Grupo de Computación Distribuida del IFCA

MPI y sus aplicaciones en infraestructuras Grid

Dr. Isabel Campos Plasencia

Instituto de Física de Cantabria-IFCA

Santander, Spain

Esquema

- Introducción a MPI
- MPI en el Grid
- Aplicaciones
- Ejercicios

Introducción
Contexto

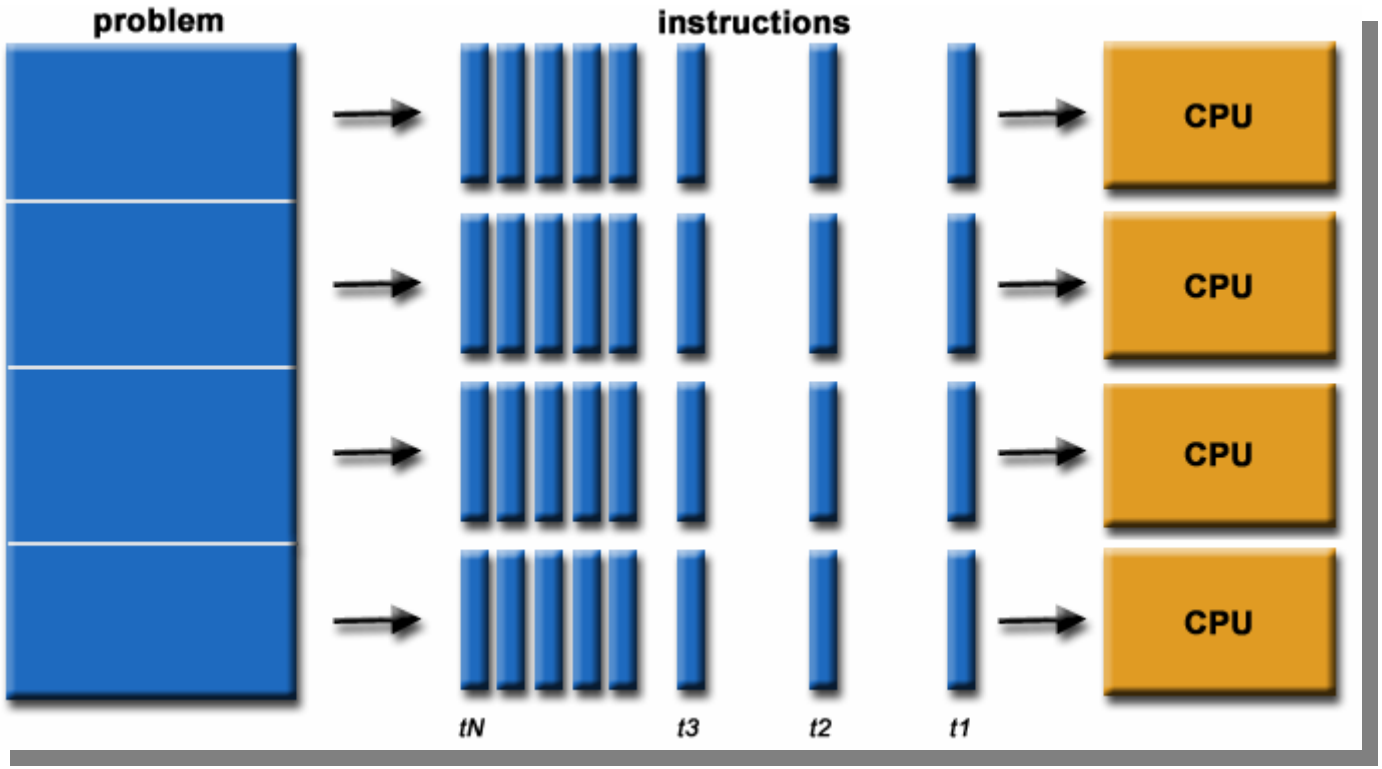
- High throughput computing (HTC)
  - Se utilizan muchos recursos computacionales durante largos periodos de tiempo
    - Acceso a mucho tiempo de cpu promedio durante largos periodos de tiempo (meses)
    - Optimizar el número de trabajos ejecutados por unidad de tiempo.
    - Computación en modo granja, o procesos independientes
  - **El Grid** se diseñó con la idea de ser una fuente de HTC

- High performance computing (HPC)
  - Disponer simultaneamente de una gran cantidad de recursos computacionales
  - Lo importante es que la aplicación se ejecute en el menor tiempo posible.
  - Para ello es necesario que los procesadores individuales que participan en el cálculo cooperen

¿Cómo hacer que los procesadores cooperen?

Computación paralela

- Ejecución simultánea de una misma tarea
- Dividida y adaptada al entorno de computación
- Obtener los resultados de una forma más rápida



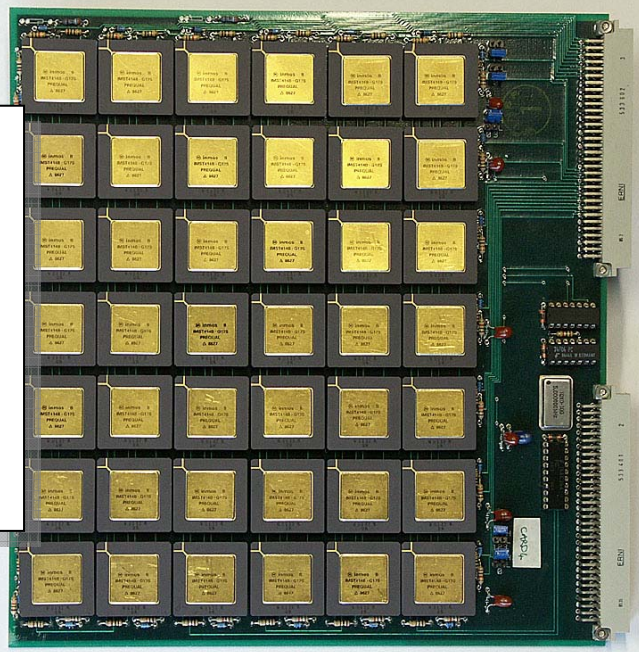
Introducción
Cómo hacer que los procesadores cooperen

- En principio, dependerá de la arquitectura del sistema y del procesador
  - Cada fabricante de hardware tenía su propia forma
    - Librerías específicas
    - Lenguajes específicos: Apesse, Occam, etc...
  - Ventajas
    - Se obtiene el mejor rendimiento posible de cada plataforma
  - Inconvenientes
    - No es portable. Repetir el mismo esfuerzo de programación para cada arquitectura.
    - A veces era necesario un conocimiento profundo del hardware de la máquina

Ejemplo INMOS
Máquinas basadas en transputers: Occam

```
WHILE next <> EOF
 SEQ
     x := next
 PAR
     in ? next
     out ! x * x
```

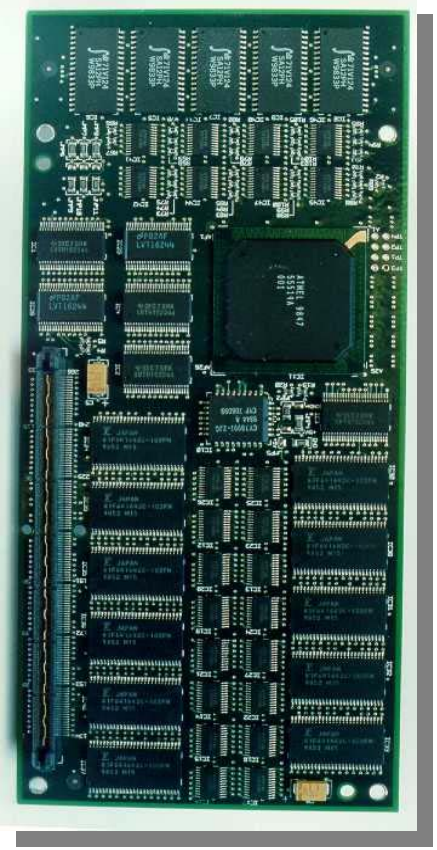
```
CHAN OF [36]BYTE message
CHAN OF COMPLEX32 imp
CHAN OF INT::[]BYTE link

message ! “Hello, World!”
link ! len::[buffer FROM start]
```



