



Universidad de Castilla-La Mancha
Escuela Superior de Ingeniería Informática

Trabajo Fin de Grado
Grado en Ingeniería Informática
Tecnología específica de Ingeniería de Computadores

**Modelado del protocolo RDMA en un simulador
de redes de interconexión basado en
OMNeT++**

Víctor Manuel Romero Ramírez

29 de junio de 2021



TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Tecnología específica de Ingeniería de Computadores

Modelado del protocolo RDMA en un simulador de redes de interconexión basado en OMNeT++

Autor: Víctor Manuel Romero Ramírez

Tutor: Jesús Escudero Sahuquillo

Co-Tutor: Francisco José Alfaro Cortés

29 de junio de 2021

*A todas las personas que me han ayudado a llegar hasta aquí, sobre todo a mi familia y a
mis amigos*

Declaración de autoría

Yo, Víctor Manuel Romero Ramírez con DNI 49215454 , declaro que soy el único autor del trabajo fin de grado titulado "Modelado del protocolo RDMA en un simulador de redes de interconexión basado en OMNeT++", que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual, y que todo el material no original contenido en dicho trabajo está apropiadamente atribuido a sus legítimos autores.

Albacete, a 29 de junio de 2021

Fdo.: Víctor Manuel Romero Ramírez

Resumen

RDMA es un concepto que usa DMA, es decir, permite a los dispositivos de la placa base de la computadora enviar datos directamente a memoria cuando dos o más computadoras se comunican, accediendo a la memoria de otro host directamente desde la memoria de un host.

Emplearemos la herramienta OMNeT++ que es una biblioteca y un marco de simulación C++ extensible, modular y basado en componentes, principalmente para la construcción de simuladores de red. Aunque OMNeT++ no es un simulador de red en sí mismo, ha ganado una gran popularidad como plataforma de simulación de red en la comunidad científica pues proporciona una arquitectura de componentes para modelos. Los componentes (módulos) se programan en C++ y luego se ensamblan en componentes y modelos más grandes utilizando un lenguaje de alto nivel (NED). La reutilización de modelos es gratuita. OMNeT++ tiene un amplio soporte de GUI y, debido a su arquitectura modular, el kernel de simulación (y los modelos) se pueden integrar fácilmente en sus aplicaciones. [Omn, 2019]

Teniendo en cuenta lo anterior, este TFG tratará de modelar RDMA en OMNeT++, pues la herramienta no tiene esta funcionalidad y compararlo con TCP/IP con el objetivo de demostrar que tiene un mejor funcionamiento

Agradecimientos

Me gustaría agradecer a mis padres, por haberme pagado la carrera y haberme apoyado en todo momento con todo lo relacionado a los estudios, a todos las personas que he conocido a lo largo de mi vida, tanto a aquellas que te ayudan a crecer estando a tu lado, como a aquellas que suponen un reto para ti y te ayudan a superar momentos y superarte a tí mismo. Por supuesto, también agradecer a todos los profesores que me han ayudado a llegar hasta aquí y, sobre todo, a Jesús y a Paco, por confiar en mí y proponerme la realización de este TFG.

Índice general

1	Introducción	1
1.1	Motivación	2
1.2	Objetivos	3
1.3	Estructura de la memoria	3
2	Antecedentes y estado de la cuestión	5
2.1	Redes de interconexión	5
2.1.1	<i>Topología</i>	6
2.1.1.1	Redes de conexión de medio compartido	7
2.1.1.1.1	Redes de área local	7
2.1.1.1.1.1	Bus de contención (CSMA/CD)	8
2.1.1.1.1.2	Token Bus	8
2.1.1.1.1.3	Token Ring	8
2.1.1.2	Redes de conexión directas o estáticas	9
2.1.1.2.1	Estrella	9
2.1.1.2.2	Anillo	9
2.1.1.2.3	Malla	10
2.1.1.2.4	Toro	10
2.1.1.2.5	Hipercubo	11
2.1.1.3	Redes de conexión indirectas o dinámicas	11
2.1.1.3.1	Redes en bus	11
2.1.1.3.2	Redes multietapa	12
2.1.1.3.2.1	Redes Omega	12
2.1.1.3.2.2	Redes de línea base	13
2.1.1.3.2.3	Redes CLOS	13
2.1.1.3.2.4	Red de Benes	14
2.1.1.3.3	Crossbar switches	14
2.1.1.4	Redes de conexión híbridas	15

2.1.2	<i>Estructura organizacional</i>	15
2.1.3	<i>Encaminamiento</i>	16
2.1.3.1	Problemas relacionados con el encaminamiento	17
2.1.3.1.1	Deadlock	17
2.1.3.1.2	Livelock	18
2.1.3.1.3	Starvation	18
2.1.4	<i>Diseño del switch</i>	18
2.1.5	<i>Control de flujo</i>	19
2.1.5.1	Basado en créditos	19
2.1.5.2	Basado en marcas (Stop & Go)	19
2.2	Protocolo de transporte Remote Direct Memory Access (RDMA)	20
2.2.1	<i>Ventajas de RDMA</i>	21
2.2.2	<i>Protocolos de red compatibles con RDMA</i>	21
2.3	Entorno de simulación Omnet++.	22
3	Metodología y desarrollo	23
4	Experimentos y resultados	25
5	Conclusiones	27
5.1	Configuración.	27
5.2	Colores	27
5.3	Listas y enumeraciones	28
5.4	Elementos flotantes	29
5.4.1	<i>Tablas</i>	29
5.4.2	<i>Figuras</i>	29
5.5	Algoritmos y código	31
5.6	Bibliografía	33
5.7	Notación matemática.	33
6	Marcas y ayudas	35
6.1	Figuras	35
A	Anexo 1	37
	Referencia bibliográfica	39

Índice de figuras

2.1	Red de Interconexión	6
2.2	Red de área local	7
2.3	Topología token bus	8
2.4	Topología token ring	8
2.5	Topología estrella	9
2.6	Topología anillo	10
2.7	Topología Malla	10
2.8	Topología toro	11
2.9	Topología hipercubo	11
2.10	Red en bus	12
2.11	Topología multietapa	13
2.12	Topología de línea base de 8 entradas	13
2.13	Topología CLOS	14
2.14	Topología de Benes	14
2.15	Switches transversales	15
2.16	Topología híbrida	15
2.17	Deadlock	17
2.18	Arquitectura interna de un switch	19
2.19	RDMA	20
5.1	Escuela Superior de Ingeniería Informática I	30
5.2	Escuela Superior de Ingeniería Informática II	30
5.3	Ejemplo de subfiguras	31

Índice de tablas



5.1 Asignaturas del Módulo I 31

Índice de algoritmos

5.1	Tree-Search exploration	32
-----	-----------------------------------	----

Índice de listados de código

5.1	Ejemplo de código en Python	32
-----	---------------------------------------	----

1. Introducción

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

1.1. Motivación

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque

tesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

1.2. Objetivos

El objetivo de este Trabajo de Fin de Grado es el de modelar el protocolo de transporte RDMA en el modelo INET del simulador Omnet++ y realizar las comparaciones necesarias para demostrar la diferencia de rendimiento de una red según usemos RDMA o TCP/IP. Para ello vamos a seguir los siguientes pasos.

