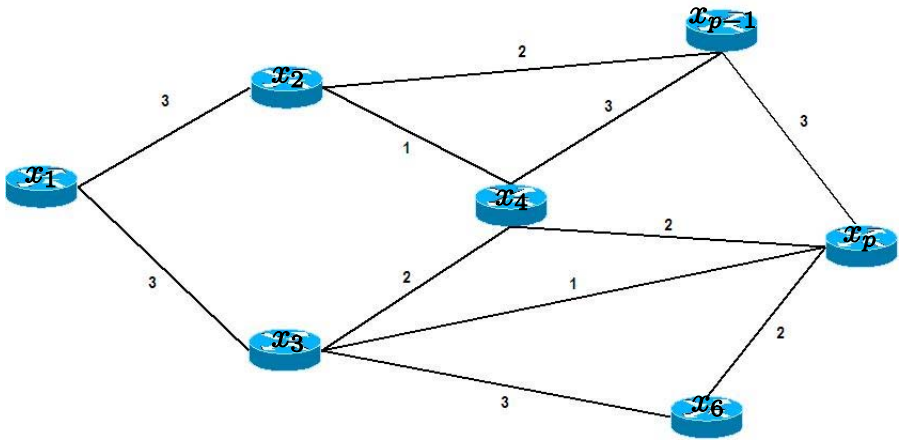


Fig. 2.7: Topología de siete nodos

El principal objetivo de un algoritmo de ruteo es identificar el camino con menor costo entre un nodo inicial (o fuente x_1) y un nodo final (o destino x_p). Definamos que el camino en un grafo $G = (N, E)$ es una secuencia de nodos $(x_1, x_2, x_3, \ldots, x_p)$ tales que para cada par de nodos $(x_1, x_2), (x_1, x_3), (x_{p-1}, x_p)$ pertenezcan a los arcos E. El costo del camino $(x_1, x_2, \ldots, x_p)$ es simplemente la suma de todos los costos de los arcos a lo largo del camino, esto es $c(x_1, x_2) + c(x_1, x_3) + c(x_{p-1}, x_p)$. Dados cualquier par de nodos x y y, usualmente hay muchos caminos entre estos nodos, y cada camino con su respectivo costo asociado [Kurose and Ross, 2005].

Una manera de clasificar a los algortimos de ruteo es de acuerdo a si estos son globales o descentralizados [Kurose and Ross, 2005].

Algoritmo de ruteo global: Un algoritmo de ruteo global calcula el camino con menor costo entre una fuente y un destino utilizando información global de la red de comunicaciones. Esto es, el algoritmo toma la conectividad entre todos los nodos y todos los costos como entradas. Esto requiere que el algoritmo de alguna manera obtenga esta información antes de realizar alguno de los cálculos requeridos. Los mismos cálculos pueden ser corridos en algún sitio (un algoritmo de ruteo global centralizado) o calculado en múltiples sitios. La clave para distinguirlos es que el algoritmo global posee la información completa acerca de la conectividad y de los costos de cada enlace en la red de comunicaciones. En la práctica, los algoritmos de información global son comúnmente referidos como *algoritmos de estado-enlace*, pues el algoritmo debe tener en cuenta el costo de cada enlace en la red de comunicaciones.

Algoritmo de ruteo descentralizado: El cálculo del camino con menor costo es llevado a cabo a base de iteraciones y de un modo distribuido. Ningún nodo en la red tiene información completa de los costos de los enlaces de toda la red de comunicaciones. Por el contrario, cada nodo comienza solamente con el conocimiento de los costos de los enlaces de los nodos que están directamente conectados a él. Entonces, a través de un proceso de iteraciones de cálculo y de intercambio de información con los nodos vecinos (o sea los nodos que están directamente conectados al nodo en cuestión). Un nodo gradualmente calcula el camino con menor coste hacia un destino o un grupo de destinos. Son comúnmente referidos como *algoritmos de vector-distancia*. Son llamados así porque cada nodo mantiene un vector con las estimaciones de los costos hacia todos los demás nodos en la red.

Una segunda manera general de clasificar a los algoritmos de ruteo es de acuerdo a si estos presentan características dinámicas o estáticas. En los algoritmos de ruteo estáticos las rutas cambian muy lentamente con respecto al tiempo, a menudo como resultado de intervención humana (por ejemplo cuando alguien modifica las tablas de ruteo de manera manual). Los algoritmos de ruteo dinámicos cambian el camino según los cambios en el tráfico de la red o de la topología. Un algoritmo dinámico puede ser ejecutado ya sea de manera periódica o de manera directa con respuesta a los cambios de topología o cambio en los costos del enlace. Mientras que los algoritmos de ruteo dinámicos responden más con los cambios en la red, también son más susceptibles a problemas tales como oscilaciones en las rutas, y lazos en las rutas.

Otra manera de clasificar a los algoritmos de ruteo es de acuerdo a si son sensibles a la carga o si son insensibles a la carga. En los algoritmos que son sensibles a la carga, los costos de los enlaces varían dinámicamente para reflejar el nivel actual de congestionamiento de la red. Si un costo alto es asociado con un enlace que actualmente está congestionado, el algoritmo tratará de elegir rutas alternas para evitar pasar por el enlace congestionado.

Hoy en día los algoritmos de ruteo para Internet (tales como RIP, OSPF y BGP) son insensibles a la carga, pues en el pasado se encontró con muchos problemas al implementar los algoritmos sensibles a la carga.

2.3.1. Algoritmo de Dijkstra

En un algoritmo de estado-enlace, la topología de la red y todos los costes de enlaces son conocidos. Esto se puede obtener haciendo que cada nodo realice una transmisión de paquetes estado-enlace hacia todos los demás nodos en la red. Cada paquete estado-enlace contiene las identificaciones y los costos de los enlaces que se encuentran directamente enlazados a ese nodo. El resultado de la retransmisión es que todos los nodos obtienen información idéntica y completa de la red [Kurose and Ross, 2005].

El algoritmo de *Dijkstra* calcula el camino con menor costo desde un nodo fuente hacia todos los demás nodos en la red. El algoritmo de *Dijkstra* trabaja a base de iteraciones y tiene la propiedad de que después de la iteración k, el camino más corto es conocido para k nodos destino.

Definimos:

- $D(v)$: Costo del camino con menor coste desde el nodo fuente al nodo destino v.
- $p(v)$: Nodo previo (vecino v) a lo largo del actual camino con menor costo desde el nodo fuente v.
- N': Subgrupo de nodos; v está en N' si el camino con menor costo desde el nodo fuente es conocido.

El algoritmo de ruteo global consiste de un paso de inicialización seguido por un lazo. El número de veces que el lazo es ejecutado es igual al número de nodos

en la red. Cuando el algoritmo termina éste habrá calculado el camino con menor costo desde el nodo fuente u hacia cada nodo en la red.

Algoritmo estado-enlace para un nodo fuente u [Kurose and Ross, 2005]:

1. **Inicialización**
2. $N' = u$
3. para todo v.
4. Si v es vecino de u
5. entonces $D(v) = c(u, v)$
6. Si no $D(v) = \infty$
7. **Lazo**
8. encuentra w que no este en N' tal que $D(w)$ es el menor
9. añade w a N'
10. actualiza $D(v)$ para cada vecino v de w y que no este en N'
11. $D(v) = \text{mín}[D(v), D(w) + c(w, v)]$
12. /*Nuevo costo de v es o el antiguo costo de v o el costo del camino mas corto conocido a w más el costo desde w a v */
13. Hasta que $N' = N$

N'	**D(v),p(v)**	**D(w),p(w)**	**D(x),p(x)**	**D(y),p(y)**	**D(z),p(z)**
U	2,U	5,U	1,U	∞	∞
UX	2,U	4,X		2,X	∞
UXY	2,U	3,Y			4,Y
$UXYV$		3,Y			4,Y
UXYVW					4,Y
UXYVWZ					

Tab. 2.2: Resultados del algoritmo de Dijkstra

Para comprender mejor el algoritmo se presenta un ejemplo, consideramos una red como se muestra en la Figura 2.8 y calcularemos el camino con menor costo desde u hacia todos los destinos posibles. El resultado de todos los cálculos del algoritmo se presentan en la Tabla 2.2 donde cada línea de la tabla muestra los valores de las variables al final de cada iteración al ejecutar el algoritmo.

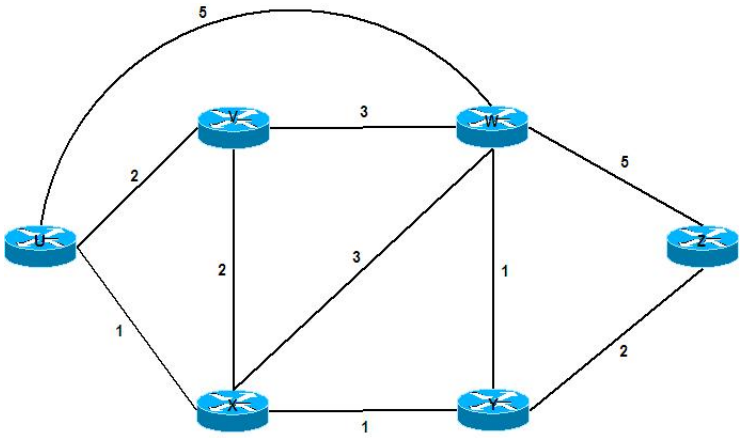


Fig. 2.8: Red

En el paso de inicialización el actual valor del camino con menor costo conocido desde u hasta sus vecinos que se encuentran directamente enlazados a él: u, x y y son inicializados en 2, 1 y 5 respectivamente. Note en particular que el costo a w es puesto en 5 pues es el costo del enlace directo de u a w. Los costos de y a z son puestos en infinito porque no están directamente conectados a u.

En la primera iteración se analiza aquellos nodos que todavía no han sido añadidos al grupo N' y se encuentra aquel nodo con el menor costo en el final de la iteración anterior. Ese nodo es x y su costo es 1, es por eso que x es añadido al grupo N'. La línea número 11 del algoritmo es ejecutada para actualizar $D(v)$ para todos los nodos v. Es así como se obtienen los resultados que se muestran en la segunda línea de la tabla mostrada en la Tabla 2.2. El costo del camino a v ya no será cambiado. El costo del camino a w (el cual era 5 al final del proceso de inicialización) a través del nodo x es igual a un costo de 4. El algoritmo selecciona este nuevo camino de costo más bajo y el predecesor de w a lo largo del camino más corto desde u es igual a x. De forma similar el costo a y a través de x según los cálculos es igual a 2 y la tabla es actualizada de acuerdo a estos resultados.

En la segunda iteración los nodos v y y resultan tener los caminos más cortos (2) y el algoritmo añade a y al grupo N' así N' ahora contiene a u, x y a y. El costo de los nodos restantes que todavía no están en el grupo N' (nodos v,w y z) son actualizados en la línea 11 del algoritmo mostrando los resultados en la la Tabla 2.2.

El algoritmo continúa de manera similar las iteraciones restantes. Cuando el algoritmo termina se tiene para cada nodo su predecesor junto con el camino de

menor costo desde el nodo fuente. Para cada predecesor también se tiene a su predecesor y así de esta manera se puede construir el camino entero desde la fuente a todos los nodos. La tabla de envíos en un nodo por ejemplo el nodo u puede ser construido por medio de la información guardando para cada destino el salto al próximo nodo en el camino de costo con menor costo desde u hasta su destino.

2.3.2. Algoritmo de Bellman-Ford

Mientras que el algoritmo de estado-enlace utiliza información global, el algoritmo de vector-distancia es iterativo, asíncrono y distribuido. Se dice que es distribuido porque cada nodo recibe información de uno o más de sus nodos vecinos directamente enlazados a él, realiza el cálculo y luego distribuye de regreso el resultado de los cálculos a sus nodos vecinos. Se dice que es iterativo en el sentido de que éste proceso continua hasta que no se intercambia información entre los vecinos. Es interesante notar que este algoritmo se termina por sí mismo; no existe una señal que le indique al algoritmo cuando parar, simplemente se para. Por último se dice que el algoritmo es asíncrono, pues no requiere que los nodos operen de manera conjunta entre ellos [Kurose and Ross, 2005].

Se analiza la relación que existe entre los costos y el camino con menor costo. Se hace que $d_x(y)$ sea el costo del camino con menor costo del nodo x al nodo y.

Entonces, los costos mínimos están relacionados por la ecuación *Bellman-Ford*:

$$d_x(y) = \min_v [c(x, v) + d_v(y)]$$

Donde: $c(x, v)$ es el costo acumulado. $d_v(y)$ es el costo a cada vecino.

Donde el $\min_v$ en la ecuación es obtenido de todos los vecinos x. La ecuación de *Bellman-Ford* es un tanto intuitiva. De hecho, después de haber viajado desde x a v, y si después se toma el camino con menor costo de v a y el costo del camino será $c(x, v) + d_v(y)$ dado que debemos comenzar viajando a un vecino v el menor costo desde x hasta y es el mínimo de $c(x, v) + d_v(y)$ obtenidos de todos los vecinos v.

La ecuación Bellman-Ford provee las entradas en la tabla de envíos del nodo x. Para ejemplificar esto, se dice que v^* es cualquier nodo vecino que es el menor en la ecuación de Bellman-Ford. Entonces, si el nodo x quiere mandar un paquete al nodo y a lo largo del camino con menor costo, entonces este debería enviar el paquete hacia el nodo v^*. Así, la tabla de envíos del nodo x debe especificar como nodo siguiente (o brinco inmediato) al nodo v^* para el destino último del nodo y.

La idea básica es la siguiente: cada nodo x comienza con un estimado del camino con menor costo hacia el nodo $y : D_x(y)$. Para todos los nodos N se hace que $D_x = [D_x(y) : y \in N]$ sea el vector distancia del nodo x. Este vector contiene los costos estimados desde el nodo x hacia todos los demás nodos y en N. En el algoritmo de vector distancia cada nodo x mantiene los siguientes datos de ruteo. Para cada vecino v, el coste $c(x, y)$ desde x hacia sus vecinos directamente ligados v. El vector distancia del nodo x. Este es: $D_x = [D_x(y) : y \in N]$ que contiene los

estimados del nodo x hacia todos los destinos y en N. Los vectores distancia para cada vecino $D_v = [D_v(y) : y \in N]$ para cada vecino v de x.