Ejemplo ALENIA
TAO para las máquinas APE

APE100

en DESY



Ejemplo ALENIA
TAO para las máquinas APE

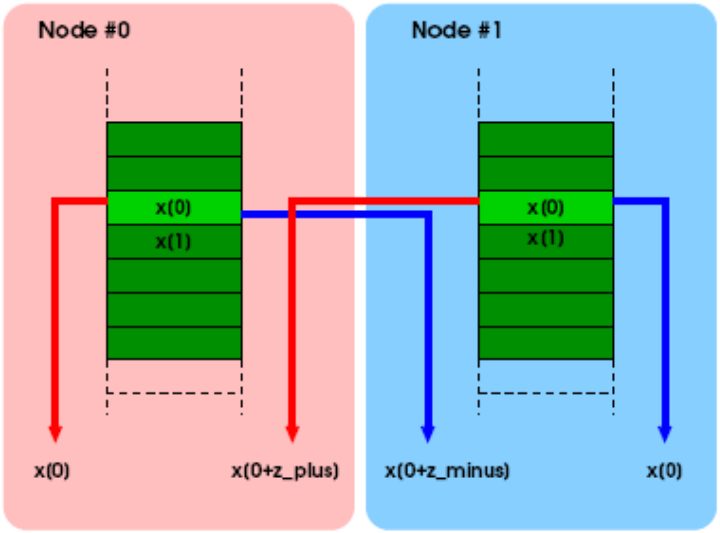
CSIC

- Select different node by node specification
  $[mx, my, mz]$

- Multinode I/O needs slice specification
  $[mx, my, mz]\ [nx, ny, nz]$

```
integer lxm, lym, lzm
real x

lxm = i_machine_lx-1
lym = i_machine_ly-1
lzm = i_machine_lz-1

x = node_abs_id

write [0,0,1] x              !! write from node #1
write [0,0,0][1,1,1] x       !! write from 8 nodes
write [0,0,0][lxm,lym,lzm] x !! write all
```

```
real x[4]                !! defined on each node

x[0] = node_abs_id       !! initialized differently
                         !! on each node

x[1] = x[0+z_plus]       !! read data from z+
x[2] = -x[1]             !! modify data
x[2] = x[2+z_minus]      !! read data from z-
```



Introducción
Message Passing Interface (MPI) Forum

CSIC

En 1992 se creo el MPI Forum para tratar crear una prescripción independiente del sistema y del procesador

- Ver información en http://www.mpi-forum.org
- Participan los más representativos fabricantes de hardware (IBM, Intel, HP, AMD,...) junto con investigadores en ciencia computacional de todo el mundo

- Objetivos del MPI Forum
  - Definir un interfaz de programación (API) uniforme
  - Enfocado a conseguir una comunicación entre procesos lo más eficiente posible
  - Asumiendo que el interfaz de comunicación hardware es fiable
  - Permita la programación en C y en Fortran de forma cómoda para el usuario

Introducción
Evolución del trabajo del MPI Forum

CSIC

- **1994 release de MPI 1.0**
  - Release inicial
- **1995 release of MPI 1.1**
- **1997**
  - Release de MPI 1.2
  - Release de MPI 2.0
- **Futuro**
  - Hacia el MPI 3.0

MPI - 1

- Comunicaciones Point-to-point
- Operaciones Collectiva
- Topologías
- Manejo de las tipologías de datos

MPI - 2

- Input/Output de ficheros
- Comunicaciones one-sided
- Extensión de macros para C++ y F90

Introducción
¿Qué es MPI ?

- **Es la especificación de un interfaz de programación (API)**: Define la forma del Interfaz y su semántica
- MPI es vendor neutral y (de facto) **Standard**
- MPI lo define el MPI-Forum
- MPI especifica la forma de los interfaces para C, C++ y Fortran 70/90
- **MPI no está acabado**. Hay un progreso continuado hacia nuevas versiones.

Introducción
¿Qué NO es MPI ?

CSIC

- MPI **no es Magia**

  "He *instalado XXX-MPI in 4 de mis maquinas y mi aplicación no se ejecuta más rápido !?!"* *(pregunta típica)*

- MPI **no es una implementación de nada, es sólo un estándar**
  - **OpenMPI, MPICH, LAMMPI son implementaciones del estándar MPI**

- MPI **no es el Santo Grial**
  - Hay muchos problemas cuya solución no se adecúa al paradigma de paso de mensajes entre procesos
  - Es muy fácil escribir una aplicación MPI cuyo rendimiento sea peor que su versión secuencial
  - MPI funciona bien en sistemas de memoria compartida pero, tal vez Open MP es mejor solución, dependiendo del problema

Introducción
Conceptos básicos

- Todas las funciones/símbolos MPI functions/symbols contienen el prefijo **"MPI_"**
  - Para usar MPI en los programa en C hay que incluir mpi.h
  - Para usar MPI en Fortran hay que incluir mpif.h
- La compilación y linkado es dependiente de la implementación
  - Las implementaciones más populares de MPI ofrecen al usuario compiladores preparados
    - mpicc, mpicxx, mpiCC, mpif70, ...
    - Cargan automaticamente los flags y las librerías necesarias
- Tipicamente en un Makefile se sustituye el nombre del compilador
  - "mpicc" en lugar de "gcc"

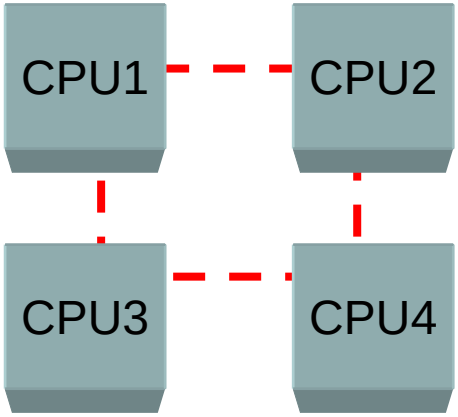
Introducción
Grupos y procesos

- En el paradigma MPI la ejecución de un binario (ej. una aplicación) se divide en N partes, denominadas procesos
  - Los procesos a su vez están agrupados en grupos
  - Cada proceso dentro un grupo está identificado por un número, el rango (rank) del proceso
  - Rank = 0,...,(N-1)

Introducción
Grupos y procesos

- Notar la diferencia entre Proceso y Procesador
  - Un mismo procesador puede ejecutar varios procesos
- Cada proceso puede en principio ejecutar un binario diferente
  - MIMD (multiple instruction multiple data)
- En general es siempre el mismo binario el que ejecuta código diferente basándose en su rango
- El caso más común es que todos los procesos ejecuten el mismo binario

Ejemplo:
Distribución de 8 procesos en 4 procesadores



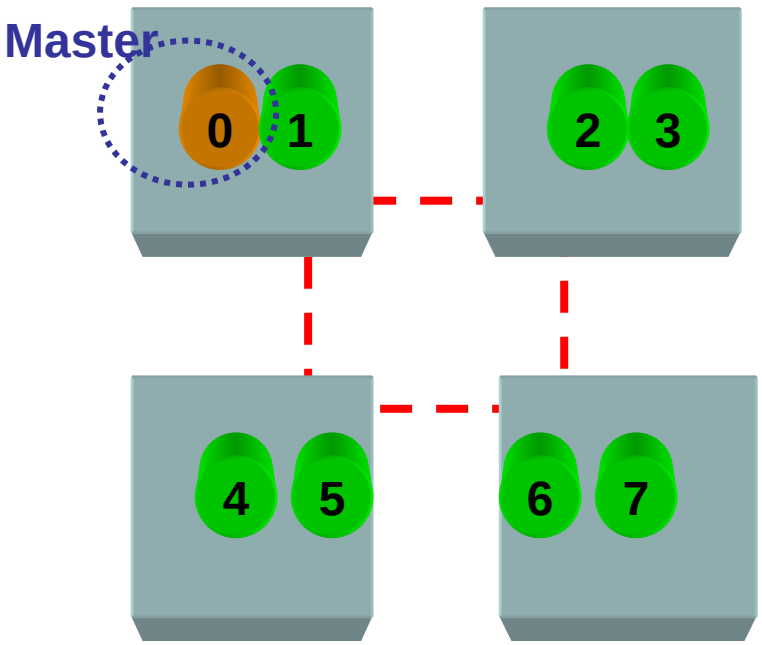

```
mpirun –np 8
        –machinefile hostnames.txt
         ./miprog
```