1. Conocer el estado del arte de las redes de interconexión en general, y del protocolo RDMA en particular.
2. Estudiar las herramientas de simulación existentes basadas en OMNeT++, como el modelo INET que ya dispone del protocolo de transporte TCP/IP.
3. Modelar el protocolo RDMA en el modelo INET de OMNeT++.
4. Realizar experimentos de simulación comparando varias configuraciones de red de interconexión que usen TCP/IP y RDMA, y hacer un estudio comparativo de las prestaciones

1.3. Estructura de la memoria

La memoria esta organizada de la siguiente manera:

- **Introducción:**
- **Antecedentes y estado de la cuestión:** se enfoca en ofrecer una introducción sobre la redes de interconexión. En ella se informa sobre las diferentes topologías de redes de interconexión y características específicas y generales de ellas, también se indican los componentes de una red de interconexión, la arquitectura interna del switch, los diferentes tipos de encaminamiento de los mensajes por la red y problemas relacionados con ellos y control de flujo. También nos enfocaremos en el que será la prioridad de este trabajo, el protocolo de transporte RDMA. Por último, se describen las ideas principales del simulador Omnet++.
- **Metodología y desarrollo:**
- **Experimentos y resultados:**
- **Conclusiones:**

2. Antecedentes y estado de la cuestión

En este capítulo vamos a conocer el estado del arte desde un punto de vista general de las redes de interconexión y, dentro de todos los protocolos de transportes que existen y son usados en estas, vamos a centrarnos en el protocolo RDMA (Remote Direct Memory Access) que nos permite transferir datos de forma más rápida a los protocolos tradicionales como TCP/IP. Una vez desarrollados ambos apartados, vamos a conocer también el entorno que utilizaremos para la simulación del tráfico empleando el mismo escenario y variando entre los dos protocolos de transporte que hemos nombrado y a partir de los cuales obtendremos unos datos que nos servirán para realizar una comparativa del rendimiento de ambos protocolos.

2.1. Redes de interconexión

Las arquitecturas de computación de alto rendimiento utilizan redes de interconexión como base de comunicación entre los núcleos del procesador, los módulos de memoria y los dispositivos de E/S. En su forma más general, las redes de interconexión son un componente central de todos los sistemas informáticos y de comunicación, desde las interconexiones internas de las arquitecturas integradas a nivel de chip hasta los sistemas a escala geográfica, como las redes WAN e Internet.

El objetivo de todas las redes de interconexión es transferir información desde cualquier nodo origen a cualquier nodo destino que se desee con la menor latencia posible y permitiendo un gran número de transferencias en ejecución simultánea.

Estas redes están compuestas por enlaces y conmutadores, que ayudan a enviar la información entre ambos nodos. Una red está especificada por su topología, algoritmo de enrutamiento, estrategia de conmutación y mecanismo de control de flujo.[Int, 2021]

En la figura 2.1 podemos ver seis terminales conectados a una red. Si uno de esos terminales quiere comunicarse con otro de los terminales, envía el mensaje del terminal origen con los datos a través de la red y esta entrega el mensaje al terminal destino.

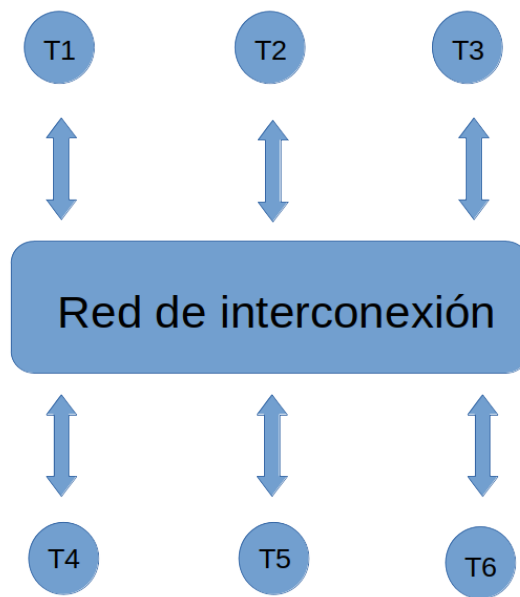


Figura 2.1: Red de Interconexión

2.1.1. Topología

Las redes de interconexión están compuestas por elementos de conmutación. La topología de la red se refiere a la disposición estática de los enlaces y los nodos de una red de interconexión: los caminos por los que viajan los paquetes. Esta puede representarse mediante un grafo $G(N,C)$, donde N es el conjunto de los nodos de la red, y $C:N \times N$ es el conjunto de enlaces que conecta un nodo o switch con sus vecinos.

Para comprobar que una topología es eficiente, deben evaluarse diferentes parámetros:

- **Ancho de bisección:** se refiere al mínimo número de enlaces que hay que eliminar para dividir la red en dos partes iguales. Un ancho de bisección alto puede reducir el tiempo de comunicación cuando el movimiento de los datos es crítico, ya que la información puede viajar por caminos alternos y evitar así o reducir la congestión en ciertos nodos. Además, esto también hace que el sistema sea más tolerante ante fallos, ya que un nodo defectuoso no hace inoperable al sistema.
- **Grado:** se llama grado de un nodo al número de enlaces que tiene con otros nodos. Es conveniente que en una red el grado sea igual para todos los nodos, siendo en ese caso *regular*. Además, el grado está relacionado directamente con el coste, ya que indica el número de puertos que necesita.
- **Distancia:** Número mínimo de nodos que hay que atravesar para viajar de un nodo origen a un nodo destino. La distancia crea latencia y reducir la distancia entre los distintos nodos es la única forma de poder minimizarla.
- **Distancia media:** representa la media de las distancias de cada par de nodos de la red.

- **Diámetro:** el diámetro es, para par de nodos, el camino más largo de todos los caminos posibles de la red, es decir, el mayor número de enlaces que hay que recorrer en la red para comunicar dos de los nodos que están en ella. Un menor diámetro indica mayor habilidad de comunicación en la red.
- **Longitud de los enlaces:** determina la velocidad a la que la red puede operar y la potencia disipada en ella.
- **Simetría:** se refiere a que desde cualquier nodo la vista de la red presenta el mismo aspecto. Esto implica que una red simétrica tiene que ser regular.
- **Regularidad:** una red es regular si todos los nodos tienen el mismo grado. Es conveniente de cara a la modularidad y escalabilidad.
- **Conectividad:** mínimo número de enlaces o nodos que hay que eliminar para obtener dos gráficos independientes. Cuanto mayor sea el número de enlaces que hay que eliminar para poder dividir la red en dos, mayor es la tolerancia de fallo.
- **Escalabilidad:** facilidad con la que la red puede expandirse manteniendo sus prestaciones sin aumentar su coste en gran medida.

Una vez hemos diferenciado los diferentes parámetros que hay que tener en cuenta para comprobar la eficiencia de una red vamos a diferenciar también los diferentes tipos de redes que hay teniendo en cuenta la forma en que se comunican los diferentes nodos.

2.1.1.1. Redes de conexión de medio compartido

Estas redes están formadas por un único elemento común de interconexión al que se conectan directamente todos los nodos de la red. No son muy utilizadas en computación paralela debido a sus bajas prestaciones. Estas redes se pueden clasificar en:[sha,]

2.1.1.1.1. Redes de área local

Permiten comunicar equipos distanciados varios metros entre sí.

