Cálculo distribuido en clusters Beowulf

por Mario Storti

Centro Internacional de Métodos Numéricos

en Ingeniería - CIMEC

INTEC, (CONICET-UNL), Santa Fe, Argentina

mstorti@intec.unl.edu.ar

http://venus.ceride.gov.ar/cimec

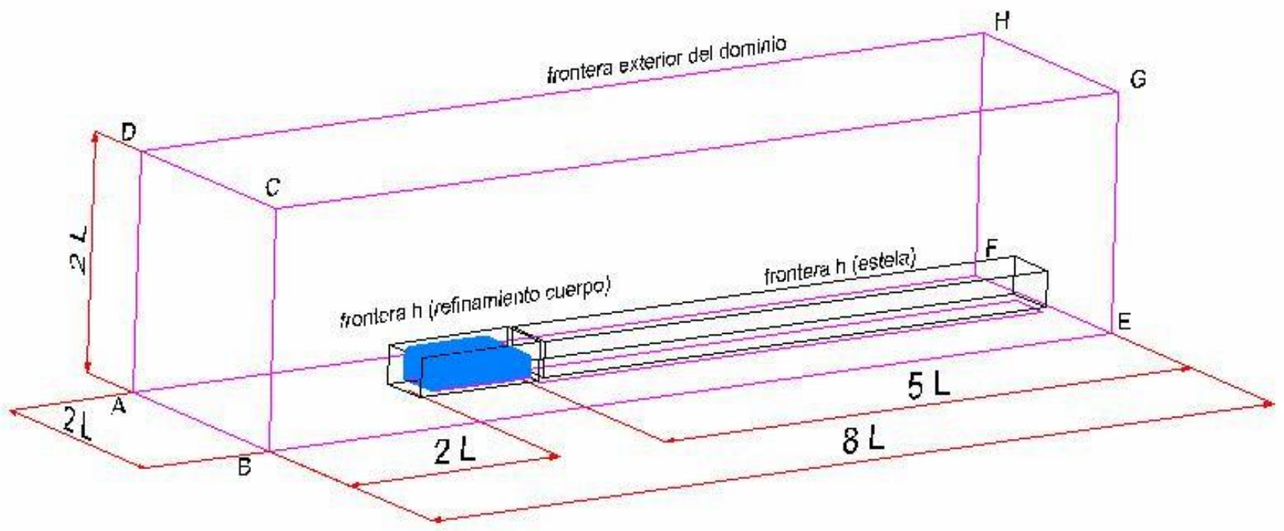
22 de marzo de 2004

Que es la Mecánica Computacional?

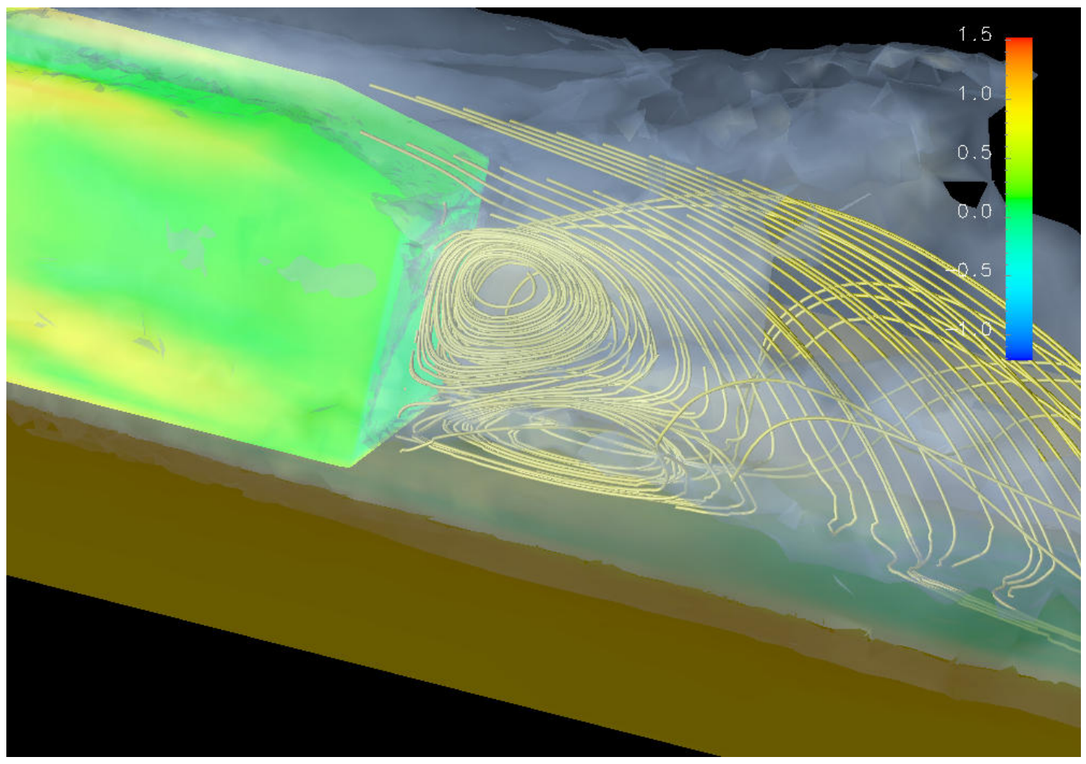
- **Resoluci’on de problemas de la Mecánica del Continuo mediante métodos computacionales.**

Los siguientes ejemplos muestran casos típicos de un problemas de ingeniería.

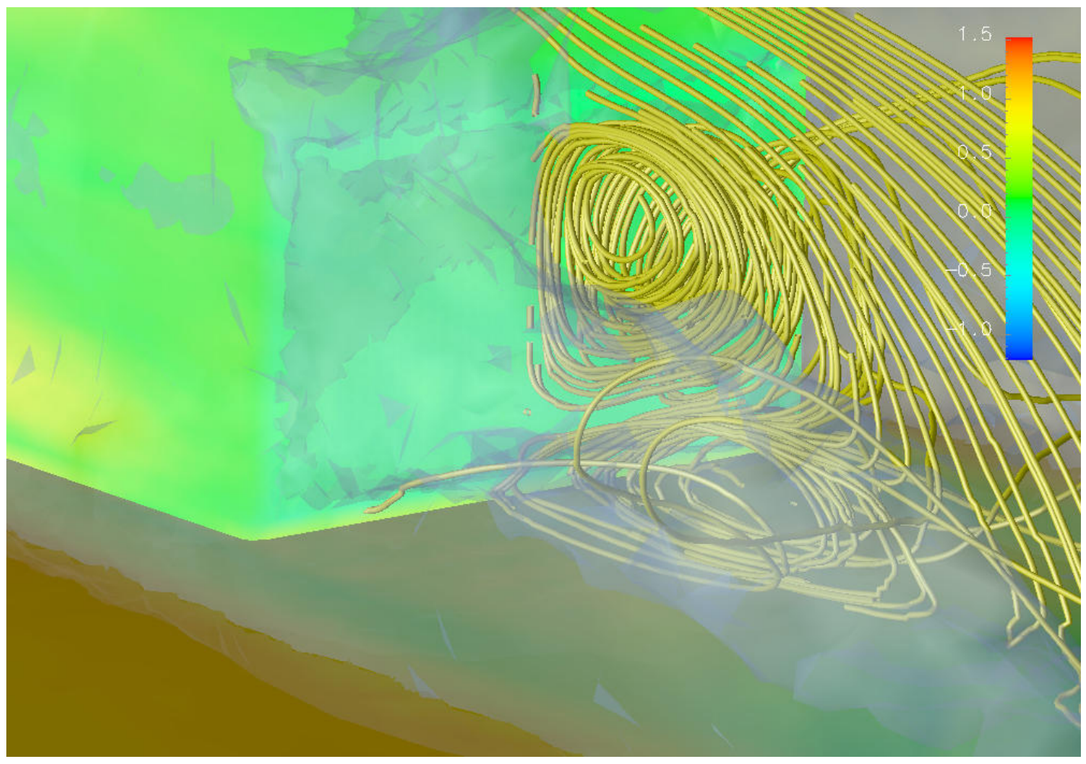
Aerodinámica exterior. Cuerpo de Ahmed



Aerodinámica exterior. Cuerpo de Ahmed (cont.)



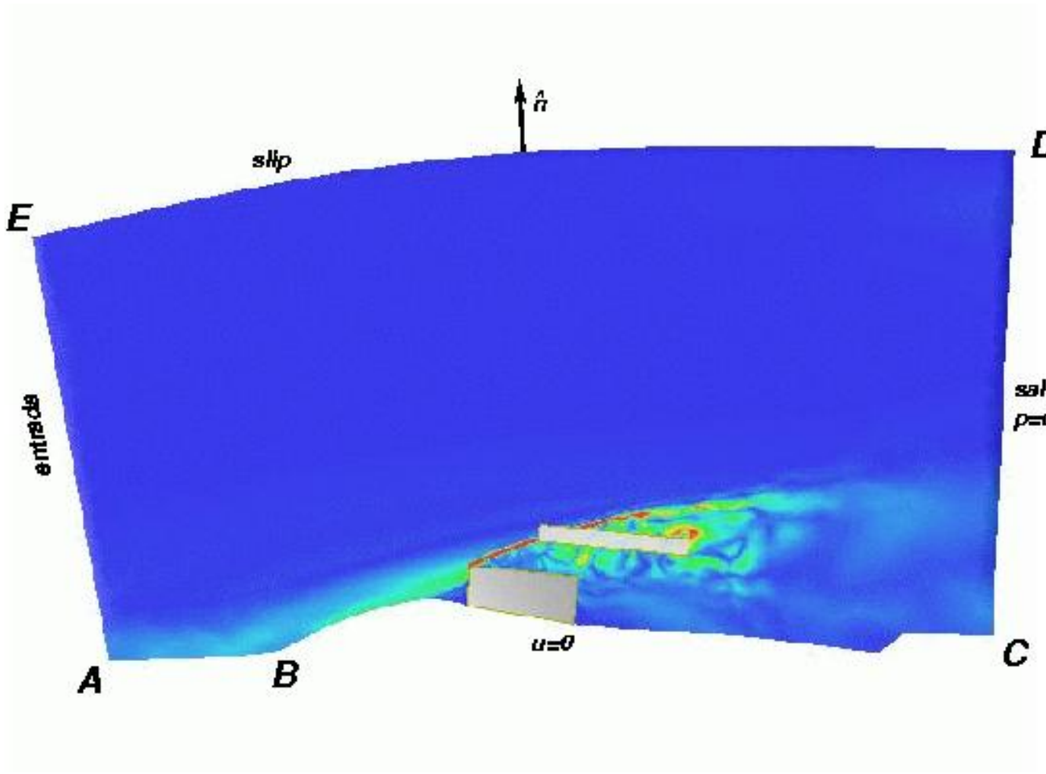
Aerodinámica exterior. Cuerpo de Ahmed (cont.)