Genera 8 procesos elementales
Distribuídos entre 4 CPUs

Introducción
Grupos y procesos

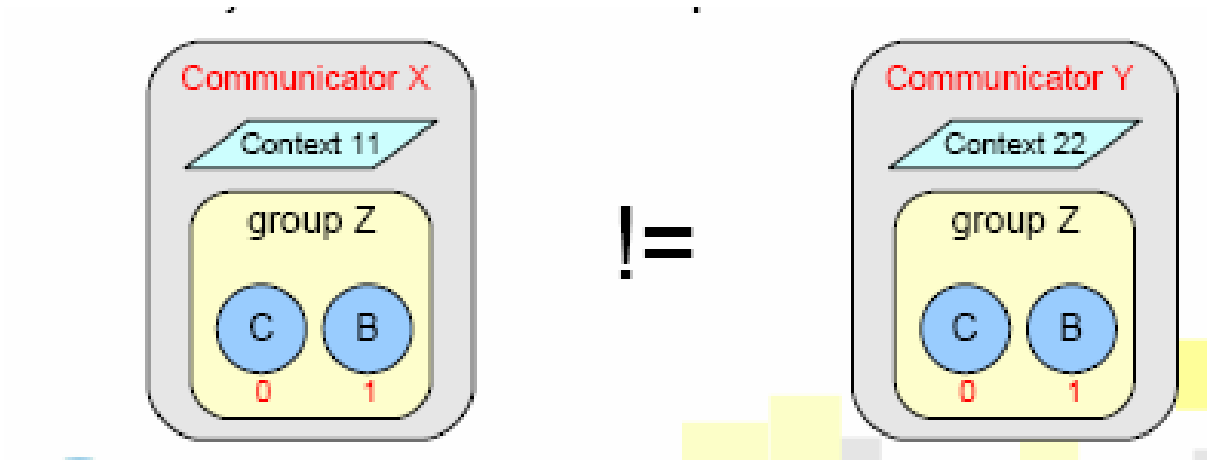
- Notar la diferencia entre Proceso y Procesador
  - Un mismo procesador puede ejecutar varios procesos
- Cada proceso puede en principio ejecutar un binario diferente
  - MIMD (multiple instruction multiple data)
- En general es siempre el mismo binario el que ejecuta código diferente basándose en su rango
- El caso más común es que todos los procesos ejecuten el mismo binario

Ejemplo:
Distribución de 8 procesos en 4 procesadores



Introducción
Comunicadores

- Un comunicador es el objeto que sirve para comunicar (~ socket)
  - Un grupo
  - Un contexto de comunicación
    - Los contextos son únicos
    - Dividen toda la comunicación MPI en espacios disjuntos



Introducción
Comunicadores predefinidos

CSIC

Comunicadores predefinidos

- MPI_COMM_WORLD
  - Envuelve a todos los procesos de la ejecución MPI
- MPI_COMM_SELF
  - Envuelve solo a los procesos que están en un determinado contexto
- **MPI_COMM_NULL**
  - Ningún proceso
- El usuario puede crear sus propios comunicadores de acuerdo a las necesidades del programa

Funciones importantes

- **MPI_comm_size**
  - Devuelve el número de procesos en el grupo
- **MPI_comm_rank**
  - Devuelve el rango del proceso que hace la llamada
- **MPI_comm_free**
  - Elimina un comunicador

Introducción
Hello World

CSIC

```
#include <mpi.h>    /* for MPI functions */
#include <stdio.h>  /* for printf */
int main(int argc, char *argv[])
{
        int rank = 0, size = 0;
        MPI_Init(&argc, &argv);
        MPI_Comm_size(MPI_COMM_WORLD, &size);
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);

        printf("Hello my rank is %i of %i\n", rank, size);

        MPI_Finalize();
        return 0;
}
```

```
Hello my rank is 1 of 4
Hello my rank is 0 of 4
Hello my rank is 2 of 4
Hello my rank is 3 of 4
```

Introducción
Hello World

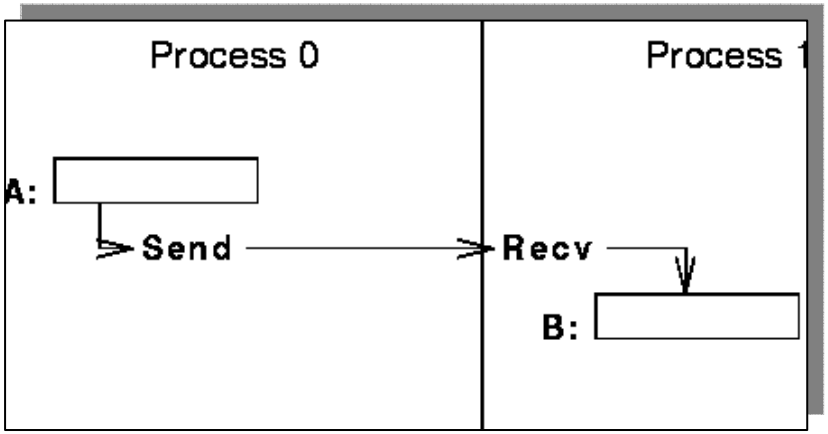
CSIC

Comentarios

- MPI_Init
  - Inicializa el sistema MPI
  - Es necesario llamarlo al principio del programa, antes de invocar ninguna otra función MPI
- MPI_Finalize
  - Deshabilita el sistema MPI
  - Tras esta llamada no se puede llamar a MPI
- Las funciones MPI devuelven siempre un código de error que se puede chequear

Introducción
Tipos de comunicación: punto-a-punto

- **Intercambio de datos entre dos procesos**
  - Se produce mediante el envío de un mensaje
  - El proceso fuente envía el mensaje
  - El proceso receptor tiene que recibirlo (activamente)

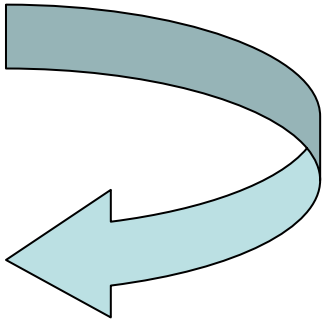



```
int MPI_Send(void *buf, int count, MPI_Datatype datatype,
                                int dest, int tag, MPI_Comm comm);

int MPI_Recv(void *buf, int count, MPI_Datatype datatype,
                        int source, int tag, MPI_Comm comm, MPI_Status *status )
```

Introducción
Tipos de comunicación: punto-a-punto

- ***buf**: puntero a los datos a enviar
- **count**: número de elementos a
- enviar
- **datatype**: tipo de dato
- **dest**: Identificación del proceso
- destino
- **tag**: etiqueta de la comunicación
- **comm**: Identificación del comunicador

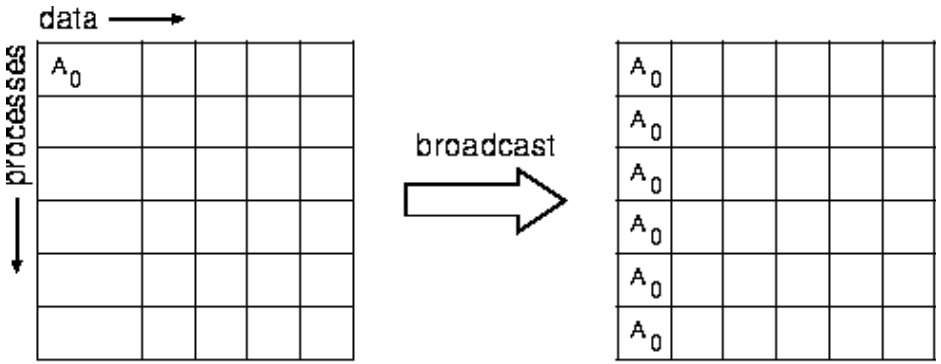
Envío: MPI_Send()



- ***buf**: puntero para la recepción de los datos
- **count**: número de elementos
- **datatype**: tipo de dato
- **source**: Identificación del proceso origen
- **tag**: etiqueta de la comunicación
- **comm**: Identificación del comunicador
- ***status**: puntero para acceso a información sobre mensaje

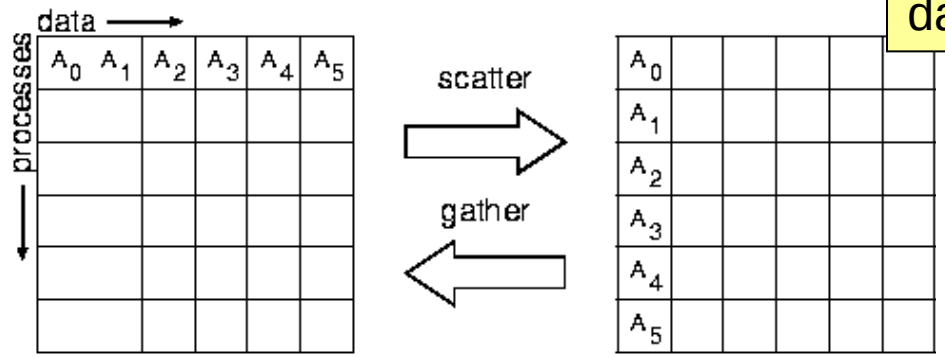
Recepción: MPI_Recv()