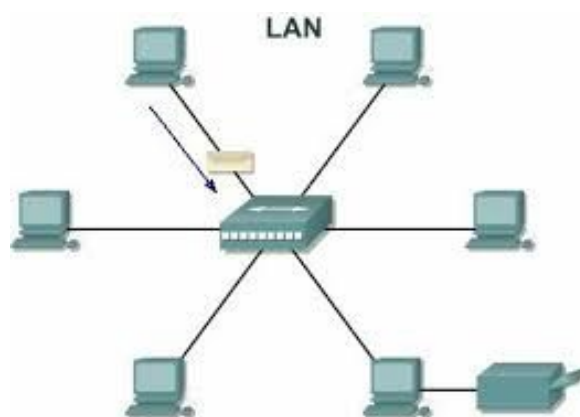


Figura 2.2: Red de área local

2.1.1.1.1. Bus de contención (CSMA/CD)

Acceso al bus compartido mediante competencia entre los dispositivos y resolución de colisiones. Es un algoritmo no determinista, por lo que no se puede garantizar el tiempo de espera de un dispositivo hasta obtener acceso al bus.

2.1.1.1.1. Token Bus

Dispositivos conectados en modo broadcast al bus. Se utiliza un token que circula entre los dispositivos en orden para habilitar el uso del bus.

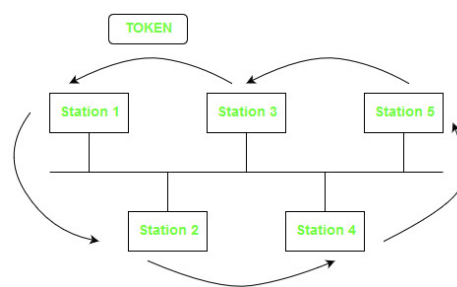


Figura 2.3: Topología token bus

2.1.1.1.1. Token Ring

Dispositivos conectados en anillo físico. Uso de un token que circula entre los dispositivos para dar acceso al bus.

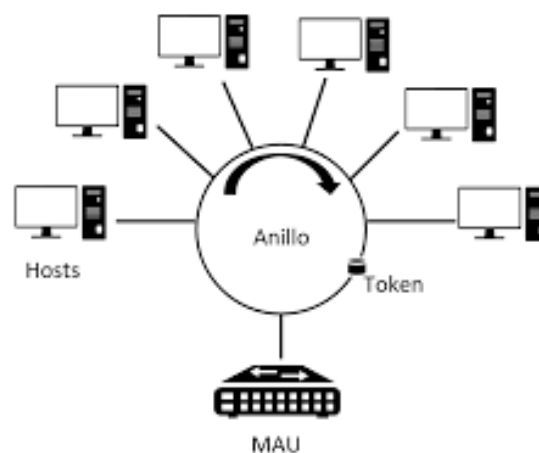


Figura 2.4: Topología token ring

2.1.1.2. Redes de conexión directas o estáticas

Una red directa está formada por un conjunto de nodos que están conectados mediante enlaces point-to-point a un pequeño subconjunto de nodos. Cada uno de los nodos que componen la red realiza tanto el enrutamiento como la computación.

Entre las propiedades de las redes directas tenemos las especificadas en la sección 2.1.1 y, además, la propiedad ortogonal, es decir que todos los nodos y enlaces que componen la red pueden disponerse en n dimensiones, de manera que cada enlace está situado exactamente en una dimensión. En las redes directas, los caminos para la transmisión de mensajes se seleccionan mediante algoritmos de encaminamiento. mientras que los mecanismos de conmutación determinan cómo se conectan las entradas a las salidas en un nodo.

Dentro de las redes de conexión directas podemos distinguir distintos tipo dependiendo de como se distribuyan y se conecten los nodos entre sí:

2.1.1.2.1. Estrella

En esta red existe un nodo central que se conecta con todos los demás nodos de la red. El inconveniente de esta red es que, cuando el número de nodos que se conectan al nodo central es alto, esta topología ya no es práctica ya que, al tener que atravesar todos los mensajes el nodo central, se sobrecarga la red.

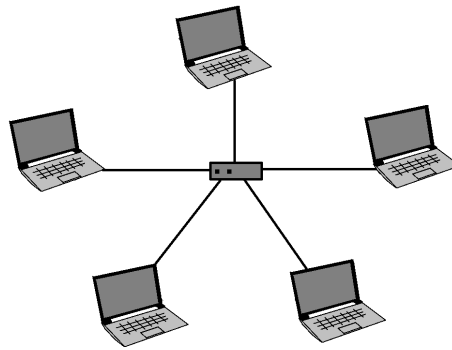


Figura 2.5: Topología estrella

2.1.1.2.2. Anillo

Al igual que la topología estrella, es una de las formas de conexión más simples pues se conectan los nodos que situándose todos en una fila, por lo que cada nodo tiene dos nodos vecinos a los que se conecta formando una circunferencia de nodos conectados. Para enca minar un mensaje de un nodo a otro, una vez determinado por qué enlace debe enviarse, únicamente hay que transmitirlo hasta que llegue al nodo destino.

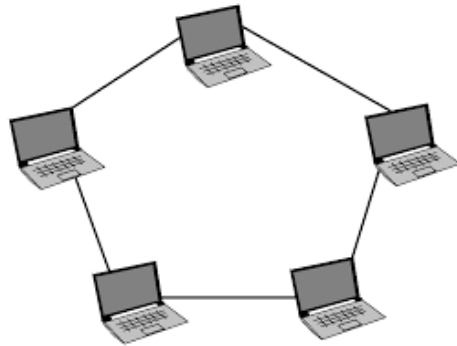


Figura 2.6: Topología anillo

2.1.1.2.3. Malla

Una malla es una topología de red directa que posee n dimensiones en la que cada una de ellas hay k nodos, por ello el número de nodos de la red es $k * n$ y cada uno de estos vértices está conectado a todos aquellos que difieren en una coordenada de él en cualquiera de las dimensiones. Esta topología no es regular, ya que los nodos interiores tienen grado $2n$ y los nodos exteriores tienen grado n .

Una de las características esenciales de la malla es que es infinitamente extensible según k y según n hasta el límite de las interconexiones disponibles; en este sentido, es la más modular de las topologías.

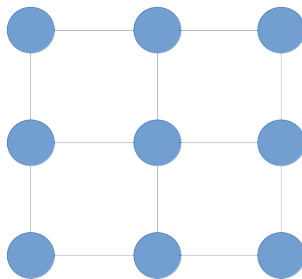


Figura 2.7: Topología Malla

2.1.1.2.4. Toro

Las redes en toro son mallas en que sus filas y columnas tienen conexiones en anillo, lo que contribuye a disminuir su diámetro. Esta modificación convierte a las mallas en estructuras simétricas y además, reduce su diámetro a la mitad.

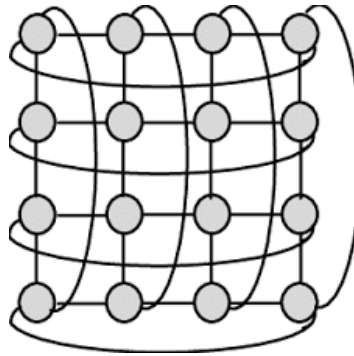


Figura 2.8: Topología toro

2.1.1.2.5. Hipercubo

Las redes con topología hipercubo están formadas por 2^n nodos, que forman los vértices de los cuadrados para crear una conexión entre redes. Esta conexión se realiza conectando los nodos que solo difieren en un bit en su representación binaria (como vemos en la figura 2.9). Un hipercubo es básicamente una red de malla multidimensional con dos nodos en cada dimensión.