En el algoritmo de Bellman-Ford, cada nodo manda una copia de su vector distancia a cada uno de sus vecinos. Cuando un nodo x recibe un nuevo vector distancia de cualquiera de sus vecinos v, este guarda el vector distancia de v y luego usa la ecuación de Bellman-Ford para actualizar su propio vector distancia mediante la siguiente ecuación:

$D_x(y) = \min_v[c(x, v) + D_v(y)]$ Para cada nodo y en N

Si el vector distancia de x cambia como resultado de este paso de actualización, el nodo x entonces manda su vector distancia actualizado a cada uno de sus vecinos, los cuales actualizan su propio vector distancia. Mientras todos los nodos continúen intercambiando sus vectores distancia en un modo asíncrono cada costo estimando $D_x(y)$ converge a $d_x(y)$ que es costo del camino de menor costo desde el nodo x hacia el nodo y.

Algoritmo Bellman-Ford [Kurose and Ross, 2005]:

1. **Inicialización**
2. Para todos los destinos en N
3. $D_x(y) = c(x, y)$ /* si y no es un vecino $c(x, y) = \infty$ /*
4. Para cada vecino w

5. $D_x = \infty$ para todos los destinos y en N
6. Para cada vecino w
7. mandar vector distancia $D_x = [D_x(y) : y \in N]$ a w
8. **Lazo**
9. esperar (hasta que vea el cambio del costo de enlace en un vecino w o hasta que reciba un vector distancia de algún vecino w)
10. Para cada y en N:
    $D_x(y) = \text{mín}_v[c(x, y) + D_v(y)]$
11. Si $D_x(y)$ cambia para cualquier destino y
12. mandar vector distancia $D_x = [D_x(y) : y \in N]$ a todos los vecinos

El algoritmo vector distancia muestra como un nodo x actualiza su vector distancia cuando ve un cambio en el costo de alguno de sus vecinos que estén directamente conectados o reciba la actualización del vector distancia de alguno de sus vecinos que estén directamente conectados a él. Pero para actualizar su propia tabla de envíos para un destino dado y, lo que el nodo x realmente necesita saber no es el camino con menor distancia a y sino el nodo vecino v^* que es el router de siguiente salto a lo largo del camino más corto a y. Como se puede esperar el router de siguiente salto v^* es el vecino v que obtuvo el mínimo en la línea 10 del algoritmo. Entonces en las líneas 9 y 10, para cada destino y, el nodo x también determina a v^* y actualiza su tabla de envíos para el destino y.

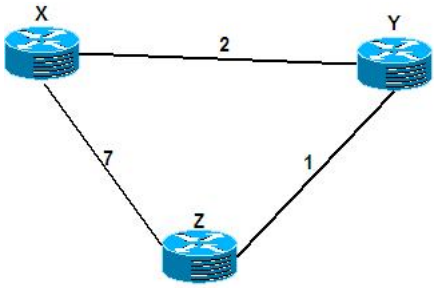


Fig. 2.9: Ejemplo Bellman-Ford Tres nodos

Un ejemplo de una simple red de tres nodos como se muestra en la Figura 2.9. Las tablas correspondientes se encuentran en la Tabla 2.10. Esta tabla contiene tres tablas de ruteo para cada uno de los tres nodos. Por ejemplo, la tabla que se encuentra en la esquina superior izquierda es la tabla inicial de ruteo del nodo x. Dentro de una tabla de ruteo específica, cada fila es el vector distancia. Cada tabla de ruteo incluye su propio vector distancia y el vector distancia de cada uno de sus vecinos. Así, la primera fila en la tabla inicial de ruteo del nodo x es $D_x = [D_x(x), D_x(y), D_x(z)] = [0, 2, 7]$. La segunda y tercera fila en esta tabla representan los vectores distancia recientemente recibidos desde los nodos y ó z, las entradas en la segunda y tercera fila son inicializados en infinito.

Después de su inicialización, cada nodo manda su propio vector distancia a cada uno de sus nodos vecinos como se muestran en la Tabla 2.10 por las flechas desde la primera columna de las tablas hasta la segunda columna de las tablas. Por ejemplo, el nodo x manda su vector distancia $D_x = [0, 2, 7]$ a ambos nodos: y y z. Una vez recibidas las actualizaciones, cada nodo calcula nuevamente su propio vector distancia. Por ejemplo el nodo x calcula:

$$D_x(x) = 0$$

$$D_x(y) = \text{mín}\left\{c(x,y) + D_y(y), c(x,y) + D_z(y)\right\} = \text{mín}\left\{2+0, 7+1\right\} = 2$$
$$D_x(z) = \text{mín}\left\{c(x,y) + D_y(z), c(x,z) + D_z(z)\right\} = \text{mín}\left\{2+1, 7+0\right\} = 3$$

La segunda columna de esta manera ahora muestra, para cada nodo, el nuevo vector distancia del nodo junto con los vectores distancia recién recibidos de sus vecinos. Note, por ejemplo, que el menor costo estimado del nodo x al nodo z: $D_x(z)$, cambió de 7 a 3. También note que para ambos nodos y y z, el nodo y adquiere el mínimo. Así que en esta etapa, el router de siguiente salto $v^*(y) = y$ y el $v^*(z) = y$.

Después de que los nodos recalculan sus vectores distancia, estos mandan sus vectores distancia actualizados a sus vecinos. Esto es ilustrado en la Tabla 2.10 por las flechas de la segunda columna de las tablas hacia la tercera columna de las tablas. Note que solamente los nodos x y z mandan sus actualizaciones, pues el vector distancia del nodo y no cambió. Así el nodo y no manda su vector distancia pues no recibió cambio alguno. Después de recibir esta actualización, los nodos recalculan sus vectores distancia y actualizan sus tablas de ruteo, las cuales se muestran en la tercera columna.

Hasta aquí se tiene el conocimiento necesario para desarrollar el diseño del controlador difuso para el control de congestión en los routers de internet, con retardos variables en la ruta mínima de la fuente al destino. Esto se presenta en el próximo capítulo.

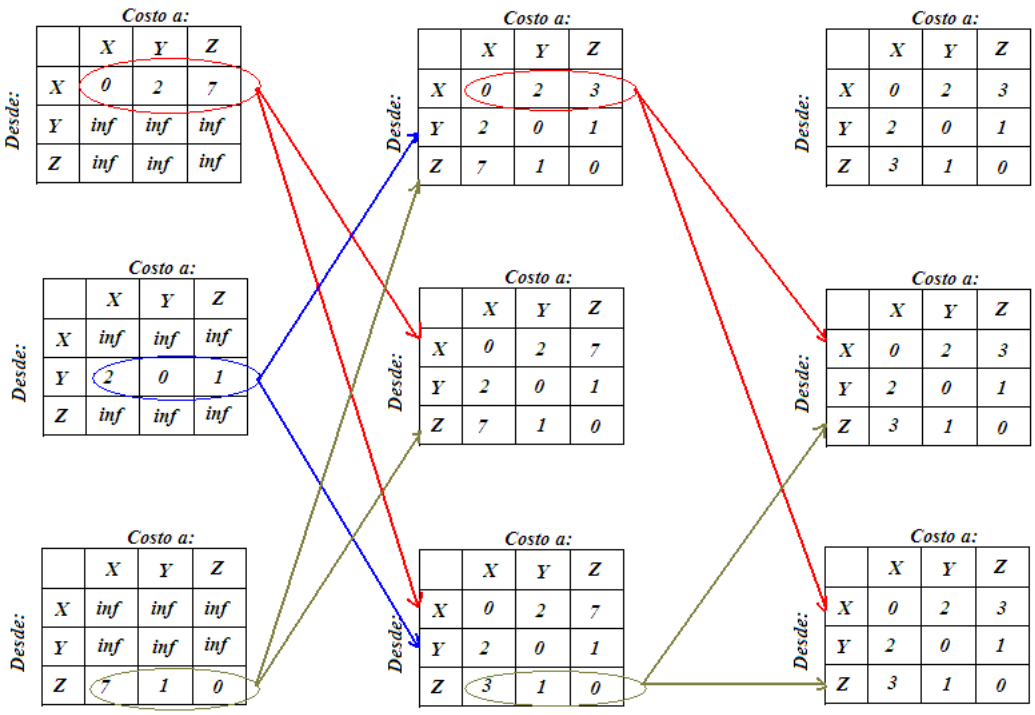


Fig. 2.10: Tabla Bellman-Ford para el nodo X

3. DISEÑO DEL CONTROLADOR DIFUSO

En este capítulo se presenta el diseño de un controlador difuso para una red de cómputo TCP/AQM, como una propuesta para llevar a la cola del router a un valor deseado tomando en cuenta el tiempo de viaje redondo (RTT) variable debido a las diferentes rutas elegidas por el protocolo de ruteo. Este trabajo se realizó en el software MATLAB-SIMULINK®

3.1. Retardo de Propagación

El gran desarrollo tecnológico que en los últimos años ha sufrido la red Internet ha permitido que amplíe su horizonte de aplicaciones, sin embargo, esto ha llevado a una serie de problemas por resolver en este campo, uno de ellos es el retardo en la comunicación.

Un paquete puede sufrir retardos a lo largo de su recorrido entre el host de origen y el host destino. Los tipos de retardos más importantes se explican a continuación, agrupándose todos ellos para dar lugar al retardo nodal total.

Supongamos que un paquete se envía desde un nodo A a un nodo B.

- **Retardo de procesamiento**: es el tiempo necesario para examinar la cabecera del paquete y determinar el enlace por el que hay que enviarlo. Pero no es el único factor, hay que añadirle el tiempo que tarda en comprobar los posibles errores en el envío desde el nodo anterior al nodo A. Puede ser del orden de microsegundos o menos.
- **Retardo de cola**: Después del procesamiento, el paquete se introduce en la cola de salida del enlace adecuado. Este retardo variará en función del número de paquetes anteriores que estén encolados a la espera de ser transmitidos. Esta espera suele ser del orden de microsegundos a milisegundos.
- **Retardo de transmisión**: Si consideramos L el número de bits del paquete y R la tasa de transmisión en bits/segundo del enlace entre A y B, el retardo de transmisión (o retardo de almacenar y enviar) es de L/R. Es del orden de milisegundos.
- **Retardo de propagación**: Es el tiempo necesario para propagar el paquete desde el nodo A hasta el nodo B. Es el resultado de la división de la distancia entre ambos nodos y la velocidad de propagación del enlace.

3.1.1. Retardo de Transmisión y Propagación

A veces se confunden ambos términos. La transmisión hace referencia al tiempo que necesita el nodo A para enviar el paquete, mientras que la propagación al tiempo que requiere un bit para viajar entre el nodo A y el nodo B. El primero depende de la longitud del paquete pero no de la distancia y el segundo al contrario.

Se considera la analogía con una autopista. El retardo de transmisión sería el tiempo requerido para que el operario del peaje cobre (uno a uno) al grupo de coches que está esperando; mientras que el de propagación es el tiempo que uno de esos coches necesita para viajar desde este peaje hasta el siguiente.

Se presenta el caso en que el tiempo de transmisión sea considerablemente mayor que el de propagación, lo que induce a que, mientras que algunos bits ya hayan llegado al nodo B; aún queden algunos esperando en el A para ser transmitidos.

El retardo nodal total viene dado por la suma de los retardos anteriores. Estos componentes de la suma pueden variar significativamente, desde ser despreciables hasta contribuir en gran medida al retardo. El retardo de proceso suele ser despreciable, pero influye mucho en la capacidad de transmisión de un router.

3.1.2. Retardo de Cola y Pérdida de Paquetes

El componente más interesante y complicado es el retardo de cola, que varía mucho de un paquete a otro. Cuanto antes llegue un paquete y menos paquetes se encuentre ya encolados por delante de él, menos retardo de cola tendrá. Por tanto, al analizar el retardo de cola, se realiza mediante medidas estadísticas. El concepto de “mucho” o “poco” hablando de este retardo depende de la tasa de entrada a la cola, de la velocidad de transmisión del enlace y de si el tráfico es continuo o a rachas.

Los algoritmos de colas en los enrutadores intentan adaptar éstos retardos a ciertas preferencias, o imponer un uso equitativo.

Supongamos α la tasa de llegada de paquetes, ρ la velocidad de transmisión del enlace y λ la longitud de los paquetes (supondremos que todos los paquetes tienen la misma longitud, por simplificar). A la relación $\frac{\lambda\alpha}{\rho}$ se le denomina *intensidad del tráfico*. Si dicha intensidad es mayor que 1, la tasa de llegada es mayor que la velocidad de transmisión, con lo que la cola crecería indefinidamente. Se debe intentar que la intensidad sea menor o igual a 1. A partir de ahora vamos a suponer éste último valor.

Si los paquetes llegasen a razón de $\frac{\lambda}{\rho}$, no encontraría nada de retardo, pero si llegan a rachas periódicas, puede producirse un importante retardo. Si estas rachas de N paquetes llegan cada $\frac{\lambda N}{\rho}$ segundos, la primera racha tendrá $\frac{\lambda}{\rho}$ de retardo y $(n-1)(\lambda-\rho)$ segundos la racha n-ésima.

Sin embargo, en la realidad la llegada de las rachas es aleatoria. Por tanto, la fómula anterior de la intensidad no es suficiente. Si la intensidad se aproxima a 0, significa que los paquetes llegan muy espaciados, con lo que no encontrarán retardo. Pero si se aproxima a 1, habrá intervalos en los que la tasa de llegada es mayor que la de transmisión, con lo que aumenta el retardo por la acumulación de paquetes.

Se puede asumir también que el arribo de paquetes a la cola es un proceso de Poisson, que el tamaño de paquetes tiene distribución exponencial, y que el sistema se puede pensar como una cola M/M/1 (la primera «M» indica que los tiempos

entre llegadas de clientes son variables aleatorias exponenciales-Markoviano-, la segunda «M» indica que los tiempos de servicio de los clientes son variables aleatorias exponenciales; y el «1» final indica que la cola tiene 1 servidor). Con estas hipótesis, dado un tamaño promedio de tamaño de paquetes de L bits/paquete y una capacidad del enlace de R bps, la tasa de servicio promedio del enlace es $\mu = R/L$ pps (paquetes por segundo). Si se considera la tasa de arribo promedio de paquetes de «a» pps, se cumple que el retardo en segundos por paquete, en promedio, está dado por la ecuación:

$$D = \frac{1}{\mu - a}$$

Si consideramos la utilización media $\rho = \frac{a}{\mu} = \frac{a}{\frac{R}{L}} = L\frac{a}{R}$, y escribimos la ecuación del retardo como:

$$D = \frac{1}{\frac{R}{L}\left(1 - L\frac{a}{R}\right)}$$

Se puede ver que el retardo por paquete tiende a infinito cuando la utilización del enlace tiende al 100 % (es decir cuando $L\frac{a}{R} \to 1$).