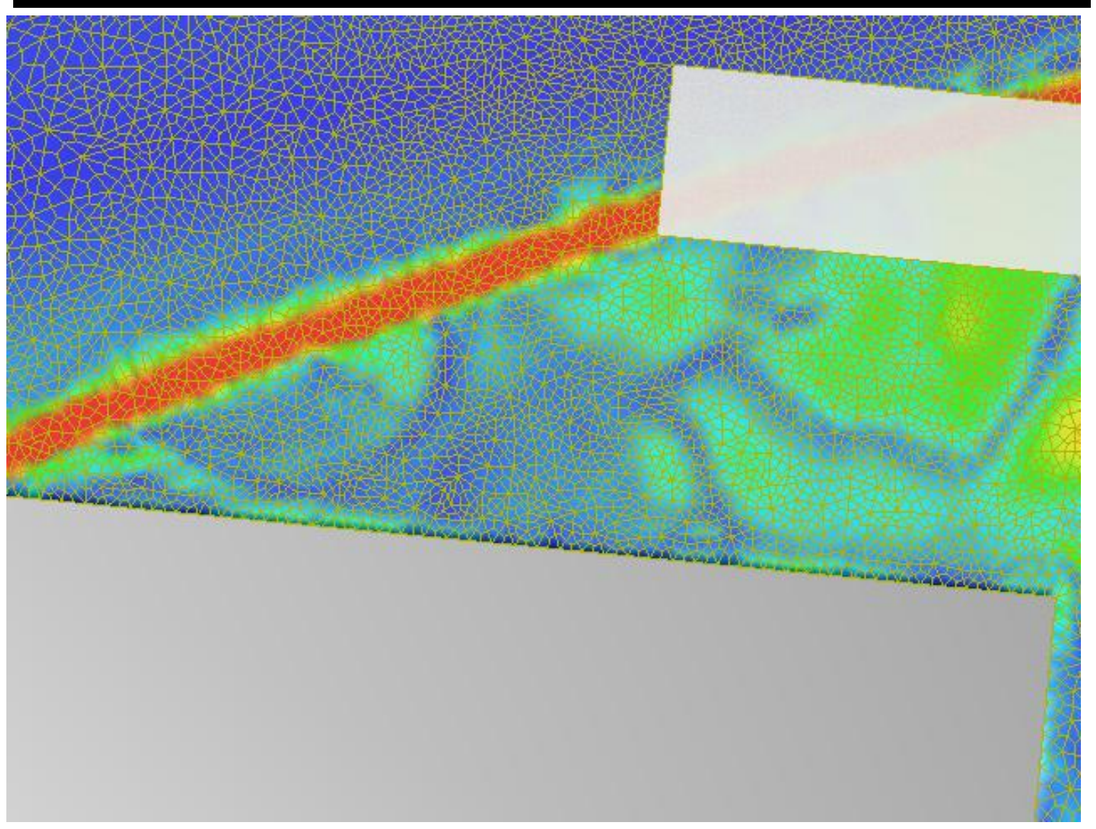

```
$ ahmed
```

Estudio de impacto ambiental en Yacyreta.

Impacto de obra civil en Puerto San Lorenzo. Rosario



Impacto de obra civil en Puerto San Lorenzo. Rosario



$ muelle

Centro Internacional de Métodos Numéricos en Ingeniería, CIMEC

Fuerzas sobre un transporte de líquidos

```
$ tank
```

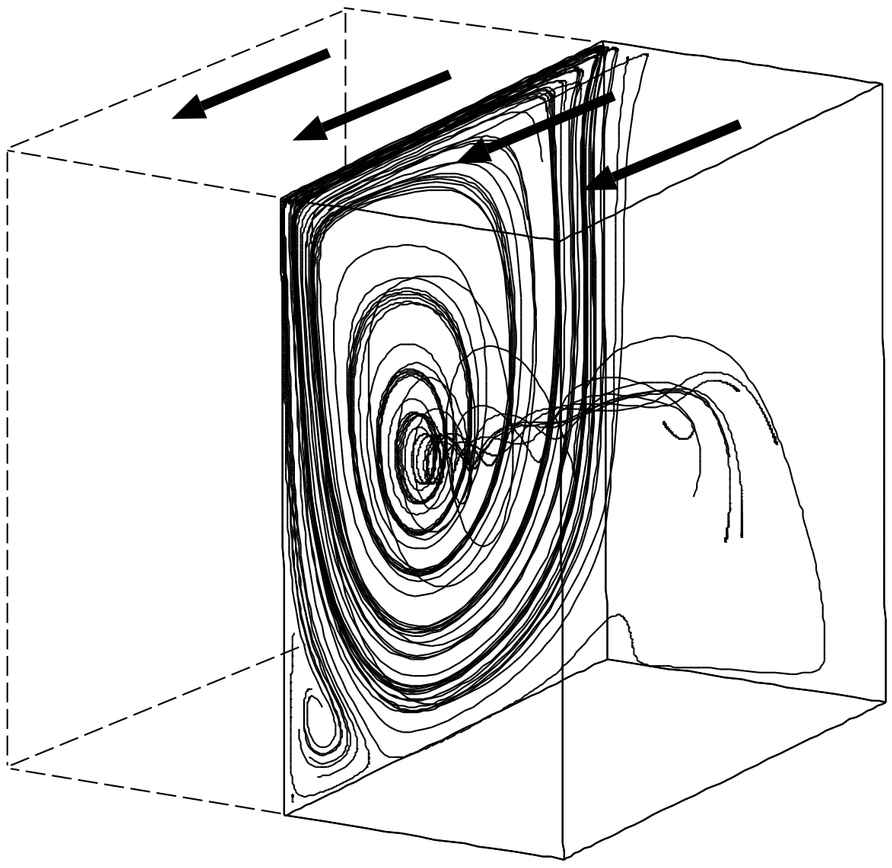
Simulación experimental

- **Para cada una de las formas propuestas crear un modelo y medir en laboratorio la perturbación en el muelle.**
- **Es un procedimiento costoso.**
- **En ciertos casos no es factible, por razones económicas, construir un modelo a escala completa sino que se debe recurrir a modelos a escala reducida, introduciendo un error en la estimación.**
- **En algunos problemas no es posible medir en forma directa: por ejemplo medir el campo de velocidades en un proceso siderúrgico con acero líquido a 1600 $^{\circ}$C.**

Simulación numérica

- **Resolver las ecuaciones que gobiernan el fluido (Ecuaciones de Navier-Stokes/Euler/Flujo Potencial).**
- **Dividir el dominio en pequeños “elementos”.**
- **Asumir variaciones simples de las variables dentro de cada elemento.**
- **Esto lleva a sistemas de ecuaciones lineales con tantas incógnitas como grados nodos por campos incógnita existen.**
- **No confundir simulación numérica con los (hoy muy comunes) efectos especiales usados en películas, etc...!!**
- **La simulación numérica es uno de las disciplinas relacionadas con la computación más antiguas. De hecho, uno de los principales usos de las primeras computadoras fue la simulación numérica en la industria de la aviación, nuclear, etc...**

Ecs. de Navier-Stokes en una cavidad cúbica.



Ecs. de Navier-Stokes en una cavidad cúbica. (cont.)

- **Resolver el movimiento de un fluido dentro de un cubo.**
- **Dividimos cada una de las dimensiones en 100 segmentos.**
- **Se originan** $100 \times 100 \times 100$**=1.000.000 de pequeños cubitos (elementos)**
- **Por cada elemento hay cuatro incógnitas: 3 componentes de velocidad y la presión:** N**=4.000.000 de incógnitas. Todas las operaciones se hacen en doble precisión.**
- **El sistema a resolver tiene una matriz de** $N \times N = 1.6 \times 10^{13} = 1.28 \times 10^{14}$**bytes = 128 Tbytes elementos. Afortunadamente casi todos los elementos de cada línea de esta matriz son cero, y sólo debemos guardar aquellos que no lo son. Hay** $N_{\mathrm{vec}} = 4 \times 27 = 108$ **coeficientes non nulos en cada línea de la matriz lo cual hace un total de** $N \times N_{\mathrm{vec}} = 4 \times 10^6 \times 108$ **= 3.4 Gb**

Ecs. de Navier-Stokes en una cavidad cúbica (cont.)