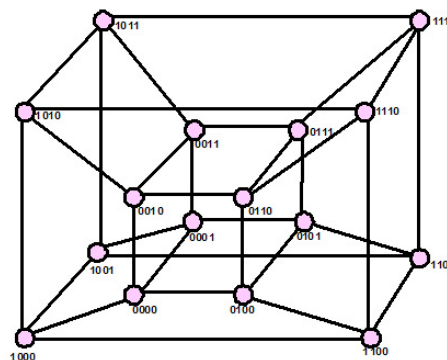


Figura 2.9: Topología hipercubo

2.1.1.3. Redes de conexión indirectas o dinámicas

Estas redes, a diferencia de las redes de conexión directa, no tienen vecinos fijos. La topología de comunicación puede cambiarse dinámicamente en base a las demandas de la aplicación.

2.1.1.3.1. Redes en bus

Estas redes están compuestas por una serie de líneas de bits a las que se conectan varios recursos (Figura 2.10). El inconveniente de un bus es que sólo permite una transferencia al

mismo tiempo, por ello, cuando hay varios recursos que tratan de conectarse al bus, se necesita un árbitro que será de ir dando paso, una a una, a las diferentes peticiones. Los buses son una forma barata de comunicación que tienen la ventaja de ser muy fácilmente reconfigurables, pero con la desventaja de un bajo ancho de banda y una gran latencia debida a las continuas esperas de las peticiones.

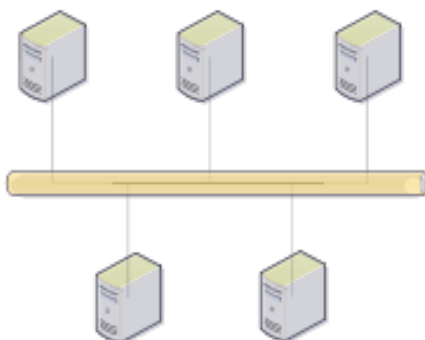


Figura 2.10: Red en bus

2.1.1.3.2. Redes multietapa

Estas redes están formadas por múltiples etapas de switches. Se compone de interruptores 'axb', siendo 'a' el número de entradas del switch y 'b' el de salidas, que se conectan mediante un patrón de conexión interetapas (ISC) determinado. La ventaja de estas redes sobre la redes crossbar es el menor número de conmutadores, que varía dependiendo del tipo de red, sin embargo, algunas de ellas son bloqueantes. Entre las redes multietapa tenemos, entre otros, los siguientes tipos:

2.1.1.3.2. Redes Omega

Estas redes están formadas por N etapas siendo N el número de entradas de la red. En este tipo de red es posible llegar a varios destinos a partir de un solo origen, permitiendo realizar difusiones en la red. Por el contrario, estas redes son sensibles a fallos ya que, si un hilo se corta, no serán posibles muchas de las rutas.

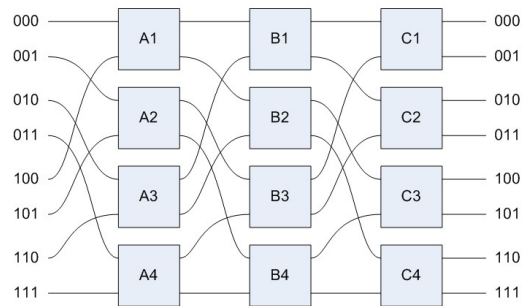


Figura 2.11: Topología multietapa

2.1.1.3.2. Redes de línea base

Las redes de línea base están formadas por conmutadores de líneas cruzadas de 2x2 que se pueden generar recursivamente uniéndose las entradas de cada pareja de conmutadores del último nivel, mediante un perfect shuffle¹, a las salidas de los conmutadores de la etapa anterior. Se repite este proceso hasta llegar a un único perfect shuffle que una las salidas de todos los conmutadores del primer nivel.

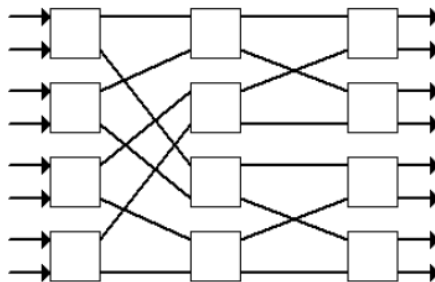


Figura 2.12: Topología de línea base de 8 entradas

2.1.1.3.2. Redes CLOS

Estas redes constan de tres etapas construidas con conmutadores de líneas cruzadas pero de diferentes tamaños en cada etapa. Estas redes están formadas por tres etapas: la primera de ellas tiene r conmutadores de n entradas y m salidas, la segunda etapa tiene m conmutadores de r entradas y salidas y la tercera etapa tiene r conmutadores de m entradas y n salidas. Una de las ventajas de las redes CLOS es que, dimensionándolas convenientemente, se puede conseguir que sea reacondicionables, o incluso, no bloqueantes.

¹Perfect Shuffle: consiste en dividir el número de entradas totales para los conmutadores con los que estamos trabajando en dos mitades iguales y coger, alternativamente cada vez, una entrada de cada mitad y enlazarla con la salida libre de número más bajo.

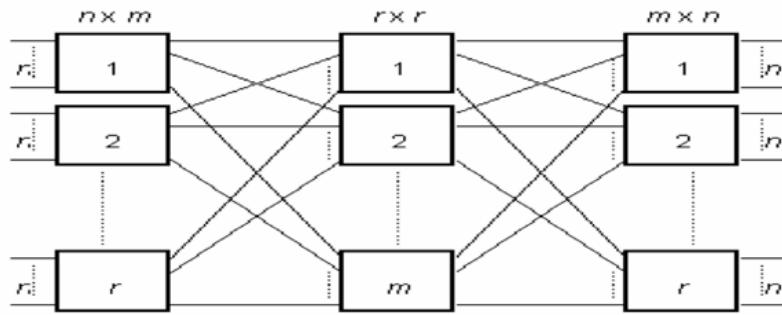


Figura 2.13: Topología CLOS

2.1.1.3.2. Red de Benes

Es una red CLOS con $n = m = 2$, donde los conmutadores centrales son, a su vez, redes CLOS con $n = m = 2$ y r la mitad de la red completa. Además, las permutaciones entre la primera y la segunda etapa son perfect shuffle inverso², y, entre la segunda y la tercera, perfect shuffle.

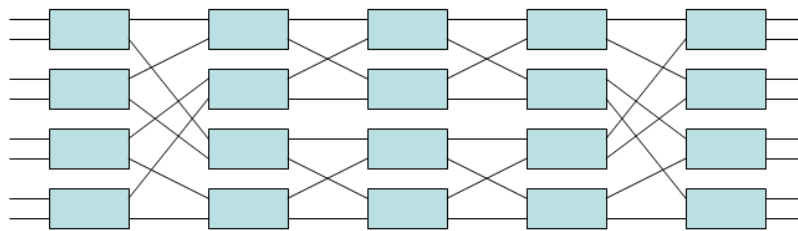


Figura 2.14: Topología de Benes

2.1.1.3.3. Crossbar switches

Contienen una matriz de elementos de conmutación simples que pueden encenderse y apagarse para crear o romper una conexión (Figura 2.15). Estos tipos de redes presentan las ventajas de que no son bloqueantes y que son fácilmente escalables, pero son necesarios un gran número de conmutadores (N^2 para una red cuadrada de lado N).