3.1.3. Pérdida de Paquetes

Hay que tener en cuenta que: la capacidad de la cola no es infinita. Esto significa que si al estar llena, llegan más paquetes, se tendrán que descartar. Por tanto, la intensidad del tráfico no es la única medida de referencia, sino también la tasa

de pérdida de paquetes, que será presumiblemente mayor a medida que aumenta la intensidad.

3.1.4. Retardo Terminal a Terminal

Ahora vamos a comenzar a considerar el retardo no entre dos nodos, sino entre el nodo origen y el nodo destino; es decir, de todo el recorrido. Consideraremos que hay N-1 routers entre ambos y que no hay congestión en la red. Supondremos también que la velocidad de transmisión de todos los enlaces es ρ, que el retardo de proceso es d_{proc}, que el retardo de transmisión es d_{trans} y que la velocidad de propagación de cada enlace es d_{prop}.

El retardo terminal a terminal sería:

$$d_{term} = N(d_{proc} + d_{trans} + d_{prop})$$

recordando que:

$$d_{trans} = \frac{\lambda}{\rho}$$

Determinar el retardo que sufre un paquete en una red de comunicaciones es un problema que puede parecer sencillo, sin embargo, no es así.

Los protocolos TCP/IP utilizados en internet no garantizan que este retardo que sufren las señales durante el paso por la red sea un parámetro constante.

El valor del retardo de las señales en Internet está ligada a muchos factores (Fig. 3.1):

- La velocidad de conmutación de los nodos
- La carga de los nodos
- La capacidad de tráfico de las redes
- La cantidad de transferencia de datos
- La velocidad de la entrega, etc.

Pero los factores dominantes que determinan la magnitud del retardo principalmente son: la velocidad de la red y la carga de los nodos, por esta razón esta magnitud es un valor imprevisible.

Una posible solución para reducir el retardo de propagación consiste en incrementar el ancho de banda, pero hay que tener en cuenta que existe un compromiso entre ambos factores, ya que a mayor ancho de banda se requieren mayores buffers y mayor tiempo para vaciarlos, lo que aumenta el retardo. Con respecto al retardo producido por un encaminamiento ineficiente de los paquetes la solución consiste en mejorar el diseño de la red para evitar la congestión y reducir el número de saltos a dar hasta alcanzar un destino determinado.

Es por ello que nuestro interés es minimizar el retardo de propagación en la transmisión de información en Internet controlando la congestión en los routers mediante un controlador difuso y asignar las mejores rutas. Las rutas son elegidas minimizando el número de saltos al destino mediante el algoritmo de Dijkstra, por ejemplo.

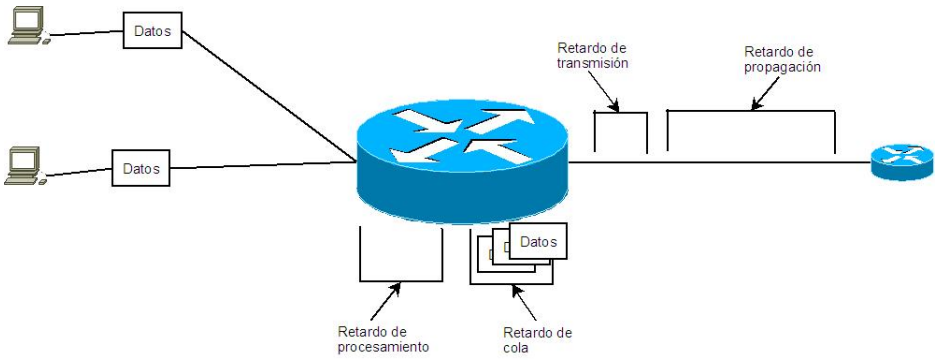


Fig. 3.1: Retardos en la red

En este trabajo presentamos al controlador difuso afectado por la tabla de ruteo. Se supone que de alguna forma se obtienen los retardos en los enlaces. Así, el retardo es el costo del enlace del router i al router j (Fig. 3.2). El protocolo de ruteo elige la ruta mínima; a ésta ruta le corresponde un retardo. Simulamos un retardo periódico con dinámica tipo senoide con frecuencia de 10 Hz, como se muestra en la Figura 3.3, y del tipo diente de sierra como en la Figura 3.4. El algoritmo de ruteo cambia constantemente la ruta elegida, por lo tanto, el retardo varía también de acuerdo a la ruta elegida, y éste retardo se toma como el antecedente para elegir al controlador a aplicar mediante lógica difusa. El retardo en las diferentes rutas elegidas por el protocolo de ruteo fue simulado como se ve en la Fig. 3.5. Se puede observar el cambio de ruta en los tiempos 400, 800, 1200 y 1600 de la Fig. 3.5. Además se simuló el retardo en cada ruta no solo con dinámica senoidal, sino con dinámica aleatoria uniformemente distribuida como se observa en la Fig. 3.6.

Así, teniendo el retardo en la ruta elegida, se incorpora al modelo de la red para el diseño del controlador difuso.

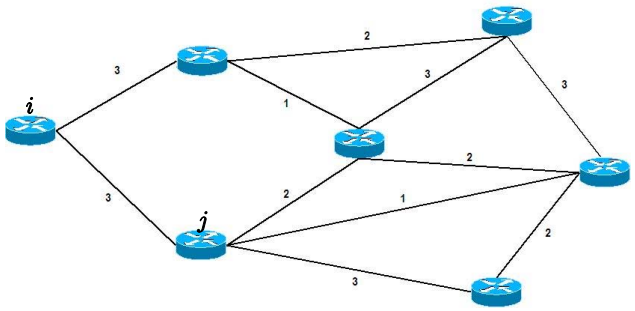


Fig. 3.2: Retardo como Coste en los Enlaces

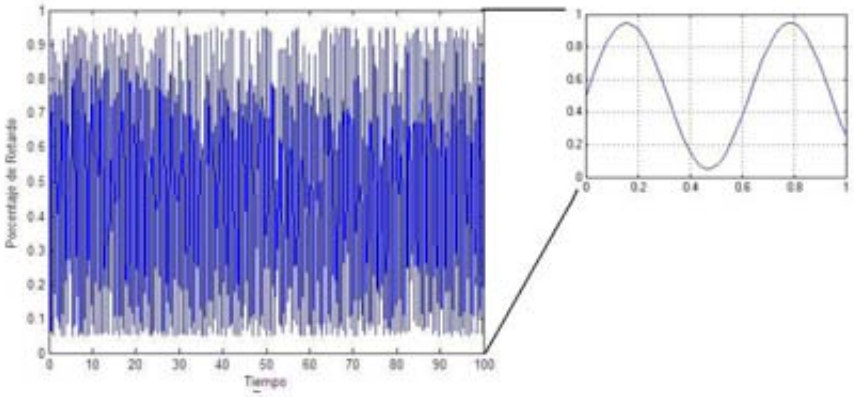


Fig. 3.3: Dinámica del retardo tipo senoidal

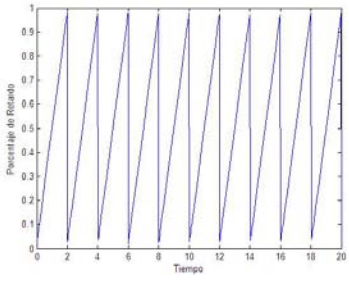


Fig. 3.4: Dinámica del retardo tipo sierra

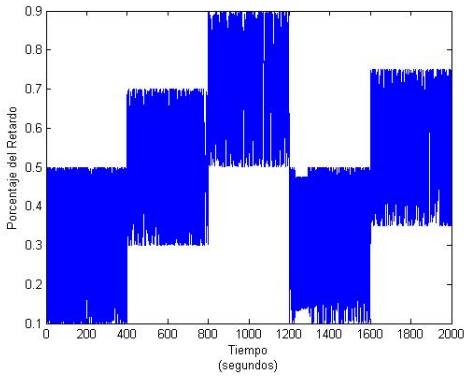


Fig. 3.5: Retardo en Rutas Elegidas por el Protocolo de Ruteo

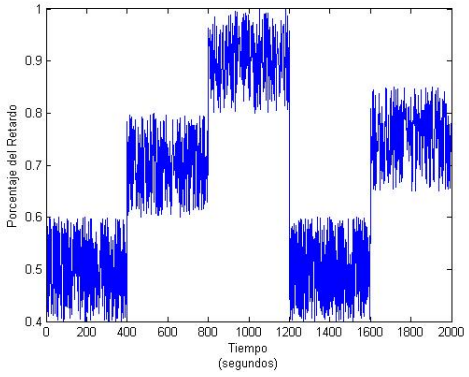


Fig. 3.6: Retardo en Rutas Elegidas por el Protocolo de Ruteo

3.2. Modelo de la Red

Para modelar la red se considera al modelo de flujo de fluidos dinámico introducido en [Hollot et al., 2002] para describir la dinámica de una red TCP/AQM, donde se ignora el time-out y el inicio lento de TCP. El modelo presenta las relaciones del valor promedio de variables claves de una red de N fuentes homogéneas controladas por TCP y un único ruteador congestionado (Fig. 3.7) mediante las

ecuaciones diferenciales nolineales acopladas Ecuac. (3.1 - 3.2).

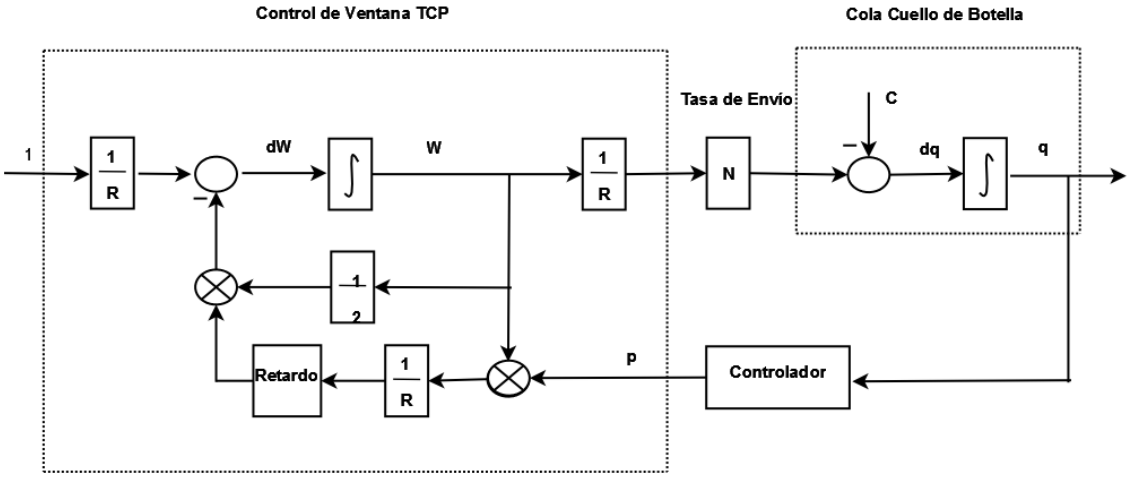


Fig. 3.7: Diagrama a Bloques del Control de Congestión de TCP

$$\dot{w} = \frac{1}{R(t)} - \frac{1}{2}\frac{w(t)w(t-R(t))}{R(t-R(t))}p(t-R(t)) \tag{3.1}$$

$$\dot{q(t)} = \begin{cases} \frac{w(t)}{R(t)} - C & : \quad q > 0 \\ \text{máx}\left(0, N\frac{w(t)}{R(t)} - C\right) & : \quad q = 0 \end{cases} \tag{3.2}$$

$$R(t) = \frac{q(t)}{C} + \tau_p$$

donde $w(t)$ denota el promedio del tamaño de la ventana TCP (paquetes), q es el promedio de la longitud de la cola (paquetes), $R(t)$ es el tiempo promedio de viaje redondo (segundos) de donde τ_p representa el retardo de propagación, c es la capacidad de transmisión (paquetes/segundo), N es el número de seciones TCP y

$p(\cdot)$ es la probabilidad de marcado de paquetes. El tamaño de la ventana $(w(t))$, la longitud de la cola $(q(t))$ y la probabilidad de marcado son cantidades positivas y acotadas, i.e., $w(t) \in [0, w_{max}]$, $q(t) \in [0, q_{max}]$ y $p(\cdot) \in [0, 1]$ respectivamente. Este modelo evita que $\dot{q} < 0$ cuando $q = 0$, con lo cual previene que la longitud promedio de la cola $q(t)$ tome valores negativos.

La ecuación (3.1) describe la dinámica de control de ventana de TCP. El incremento del tamaño de la ventana en uno por cada RTT queda determinado por el primer término del lado derecho de la Ecuac. (3.1) y el segundo término indica que la ventana se divide a la mitad en el instante que ocurre una pérdida $(p(\cdot) = 1)$. Así también la longitud promedio de la cola queda definida por Ecuac. (3.2) como la diferencia entre la tasa de llegada de paquetes y la capacidad de transmisión, asumiendo que no hay dinámica interna en el cuello de botella. Así la dinámica de la ventana queda determinada por el emisor, sin embargo, en los esquemas AQM son estrategias de control a nivel servidor, entonces bajo este esquema, la confirmación de que un paquete ha sido marcado como perdido es recibida por el emisor después de un cierto tiempo de retardo, modificando así la dinámica de la ventana. Esto es evidente en Ecuac. (3.1), en donde la función de marcado viene retardada.

La estrategia de control retroalimentado del sistema queda representada por la función de marcado de paquetes $p(\cdot)$, cuya dinámica depende de q. Se observa de Ecuac. (3.1) una característica inherente de los sistemas de comunicación y de los sitemas de redes TCP/AQM, el retardo no puede ser cero.

Sin embargo, de [Melchor and Castillo, 2007] se tiene un modelo más simplificado, donde RTT $\approx \tau_p = \tau$:

$$\dot{w} = \frac{1}{R(t)} - \frac{1}{2}\frac{w(t)w(t-R(t))}{R(t)}p(t-R(t)) \tag{3.3}$$

$$\dot{q(t)} = \begin{cases} N\frac{W(t)}{R(t)} - C & : \quad q > 0 \\ \text{máx}\left(0, N\frac{W(t)}{R(t)} - C\right) & : \quad q = 0 \end{cases} \tag{3.4}$$

$$R(t) \approx \tau_p = \tau \tag{3.5}$$

Esto cuando el retardo en la cola es mucho menor que el retardo de propagación, lo cual es una buena suposición ya que como se argumenta en [Kelly, 2000] hay un rápido avance en el hardware de routers y la capacidad de la red, reduciendo así el retardo en la cola. En este trabajo tomamos al retardo de propagación variable ($\tau \in (0, 1)$) con la suposición que la variación tiene una dinámica peculiar.