- **Factorizacion de la matriz requiere $O(Nm^2) = O(4\times10^6 \times (4\times10^4)^2) = 6.4\times10^{15})$ operaciones de punto flotante. m es el número de grados de libertad que hay en una *"capa"* 2D de nodos ($m = 4 \times 100 \times 100$). A una velocidad de procesamiento de 2.4 Gflops (2.4×10^9 operaciones de punto flotante por segundo p.ej. Pentium III 2.4 Ghz). Esto tardaría $6.4\times10^{15}/8\times10^8/3.600$ = 740 hrs!!. En realidad la velocidad de procesamiento es menor porque la velocidad de procesamiento está limitada por la velocidad con la que los datos pueden fluir desde la memoria RAM al procesador y la velocidad de procesamiento resulta en 300 Mflops. Esto lleva la evaluación del tiempo de procesamiento a ~ 6.000 hrs!!**

Ecs. de Navier-Stokes en una cavidad cúbica (cont.)

- **Los *"métodos iterativos"* permiten reducir mucho el tiempo de cómputo, pero de todas formas se requiere realizar una cantidad $N_{\rm it}$ de iteraciones, digamos $N_{\rm it} = 100$. Cada iteracion requiere un producto matriz-vector el cual requiere al menos 2 operaciones (una suma y un producto) por elemento no-nulo de la matriz. Esto hace un total de $2N \times N_{\rm vec} \times N_{\rm it} = 2 \times 4 \times 10^6 \times 108 \times 100 = 8.6 \times 10^{10}$ operaciones (approx 5 min a 300 Mflops).**
- **En problemas temporales (que evolucionan en el tiempo) esto hay que hacerlo cada paso de tiempo, durante, digamos, varios miles de pasos de tiempo.**

Ventajas

- **El costo de las simulaciones numéricas es menor que las simulaciones experimentales y tiende a disminuir constantemente.**
- **Permite conocer el estado del fluido en todo punto del dominio.**
- **No hay problemas de escala o perligrosidad.**
- **Los mismos recursos (hardware y software) pueden ser usados para una variedad de problemas de ingeniería.**

Desventajas

- **El modelo matemático a resolver no es perfectamente conocido para ciertos problemas.**
- **El grado de refinamiento a veces no es suficiente para obtener una representación adecuada.**

Técnicas de computación de alta-performance (HPC)

- **En la búsqueda de modelizaciones cada vez más precisas, se van desarrollando técnicas para incrementar la capacidad de cálculo.**
- **Una de las técnicas más comunes y prometedoras es la de dividir el problema en subproblemas más pequeños y resolver cada uno de los problemas en un procesador por separado. A esto se le da el nombre de *"procesamiento distribuido"*.**
- **En el mejor de los casos, si cada uno de los subproblemas es independiente del otro (están *"desacoplados"*) entonces el tiempo de cálculo en n procesadores es $T_n = T_1/n$.**
- **Debido a que, en general, los problemas no están desacoplados es necesario pasar cierta información de un procesador a otro. Es decir que $T_n = T_1/n + T_{\mathrm{comm}}$.**

Técnicas de comp. de alta-performance (HPC) (cont.)

- **El factor de ganancia al paralelizar se llama *"factor de aceleración"* (*"speedup"*) y se define como $S_n = T_1/T_n$. En el caso ideal en que el problema es completamente desacoplado entonces $S_n = n$. La *"eficiencia"* de la paralelización $\eta = S_n/n < 1$.**
- **Una lista de las más veloces computadoras (la lista de los *"Top 500"*) es mantenida por Jack Dongarra en `http://www.netlib.org`. Todas estas máquinas usan procesamiento distribuido, contando con hasta miles de procesadores.**

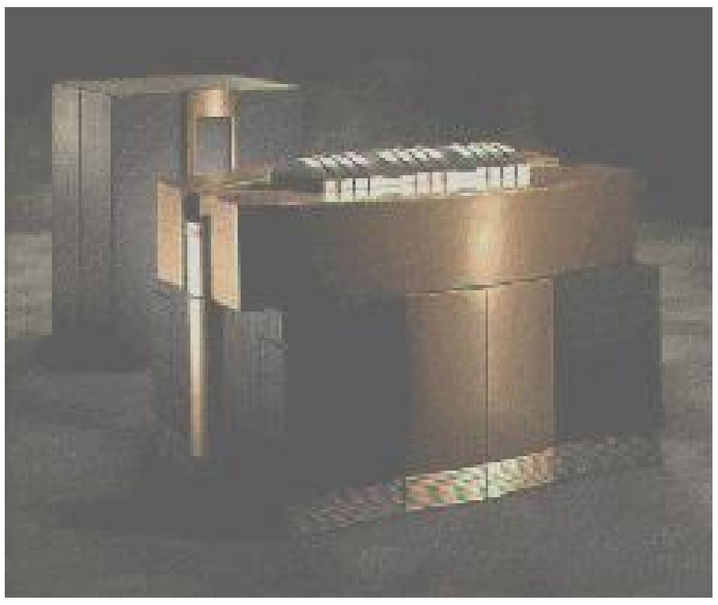
Evolución del hardware usado en el procesamiento distribuido

- **En el extremo “superior” estan las grandes firmas que venden supercomputadoras como Cray (hoy SGI) o IBM Asci-Red con miles de procesadores. Hoy las más comunes son las SGI Origin 2000.**
- **NOW/COW: A partir de los 80’s era común encontrar en los laboratorios de las universidades un cierto número de estaciones de trabajo (SUN/HP/DEC/SGI) y surgió la motivación de utilizar esta potencia de cálculo en corridas nocturnas. Surgieron las bibliotecas de *“Paso de Mensajes”* como PVM y MPI.**

Cray T94. CSC/UFRGS - Porto Alegre Brasil

- **Cray T94 en Centro de Supercomputación UFRGS, Porto Alegre Brasil.**
- **2 Procs. $\times$ 1.8 Gflops, Costo aprox. $ 5,000,000.**

  `http://www.cesup.yfrgs.br`



Clementina II, SETCIP

- **SETCIP Secretearía de la Tecnología e Innovación Productiva.**
- **Silicon Graphics Origin 2000, 40 procs.**
- **Costo aprox. $ 3,000,000**
- **`http://www.setcip.org.ar`**

Clusters Beowulf

- **Con el abaratamiento de las PC's y el advenimiento de software libre surgió la posibilidad de crear clusters de PC's completamente dedicados a cálculo. Estos son los llamados clusters Beowulf.**
- **De *"How to build a Beowulf"* (Sterling, T.L. et.al., MIT Press, 1999) un *"cluster Beowulf"* es *" Un cluster the 'mass-market commodity off-the-shelf' (M2COTS) PC's interconectadas por tecnología LAN de bajo costo corriendo un OS open source de tipo Unix y ejecutando aplicaciones en paralelo con una librería de paso de mensajes estándar en la industria."* El *"Proyecto Beowulf"* fue desarrollado originalmente en el *Goddard Space Flight Center* (GSFC). También son populares los clusters con procesadores DEC/Alpha.**