Introducción
Tipos de comunicación: Colectivas



MPI_Bcast() Un proceso distribuye un mismo dato al resto de los procesos

MPI_Scather() Un proceso distribuye datos al resto de los procesos particionando



MPI_Gather() Un proceso recoge datos de varios procesos

Introducción
Tipos de comunicación: Colectivas

CSIC

- MPI_Reduce

Realiza una operación matemática distribuida y se devuelve el resultado al root de la operación

El programador puede añadir sus propias operaciones

Ejemplos de funciones empleadas en operaciones colectivas:

MPI_MAX
MPI_MIN
MPI_SUM
MPI_PROD
MPI_LAND logical AND integer logical
MPI_BAND bit-wise AND integer,
MPI_BYTE integer, MPI_BYTE
MPI_LOR logical OR integer logical
MPI_BOR bit-wise OR integer

Introducción
Ejemplo MPI_Bcast

CSIC

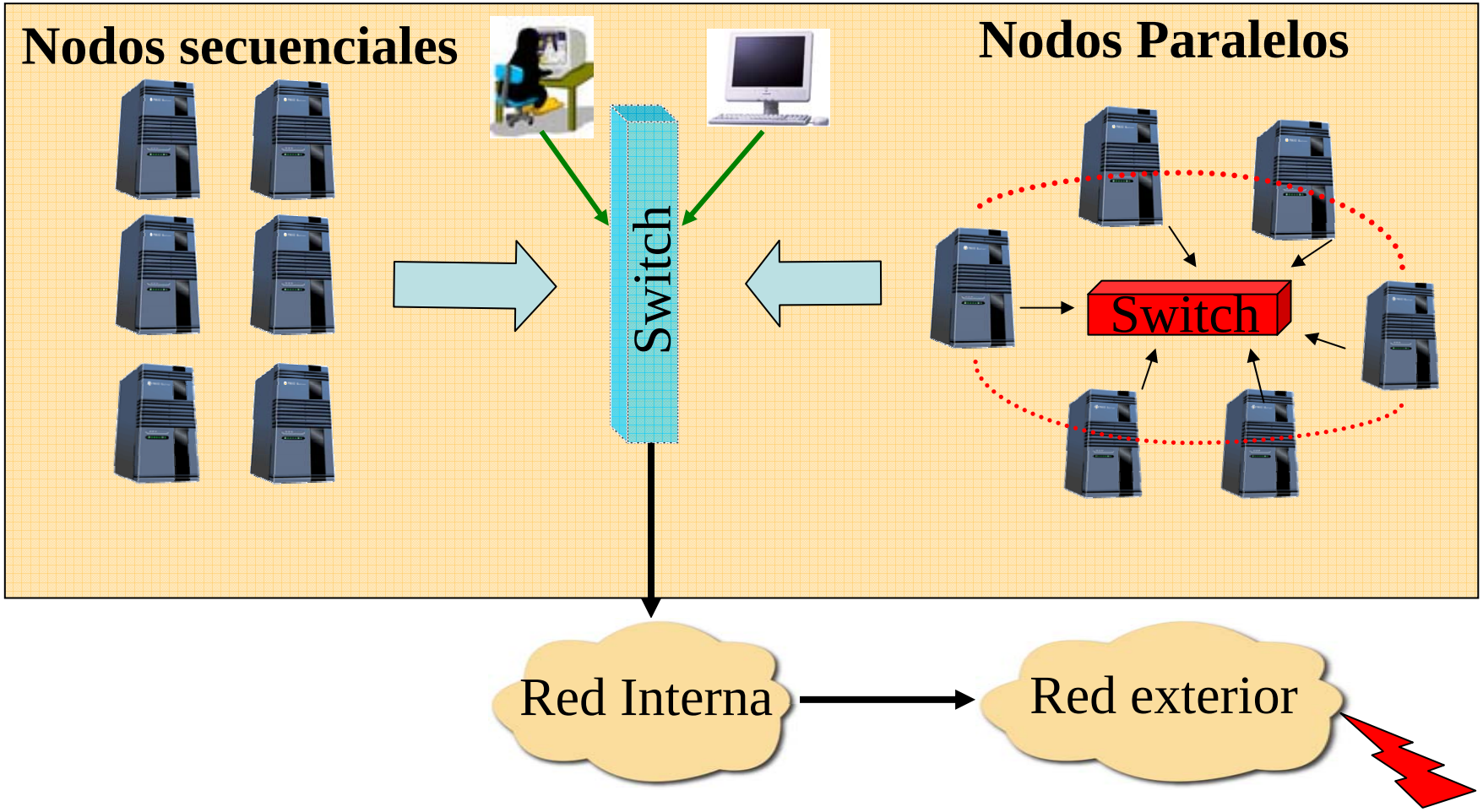
```
Char  msg[100];
if(my_rank==source)
   {

     sprintf(msg,"\n Esto es un mensaje del proceso %d a todos los demás",source);

     MPI_Bcast(msg,100,MPI_CHAR,source,MPI_COMM_WORLD);

     printf("\n Mensaje enviado a todos desde %d",source);
   }
  else
   {
     MPI_Bcast(msg,100,MPI_CHAR,source,MPI_COMM_WORLD);
     printf("\n Mensaje recibido en %d desde %d",my_rank,source);
     printf(msg);
   }
```

Introducción
Ejemplo MPI_Reduce

```
int value;
int result;



value = my_rank;

MPI_Reduce(&value,&result,1,MPI_INT,MPI_SUM,source,MPI_COMM_WORLD);

if(my_rank==source)
  {

    printf("\n Resultado de la suma colectiva %d", result);
  }
```

Sistemas, Hardware y MPI

Configuración típica de un cluster



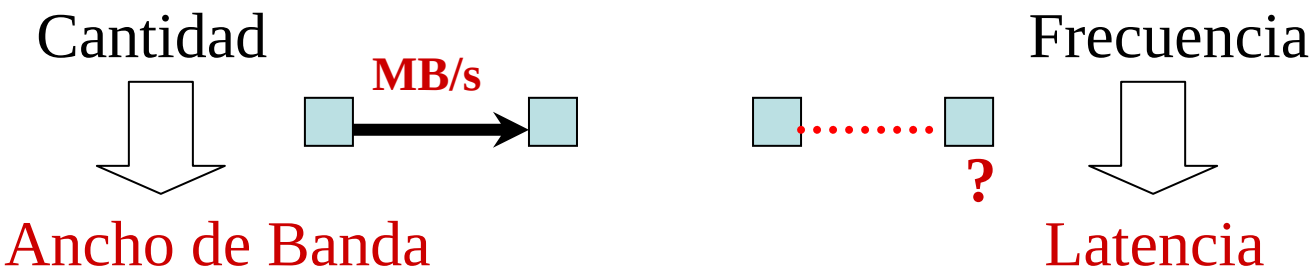
Switch para MPI
Parámetros a tener en cuenta

CSIC

Durante el cálculo necesitan intercambiar datos a través de la red de procesadores que conforma el cluster

¿ Cuánto y con qué Frecuencia necesita el programa comunicar ?

La respuesta fija los parametros de la RED que necesitamos



Alternativas hardware para MPI

CSIC

I. **Gigabit Ethernet (GE)**
   - ✓ Latencias del orden de los 100 microseg.
   - ✓ Ancho de Banda en el rango 50-100 MB/s
   - ✓ Precio: 80 € /puerto + Switch (> 1500 €)