3.3. Controlador Difuso

Habitualmente suele emplearse la lógica difusa en aplicaciones de control [Ogata, 1998], [Takagi and Sugeno, 1985]. Entonces, proponemos un controlador difuso para el control de congestión del modelo de red Ecuac. (3.3 - 3.5). El controlador mantendrá a la cola del router en un valor deseado teniendo como antecedente al retardo variable correspondiente a la ruta mínima elegida por el protocolo de ruteo desde un emisor a un receptor en la red, encontrado mediante el algoritmo de Dijkstra, por ejemplo. Con este controlador se evitará la pérdida de información por congestión, estableciendo así calidad de servicio en la red y

evitando la oscilación del envío de información de las fuentes. Esta es la diferencia entre los trabajos presentados en el capítulo 2.

El control difuso establece el algoritmo de control del proceso como un conjunto de relaciones difusas entre la variable τ y la probabilidad establecida por una combinación de controladores PI lineales con el fin de mantener una carga constante en el router, tratar de establecer una cantidad de transferencia de datos constante y mejorar la velocidad de entrega.

Como una primera aproximación para tratar el retardo variable en el modelo, se tomaron PI robustos para el consecuente de las reglas del controlador como en [Melchor and Castillo, 2007] (ver Fig. 3.8), diseñados para retardos constantes. Los consecuentes se obtuvieron de la forma en la Ecuac. (3.6):

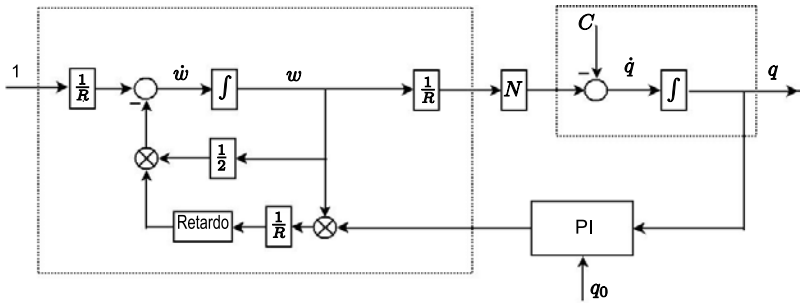


Fig. 3.8: Controlador PI para Retardo Constrante

$$p(t) = k_p q(t) + \frac{k_p}{I} \int_t^0 (q(\sigma) - q_0) d\sigma \tag{3.6}$$

Se define:

$$\delta = \int_t^0 (q(\sigma) - q_0) d\sigma \tag{3.7}$$

Se obtiene el sistema aumentado:

$$\dot{w} = \frac{1}{R(t)} - \frac{1}{2}\frac{w(t)w(t-R(t))}{R(t)} p(t-R(t)) \tag{3.8}$$

$$\dot{q(t)} = \begin{cases} N(t)\frac{W(t)}{R(t)} - C & : \quad q > 0 \\ \text{máx}\left(0, N(t)\frac{W(t)}{R(t)} - C\right) & : \quad q = 0 \end{cases} \tag{3.9}$$

$$R(t) = \frac{q(t)}{C} + \tau_p \tag{3.10}$$

$$\dot{\delta} = q(t) - q_0 \tag{3.11}$$

Se realiza la linealización en el punto de equilibrio (w_0, q_0, δ_0):

$$\dot{w} = 0 \Rightarrow w_0^2 p_0 = 2 \tag{3.12}$$

$$\dot{q} = 0 \Rightarrow w_0 = \frac{Rc}{N} \tag{3.13}$$

obteniendo:

$$\bar{q} = q(t) - q_0, \;\; \bar{\delta} = \delta(t) - \delta_0, \;\; \bar{p} = p(t) - p_0$$

y

$$\zeta(t) = \begin{pmatrix} \bar{w} \\ \bar{q} \\ \bar{\delta} \end{pmatrix}, \;\; A = \begin{pmatrix} -\frac{N}{\tau^2 c} & 0 & 0 \\ \frac{N}{\tau} & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \;\; A_1 = \begin{pmatrix} -\frac{N}{\tau^2 c} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \;\; b = \begin{pmatrix} -\frac{\tau c^2}{2n^2} \\ 0 \\ 0 \end{pmatrix}$$

Entonces:

$$\dot{\zeta(t)} = A\zeta(t) + A_1\zeta(t-\tau) + b\bar{p}(t-\tau) \tag{3.14}$$

Se considera al controlador, ahora, de la forma:

$$\bar{p}(t) = k_p\bar{q}(t) + \frac{k_p}{I}\bar{\delta} \tag{3.15}$$

donde $k_p/I \neq 0$

El sistema en lazo cerrado Ecuac. (3.14 - 3.15) queda de la forma:

$$\dot{\zeta(t)} = A\zeta(t) + B\zeta(t-\tau) \tag{3.16}$$

donde:

$$B = \begin{pmatrix} -\frac{N}{\tau^2 C} & -\frac{\tau C^2}{2N^2}k_p & -\frac{\tau C^2}{2N^2}k_p \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

El sistema Ecuac. (3.16) es asintóticamente estable, si y solo si, la función característica:

$$f(s) = s^3 + \frac{N}{\tau^2 C}s^2 + [\frac{N}{\tau^2 C}s^2 + \frac{C^2}{2N}k_p(s + \frac{1}{I})]e^{-\tau s}$$

no tiene ceros con parte real no negativa [Gu et al., 2003].

En [Melchor and Castillo, 2007] se establece un teorema de región de estabilidad para controladores PI. Este establece que dado un conjunto de parámetros (N, τ, C) el sistema Ecuac. (3.16) es asintóticamente estable, si y solo si,las ganancias del controlador (k_p, I) pertenencen a la región de estabilidad $\Phi_{(N,\tau,C)}$, Fig. 3.9, cuya frontera en el espacio de ganancias del controlador (k_p, I) está descrita por Ecuac. (3.17):

$$\partial\Phi_{(N,\tau,C)} = \left\{ (k_p, I) : I = \frac{\theta cos(\theta\tau) + \frac{N}{\tau^2 C} sen(\theta t)}{\theta(\frac{N}{\tau^2 C}(1 + cos(\theta\tau)) - \theta sen(\theta\tau))}, \right. \qquad (3.17)$$
$$\left. k_p = \frac{2N}{C^2}\theta(\theta cos(\theta\tau) + \frac{N}{\tau^2 C} sen(\theta\tau)),\ \theta \in (0, \theta^*) \right\}$$

donde θ^* es la solución de:

$$\frac{sen(\theta\tau)}{cos(\theta\tau) + 1} = \frac{N}{\tau^2 C\theta},\ \ \theta \in (0, \frac{\pi}{\tau})$$

De la Ecuac. (3.17) se observa que $I(\theta) \to \frac{\tau^2 C}{2N} + \frac{\tau}{2}$ y $k_p(\theta) \to +0$ cuando $\theta \to +0$, y $I(\theta) \to +\infty$ y $k_p(\theta) \to \frac{2N(\theta^*)^2}{C^2}$ cuando $\theta \to -\theta^*$.

Así, se obtuvieron los consecuentes de las reglas difusas de manera heurística que se muestran en la Tabla 3.1. Estos corresponden a las distintas regiones de estabilidad (ver Figura 3.10 y Figura 3.11). En este caso la variable controlada es la cola $(q(t))$.

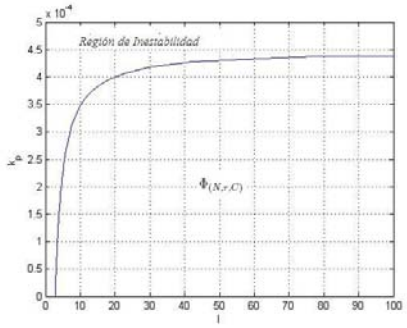
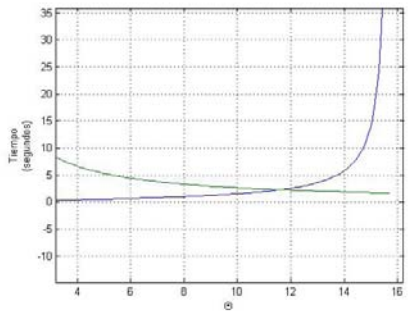


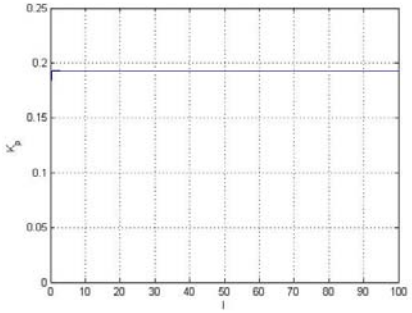
Fig. 3.9: Región de Estabilidad $\Phi_{(N,\tau,C)}$

PI	τ	k_p	I
1	0.1	0.009	57
2	0.4	1e-3	29
3	0.6	1e-4	12
4	0.8	0.8e-4	18

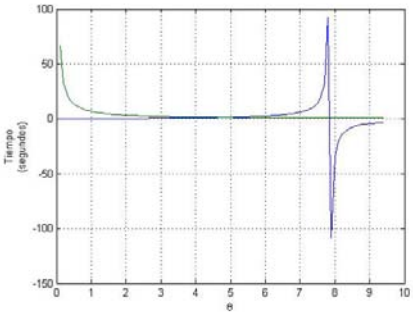
Tab. 3.1: Valores de los Consecuentes Difusos



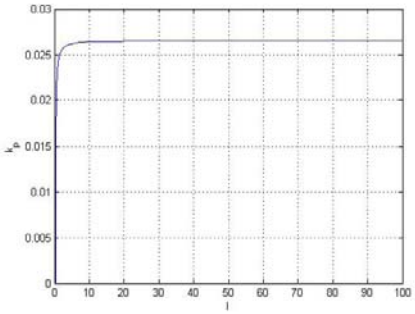
(a) Solución numérica de la ecuación 3.3 para $\tau = 0{,}1$



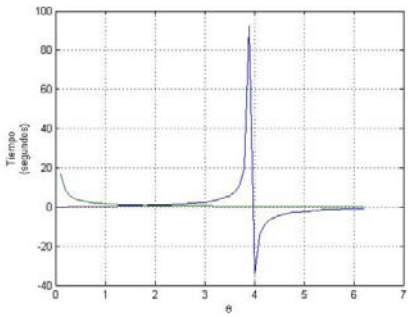
(b) Región de estabilidad para $\tau = 0{,}1$



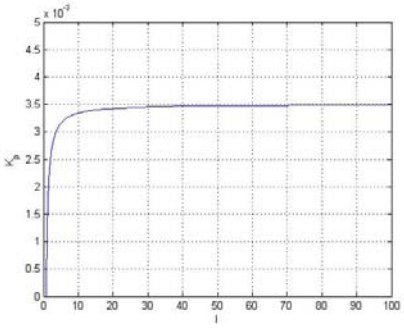
(c) Solución numérica de la ecuación 3.3 para $\tau = 0{,}2$



(d) Región de estabilidad para $\tau = 0{,}2$

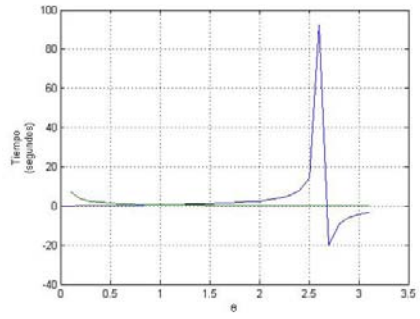


(e) Solución numérica de la ecuación 3.3 para $\tau = 0{,}4$

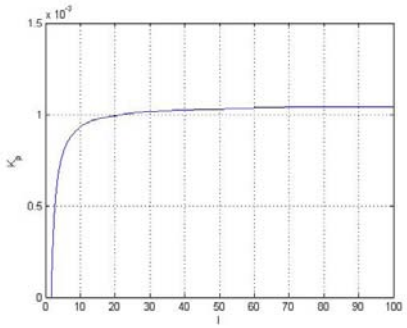


(f) Región de estabilidad para $\tau = 0{,}4$

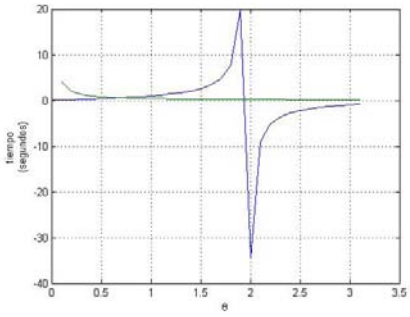
Fig. 3.10: Región de Estabilidad de Familia de PI



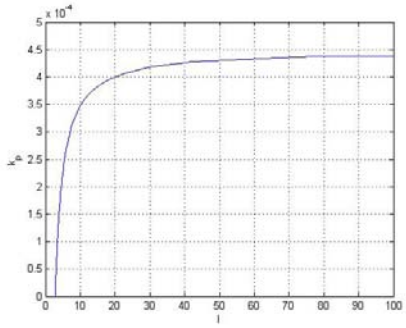
(a) Solución numérica de la ecuación 3.3 para $\tau = 0{,}6$



(b) Región de estabilidad



(c) Solución numérica de la ecuación 3.3 para $\tau = 0{,}8$



(d) Región de estabilidad

Fig. 3.11: Familia de PI Estabilizantes

Entonces el controlador difuso tendría como entradas al error, la derivada del error y el retardo (Ver Fig. 3.12). En este caso la variable controlada es la cola $(q(t))$ y cabe mencionar que el retardo es la única variable de entrada en la parte antecedente del controlador difuso.

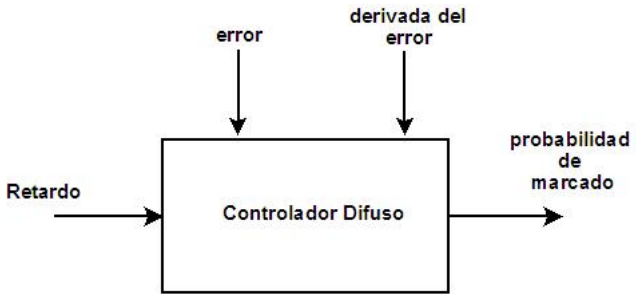


Fig. 3.12: Entradas del Controlador Difuso

Como es sabido, un controlador difuso se compone de: fuzzificador o difusor, base de conocimientos, mecanismo de inferencia y desfuzzificador o desdifusor. Para este controlador difuso se consideró como variable de salida a la probabilidad de marcado y como variable antecedente al retardo de la ruta elegida, este tiene dinámica ya que el protocolo de ruteo cambia la ruta. La fuzzificación se lleva a cabo categorizando al retardo en ocho conjuntos: Mínimo, Muy Pequeño, Pequeño, Medio Pequeño, Medio, Medio Grande, Grande y Máximo (Figura 3.13).