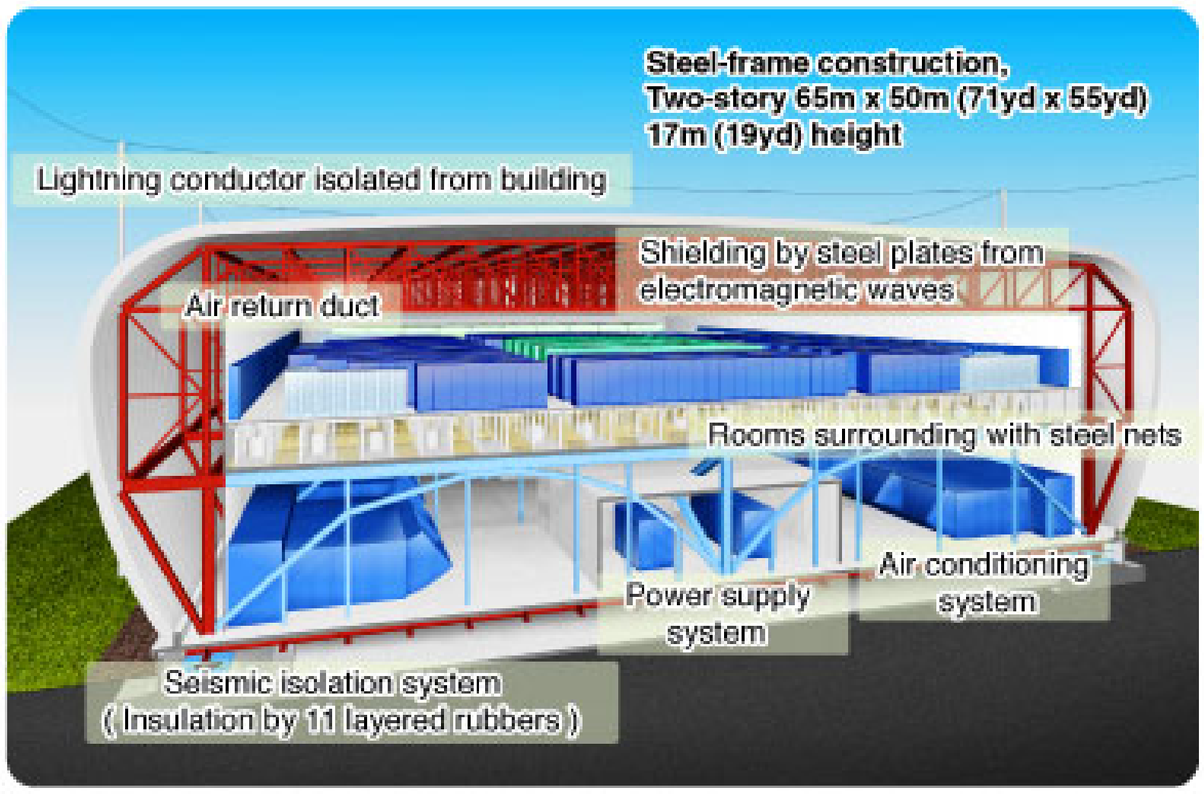
Top 10 list

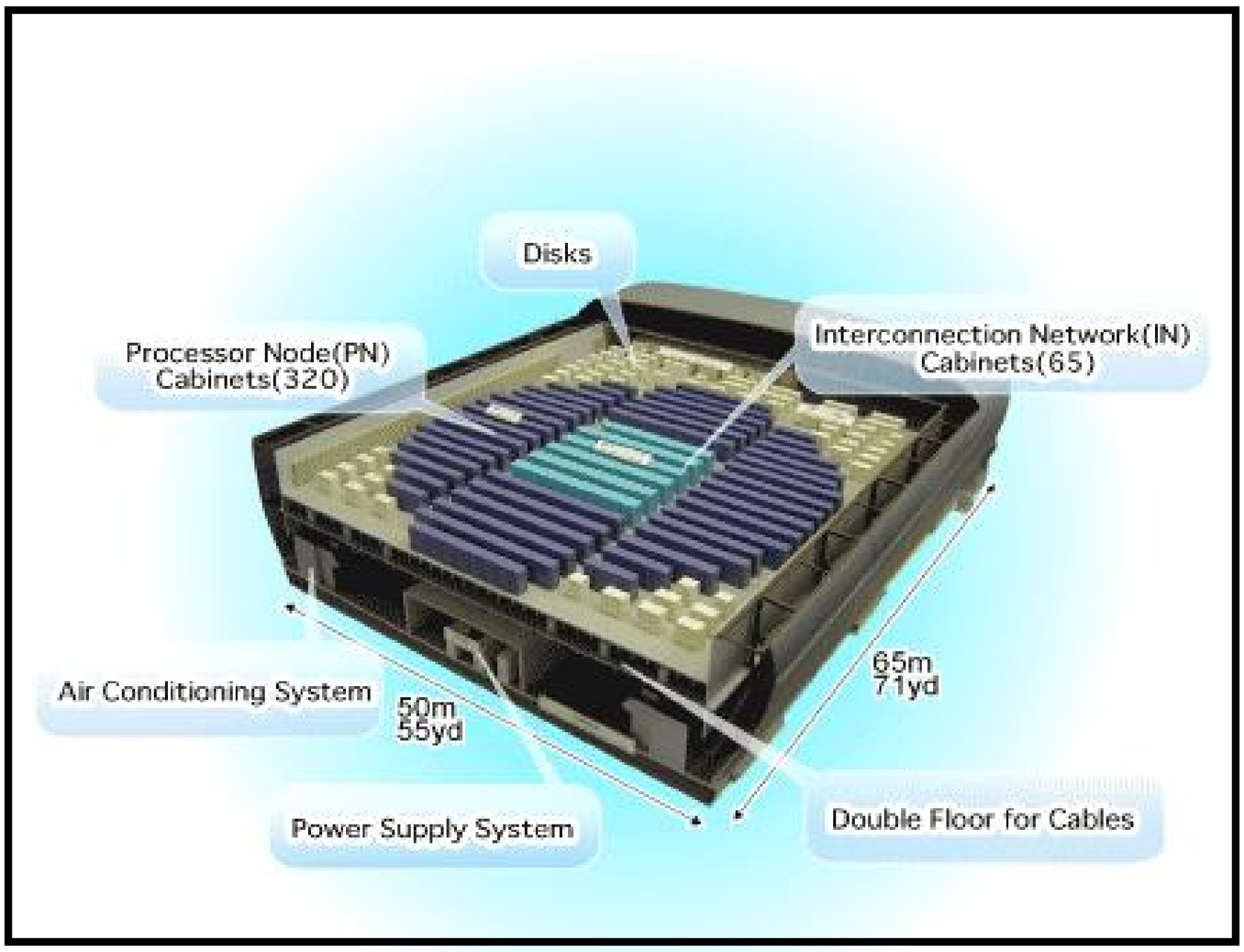
(sacado de www.top500.org)

- **The Earth Simulator, built by NEC, remains the unchallenged #1 at 40 Tflops. (4e12 flop/sc)**
- **ASCI Q at Los Alamos is still #2 at 13.88 TFlop/s.**
- **The third system ever to exceed the 10 TFflop/s mark is Virgina Tech's X measured at 10.28 TFlop/s. This cluster is built with the Apple G5 as building blocks and is often referred to as the 'SuperMac' in media reports. It uses a Mellanox network based on the new Infinband technology as interconnect.**
- **The fourth system is also a cluster. The Tungsten cluster at NCSA is a Dell PowerEdge-based system using a Myrinet interconnect. It just missed the 10 TFlop/s mark with a measured 9.82 TFlop/s.**
- **The list of clusters in the TOP10 continues with the upgraded Itanium2-based Hewlett-Packard system, located at DOE's Pacific Northwest National Laboratory, which uses a Quadrics interconnect.**

Earh simulator (1st pos.)

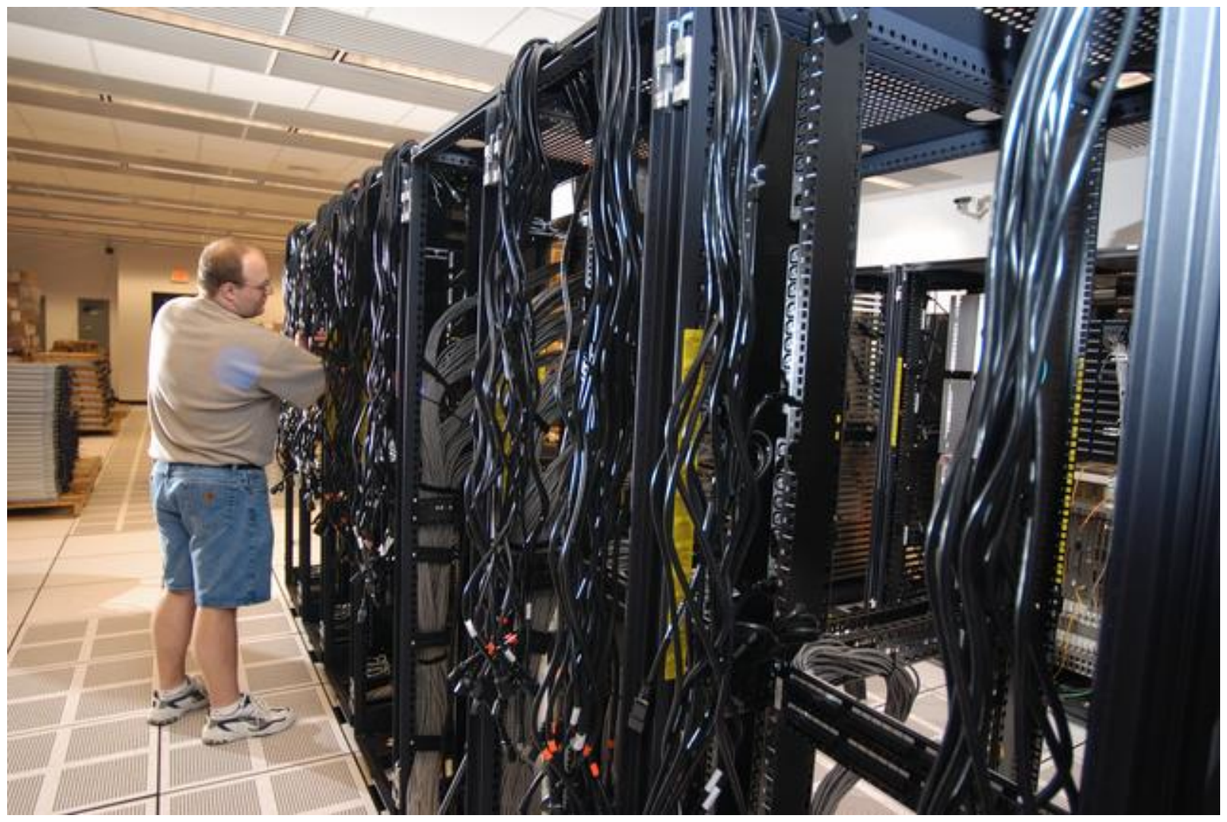
- **5,120 (640 8-way nodes) 500 MHz NEC CPUs**
- **8 GFLOPS per CPU (41 TFLOPS total)**
- **2 GB (4 512 MB FPLRAM modules) per CPU (10 TB total)**
- **shared memory inside the node**
- **640 640 crossbar switch between the nodes**
- **16 GB/s inter-node bandwidth**
- **20 kVA power consumption per node**


Steel-frame construction,
Two-story 65m x 50m (71yd x 55yd)
17m (19yd) height
Lightning conductor isolated from building
Shielding by steel plates from electromagnetic waves
Air return duct
Rooms surrounding with steel nets
Air conditioning system
Power supply system
Seismic isolation system
(Insulation by 11 layered rubbers)


Disks
Processor Node(PN)
Cabinets(320)
Interconnection Network(IN)
Cabinets(65)
Air Conditioning System
50m
55yd
65m
71yd
Power Supply System
Double Floor for Cables

Tungsten (4th pos.)

Tungsten, NCSA’s newest cluster, will employ more than 1,450 dual-processor Dell PowerEdge 1750 servers running Red Hat Linux, a Myrinet 2000 high-speed interconnect fabric, an I/O subcluster with more than 120 terabytes of DataDirect storage, and a dedicated 64-node applications development environment. Tungsten is expected to have a peak performance of 17.7 teraflops (17.7 trillion calculations per second).