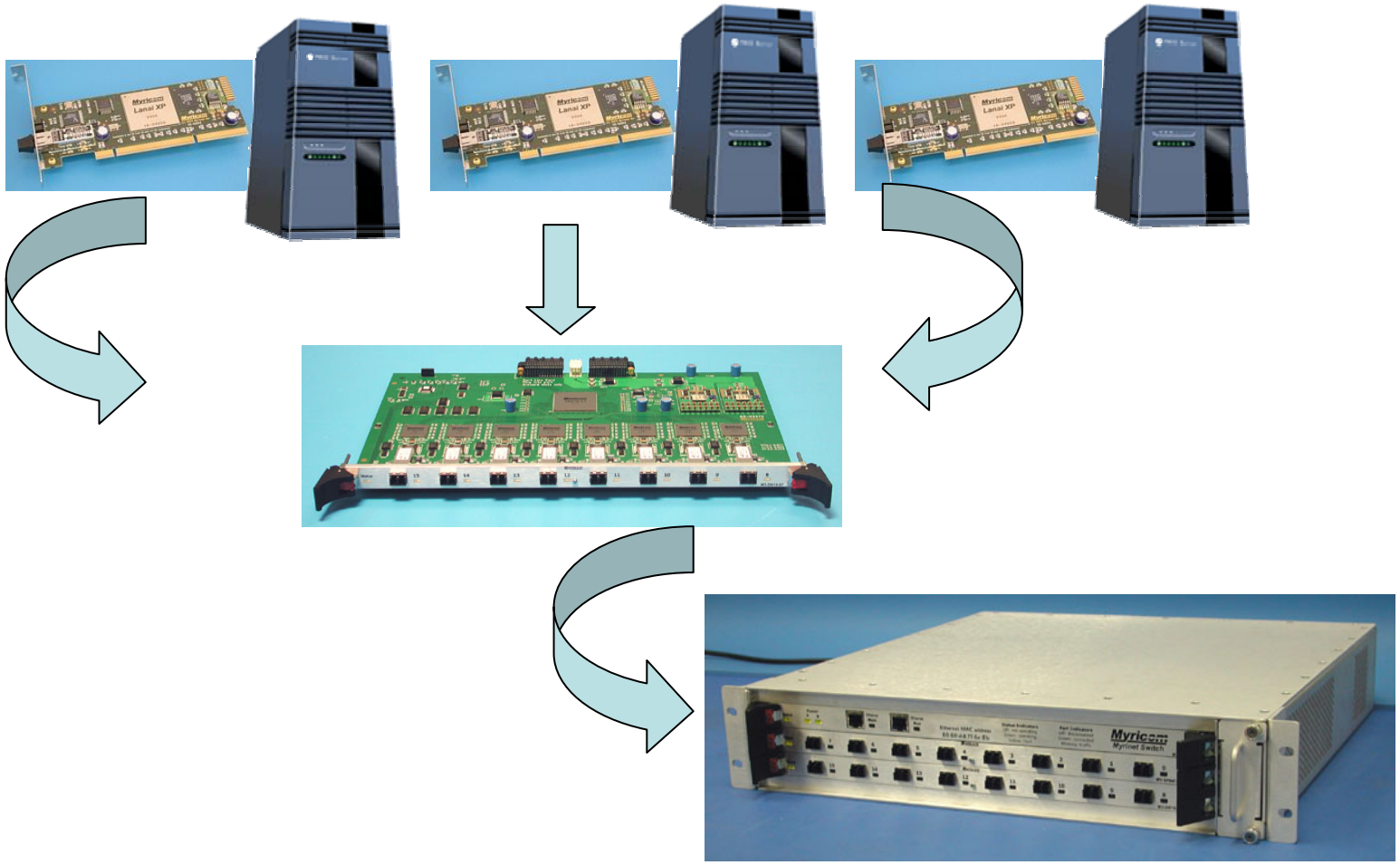
II. **Myrinet 2000**
   - ✓ Latencias entorno a 10 microseg.
   - ✓ Ancho de Banda entorno a 300 MB/s
   - ✓ Precio: 1200 € por tarjeta + Switch (~ 12,000 / 32 puertos)

III. **Infiniband**
   - ✓ Latencias entorno a 5 microseg.
   - ✓ Ancho de Banda entorno a 1GB/s
   - ✓ Precio: ~ 20,000 € / 32 puertos

Red Myrinet

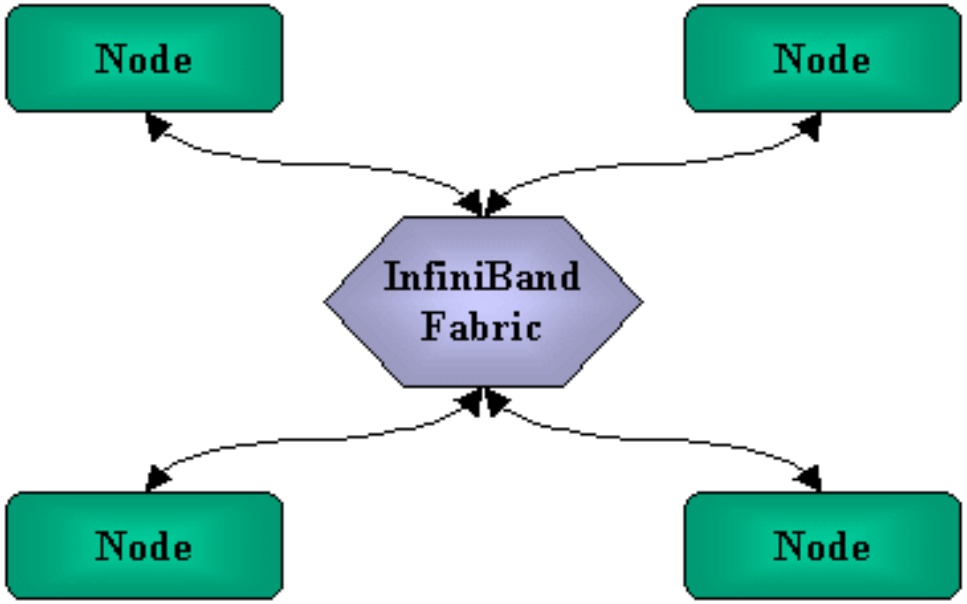
- Componentes de una red Myrinet (Myrinet 2000)
  - Cada nodo tiene una **tarjeta PCI-X** con una o dos conexiones (490MB/s y 900 MB/s respectivamente)
  - Las tarjetas se conectan con un **cable Myrinet** (fibra óptica) a un switch multi-puerto (hasta 32 puertos por switch).
  - Para redes mayores: combinaciones de switches (**Myrinet Network in a box**)
- Software Myrinet (libre)
  - **Detecta la red Myrinet** presente de manera automática (no hay que configurar el switch)
  - **GM**: software de paso de mensajes de bajo nivel sobre el que funcionan aplicaciones de alto nivel como MPI
- Myrinet Express (MX)
  - Conectado a un slot PCI-Express