Estos conjuntos difusos se eligieron sigmoides para Mínimo y Máximo, y gaussianas para los conjuntos restantes como se muestra a continuación, esto para tener evitar puntos de inflexión que podrian producir inestabilidad y cubriendo casi todo el universo de discurso para evitar cambios abruptos en la elección del consecuente.

$$
\begin{aligned}
\text{Mínimo} &= \frac{1}{1+e^{23(\tau-0{,}03)}} \\
\text{Muy pequeño} &= e^{-(\frac{\tau-0{,}1}{0{,}4})^2} \\
\text{Pequeño} &= e^{-(\frac{\tau-0{,}2}{0{,}4})^2} \\
\text{Medio Pequeño} &= e^{-(\frac{\tau-0{,}4}{0{,}4})^2} \\
\text{Medio} &= e^{-(\frac{\tau-0{,}6}{0{,}4})^2} \\
\text{Medio Grande} &= e^{-(\frac{\tau-0{,}8}{0{,}4})^2} \\
\text{Grande} &= e^{-(\frac{\tau-0.}{0{,}9})^2} \\
\text{Máximo} &= \frac{1}{1+e^{-23(\tau-0{,}8)}}
\end{aligned}
$$

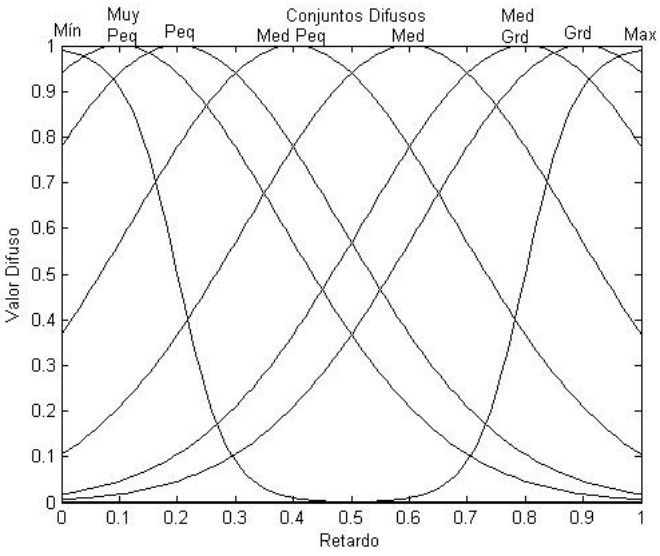


Fig. 3.13: Conjuntos Difusos para el Retardo

Base de Reglas
Si τ es *Mínimo* **Entonces** PI-3
Si τ es *Muy Pequeño* **Entonces** PI-1
Si τ es *Pequeño* **Entonces** PI-2
Si τ es *Medio Pequeño* **Entonces** PI-2
Si τ es *Medio* **Entonces** PI-3
Si τ es *Medio Grande* **Entonces** PI-3
Si τ es *Grande* **Entonces** PI-3
Si τ es *Máximo* **Entonces** PI-4

Tab. 3.2: Reglas Difusas

El mecanismo de inferencia quedó establecida mediante el conjunto de expresiones presentadas en la Tabla 3.2 y el desdifusor fue tipo centroide como se verá más adelante.

La variable τ toma valores lingüísticos mencionados anteriormente (Mínimo, Muy Pequeño, etc.), y se forman predicados difusos. Así, en estas reglas de control difuso, la parte izquierda es el *antecedente* y la parte derecha es el *consecuente*. Cuando se proporciona el valor actual de la variable de entrada τ se hace la fuzzificación de acuerdo a los conjuntos establecidos y se obtienen valores lingüísticos. Estos valores son evaluados mediante el mecanismo de inferencia para obtener un salida con valor lingüístico de la probabilidad de marcado.

De esta forma compilamos la información provista por cada regla y se toma una desición. Como se mencionó antes, la decisión tomada como salida del con-

trolador es una variable difusa, por lo cual se lleva a cabo la defuzzificación. El método de defuzzificación utilizado en este controlador es el llamado *Método del centroide*, definido de la forma Ecuac. (3.18), teniendo ocho conjuntos difusos como en la Figura 3.13.

$$p = \frac{\sum_{j=1}^{N_r} \beta_j PI_j}{\sum_{j=1}^{N_r} \beta_j} \tag{3.18}$$

$$\tag{3.19}$$

donde:

$$\beta_j = \prod_{i=1}^{n} A_{i,j}$$

β el producto del grado de membresía.
$A_{i,j}$ define la función de membresía de la entrada i, usada en la regla j.

Así se obtiene el modelo de red con el controlador difuso mostrado en la Figura 3.14 manteniendo a la cola del router en el valor deseado impidiendo la congestión teniendo como antecedente al retardo en la ruta mínima obtenida por el algoritmo de Dijkstra. Las respuestas del controlador ante un retardo variable se muestran en el capítulo siguiente.

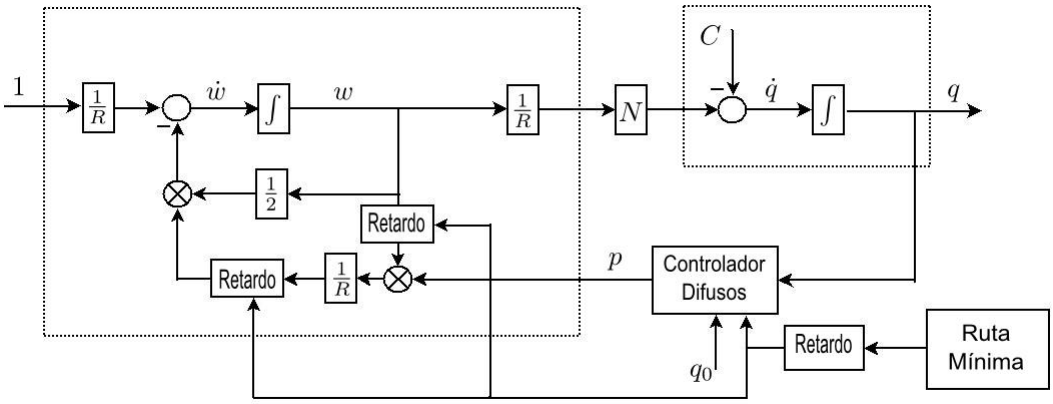


Fig. 3.14: Diagrama del Sistema con Controlador Difuso

4. RESULTADOS

El sistema fue simulado en el paquete MATLAB-SIMULINK® con la estructura mostrada en la Figura 4.1.

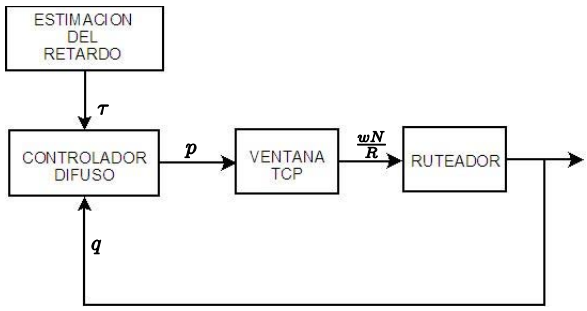


Fig. 4.1: Diagrama del Sistema

Los métodos de retardo usados son sencillos. No se recurrió a herramientas como *tracerouter* que permite generar retardos con mayor verosimilitud, pues se consideró que para probar el controlador difuso los modelos simples eran suficientes.

Dado el retardo asociado al cambio de ruta realizado por el protocolo de ruteo, siendo este la ruta mínima, se diseñó el controlador difuso para controlar la congestión en la cola del router manteniéndola en un punto de operación.

De esta forma resolvemos el problema planteado en el capítulo 1, que recapitulando, establece que la pérdida de información en la red se debe principalmente a que se transmiten paquetes sin conocer el estado de la red. En este trabajo la señal de control, que es la probabilidad de marcado, solamente se está tomando para controlar la congestión en el router estableciendo el tamaño de la ventana de la fuente. La probabilidad de marcado es dependiente de la dinámica del retardo debido al cambio de ruta. Así, se controla la congestión en un router para evitar la pérdida de información debido a este fenómeno.

En la Fig. 4.2 se presenta el diagrama del sistema general en Simulink para usarse en la simulación, distinguiéndose los subsistemas principales del modelo: Ventana de TCP, Router, Controlador y el bloque de Retardo. Este último bloque es el que sufre modificaciones de acuerdo a la dinámica del retardo propuesto. Se observa un bloque de saturación únicamente para mostrar que la cola no toma valores negativos y los bloques a la entrada del controlador difuso son necesarios únicamente para el consecuente en las reglas del controlador. El subsistema "TCP" se muestra en la Figura 4.3, los distintos tipos de retardo con dinámicas propuestas para simular los cambios de ruta establecidas por el router se muestran en la Figura 4.4. El bloque "controlador difuso" es una archivo en código "m" indexado en el apéndice. El subsistema "cola" no se detalla pues se trata de un integrador solamente.

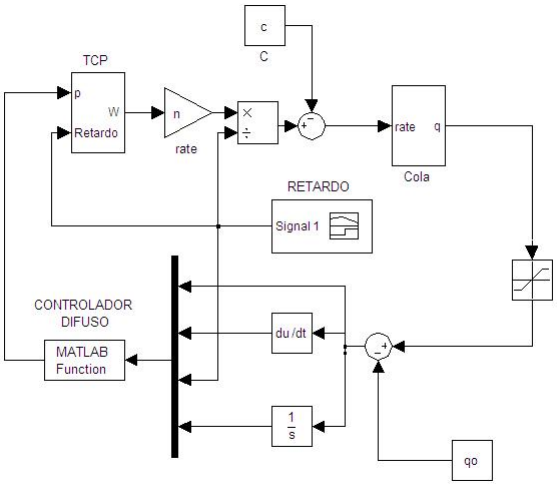


Fig. 4.2: Diagrama del Sistema General

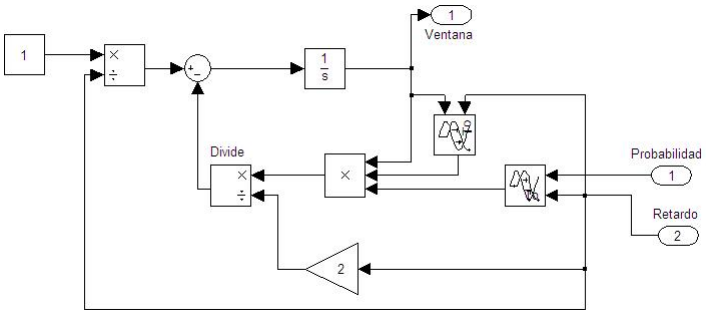
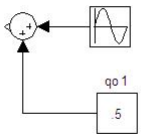
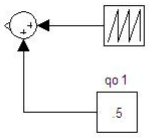


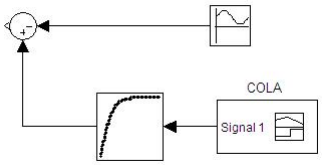
Fig. 4.3: Ventana TCP



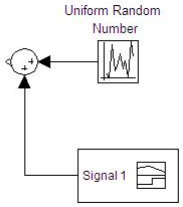
(a) Retardo Senoidal



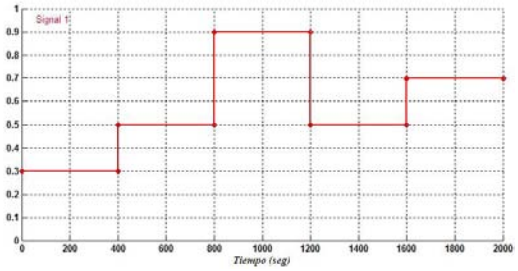
(b) Retardo Diente de Sierra



(c) Retardo Afectado por Cola



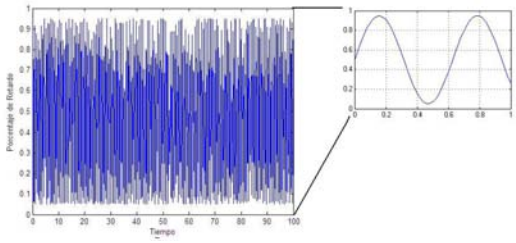
(d) Retardo Aleatorio



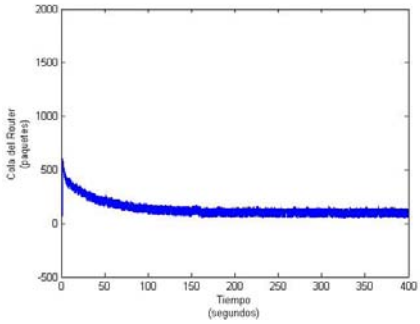
(e) Bloque Signal 1

Fig. 4.4: Retardos en Simulink

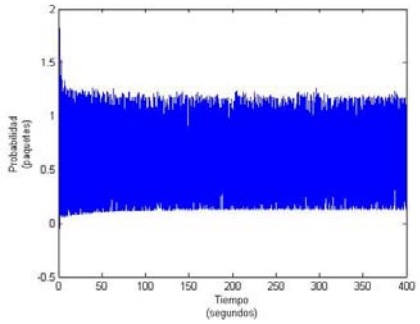
El retardo asociado a cada ruta es aplicado a la entrada del controlador difuso para ser tomado como antecedente en las reglas difusas. Para las simulaciónes se tomaron los siguintes valores para los parámetros del modelo: C=300, N=40, $q_0 = 100$ y $q_0 = 300$ para algunos ejemplos. En la Figura 4.5 se muestran las respuestas obtenidas con el controlador difuso suponiendo que los cambios de rutas son suaves de tipo senoidal, es decir, el ruteador elige la ruta con menor retardo cambiando hasta la ruta con mayor retardo, en forma ascendente, para despues elegir las rutas en forma descendente. Ante esto, el controlador proporciona una probabilidad poco variante lo cual produce un cambio de la ventana casi constante que se traduce a su vez en llevar a la cola del router al punto de operación. Se observa que el buffer del router se mantiene en una región del punto de operación evitando la pérdida de información en la red y, por lo tanto, mejora la calidad de servicio.