MPP2/PNNL

- **System Name: Mpp2**
- **Purpose: Production**
- **Platform: HP/Linux**
- **Nodes: 980**
- **Processors: 1960**
- **Operating System: Red Hat Linux 7.2**
- **Cluster Management: RMS**
- **File System: NFS (/msrc, /home), Lustre* (/dtemp) and local (/scratch)**
- **Compilers: C (ecc), F77/F90/F95 (efc), G++**
- **Batch Scheduler: LSF**
- **Code Development: TotalView, Vampir, gdb**

CLIC. Chemniz University Cluster

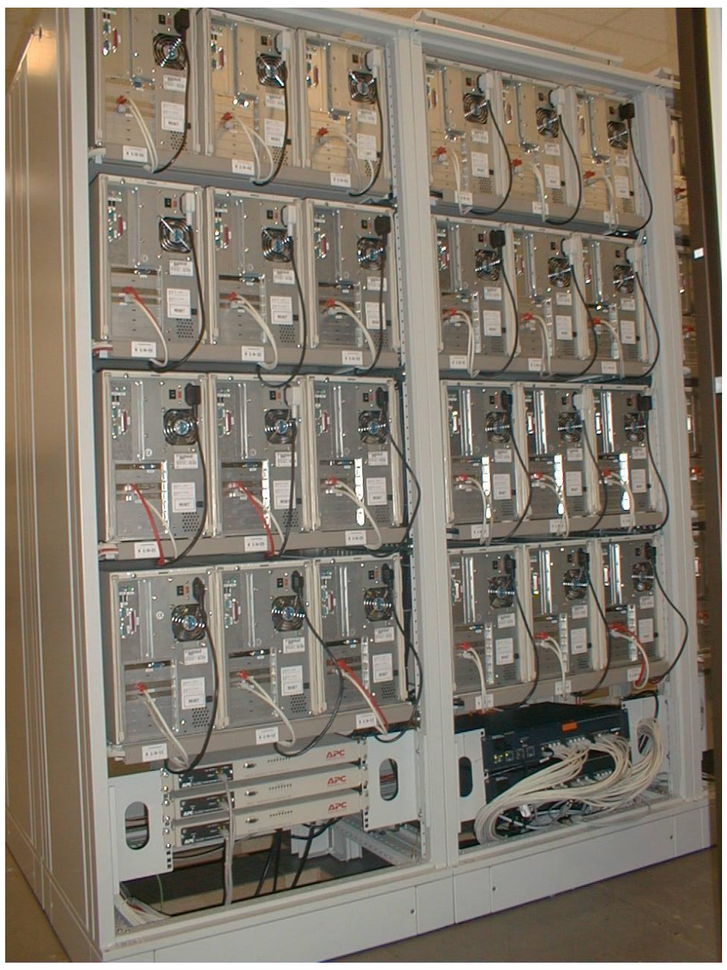
Hardware: Compute Nodes, 528 units, each equipped with

- **Intel PentiumIII, 800 MHz**
- **ASUS-Mainboard CUBX**
- **512 MB SDRAM PC100**
- **harddisk Seagate Barracuda II 20,4 GB**
- **floppy**
- **gfxcard ATI 3D Carger 4MB**
- **2 x network adapter Level One FNC 0108TX**

CLIC. Chemniz University Cluster (cont.)

Server Nodes, 2 units, each equiped with

- **Intel PentiumIII, 800 MHz**
- **ASUS-Mainboard CUBX**
- **1 GB SDRAM PC100**
- **harddisk Seagate Barracuda II 20,4 GB**
- **floppy**
- **gfxcard ATI 3D Carger 4MB**
- **2 x Gigabit Ethernet Server Adapter SK 9843-SK-Net GE SX**
- **CD-ROM**