Red Myrinet

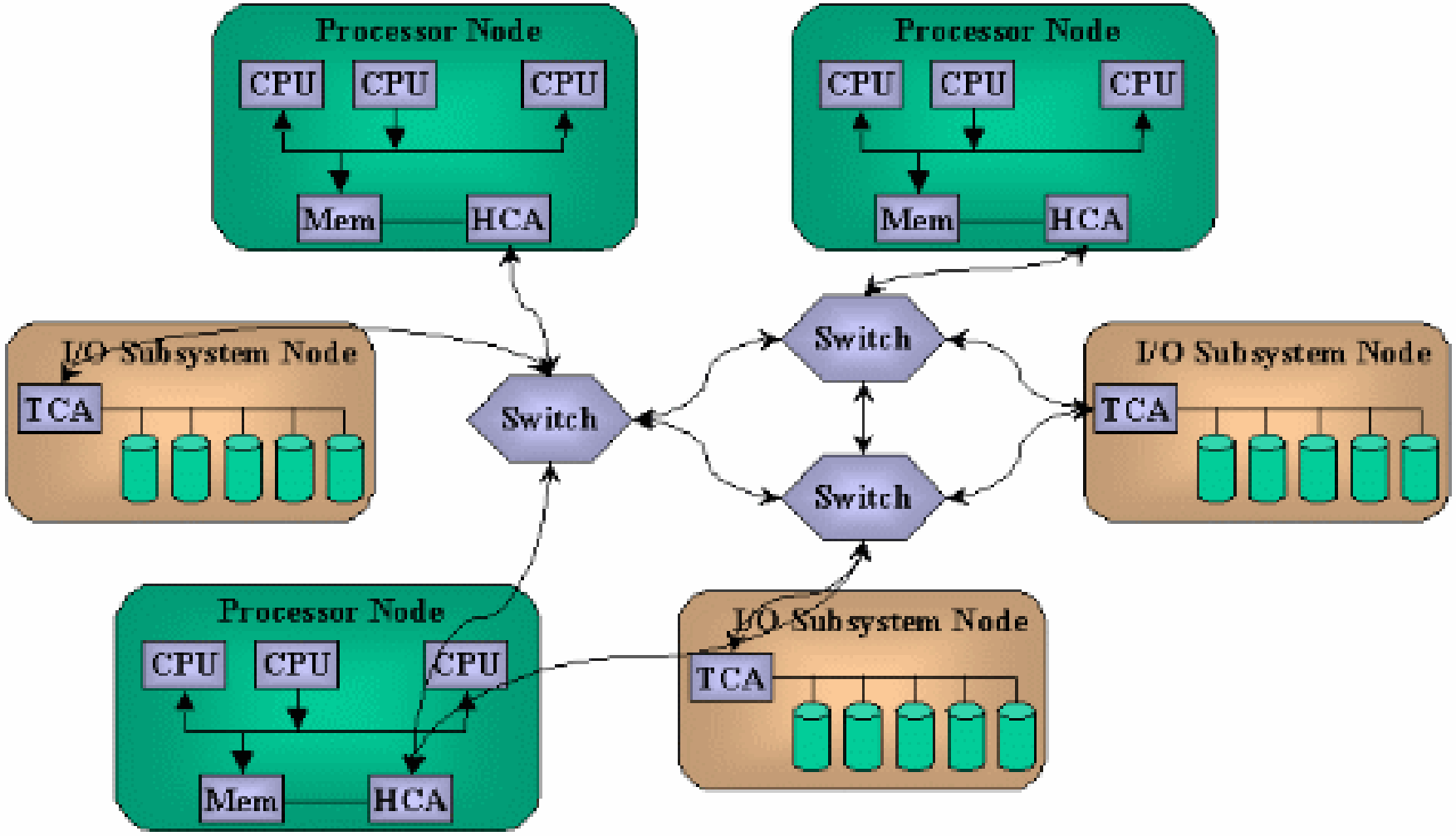


Red Infiniband

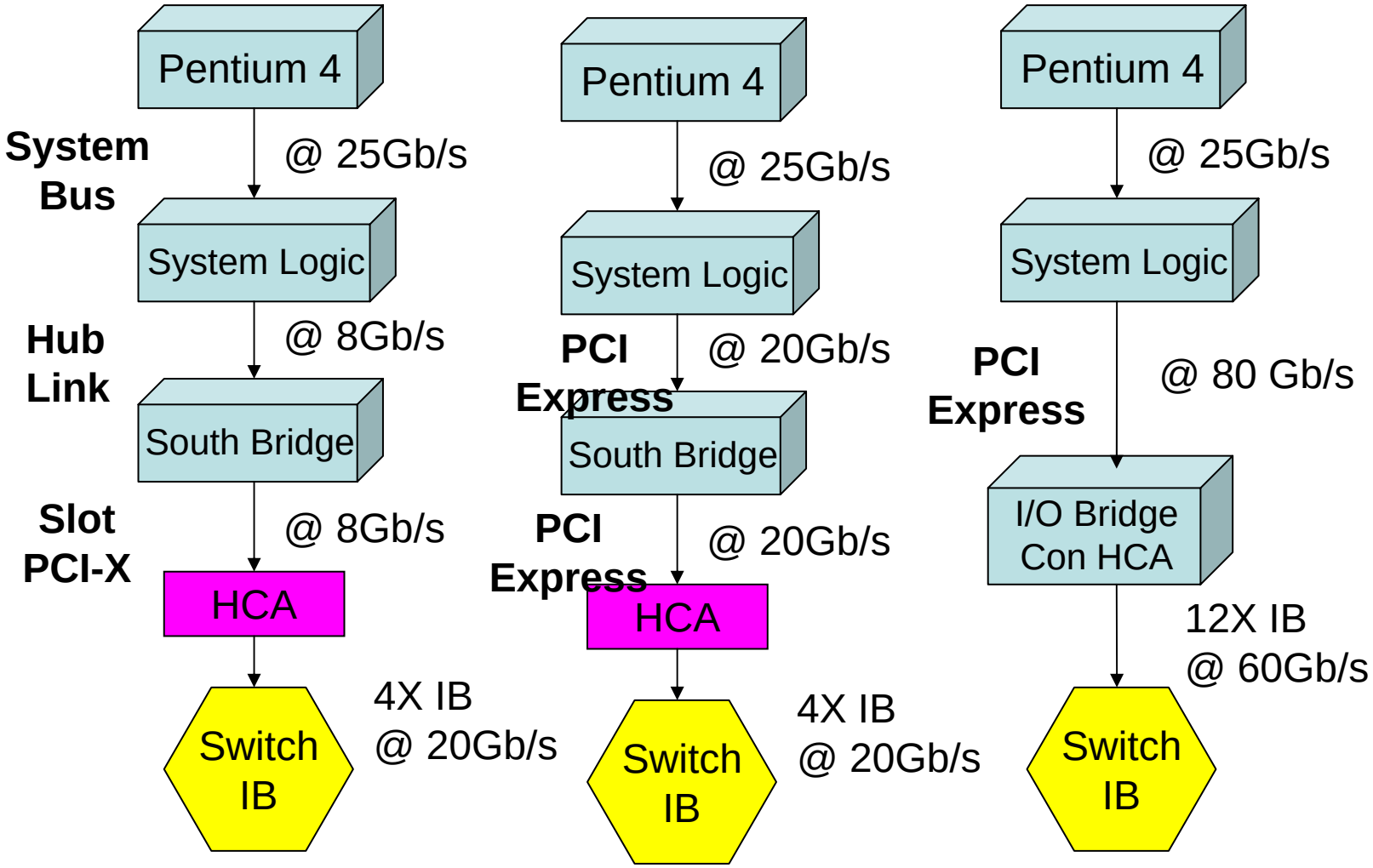
Tecnología que trata de dar respuesta a las necesidades de I/O Compartidas en clusters de ordenadores conectados a Storage Area Networks ➔ desconectar el I/O al exterior, del resto de procesos de la máquina