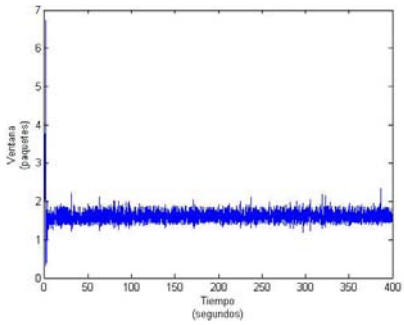
(a) Retardo con Dinámica Senoidal



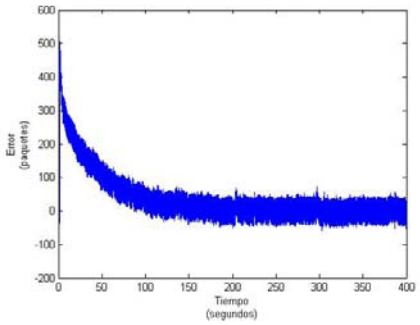
(b) Dinámica de la Cola del Router



(c) Probabilidad



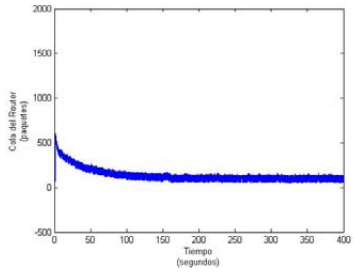
(d) Ventana



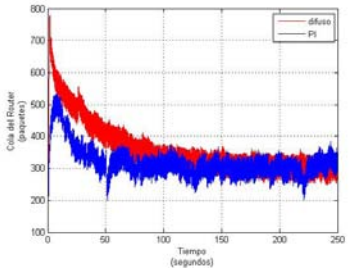
(e) Error

Fig. 4.5: Respuesta del Sistema con Retardo Tipo Senoidal con Controlador Difuso

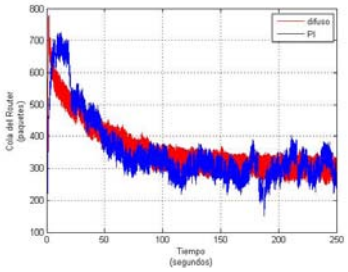
Se hizo la comparación con controladores PI robustos propuestos en [Melchor and Castillo, 2007], en todas las simulaciones, notando que aquellos PI que no aparecen es porque fueron inestables. Se tomaron los mejores PI de forma hurística entre la familia estabilizante. En la Fig. 4.6 se presentan estas comparaciones de diferentes comportamientos de la cola utilizando los PI y la obtenida con el controlador difuso. En la Fig. 4.6(b) y (c) se observa mejor respuesta con el PI en sobrepaso y el tiempo en el que alcanza el valor deseado, en (d),(e) y (f) evidentemente se tiene mejor respuesta con el difuso por el sobrepaso que presentan los PI, así como valores negativos, lo cual se refleja teniendo la cola en cero, significando la subutilización del buffer.



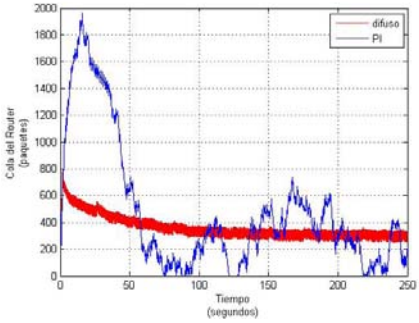
(a) Respuesta con Controlador Difuso



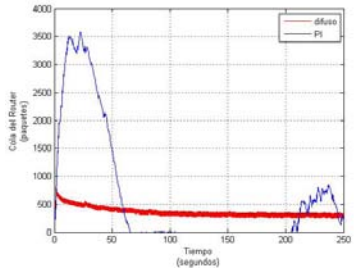
(b) Controlador diseñado para $\tau = 0{,}3$



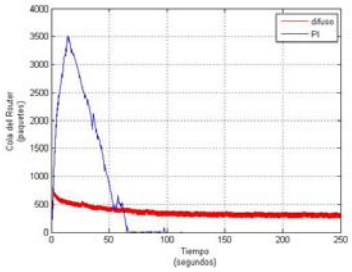
(c) Controlador diseñado para $\tau = 0{,}4$



(d) Controlador diseñado para $\tau = 0{,}6$



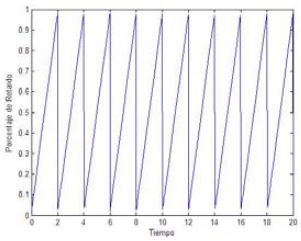
(e) Controlador diseñado para $\tau = 0{,}7$



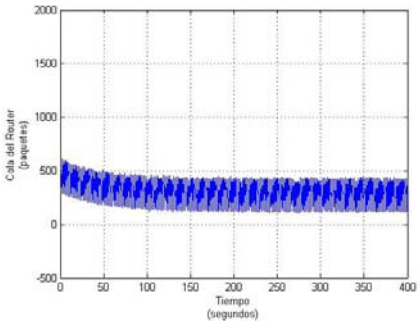
(f) Controlador diseñado para $\tau = 0{,}8$

Fig. 4.6: Comparación de Controladores para Retardo con Dinámica Senoidal

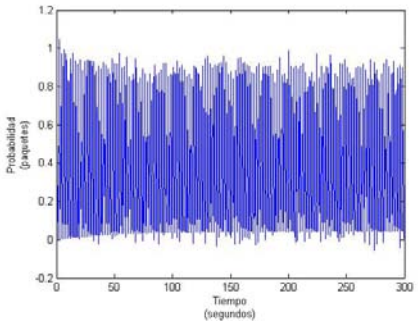
Después de proponer que los cambios de ruta tengan una dinámica tipo senoidal, se probó y comparó al controlador difuso ante un retardo períodico ahora con dinámica más rápida, con un retardo con dinámica tipo diente de sierra. En la Figura 4.7 se muestran las respuestas obtenidas con el controlador difuso. De igual forma, se hizo la comparación con los controladores PI y en la Fig. 4.8 se presentan las respuestas obtenidas. La respuesta del PI en la Fig. 4.8(a) es mejor que la obtenida con el difuso en la magnitud de oscilación y tiempo en el que alcanza el valor deseado. En la Fig. 4.8(b) la mejor respuesta se obtiene con el difuso por el tiempo en el que alcanza el valor deseado. En la Fig. 4.8(c),(d) y (e) el control del difuso es evidentemente mejor que el de los PI por los sobrepasos y la subutilización de la cola.



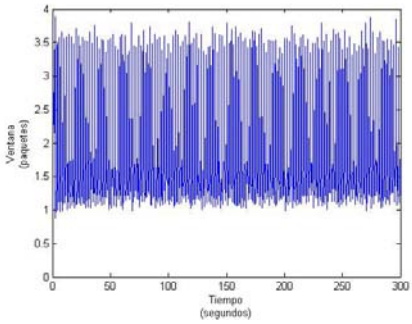
(a) Retardo con Dinámica Diente de Sierra



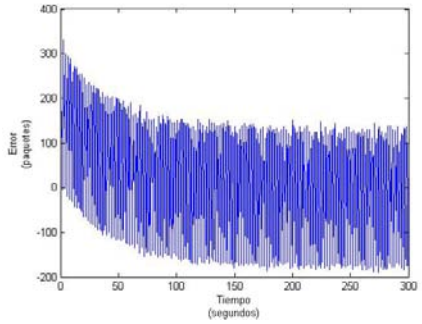
(b) Dinámica de la Cola del Router



(c) Probabilidad

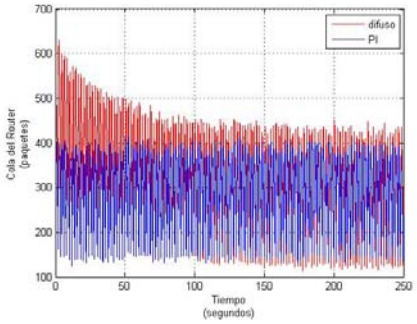


(d) Ventana

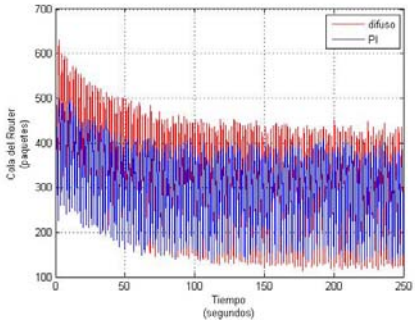


(e) Error

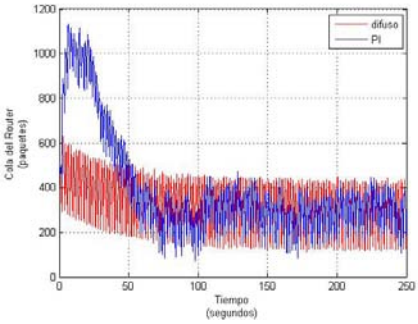
Fig. 4.7: Respuesta del Sistema con Retardo Diente de Sierra con Controlador Difuso



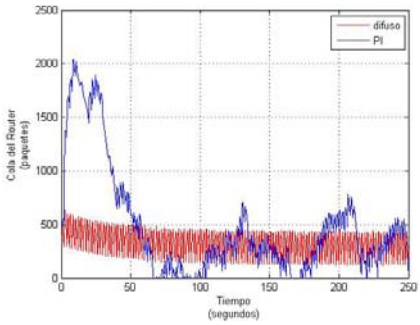
(a) Controlador diseñado para $\tau = 0{,}3$



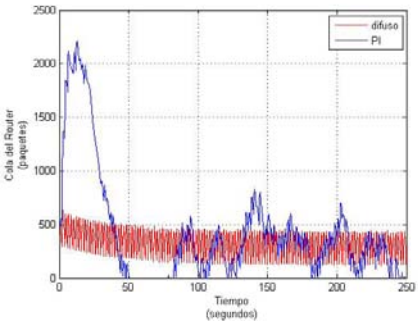
(b) Controlador diseñado para $\tau = 0{,}4$



(c) Controlador diseñado para $\tau = 0{,}6$



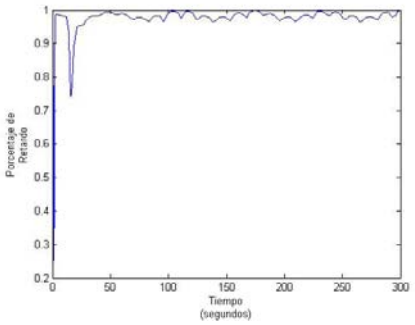
(d) Controlador diseñado para $\tau = 0{,}7$



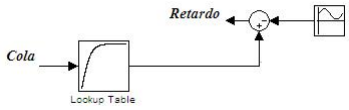
(e) Controlador diseñado para $\tau = 0{,}8$

Fig. 4.8: Dinámica de la Cola con Distintos Controladores Robustos para Retardo con Dinámica Diente de Sierra

Debido a que la motivación del trabajo de investigación es establecer una relación entre el tamaño de la cola y el retardo en las rutas para establecer la asignación de ruta, se hizo una aproximación mediante una relación no lineal entre la cola y el retardo mostrada en la fig. 4.9(a) donde se puede observar que al mantener a la cola en un valor deseado daría un retardo constante en la ruta elegida, con la suposición fuente-router, es decir, no hay más routers que contribuyan a la generación de retardos. Esto se simuló mediante una relación no lineal en el bloque *Lookup Table* de Simulink. También se supuso una contribución en la variación del retardo mostrado en la Fig. 4.9(b). La dinámica de la cola obtenida con el controlador difuso se muestra en la Fig. 4.10 y en la Fig. 4.12 las comparaciones, donde se puede observar un mejor control de la cola con el difuso por las oscilaciones aunque, presenta un sobrepaso grande que, sin embargo, el tamaño de la cola lo soportaría.



(a) Retardo con Dinámica Afectada por la Cola

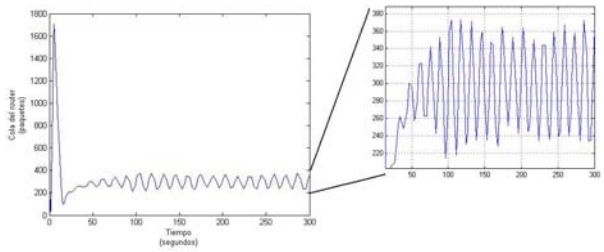


(b) Bloque de Simulink

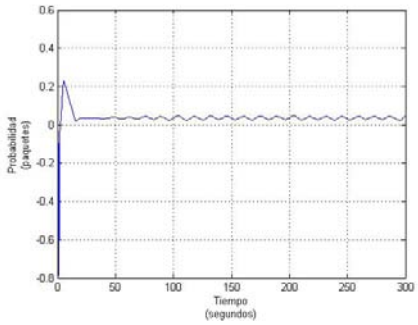
Fig. 4.9: Retardo Afectado por la Cola

Cola	Retardo
0	0
1	0.01
2	0.02
3	0.03
4	0.04
5	0.05
6	0.0599
7	0.0699
8	0.0798
9	0.0898
⋮	⋮
100	0.7616
⋮	⋮
110	0.8005
120	0.8337
130	0.8617
140	0.8854
150	0.9051
160	0.9217
170	0.9354
180	0.9468
190	0.9562
200	0.9640
⋮	⋮
300	0.9951
400	0.9993
500	0.9999
600	1
700	1

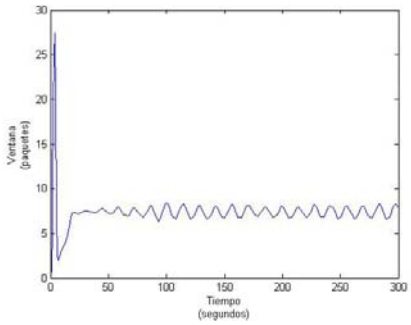
Tab. 4.1: Relación no Lineal Cola-Retardo