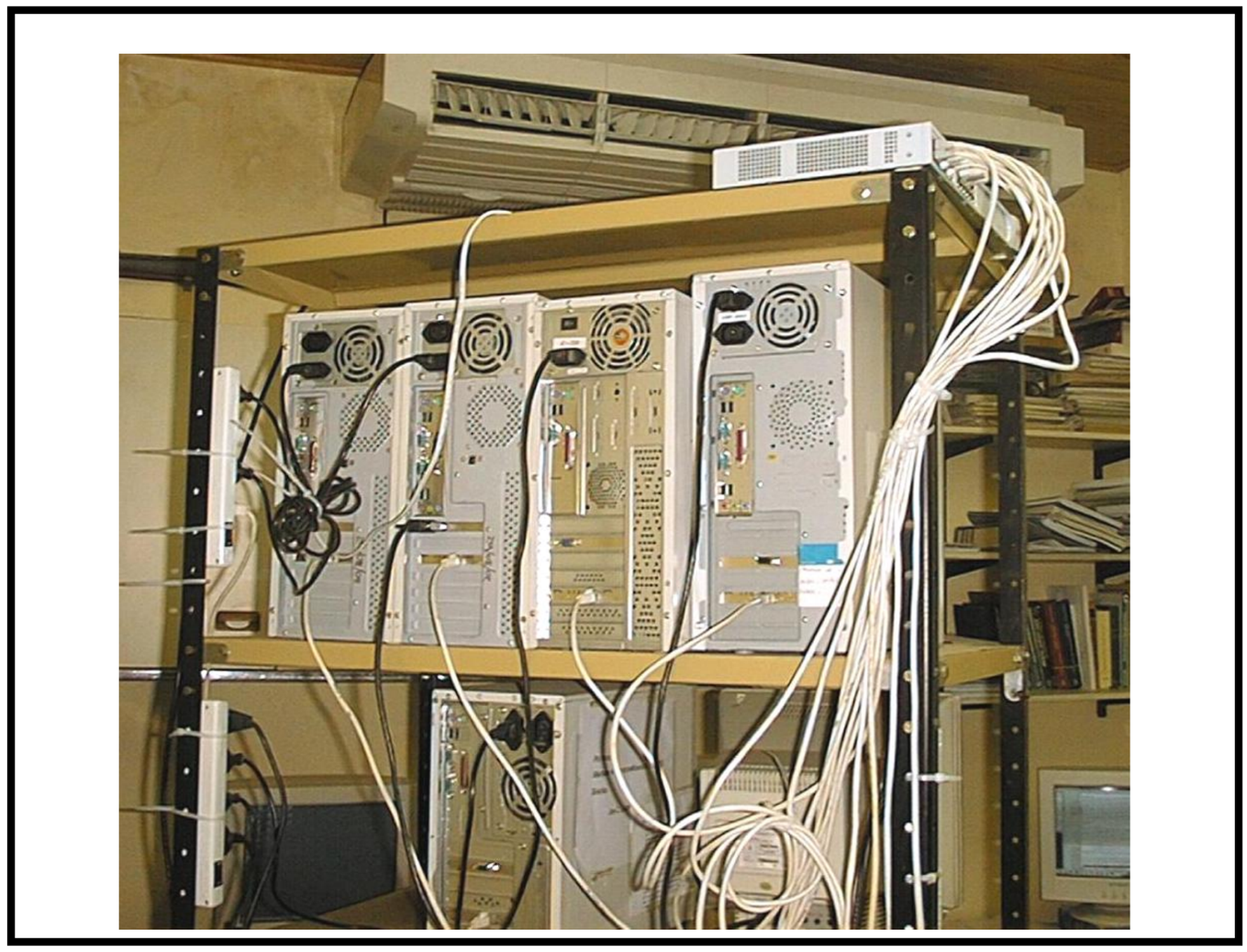


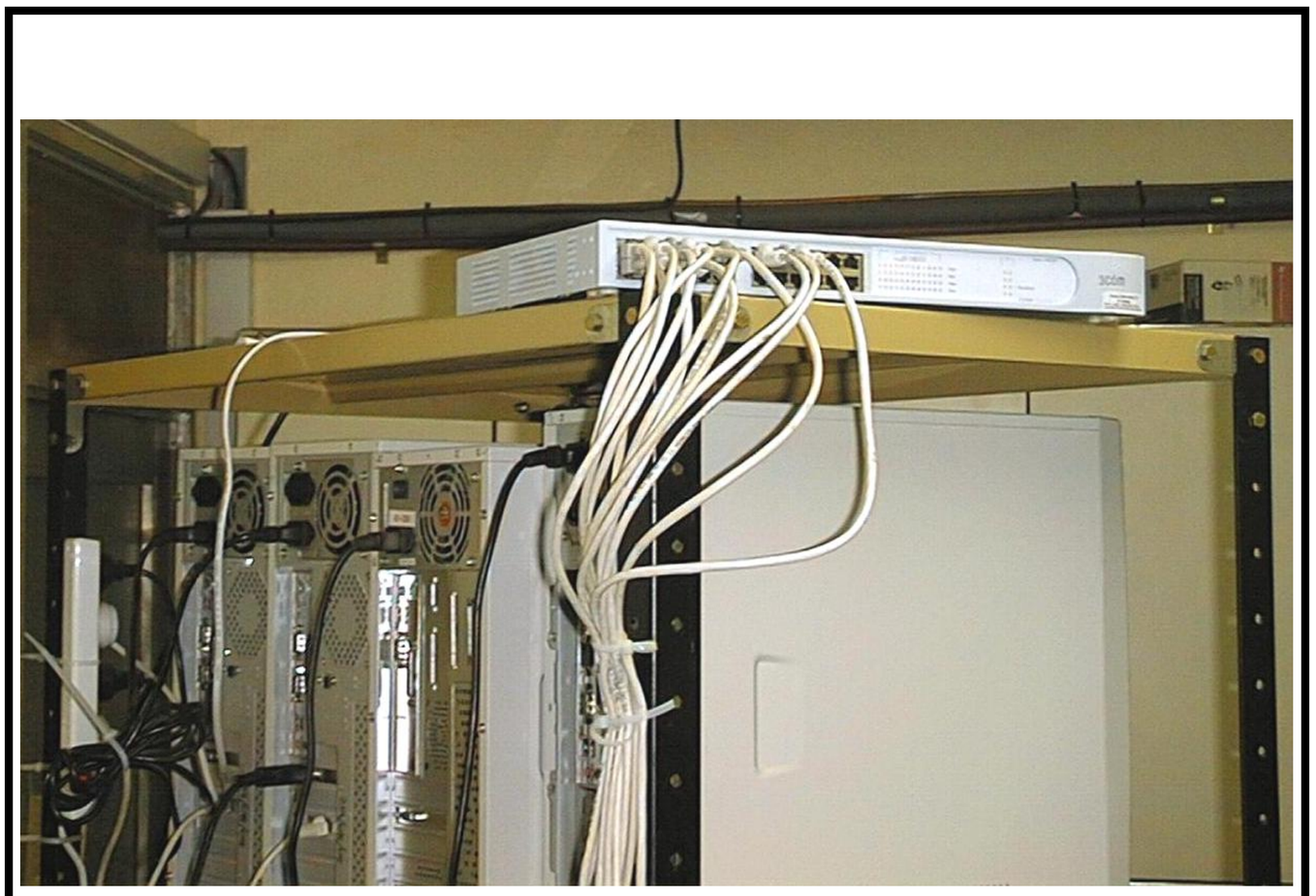
El cluster “Geronimo” en el CIMEC

- **El CIMEC (Centro Internacional de Métodos Numéricos en Ingeniería, ubicado en Santa Fe, dependiente del CONICET) desarrolla tareas de investigación en métodos numéricos desde 1982.**
- **Desde el año 1997 se vienen desarrollando experiencias en procesamiento distribuido. Originalmente en 4 procesadores DEC/Alpha 500/333 Mhz.**
- **Desde 1999 este esfuerzo se ha orientado a los cluster de PC corriendo bajo GNU/Linux. Actualmente, el cluster cuenta con**
  - **Frontend: P IV 1.7 GHz con 256MB RAM RIMM. 80GB HD, 2 3COM NIC cards.**
  - **10 compute nodes P IV 2.4 GHz HT con 1Gb RAM DDR 400 MHz**
  - **6 compute nodes P IV 1.7 GHz con 512MB RAM RIMM**
  - **All compute nodes have 3COM NIC Cards**
  - **Switch Encore ENH924-AUT+ Fast Ethernet 100MB/sec switch**

Arquitectura

- **La configuración del cluster es “disk-less” es decir que los nodos no tienen disco duro.**
- **Los nodosCargan el kernel de un diskette y montan el root filesystem via NFS en el server.**
- **El uso de la configuración disk-less permite una administración mucho más simple e implica un considerable ahorro ya que no requiere disco duro.**
- **Nuevos nodos pueden ser instalados en menos de 5 minutos/nodo.**





Software utilizado

- **OS: GNU/Linux RedHat 7.1.**
- **Desarrollamos un código de elementos finitos escrito en C++ llamado PETSc-FEM. Actualmente tiene +81,000 líneas de código.**
- **Como librería de paso de mensajes usamos MPI (*"Message Passing Interface"*, `http://www.mcs.anl.gov/mpi`) en su implementacion MPICH (versión 1.2.2, `http://www.mcs.anl.gov/mpich`) desarrollados en el *Argonne National Laboratory* (ANL).**

Software utilizado. (cont.)

- **En general MPI no es llamado directamente sino a través de PETSc (versión 2.1.6) *Parallel Extensible Toolkit for Scientific Computations* que es un paquete orientado a métodos numéricos en procesamiento distribuido, y permite operaciones abstractas como definir vectores y matrices distribuidos y resolver los sistemas lineales asociados.**
- **Para el álgebra lineal se utiliza los paquetes estándar Lapack y Blas distribuidos por Netlib (`http://www.netlib.org/lapack/`).**
- **Sistema de manejo de colas PBS (Portable Batch System.)**

Características de PETSc-FEM

- **GPL, accesible en `http://venus.ceride.gov.ar/petscfem`, versión actual es `petscfem-beta-3.29`**
- **Programa de elementos finitos de uso general y multi-física.**
- **Procesamiento en paralelo via uso de PETSc/MPI/Metis.**
- **Escrito en C++ con una concepción OOP, con especial énfasis en la eficiencia.**
- **Muchos tipos de elementos pueden ser usados en la misma aplicación, agrupando los elementos del mismo tipo en "elemsets".**
- **Propiedades de elementos pasadas a las rutinas de elementos via correspondencias *string*$\longrightarrow$*string* genéricas.**

Características de PETSc-FEM (cont.)

- **Condiciones de contorno Dirichlet/Newman/mixtas/periódicas o restricciones generales.**
- **Cálculo de jacobianos numéricos.**
- **Actualmente implementados módulos de Navier-Stokes, shallow-water, Euler, sistemas advectivos generalizados y ec. de Laplace.**
- **Perfiles de sistemas de ecuaciones calculados automáticamente.**
- **Librerías de matrices rápidas usando “caches” para las direcciones de memoria.**
- **Balance de carga**

Rendimiento en clusters heterogéneos

- **Si un procesador es más rápido y se asigna la misma cantidad de tarea a todos los procesadores independientemente de su velocidad de procesamiento, cada vez que se envía una tarea a todos los procesadores, los más rapidos deben esperar a que el más lento termine, de manera que la velocidad de procesamiento es a lo sumo n veces la del procesador más lento. Esto produce un deterioro en la performance del sistema para clusters heterogéneos. El concepto de speedup mencionado anteriormente debe ser extendido a grupos heterogéneos de procesadores.**

Rendimiento en clusters heterogéneos (cont.)

- **Supongamos que el trabajo W (un cierto número de operaciones) es dividido en n partes iguales $W_i = W/n$**
- **Cada procesador tarda un tiempo $t_i = W_i/s_i = W/ns_i$ donde s_i es la velocidad del procesador i (por ejemplo en Mflops). El hecho de que los t_i sean distintos entre si indica una pérdida de eficiencia.**
- **El tiempo total transcurrido es el mayor de los t_i, que corresponde al menor s_i:**

$$T_n = \max_i t_i = \frac{W}{n \min_i s_i} \qquad (1)$$

Rendimiento en clusters heterogéneos (cont.)

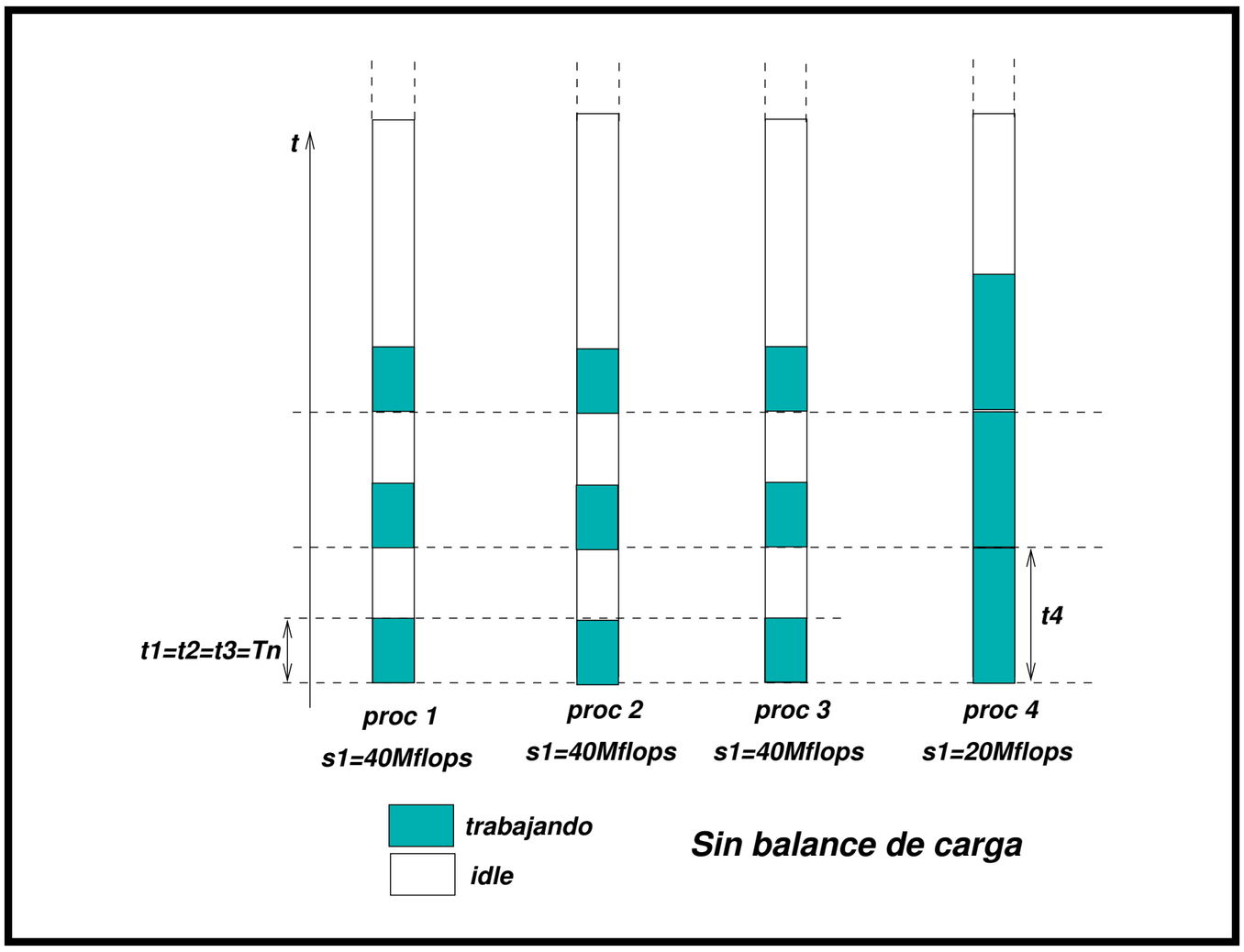
- **El tiempo T_1 para hacer el speedup podemos tomarlo como el obtenido en el más rápido, es decir**

$$T_1 = \min t_i = \frac{W}{\max_i s_i} \tag{2}$$

- **El speedup resulta ser entonces**

$$S_n = \frac{T_1}{T_n} = \frac{W}{\max_i s_i} \Big/ \frac{W}{n \min_i s_i} = n \frac{\min_i s_i}{\max_i s_i} \tag{3}$$