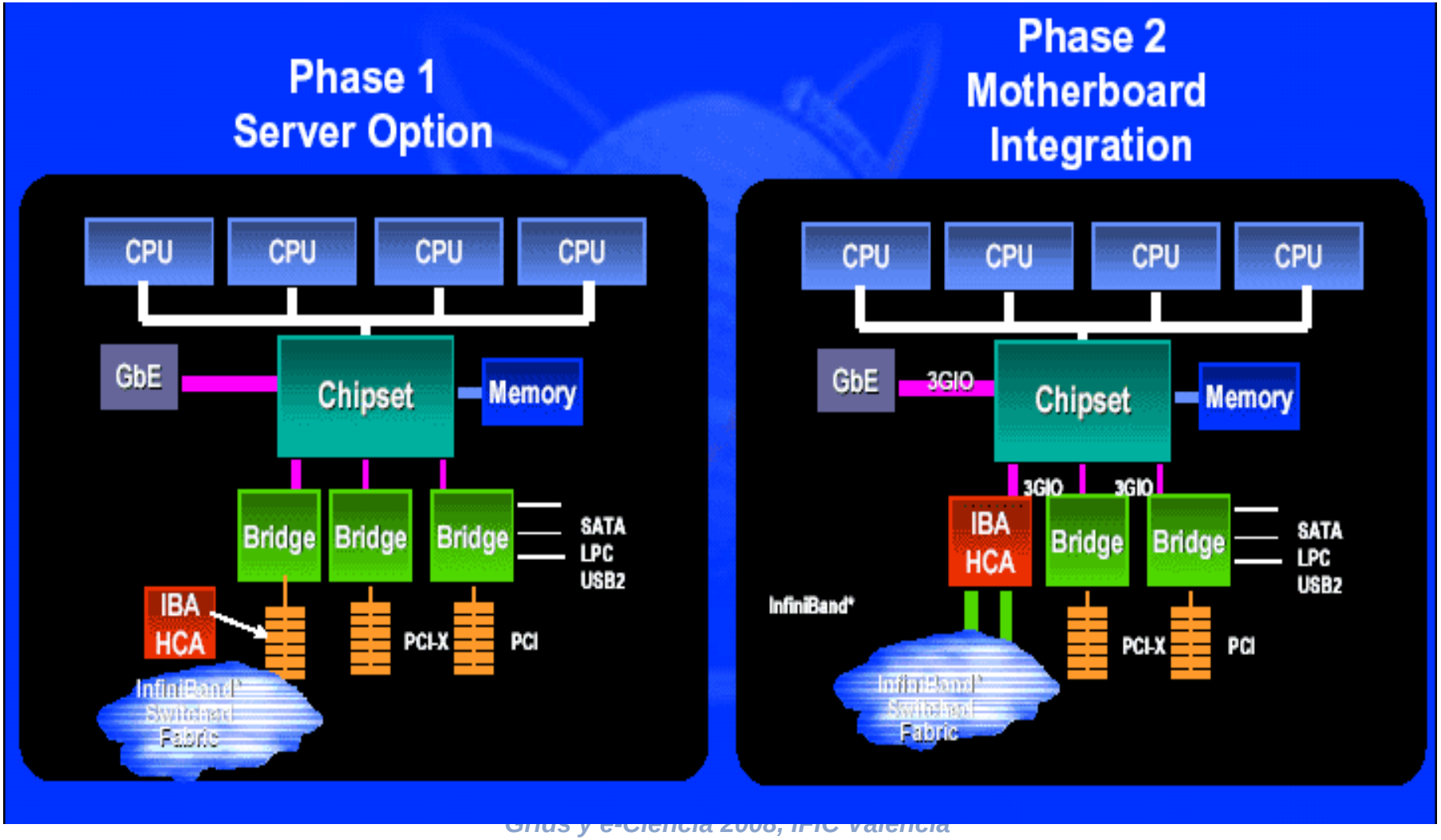
Arquitectura de Infiniband



Infiniband

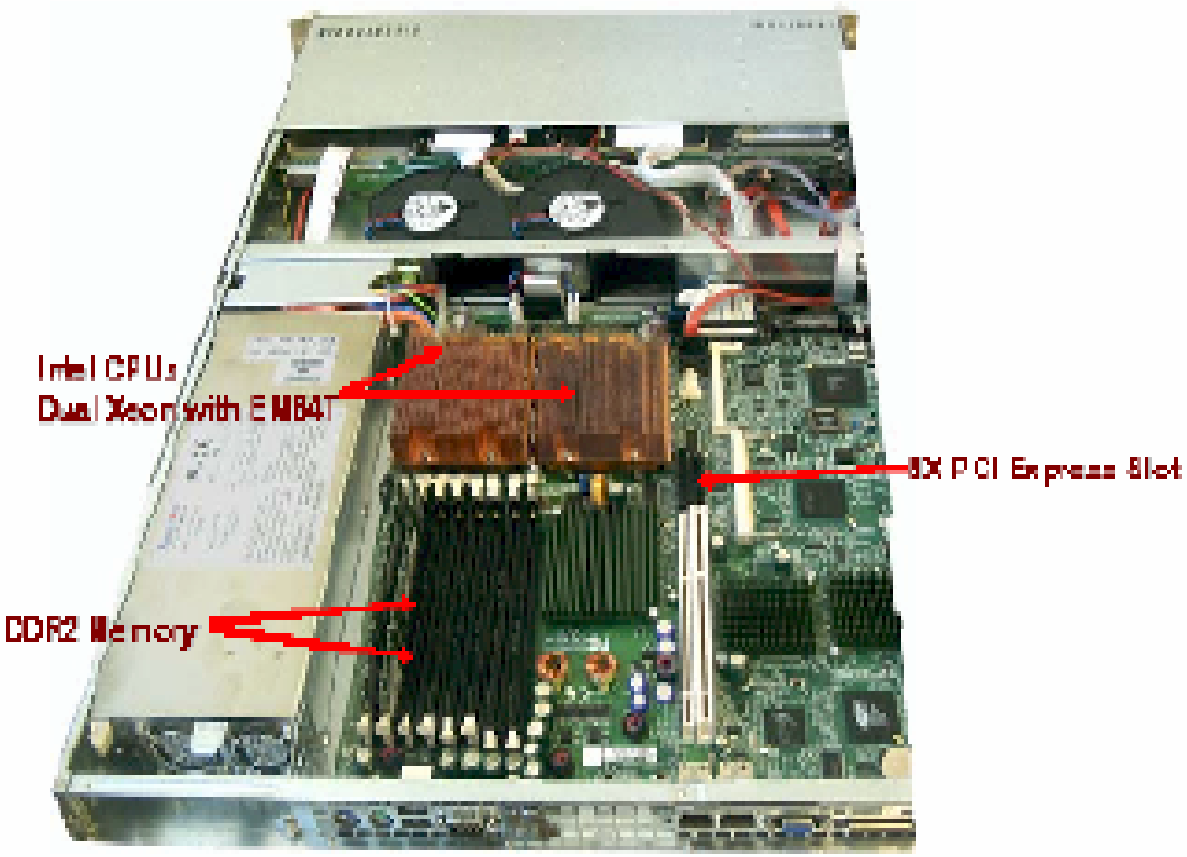

Pentium 4
System Bus
@ 25Gb/s
System Logic
Hub Link
@ 8Gb/s
South Bridge
Slot PCI-X
@ 8Gb/s
HCA
Switch IB
4X IB @ 20Gb/s
Pentium 4
@ 25Gb/s
System Logic
PCI Express
@ 20Gb/s
South Bridge
PCI Express
@ 20Gb/s
HCA
Switch IB
4X IB @ 20Gb/s
Pentium 4
@ 25Gb/s
System Logic
PCI Express
@ 80 Gb/s
I/O Bridge Con HCA
12X IB @ 60Gb/s
Switch IB

Integración de Infiniband



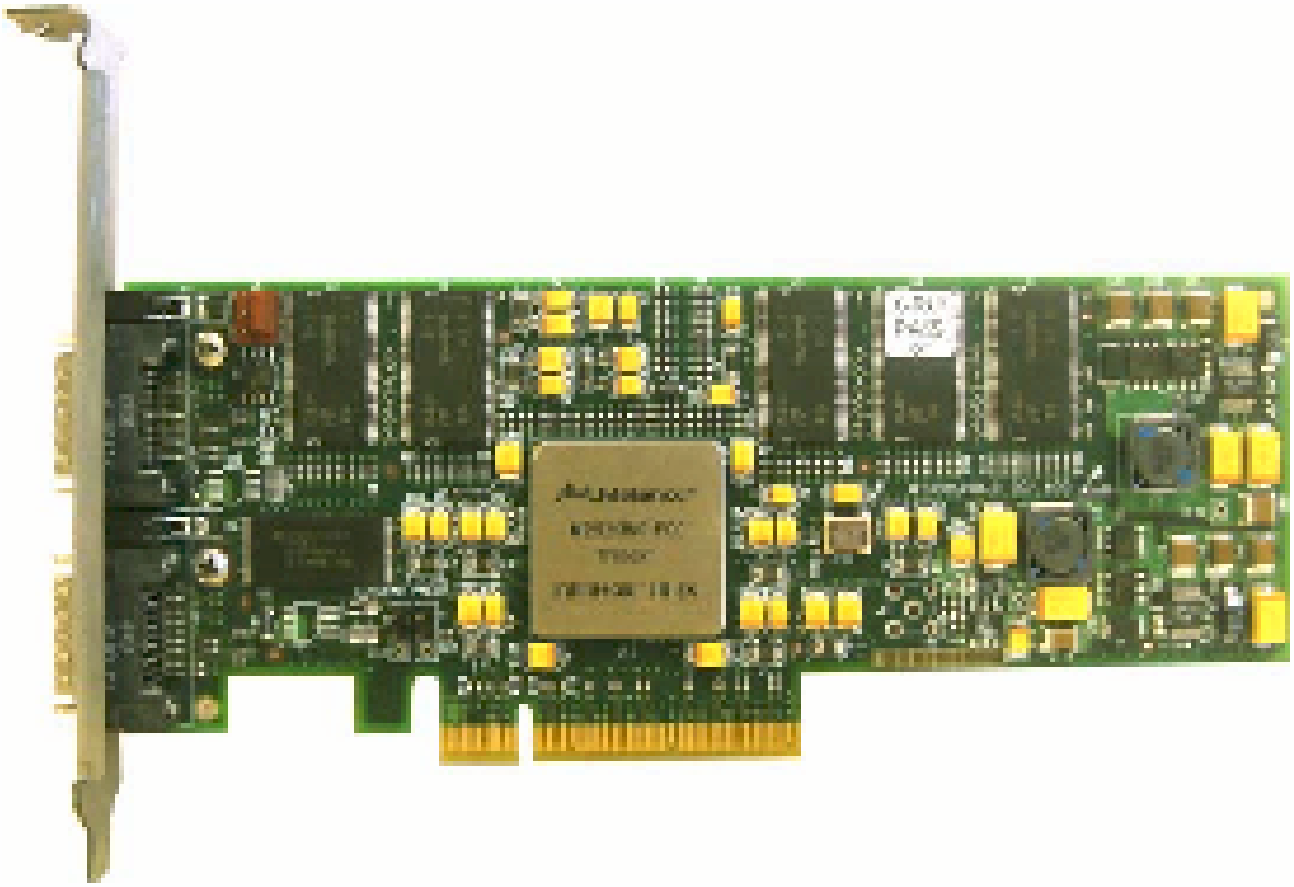
Ejemplo de Hardware Infiniband

Figure 1 Supermicro Platform 6014H-82 - 1U DP High Performance Server with PCI Express Slot



Hardware Infiniband

Figure 2. The InfiniHost III Ex device based Low Profile 8x PCI Express HCA Adapter Card



Ejercicio: el benchmark de Intel MPI Benchmark

Descripción del Benchmark

CSIC

Medida de la eficiencia de la intranet de un cluster con respecto a la ejecución de procesos MPI