²Perfect shuffle inverso: similar al perfect shuffle, pero en este caso se tienen en cuenta las salidas donde previamente se tenían en cuenta las entradas, y se relaciona con la entrada de número más bajo

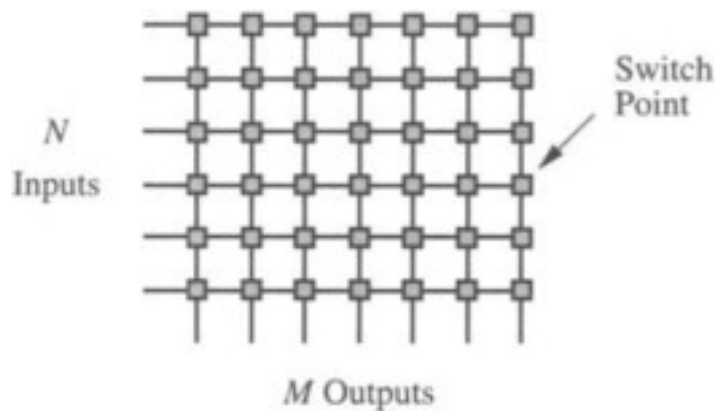


Figura 2.15: Switches transversales

2.1.1.4. Redes de conexión híbridas

Estas redes combinan mecanismos de las redes de medio compartido y de las redes directas e indirectas. En cuanto a las redes de medio compartido, se trata de incrementar el ancho de banda, con respecto a las redes directas se trata de reducir el diámetro de la red y, con respecto a las indirectas, se reduce tanto el número de switches como de enlaces.

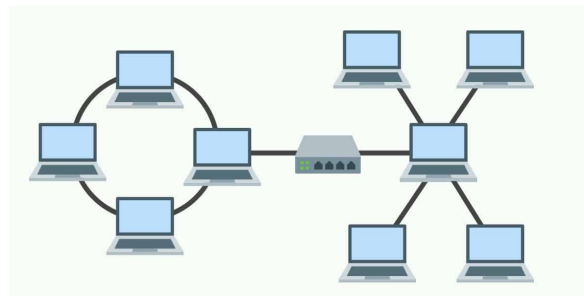


Figura 2.16: Topología híbrida

2.1.2. Estructura organizacional

Las redes de interconexión están compuestas por los siguientes tres componentes:

- **Enlaces:** cable de una o varias fibras ópticas o hilos eléctricos con un conector en cada extremo conectado a un conmutador o puerto de interfaz de red. A través de él se transmite una señal analógica desde un extremo y se recibe en el otro para obtener el flujo de información digital original.
- **Switches:** se componen de un conjunto de puertos de entrada y salida (normalmente el número de ambos puertos es el mismo), una "cross-bar" interna que conecta toda

la entrada con toda la salida, un buffer interno y una lógica de control para efectuar la conexión de entrada-salida en cada momento

- **Interfaces de red:** se comportan de forma muy diferente a los nodos de conmutación y pueden estar conectadas a través de enlaces especiales. La interfaz de red formatea los paquetes y construye la información de enrutamiento y control. Puede tener un buffer de entrada y de salida, en comparación con un conmutador. Puede realizar una comprobación de errores de extremo a extremo y un control de flujo. Por lo tanto, su coste depende de la complejidad del procesamiento, la capacidad de almacenamiento y el número de puertos.

2.1.3. Encaminamiento

El algoritmo de encaminamiento determina cuál de los posibles caminos desde el origen hasta el destino se utiliza como ruta y cómo se determina la ruta que sigue cada paquete en particular. El proceso para seleccionar la mejor ruta para nuestros paquetes lo realizará un protocolo de encaminamiento. El protocolo de encaminamiento que se utilizará en red no está previamente estipulado sino que, dependiendo de diversos factores como la topología, la aplicación, etc, se elegirá el protocolo que más se adecue a nuestro escenario. El principal objetivo del algoritmo de encaminamiento es seleccionar el camino que suponga un menor retardo para cada paquete, intentando elegir rutas, evitando la sobrecarga.

Los algoritmos de encaminamiento deben cumplir las propiedades de **simplicidad y rapidez** (tomar decisiones correctas en un tiempo limitado), **robustez** (capacidad de reconocer cambios de topología debido a fallos), **conectividad** (debe existir al menos un camino válido para cada par de nodos de la red), **optimalidad** (manejar la relación entre minimizar el retardo medio de los paquetes y maximizar la utilización total de la red) y **maximización de los enlaces** (garantizando la igualdad en el acceso a estos).

Podemos clasificar los algoritmos de encaminamiento en base a diferentes criterios:

- **Número de destinos:** determina el número de nodos destino a los que se envía un solo paquete. En caso de un solo receptor tenemos una comunicación *unicast* y en caso de varios nodos se denominan *multicast* o *broadcast*, dependiendo si se envía a un subconjunto de nodos o a todos los nodos de la red.
- **Ubicación del algoritmo:** determina quién y dónde se toman las decisiones de encaminamiento. Puede ser *centralizada*, en el caso de que un nodo centralice las decisiones o *no centralizada*, donde las decisiones se deciden localmente entre todos los nodos de la red. En este último caso existen dos posibilidades, debido a que la determinación del camino a seguir puede hacerse en cada nodo fuente que envía mensajes o, de manera distribuida a través de cada nodo que atraviesa el mensaje.
- **Función de encaminamiento:** la implementación de esta mediante una *tabla* que almacena los caminos posibles o mediante una *función* (lógica o aritmética) que determine la trayectoria a recorrer en cada nodo

- **Decisiones:** hace referencia a si se tiene en cuenta el tráfico en la red y/o la ocupación de los enlaces para determinar los caminos. Tenemos dos tipos de algoritmos para la toma de decisiones:
 - **Determinista:** las decisiones de encaminamiento no se basa en mediciones del tráfico existente en la red sino que se establece una trayectoria fija para el envío del paquete.
 - **Adaptativo:** son capaces cambiar las rutas que seguirán los paquetes dependiendo de las condiciones del tráfico existente en la red para evitar lo máximo posible situaciones de congestión.

2.1.3.1. Problemas relacionados con el encaminamiento

Existen fenómenos indeseables que pueden aparecer en la red y que deben ser evitados ya que provocan una degradación significativa en las prestaciones. Estos problemas aparecen por la utilización de una técnica de encaminamiento ineficiente e inadecuada.

2.1.3.1.1. Deadlock

Esta situación impide que uno o más mensajes puedan avanzar en la red (se bloquean unos a otros) para alcanzar su destino durante un tiempo indeterminado. Existen tres técnicas para tratar los *deadlock*:



Figura 2.17: Deadlock

- **Técnicas de prevención:** solo asignan un recurso a un solicitante si se asegura que la petición no podrá producir un *deadlock*, por ejemplo, reservando el camino en su totalidad con antelación.