El *"factor de desbalance"* $\min_i s_i / \max_i s_i$ **puede llegar en nuestro a** 0.7 **con lo cual el speedup teórico puede llegar a bajar de** 11 **a** 7.7**, que casi es igual a tomar solamente los 7 procesadores más rápidos sin incluir los más lentos. Esto sin tener en cuenta el deterioro en el speedup por los tiempos de comunicación.**


t
t1=t2=t3=Tn
t4
proc 1
s1=40Mflops
proc 2
s1=40Mflops
proc 3
s1=40Mflops
proc 4
s1=20Mflops
trabajando
idle
Sin balance de carga

Balance de carga

- **Si distribuimos la carga en forma proporcional a la velocidad de procesamiento**

$$W_i = W \frac{s_i}{\sum_j s_j}, \quad \sum_j W_j = W \tag{4}$$

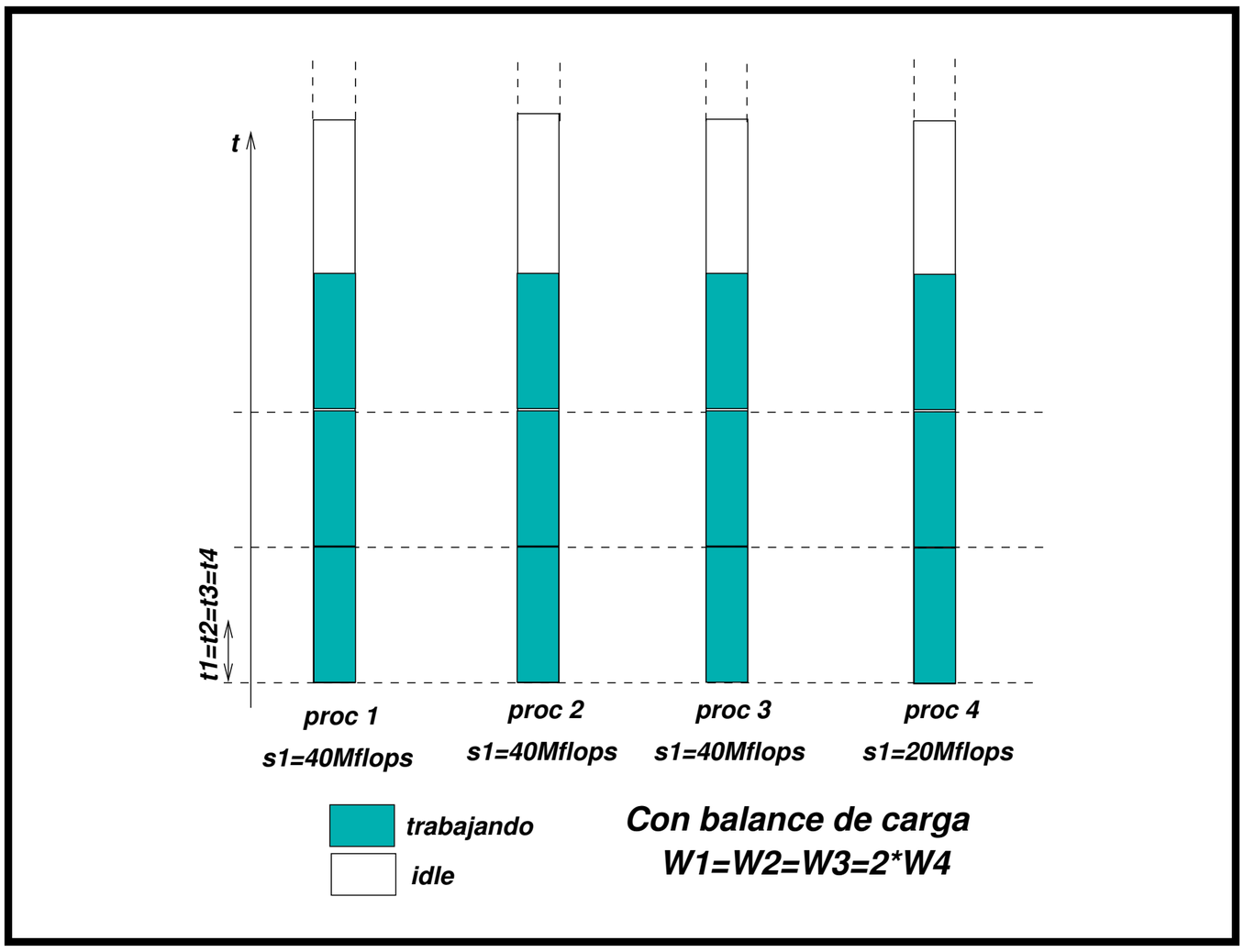
- **El tiempo necesario en cada procesador es**

$$t_i = \frac{W_i}{s_i} = \frac{W}{\sum_j s_j} \quad \textbf{(independiente de } i\textbf{!!!!!!)} \tag{5}$$

- **El speedup ahora es**

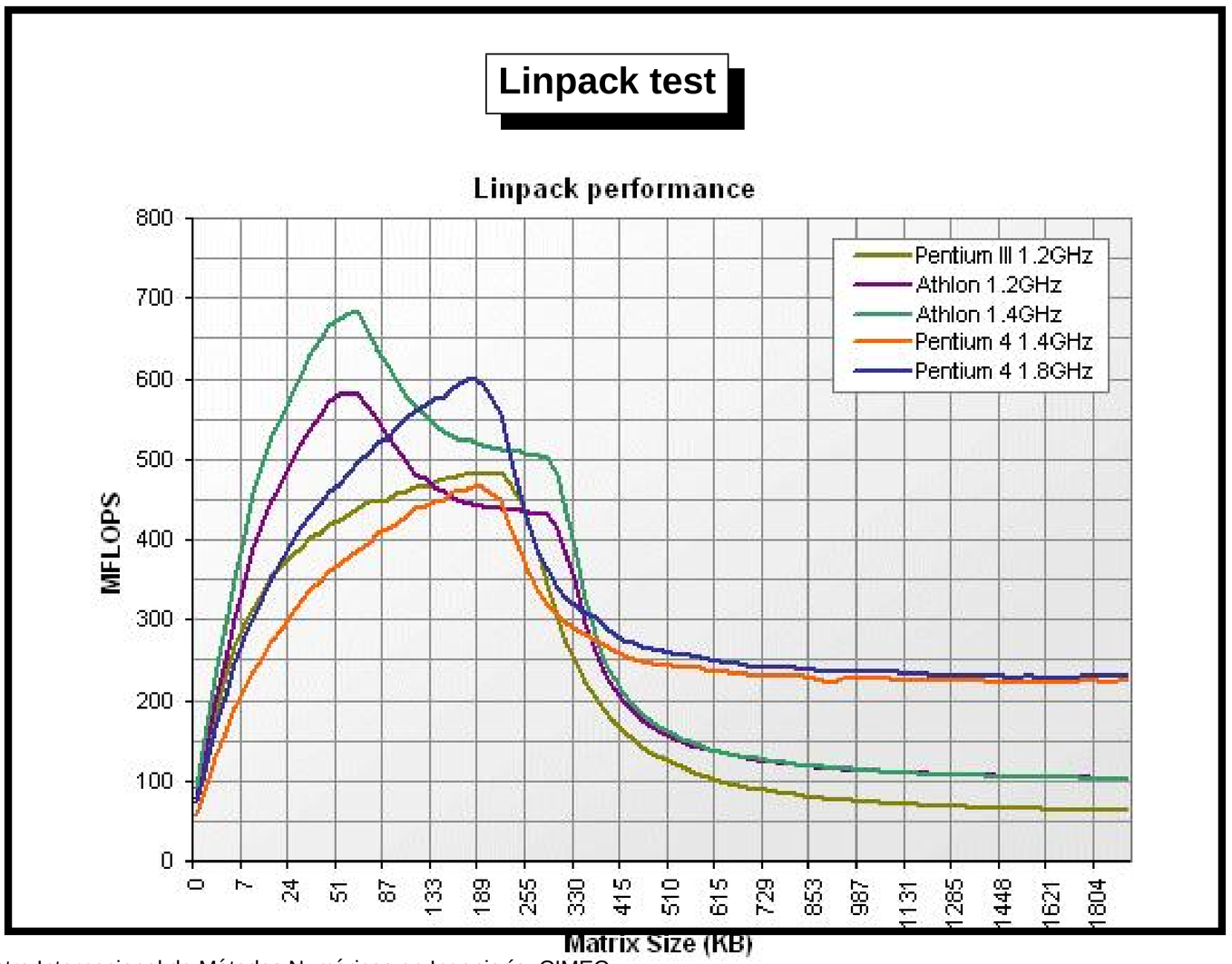
$$S_n = \frac{T_1}{T_n} = (W/\max_j s_j)/\left(\frac{W}{\sum_j s_j}\right) = \frac{\sum_j s_j}{\max_j s_j} \tag{6}$$