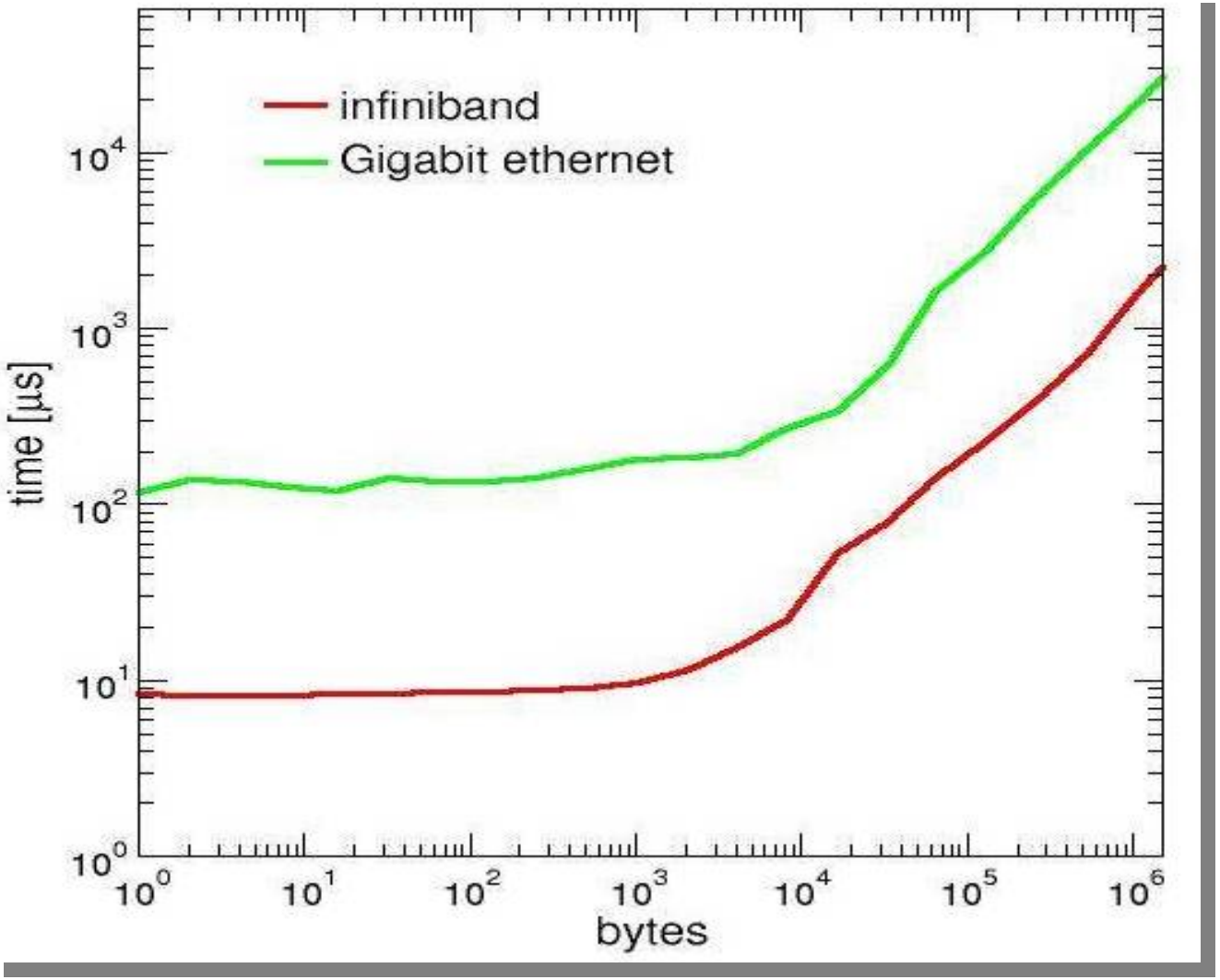
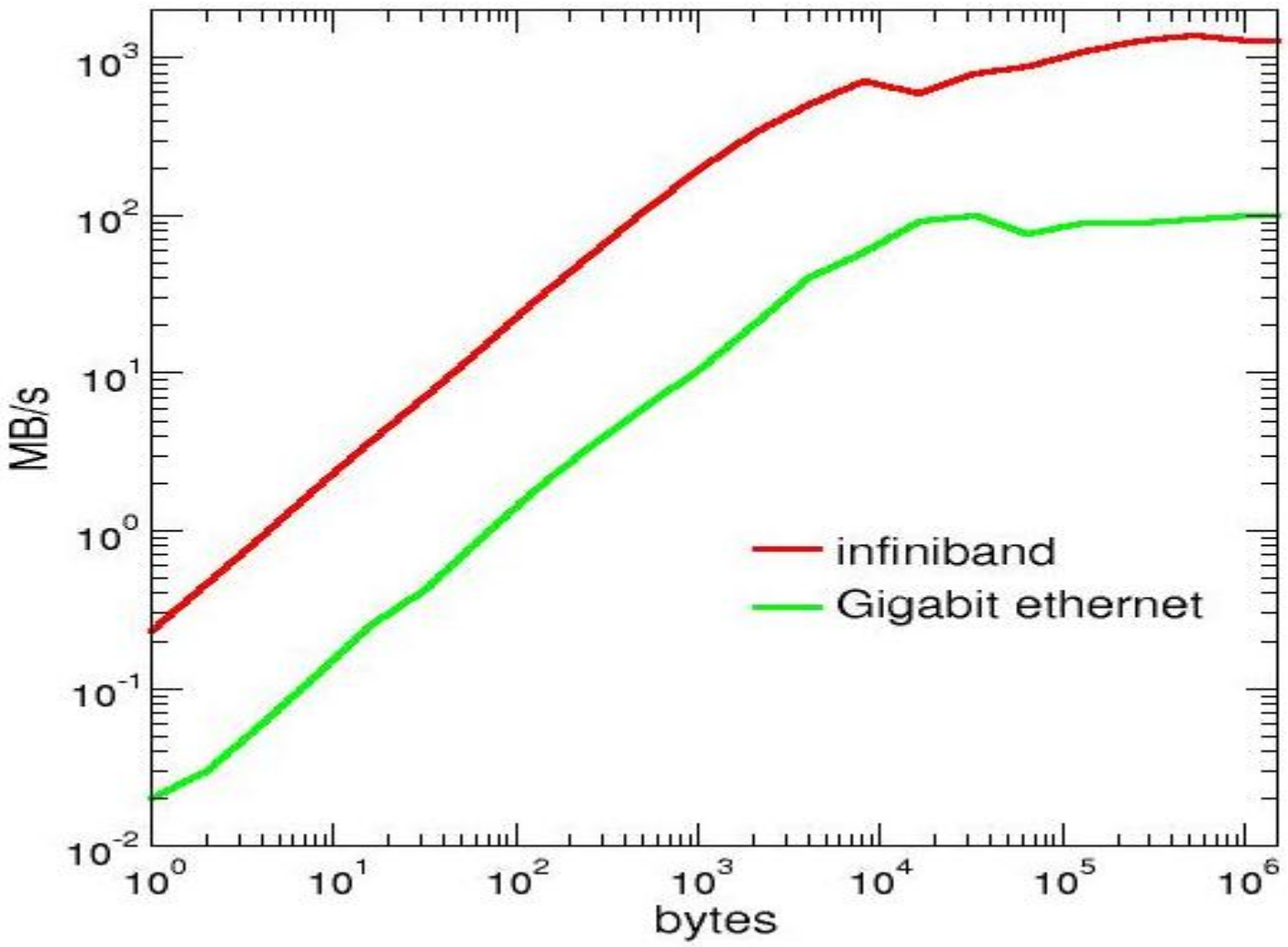
```
for test in TESTS                          # {barrier, sendrecv, ...}
  for proc in PROCS                        # {2^1,2^1, ... 2^P}
    for size in SIZES                      # {2^0, 2^1, ... 2^S}
      for repettition in REPETITIONS       # {0,1, ... N-1}
        run_test(test, size)
      }
    }
  }
}
```

Descargar de
http://wwwiscampos.ifca.es/users/iscampos/docs/GridsEciencia/IMB_3.1.tgz

Ejercicio

- **Descargar e instalar el IMB y compilarlo en el frontend de un cluster con alguna implementación de MPI instalada**
  - Compiladores de Intel
- **Usar el IMB para averiguar cual es el cluster con mejor intranet al que se tiene acceso en el Grid**

Comparación de latencias entre Infiniband y Gigabit Ethernet



Comparación de Anchura de Banda entre Infiniband y Gigabit Ethernet