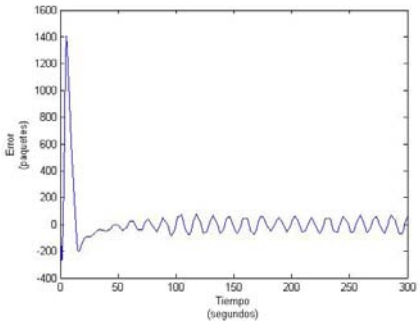
(a) Dinámica de la cola del router



(b) Probabilidad

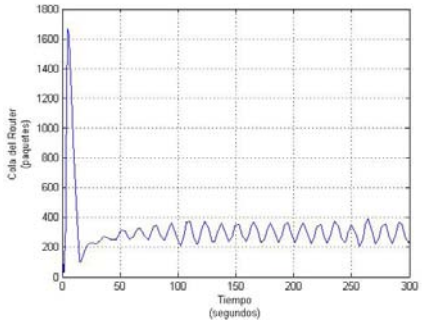


(c) Ventana

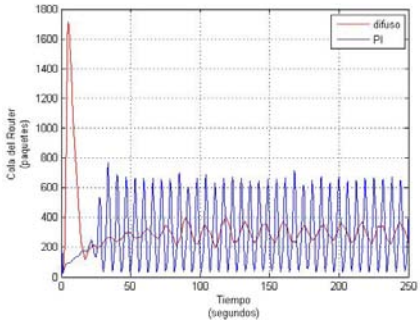


(d) Error

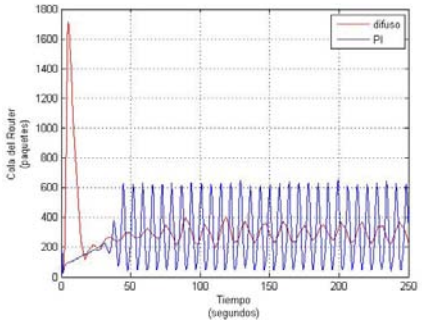
Fig. 4.10: Respuesta del Sistema con Retardo Afectado por la Cola del Router



(a) Respuesta Obtenida con el Controlador Difuso

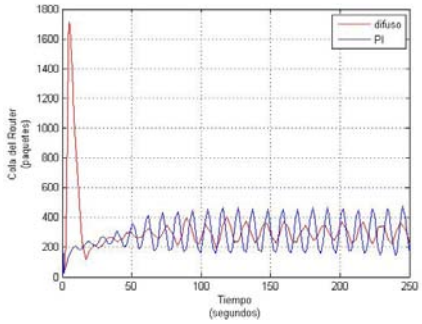


(b) Respuesta con un PI para $\tau = 0{,}3$

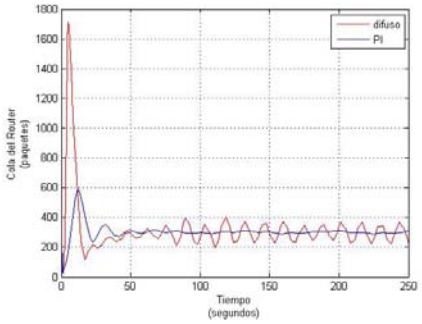


(c) Respuesta con un PI para $\tau = 0{,}4$

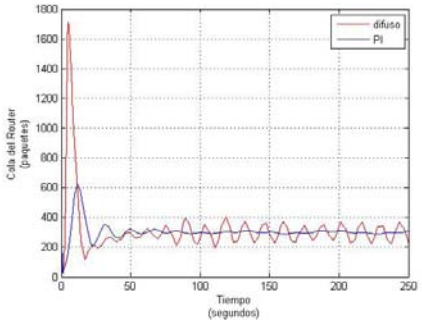
Fig. 4.11: Respuestas ante un Retardo Afectado por la Cola



(a) Respuesta con un PI para $\tau = 0{,}6$



(b) Respuesta con un PI para $\tau = 0{,}7$



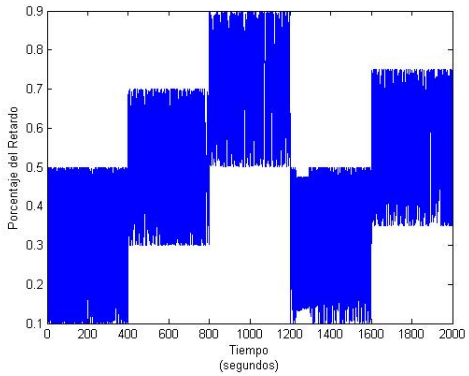
(c) Respuesta con un PI para $\tau = 0{,}8$

Fig. 4.12: Respuestas ante un Retardo Afectado por la Cola

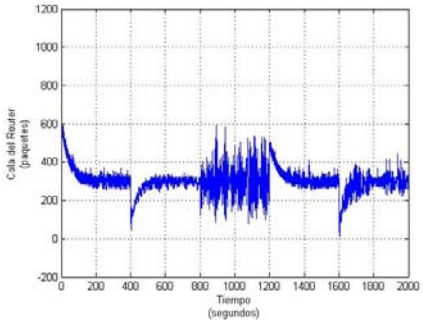
Hasta aquí se ha simulado los cambios de ruta con sus respectivos retardos de forma muy suave, sin embargo, para tener una mejor aproximación se propuso cambios de rutas abruptas que poseen retardos con pequeñas oscilaciones periódicas. También se realizaron las mismas pruebas anteriormente descritas.

La respuesta del controlador difuso con los diferentes retardos correspondientes a las diferentes rutas se presentan a continuación comparándola, como se ha venido haciendo, con los PI robustos de [Melchor and Castillo, 2007]. En las Figuras 4.13 y 4.14 se presenta la dinámica de la cola cuando la dinámica de los

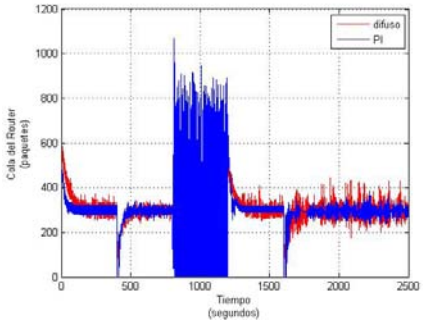
retardos en las rutas son periódicas del tipo senoidal. En estos cambios de rutas con retardos variables en cada ruta es donde se presenta mejor respuesta con el controlador difuso. En (c) y (d) se observan dinámicas parecidas pero co oscilaciones de mayor magnitud para los PI. Para el caso (e) y la Fig. 4.14 se dan picos muy grandes lo que podría interpretarse como saturaciones de la cola e incluso se observan subutilizaciones de la cola.



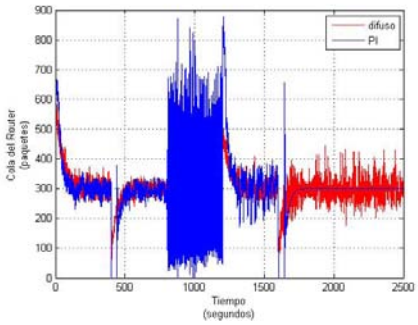
(a) Retardo



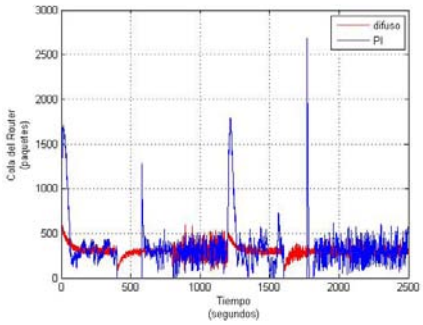
(b) Cola Obtenida con el Controlador Difuso



(c) Cola Obtenida con PI para $\tau = 0{,}3$

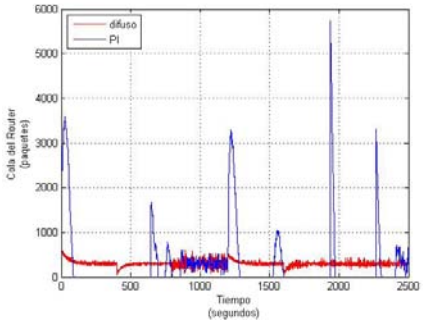


(d) Cola Obtenida con PI para $\tau = 0{,}4$



(e) Cola Obtenida con PI para $\tau = 0{,}6$

Fig. 4.13: Respuesta de los Controladores ante Cambios de Rutas con Retardo Periódico

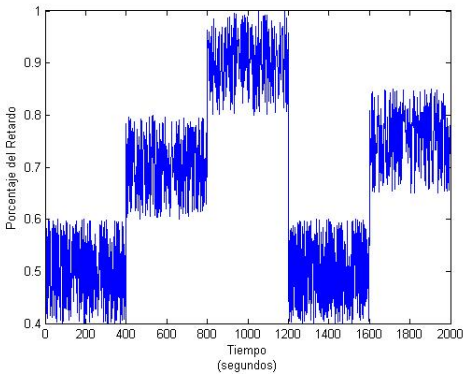


(a) Cola Obtenida con PI para $\tau = 0{,}7$

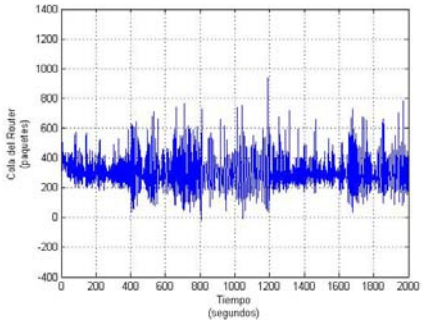
Fig. 4.14: Respuesta de los controladores ante Cambios de Rutas con Retardo periódico-2

Por otro lado, en la Fig. 4.15 se muestran los resultados obtenidos ante un retardo aleatorio uniformemente distribuido en las diferentes rutas. No se presenta la respuesta obtenida con el PI para un retardo de $\tau = 0{,}3$ porque fue inestable. Las mejores respuestas son los casos (d), (e) y (f) presentando dinámicas parecidas a la del contolador difuso. Para (d) se muestran oscilaciones grandes ante cambios grandes de retardo en las rutas elegidas, de lo contrario presentaría mejor comportamiento que con el difuso. En (e) y (f) se presentan incrementos de la cola más grandes que los obtenidos con el difuso y subutilización de ésta que no se tienen con el difuso.

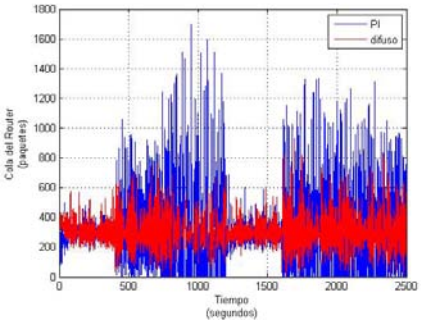
También se probó la respuesta del sistema con los controladores suponiendo que los retardos en las rutas elegidas por el protocolo de ruteo fuesen constantes y se obtuvieron las respuestas mostradas en Fig. 4.16. Se puede observar que con el controlador difuso se obtuvo la mejor respuesta en todos los casos. Los PI que no se muestran es porque fueron inestables.



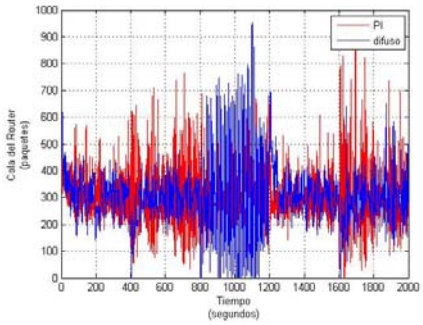
(a) Retardo



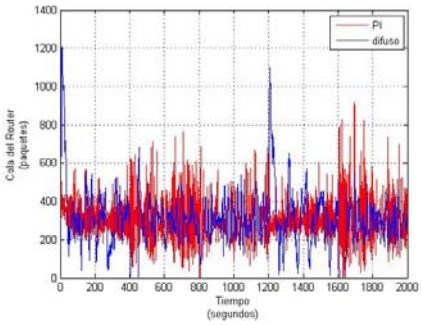
(b) Respuesta del Controlador Difuso



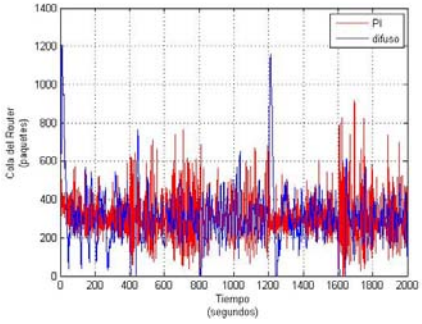
(c) Respuesta del PI para $\tau = 0{,}4$



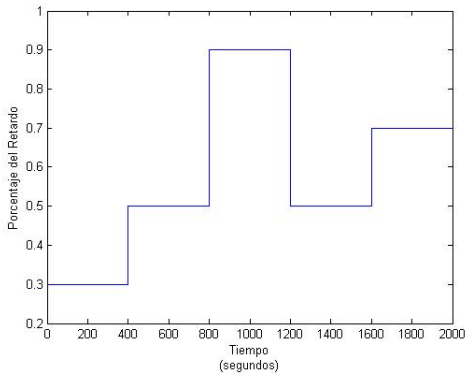
(d) Respuesta del PI para $\tau = 0{,}6$



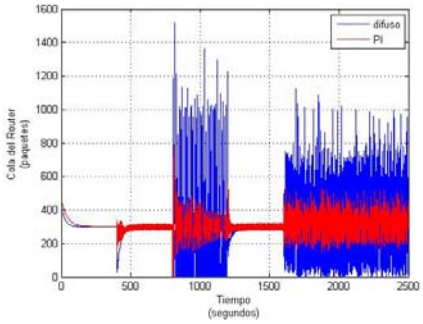
(e) Respuesta del PI para $\tau = 0{,}7$



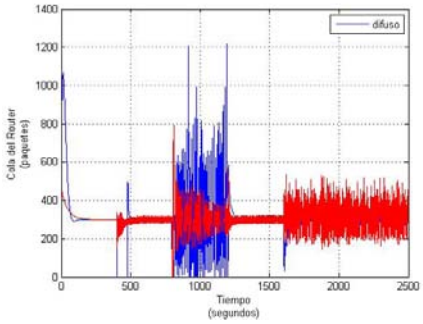
(f) Respuesta del PI para $\tau = 0{,}8$



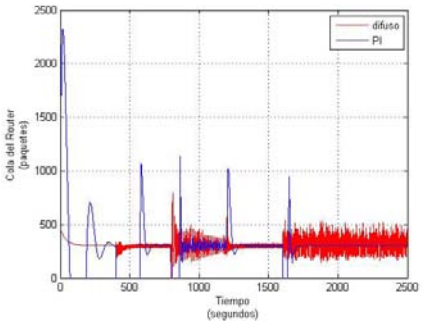
(a) Retardo



(b) Dinámica de la cola con PI para $\tau = 0{,}4$



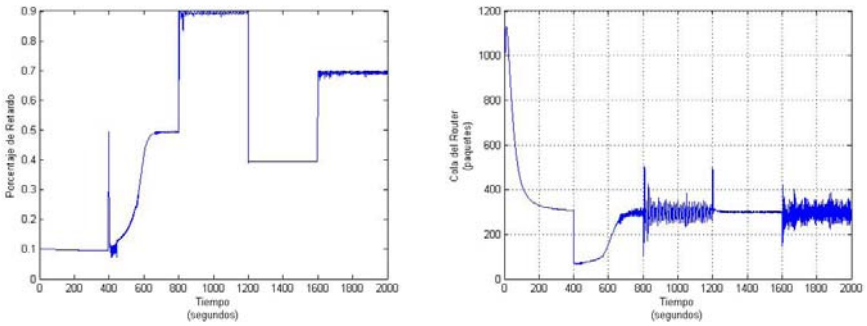
(c) Dinámica de la cola con PI para $\tau = 0{,}6$



(d) Dinámica de la cola con PI para $\tau = 0{,}7$

Fig. 4.16: Respuestas ante Retardo Constante en las Diferentes Rutas

De igual forma se propuso un retardo como una relación no lineal de la cola del router teniendo en cuenta los cambios de rutas asignadas por el protocolo de ruteo en la Figura 4.17, donde todos los PI resultaron inestables. Se puede ver en (a) que los cambios son abruptos teniendo pequeñas oscilaciones de retardo en las rutas. El comportamiento de la cola con el controlador difuso es bueno ya que no se tienen oscilaciones grandes a pesar de tener un sobrepaso grande pero sin llegar valores que se podrían considerar saturación de la cola.