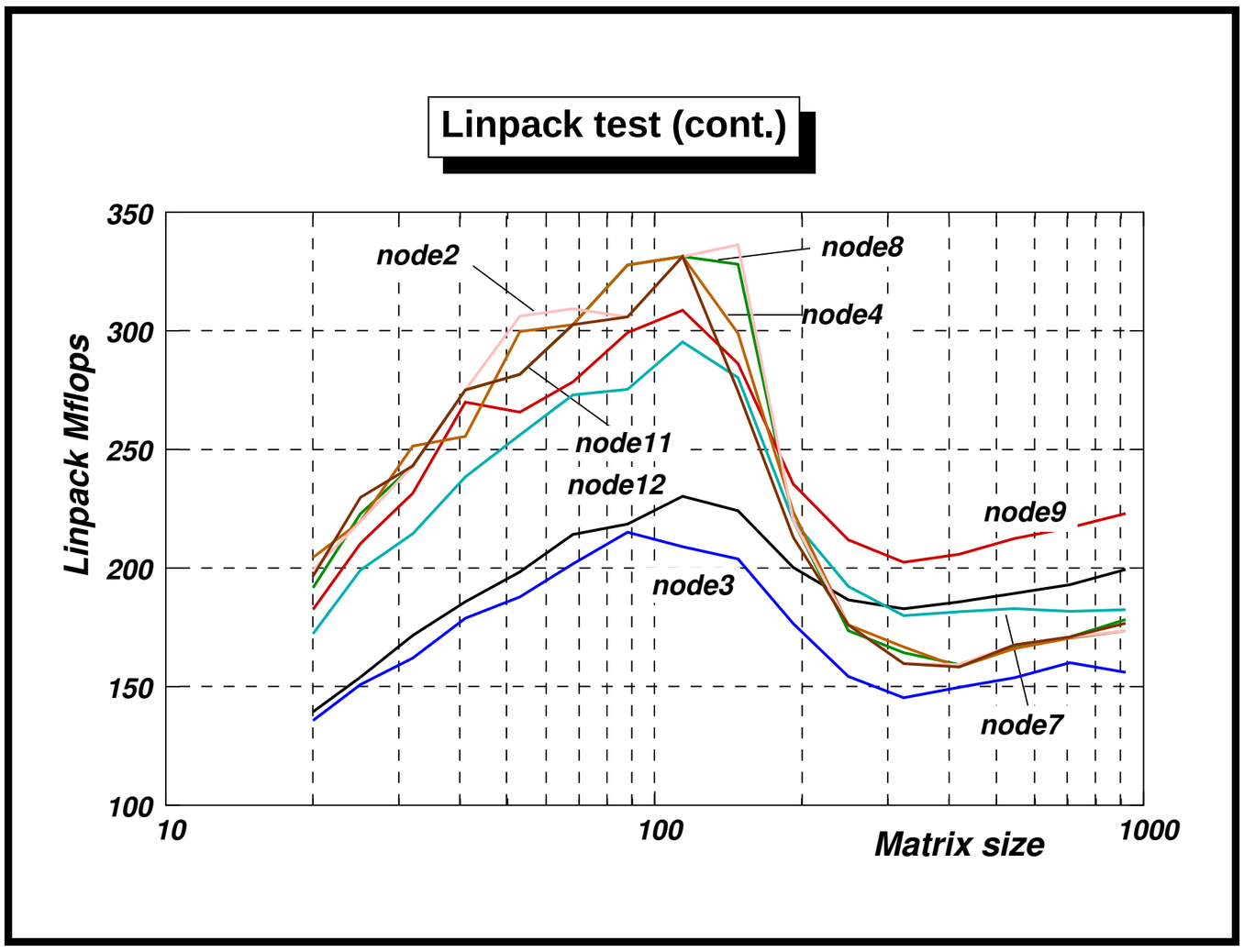
Este es el máximo speedup teórico alcanzable en clusters heterogéneos.

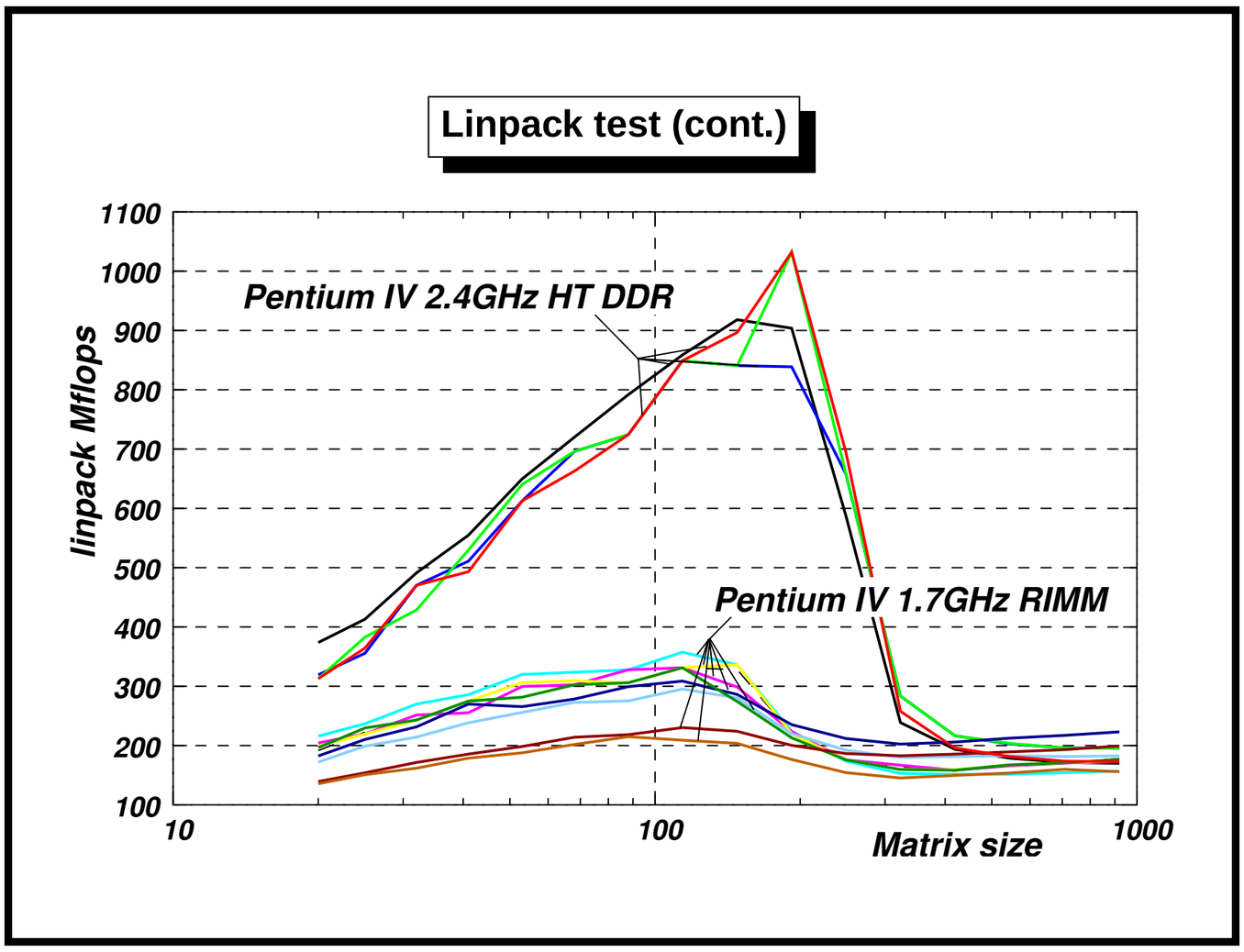

t
t1=t2=t3=t4
proc 1
s1=40Mflops
proc 2
s1=40Mflops
proc 3
s1=40Mflops
proc 4
s1=20Mflops
trabajando
idle
Con balance de carga
W1=W2=W3=2*W4

Balance de carga estático en PETSc-FEM

- **Actualmente PETSc-FEM permite balancear en forma estática la carga, esto es, una vez determinada la velocidad de los procesadores estos son leidos antes de comenzar la ejecución y la partición de la malla se realiza de manera de mantener en cada procesador la misma cantidad de “carga” (número de elementos).**
- **Desventajas del balance estático: Como tomar la velocidad del procesador?**
- **El procesador parece tener diferentes velocidades dependiendo del compromiso entre número de operaciones realizadas y cantidad de información que tiene que fluir desde la memoria al procesador.**
- **A veces el trabajo a realizar en el procesador no es el mismo para todos los elementos.**


Linpack test
Linpack performance
MFLOPS
800
700
600
500
400
300
200
100
0
0
7
24
51
67
133
189
255
330
415
510
615
729
853
987
1131
1285
1448
1621
1804
Matrix Size (KB)
Pentium III 1.2GHz
Athlon 1.2GHz
Athlon 1.4GHz
Pentium 4 1.4GHz
Pentium 4 1.8GHz


Linpack test (cont.)
Linpack Mflops
350
300
250
200
150
100
10
100
1000
Matrix size
node2
node8
node4
node11
node12
node9
node3
node7


Linpack test (cont.)
linpack Mflops
1100
1000
900
800
700
600
500
400
300
200
100
10
100
1000
Matrix size
Pentium IV 2.4GHz HT DDR
Pentium IV 1.7GHz RIMM

Proyectos

- **Balance dinámico de carga, a través de un algoritmo de tipo *“self-scheduling”*.**
- **Agregar un lenguaje de scripting (Guile? Perl? Python?)**
- ***“Functional programming”* con Scheme u otro?**
- **Mejorar la OOP y el soporte a multifísica.**

Relacionados con la física:

- **Agregar ALE.**
- **Refinamiento adaptivo.**