-
- **Técnicas de evitación:** asignan recursos a medida que los paquetes avanzan por la red. Esto se hace ordenando los recursos y asignándolos por estricto orden o limitando los caminos que los paquetes pueden tomar, asegurando que esas asignaciones nunca puedan producir *deadlock*.
 - **Técnicas de recuperación:** se centran en detectar *deadlock* y eliminarlos, removiendo un mensaje del bloqueo y reinyectándolo posteriormente.

2.1.3.1.2. Livelock

Provoca la presencia indefinida de un paquete en la red que no es capaz de alcanzar su destino en ningún momento. Esto ocurre cuando el mensaje circula alrededor del nodo destino pero no llega nunca a alcanzarlo. Este problema puede solucionarse empleando caminos mínimos de manera que los paquetes se acerquen siempre a su destino.

2.1.3.1.3. Starvation

Se produce cuando, para un mensaje, los recursos que solicita son siempre asignados a otros mensajes que también los solicitan, quedando ese mensaje a la espera de dichos recursos. Este problema puede solucionarse aplicando una política Round-Robin, que es una política que asigna el recurso por igual a todos los que lo solicitan.

2.1.4. Diseño del switch

El diseño de una red depende del diseño del switch y del modo en que estos están conectados entre sí mediante los enlaces, que son los encargados de retransmitir los paquetes. Dependiendo del grado del switch, sus mecanismos de enrutamiento interno y su almacenamiento interno, pueden admitirse o no un determinado tipo de topología y algoritmos de enrutamiento.

La función del switch es dirigir los paquetes basados en la dirección de destino, localizada en la cabecera de los mismos, transmitiéndolo por el puerto de salida que corresponda. El switch es el componente fundamental para el encaminamiento dentro de una subred, ya que, fuera de esta, es responsabilidad de los routers. En la figura 2.18 podemos ver la arquitectura interna de un switch.

Entre los elementos del switch tenemos los puertos que son las interfaces que conectan los extremos de los enlaces a los conmutadores. Dentro de cada uno de ellos existe una unidad de transmisión y otra de recepción para enviar y recibir los paquetes que circulan por la red.

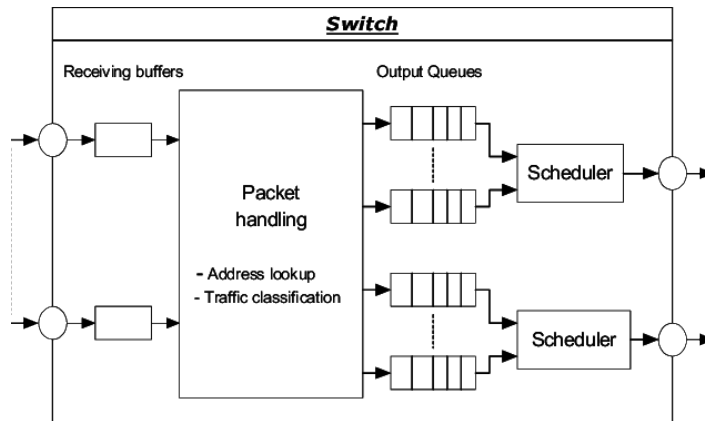


Figura 2.18: Arquitectura interna de un switch

2.1.5. Control de flujo

Los switches contienen buffers que permiten almacenar datos evitando, así, la pérdida de paquetes que ocurre cuando el puerto no es capaz de enviarlos/recibirlos. Debido a la presencia de estos, es necesario un control de flujo de los datos capaz de garantizar la capacidad de almacenamiento. Mediante el uso de un control de flujo, se notifica al emisor la disponibilidad de espacio dentro del receptor de los datos. Las políticas de control de flujo más utilizadas son el control de flujo basado en créditos y el control de flujo basado en marcas o Stop & Go.

2.1.5.1. Basado en créditos

Cada conmutador tiene varios créditos para el envío de paquetes, que dependerá de la cantidad de espacio en el buffer del nodo vecino. En cada transmisión, el emisor consume un crédito hasta que haya consumido todos, en cuyo caso el emisor dejará de enviar paquetes. Una vez se haya liberado espacio en el receptor, es decir, se haya procesado algún paquete, se envían nuevos créditos al emisor, es decir, se incrementa el número de créditos de los que el emisor dispone y continúa el envío.

2.1.5.2. Basado en marcas (Stop & Go)

El receptor detecta que su buffer de entrada está en el límite de su almacenamiento, el cuál ha sido establecido previamente, y envía una señal (*Stop*) al nodo emisor para que detenga el envío de paquetes. Una vez que la capacidad del buffer de entrada del receptor es suficiente para recibir nuevos paquetes, se envía una nueva señal (*Go*) al emisor, indicando que puede continuar con el envío de los paquetes.

2.2. Protocolo de transporte Remote Direct Memory Access (RDMA)

Con el rápido crecimiento del procesamiento de datos, la computación y la comunicación en la arquitectura de la computación de alto rendimiento (HPC), la red TCP/IP tradicional no puede satisfacer la creciente demanda de alto rendimiento y baja latencia y se convierte en el cuello de botella de todo el sistema debido a que su transporte tiene que copiar los datos en buffers internos, lo que añade latencia y consume un uso significativo de la CPU. Esto se produce a que la tecnología TCP/IP tradicional debe pasar a través del sistema operativo y otras capas de software en el proceso de procesamiento de paquetes de datos, lo que requiere una gran cantidad de recursos del servidor y ancho de banda del bus de memoria. Por lo tanto, es esencial una red de alto rendimiento que pueda cumplir los requisitos de las aplicaciones HPC.

Direct Memory Access (DMA) es la capacidad de un dispositivo para acceder directamente a la memoria del host sin la intervención de la CPU, y RDMA (Remote DMA) es un concepto que utiliza DMA y que representa la capacidad de acceder a la memoria de una máquina remota sin interrumpir el procesamiento de la CPU de ese sistema. Es decir, RDMA permite eliminar las operaciones de copia de datos y reducir las latencias al permitir que un ordenador coloque directamente la información en la memoria de otro con una demanda mínima de ancho de banda del bus de memoria y de sobrecarga de procesamiento de la CPU, al tiempo que se conserva la protección de la memoria. Esto permite la realización de sistemas de alto rendimiento así como comunicaciones de baja latencia lo cual es muy importante en sistemas MPP.

El protocolo RDMA promete una computación y un transporte de datos más eficientes y escalables dentro del centro de datos al reducir la carga de los procesadores y la memoria con respecto al protocolo TCP/IP.