(a) Retardo (b) Respuesta con el Controlador Difuso

Fig. 4.17: Dinámica de la Cola ante Retardo Afectado por el Comportamiento de la Cola en Diferentes Rutas

5. CONCLUSIONES Y TRABAJOS FUTUROS

En la literatura se han desarrollado diferentes tipos de controladores para una red de cómputo con resultados cada vez mejores, sin embargo la suposición que han hecho es que el retardo de propagación es constante, lo que está bastante alejado a la realidad. En este trabajo de tesis se propuso relajar más esa suposición para llevar a cabo un trabajo más apegado a la realidad, un retardo con dinámica relacionada al cambio de ruta realizado por el protocolo de ruteo.

Se abordó el problema de la influencia del ruteo en el control de congestión con base en la caracterización del retardo.

Controlamos la congestión en la red teniendo en cuenta que el retardo es variable mediante el diseño de un controlador difuso utilizando como consecuentes los controladores más usados en la industria, un conjunto de contoladores PIs. Con base a una idea bastante sencilla del controlador difuso, se pudieron observar buenos resultados.

Se diseñó un controlador para un sistema no lineal con retardos variables de dinámicas específicas con buenos resultados.

Se tiene como trabajo futuro el estudio de la tabla de ruteo para establecer una aproximación al modelo de la misma.

El diseño de controladores difusos podrían mejorar la respuesta de la cola del router, quiza realizando un PI difuso con restablecimiento como en [Díaz et al., 1997] o un controlador neuro-difuso, sintonización del controlador difuso utilizando técnicas clásicas para PIDs [Dormido et al.,], etc.

Así también podría hacerse la estimación del retardo, haciendo un estudio y aplicación de [Belkoura et al.,] con lo que se podrián tener buenos resultados.

Utilizar un simulador de red más real, como el "ns-2", para probar el funcionamiento del controlador es un trabajo futuro interesante.

Poder llevar a la práctica los trabajos de simulación del controlador propuesto para establecer una eficiente utilización de los recursos de la red sería un trabajo interesante.

Bibliografía

[Abonyi, 2003] Abonyi, J. (2003). "Fuzzy model identification for control". *Birkhäuser Boston*.

[Belkoura et al.,] Belkoura, L.; Jean-Pierre; and Fliess, M. "Real Time Identification of Delay Systems".

[Carrero, 2004] Carrero, N. (2004). "Reglas de ajuste del control PI-AQM". *Proyecto de grado EISULA, Universidad de los Andes*.

[Comer, 2000] Comer, D. (2000). "Internetworking with TCP/IP Volumen I: Principles, Protocols and architecture". *Prentice Hall*.

[D. Driankov and Reinfrank, 1993] D. Driankov, H. H. and Reinfrank, M. (1993). "An introduction to Fuzzy Control". *Springer-Verlag, Heildelberg, Germany*.

[Díaz et al., 1997] Díaz, H.; Borjas, R.; and A., S. (1997). "Diseño y ensayo de un controlador difuso para un motor de inducción trifásico". *Revista Facultad de Ingeniería, UTA, Chile*, 4.

[Deb and Srikant, 2006] Deb, S. and Srikant, R. (2006). "Rate-based versus queue-based models of congestion control". *IEEE Transactions on Automatic Control*, 51, pp. 606–619.

[Dormido et al.,] Dormido, S.; Santos, M.; Pérez, A.; and Morilla, F. "Autosintonía de controladores difusos utilizando técnicas clásicas basadas en reguladores PID". *http://www.dacya.ucm.es/msantos/papers/flat93.pdf*.

[Filipiak, 1988] Filipiak, J. (1988). "Modelling and control of dynamic flows in communication networks". *Berlin:Springer*.

[F.L. Lian and Tilbury, 2001] F.L. Lian, J. M. and Tilbury, D. (2001). "Performance Evaluation of Control Networks: Ethernet, Controlnet, and DeviceNet". *IEEE Control Systems Magazine*, 21(1), pp. 66–83.

[Floyd and Jacobson, 1993] Floyd, S. and Jacobson, V. (1993). "Ramdom early detection gateways for congestion avoidance". *IEEE/ACM Transsactions Networking*, 1, pp. 397–413.

[Forouzan, 2000] Forouzan, B. (2000). "TCP/IP: Protocol Suite". *McGraw-Hill*.

[Gerla et al., 2002] Gerla, M.; Cigno, R. L.; Mascolo, S.; and Weng, W. (2002). "Generalized window advertising for TCP congestion control". *Eur. Trans. Telecommun.*, 13, pp. 549–562.

[Gerla et al., 2000] Gerla, M.; R., W. W.; and Cigno, L. (2000). "Bandwidth feedback control of TCP and real time sources in the internet". In *Proc. IEEE Global Internet Symp., Globecome, San Francisco, CA,USA*.

[Gu et al., 2003] Gu, K.; Kharitonov, V.; and Chen, J. (2003). "Stability of Time-Delay Systems". *Birk auser,Boston*.

[Halevi and Ray, 1988] Halevi, Y. and Ray, A. (1988). "Integrated communica-

tion and Control systems: Part I- Analysis". *Journal of Dinamic Systems, Measurement and Control*, 110, pp. 367–373.

[Hassan and Jain, 2004] Hassan, M. and Jain, R. (2004). "High performance TCP/IP Networking: concepts, issues and solutions". *Prentice Hall*.

[Hollot et al., 2001] Hollot, C.; Misra, V.; Towsley, D.; and Gong, W. (2001). "A Control Theoretic Analysis of RED". *Proc. IEEE INFOCOM*, 3, pp. 1510–1519.

[Hollot et al., 2002] Hollot, C.; Misra, V.; Towsley, D.; and Gong, W. (2002). "Analysis and design of controllers for AQM routers supporting TCP flows". *IEEE Transactions on Automatic Control*, 47, pp. 945–959.

[Jacobsson et al., 2006] Jacobsson, K.; Hjalmarsson, H.; and Johasson, K. (2006). "Towards accurate congestion control models: Validation and stability analysis". In *Mathematical theory of networks and Systems*, pages 672–682.

[Kaplan, 2001] Kaplan, G. (2001). "EthernetŽs winning ways". *IEEE Spectrum*, 38(1), pp. 113–115.

[Kelly, 2000] Kelly, F. (2000). "Models for a self-managed Internet". *Philos. Trans. Royal Soc.*, A358, pp. 2335–2348.

[Kelly, 2001] Kelly, F. (2001). *Mathematical modeling of the internet*. Springer-Verlag.

[Kelly et al., 1998] Kelly, F.; Maullo, A.; and Tan, D. (1998). "Rate control in communication networks: shadow prices, proportional fainess and stability". *Journal of the Operational Research Society*, 49, pp. 237–252.

[Kurose and Ross, 2005] Kurose, J. and Ross, K. (2005). "Computer Networking A Top-Down Approach Featuring the Internet". *Pearson Addison Wesley*, 3^{rd}Edition.

[L.A., 1973] L.A., Z. (1973). "Outline of a new approach to the analysis of complex systems and decision processes". *IEEE Trans. SMC*, SMC-3, pp. 28–44.

[li Lian et al., 1999] li Lian, F.; Moyne, J. R.; and Tilbury, D. M. (1999). "Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet". *IEEE Control Systems Magazine*, 21, pp. 66–83.

[Lian et al., 2001] Lian, F.-L.; Moyne, J.; and Tilbury, D. (2001). "Analysis and modeling of networked control systems: MIMO case with multiple time delays". volume 6, pages 4306–4312 vol.6.

[Long C. and Yang, 2005] Long C., Zhao, B. G. X. and Yang, J. (2005). "The yellow active queue management algorithm". *http:www.sciencedirect.com*.

[Low and Lapsey, 1999] Low, S. and Lapsey, D. (1999). "Optimization flow control i:Basic algorithm and convergence". *IEEE/ACM Transsactions on Networking*, 7, pp. 861–874.

[Low et al., 2002] Low, S.; Paganini, F.; and Doyle, J. (2002). "Internet congestion control". *IEEE Control Syst. Mag.*, 22, pp. 28–43.

[Mamdani, 1975] Mamdani, E. (1975). "An experiment in linguistic synthesis with a fuzzy logic controller". *Int. J. Man Mach. Studies*, 7, pp. 1–13.

[Melchor and Castillo, 2007] Melchor, D. and Castillo, V. (2007). "Stability

analysis of Proportional-Integral AQM controllers supporting TCP flows". *Computación y sistemas*, 10, pp. 401–414.

[Michiels et al., 2006] Michiels, W.; Melchor-Aguilar, D.; and Niculescu, S. (2006). "Stability Analisys of some classes of TCP/AQM networks". *International journal of Control*, 9(79), pp. 1136–1144.

[Misra V. and D., 2000] Misra V., G. W. and D., T. (2000). "Fluid-based analysis of a network of AQM routers, suporting TCP flows with an application to RED". *Proceedings of ACM/SIGCOMM,Estocolmo,Suecia*.

[Ogata, 1998] Ogata, K. (1998). "Ingeniería de Control Moderna". *Prentice-Hall Hispanoamericana SA*.

[Passino and Yurkovich, 1998] Passino, K. M. and Yurkovich, S. (1998). "Fuzzy Control". *Addison Wesley Longman Inc.*

[Postel, 1981] Postel, J. (1981). "Transmission Control Protocol". *RFC793*, September.

[Quet and Ozbay, 2004] Quet, P.-F. and Ozbay, H. (2004). "On the design of AQM supporting TCP flows using robust control theory". *IEEE Transsactions on Automatic Control*, 49, pp. 1031–1036.

[S. Shakkottai and Anvekar, 2001] S. Shakkottai, A. Kumar, A. K. and Anvekar, A. (2001). "TCP performance over end-to-end rate control and stochastic available capacity". *IEEE/ACM Transactions on Networking*, 9(4), pp. 377–391.

[Saltzer et al., 1981] Saltzer, J.; Reed, D.; and Clark, D. (1981).

”End to end arguments in systems design”. Disponible en http://www.web.mit.edu/Saltzer/www/publications/endtoend/endtoend.pdf.

[Srikant, 2004] Srikant, R. (2004). ”The Mathematics of Internet Congestion Control”. *Birkhauser*, Berlin, Germany:Birkhauser.

[Takagi and Sugeno, 1985] Takagi and Sugeno (1985). ”Fuzzy identification of systems and its application to modeling and control”. *IEEE Trans. Syst. Man Cybern.*, SMC-15, pp. 116–132.

[Tang et al., 2007] Tang, A.; Jacobsson, K.; Andrew, L.; and Low, S. (2007). ”An accurate link model and its aplication to stability analysis of FAST TCP”. In *IEEE INFOCOM*.

[Trick, 1998] Trick, M. A. (1998). ”Shortest Path”. *Carnegie Mellon University*, Computer Science Department.

[V., 1998] V., J. (1998). ”Congestion avoidance and Control”. *http://citeseer.ist.psu.edu/article/jacobson88congestion.html.*

[Yasunobu and Miyamoto, 1985] Yasunobu, s. and Miyamoto, S. (1985). ”Automatic Train operation by predictive fuzzy control”. *Industrial applications of fuzzy control*, Sugeno (Ed.), pp. 1–18.

[Yilmaz and Matta, 2002] Yilmaz, S. and Matta, I. (2002). ”On the Scalability-performance tradeoff in MPLS and IP routing.”. *Proceedings of SPIE ITCOM: Scalability and Traffic Control in IP Networks, Boston, MA*.

[Zampieri, 2008] Zampieri, S. (2008). ”Trends in Networked Control Systems”.

In *Proceedings of the 17th World Congress The International Federation of Automatic Control*, pages 2886–2894.

Apéndice

```
function cpi = cdifuso(u)

e=u(1);
de=u(2);
T=u(3);
inte=u(4);
val=.4;


kp=[2 .009 1e-3 .3e-3 1e-4 .8e-4];
ki=[10 57 29 8 12 18];

mf =
[(1/(1+exp(23*(T-0.03)))),exp(-((T-0.1)/val)^2),
exp(-((T-0.2)/val)^2),exp(-((T-0.4)/val)^2),
exp(-((T-0.6)/val)^2),exp(-((T-0.8)/val)^2),
exp(-((T-0.9)/val)^2),(1/(1+exp(-23*(T-.8))))];



y =
(mf(1)*((kp(5)*e)+((kp(5)/ki(5))*inte))) +
(mf(2)*((kp(2)*e)+((kp(2)/ki(2))*inte))) +
(mf(3)*((kp(3)*e)+((kp(3)/ki(3))*inte))) +
(mf(4)*((kp(3)*e)+((kp(3)/ki(3))*inte))) +
```

```
(mf(5)*((kp(5)*e)+((kp(5)/ki(5))*inte))) +
(mf(6)*((kp(5)*e)+((kp(5)/ki(5))*inte))) +
(mf(7)*((kp(5)*e)+((kp(5)/ki(5))*inte))) +
(mf(8)*((kp(6)*e)+((kp(6)/ki(6))*inte)));


p =mf(1)+ mf(2)+mf(3)+mf(4)+mf(5)+mf(6)+mf(7)+mf(8);



cpi=y/p;
```