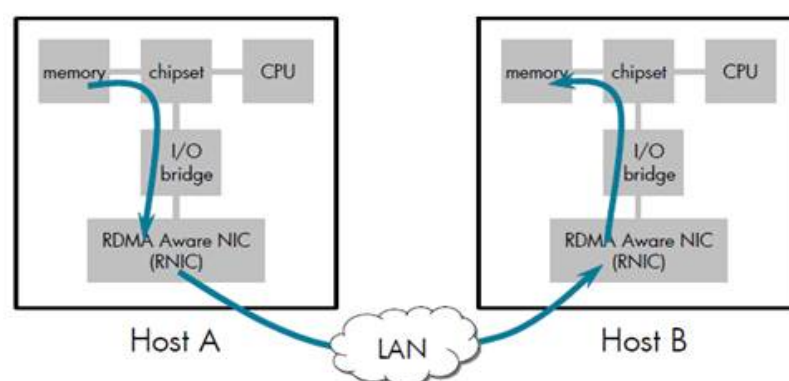


Figura 2.19: RDMA

2.2.1. Ventajas de RDMA

RDMA presenta muchas ventajas con respecto a los protocolos de transporte correspondientes:

- **Zero-copy:** las aplicaciones pueden realizar la transferencia de datos sin que intervenga la pila de software de la red y los datos se envían directamente a los buffers si que se copien entre las capas de red.
- **Kernel bypass:** las aplicaciones pueden realizar la transferencia de datos directamente desde el espacio de usuario, sin necesidad de realizar conmutaciones de contexto.
- **Sin intervención de la CPU:** las aplicaciones pueden acceder a la memoria remota sin consumir CPU en la máquina remota. La memoria de la máquina remota se leerá sin ninguna intervención de elementos remotos (proceso/procesador). Las cachés de la CPU remota no se llenarán con el contenido de la memoria accedida.
- **Transacciones basadas en mensajes:** los datos se manejan como mensajes discretos y no como un flujo, lo que elimina la necesidad de que la aplicación separe el flujo en diferentes mensajes.
- **Soporte de entrada de dispersión/recopilación:** RDMA soporta de forma nativa el trabajo con múltiples entradas de dispersión/recopilación, es decir, la lectura de múltiples buffers de memoria y su envío como un flujo o la obtención de un flujo y su escritura en múltiples buffers de memoria.

2.2.2. Protocolos de red compatibles con RDMA

- **Infiniband:** RDMA es el protocolo estándar para la conexiones de red Infiniband de alta velocidad. Este protocolo se utiliza a menudo para la comunicación entre sistemas. Se necesitan tarjetas de red y conmutadores que admitan esta tecnología.
- **RoCE:** RoCE es protocolo de red que permite el uso de RDMA sobre una red Ethernet, definiendo su funcionamiento en dicho entorno.
- **iWARP:** aprovecha el protocolo TCP o el SCTP para la transmisión de los datos. Fue desarrollado para permitir que las aplicaciones de un servidor lean o escriban directamente en las aplicaciones que se ejecutan en otro servidor sin que el sistema operativo de ninguno de los dos lo soporte.

2.3. Entorno de simulación Omnet++

Omnet++ es una biblioteca y un marco de simulación C++ extensible, modular y basado en componentes, principalmente para construir simuladores de red. El concepto *red* incluye tanto redes de comunicación cableadas e inalámbricas, redes en chip, redes de colas, etc. Este entorno de simulación ofrece un IDE basado en Eclipse, un entorno gráfico de ejecución con las opciones necesarias para la manipulación de esta.

Ofrece una arquitectura de componentes para los modelos. Los componentes, que llamaremos módulos, se programan en C++ y, posteriormente, se conectan entre ellos dando lugar a componentes más grandes utilizando un lenguaje de alto nivel como es NED. Una vez tenemos formada nuestra red en el lenguaje NED, se debe configurar un archivo .ini en el que se definirá el comportamiento de la red indicando el tipo de tráfico, el tiempo de inicio, el tiempo entre mensajes, el tiempo que durará la simulación, etc.

Dentro de Omnet++, utilizaremos INET que es una biblioteca de modelos. Proporciona protocolos, agentes y otros modelos para trabajar con redes de comunicación. Contiene modelos para la pila de Internet (TCP, UDP, IPv4, etc.), protocolos de capa de enlace cableados e inalámbricos (Ethernet, IEEE 802.11, etc.), soporte para movilidad, etc.

Para nuestro caso, en el fichero .ned, utilizaremos módulos compuestos para desarrollar la red que representarán los host (StandardHost) y los switch (EthernetSwitch) que forman parte de la biblioteca INET y, también, utilizaremos tráfico UDP para estudiar las prestaciones de TCP/IP. Además, modificaremos el módulo StandardHost de INET para modelar el protocolo RDMA y estudiar sus prestaciones.

3. Metodología y desarrollo

4. Experimentos y resultados

5. Conclusiones

5.1. Configuración


La configuración es muy sencilla. Al margen de la adaptación particular que pueda hacerse, la elaboración de un documento solamente requiere información relativa al autor, título y tutores. Ésta ha de configurarse en el archivo `include\opciones.tex`.

Además, en el archivo `include\colores.tex` puede configurarse el color tema, utilizado en títulos, elementos flotantes y, opcionalmente, tablas y negritas. Por defecto, se recomienda ponerlo a `{0,0,0}` (negro). Pueden escribirse **negritas con el color configurado** con el comando `\bft{texto}`.

Por último, en el archivo principal se pueden configurar las fuentes utilizadas en la portada, preámbulo y documento.

5.2. Colores

En el archivo `include\colores.tex` se predefinen algunos colores. También varios comandos para utilizar negritas con ellos. En concreto, se proporcionan cinco: Con el comando `\bfa{texto}` se pueden poner las cosas en negrita **azul**. Con el comando `\bfr{texto}` se pueden poner las cosas en negrita **rojo**. Con el comando `\bfm{texto}` se pueden poner las cosas en negrita **verde**. Con el comando `\bfv{texto}` se pueden poner las cosas en negrita **mostaza**. Con el comando `\bfm{texto}` se pueden poner las cosas en negrita **mostaza**.

 No es conveniente abusar los colores, salvo en alguna ocasión adicional. Por ejemplo, si el color del tema es **negro** o **azul** se puede utilizar el **rojo** para alguna advertencia.

5.3. Listas y enumeraciones

El entorno *itemize* usa, por defecto, el color del tema. Los símbolos utilizados para marcar los items están pre-definidos en el archivo de configuración, y pueden cambiarse. También es posible seleccionar, mediante `\item[símbolo]` el símbolo específico para cada línea. Esta lista muestra los símbolos por defecto:

- Primer ítem
 - Primer ítem en el segundo nivel
 - Otro ítem en el segundo nivel
 - Ítem en el tercer nivel
- Segundo ítem
- Tercer ítem

En esta se usan algunos caracteres especiales predefinidos para marcar los ítems. Existen comandos para mostrar algunos de estos caracteres con los colores azul, rojo y el color del tema.

- ✓ `\vmarka` : “v” representa el símbolo, y “a” el color
- ✗ `\xmarkr`
- 📎 `\lmarkt` : el color “t” corresponde al tema.
- 👉 `\hmarkt`
- ➔ `\fmarkt`

Las enumeraciones también están preconfiguradas en el archivo de configuración:

1. Esto es una enumeración
2. Con algunos elementos adicionales
3. Que se pueden quitar

Pueden ser modificadas, como en el siguiente ejemplo.

[Re 8] Esto es una enumeración

[Re 9] Sin elementos

[Re 10] Adicionales

5.4. Elementos flotantes

En el archivo de `include\configuracion` existen varios parámetros que se fijan en la configuración para figuras y tablas, pero se pueden alterar de manera ocasional (y excepcional) al maquetar el documento. La mayoría son relativos al título, y se pueden cambiar a través de las funcionalidades que proporciona el paquete `caption`. Cuando se cambian para un elemento particular **es buena práctica restaurar el valor original**, para que el documento tenga un aspecto uniforme.

```
% Títulos
\DeclareCaptionFont{tema}{\color{tema}} % Color
\captionsetup{labelfont={tema, bf}}      % Estilo
\captionsetup{font=normalsize}          % Tamaño
\captionsetup{width=.9\linewidth}       % Anchura del título
% Distancia de la figura al texto.
\setlength{\intextsep}{1cm}
\setlength{\textfloatsep}{1cm}
```

5.4.1. Tablas

Las tablas se hacen de modo normal, aunque se ha añadido la capacidad utilizar celdas que ocupen múltiples filas, colores en los encabezados, y anchuras fijas. Se recomienda mirar el código fuente de la tabla 5.1.

Con respecto al color de los encabezados, es conveniente utilizar el mismo en todo el documento. Una opción con respecto al fondo del encabezado es utilizar el mismo que el del texto pero con transparencia.

5.4.2. Figuras

Con respecto a las figuras, a veces es utilizar un texto descriptivo demasiado ancho puede ser antiestético. Por ejemplo en la 5.1 el rótulo ocupa toda la línea¹. En la figura 5.2 se corrige este efecto con:

```
\begin{figure}[b]
\captionsetup{width=0.8\linewidth}
...
```

Después, se restaura el valor original de la configuración:

```
...
\captionsetup{width=0.9\linewidth}
\end{figure}
```



Figura 5.1: Escuela Superior de Ingeniería Informática. Esto es un texto descriptivo algo más largo que puede producir un efecto feo.



Figura 5.2: Escuela Superior de Ingeniería Informática. Esto es un texto descriptivo algo más largo que puede producir un efecto feo.

Módulo I	Materia	Asignatura	ECTS
Formación Básica (60 ECTS)	Fundamentos Matemáticos de la Informática	Álgebra y Matemática Discreta ES	6
		Cálculo y Métodos Numéricos ES/EN	6
		Estadística ES/EN	6
		Lógica ES	6
	Fundamentos Físicos de la Informática	Fundamentos Físicos de la Informática ES/EN	6
	Ingeniería de Computadores	Estructura de Computadores ES	6
		Tecnología de Computadores ES	6
	Programación	Fundamentos de Programación I ES	6
		Fundamentos de Programación II ES/EN	6
	Gestión de las Organizaciones	Fundamentos de Gestión Empresarial ES	6

Tabla 5.1: Asignaturas del Módulo I.

Se incluye el paquete `subfigure` para la inclusión de figuras con varias imágenes o gráficas, como la que muestra la figura 5.3.



(a) Fotografía de la izquierda



(b) Fotografía de la derecha

Figura 5.3: Ejemplo de inclusión de subfiguras

5.5. Algoritmos y código

Con respecto a los algoritmos (pseudocódigo) y código pueden implementarse, respectivamente, mediante el uso de los paquetes `algorithmic` y `listings`. Pueden configurarse

¹Por defecto, ocupa el 90%, que es el valor por defecto en la configuración, pero se ha cambiado para que se vea mejor el efecto.

```

def search(initial):
    open = [initial]
    while open:
        node = open.pop()
        if testGoal(node):
            return recoverPath(node)
        successors = expand(node)
        for sucesor in successors:
            open.append(sucesor)
    return failure

```

Código 5.1: Ejemplo de código en Python

ambos entornos el archivo `include\configuracion`. Por otra parte, se han definido dos entornos flotantes, denominados `algorithm` y `code` para encapsular y mostrar ambos tipos de elemento de manera uniforme.

El algoritmo 5.1 muestra un ejemplo de pseudocódigo. Se han añadido dos comandos, `\va{variable}` para distinguir las *variables*, y `\fu{funcion}` para distinguir las FUNCIONES.

Por último, el listado 5.1 muestra el código del algoritmo 5.1 en *Python*. El paquete `listings` contiene multitud de opciones para el manejo de distintos lenguajes y para personalización.

1:	función TREE-SEARCH()	
2:	<code>open</code> \leftarrow CREATENODE(INITIAL-STATE)	▷ Creates the root
3:	mientras <code>open</code> $\neq \emptyset$ hacer	
4:	<code>node</code> \leftarrow EXTRACT(<code>open</code>)	
5:	si TESTGOAL(<code>node.STATE</code>) entonces	
6:	devolver RECOVERPATH(<code>node</code>)	▷ Solution found
7:	<code>successors</code> \leftarrow EXPAND(<code>node</code>)	
8:	para cada <code>successor</code> en <code>successors</code> hacer	▷ Processes each successor
9:	INSERT(<code>successor</code> , <code>open</code>)	
10:	devolver failure	▷ No solution have been found

Algoritmo 5.1: Tree-Search exploration

5.6. Bibliografía

Esto es una cita [?] y esto otra [?]. La bibliografía se gestiona con bibtex y un estilo pre-definido. No se han hecho modificaciones de momento.

5.7. Notación matemática

Utiliza las fuentes por defecto del tema, aunque también se pueden cambiar.

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right], \lambda > 0$$

6. Marcas y ayudas

Se han introducido algunos elementos que permiten dejar marcas para indicar que algunas cosas provisionales se han de corregir. Por ejemplo `\pcite` deja esta señal **[?]** para indicar que se ha de añadir una cita bibliográfica en un punto concreto.

[—

Las marcas `\ps` y `\fs` permiten marcar y delimitar un bloque con contenido provisional, y que debe ser revisado posteriormente.

—]

Existen métodos más avanzados para hacer anotaciones, como los que ofrece `todonotes`. Permiten insertar notas al margen .

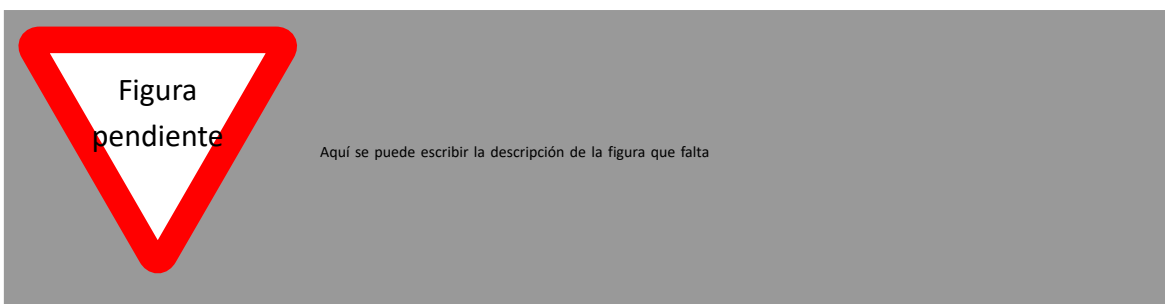
Como por ejemplo ésta.

También permite utilizar notas en el propio texto.

En internet hay combinaciones de colores interesantes para destacar distintos tipos de notas

6.1. Figuras

Este paquete tiene otros comandos como `\missingfigure` para indicar que faltan figuras.



A. Anexo 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Referencia bibliográfica

[sha,] Sistemas de procesadores paralelos. [Online; accessed 22. Jun. 2021]. 7

[Omn, 2019] (2019). What is omnet++? vii

[Int, 2021] (2021). Interconnection Network Design - Tutorialspoint. [Online; accessed 17. Jun. 2021]. 5