Mecánica Computacional Vol. XXIV
A. Larreteguy (Editor)
Buenos Aires, Argentina, Noviembre 2005

COMPUTACION DE ALTO RENDIMIENTO BAJO WINDOWS SERVER 2003 CON UN DESARROLLO DE SOFTWARE ABIERTO Y UNA PRUEBA DE CLUSTER VIRTUAL USANDO VIRTUAL PC 2004.

Juan Pablo Suárez, Alejandro Soba y Guillermo Marshall

Laboratorio de Sistemas Complejos, Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires,
Argentina.
jsuarez@dc.uba.ar , soba@cnea.gov.ar
marshallg@mail.retina.ar

Key words: Cluster Educativo, Entorno WINDOWS XP - WINDOWS SERVER 2003, HPC, Mpich-NT.1.2.5.

Con el objetivo final de obtener computación de alto rendimiento se propone ampliar la capacidad operativa de una red de cómputos existente mediante su transformación en un cluster windows de 70 nodos. En este contexto, se presentan resultados preliminares de un proyecto de diseño, construcción y operación optimizada de un cluster estructurado sobre una red de 70 PC's bajo el sistema operativo Windows XP (nodos trabajadores) y Windows Server (servidor de dominio y servidor de archivos). Los primeros ensayos se realizaron sobre dos Workstations HP con un cluster virtual bajo Virtual PC®. Posteriormente, se extendió al sistema a los 70 nodos de la red existente. La capacidad de cómputo y la performance obtenidas en situaciones de poco uso del sistema permiten inferir la conveniencia de construir un cluster dedicado bajo el sistema operativo Windows.

Here we report the construction of a Windows HPC based prototype system, its evaluation and performance. The HPC system consists of a cluster under Windows Server 2003 using Virtual Pc 2004 and Open Source MPI for Windows NT MPICHNT1.2.5. The construction is based on the document "Building a Windows High Performance Computer cluster with Virtual PC" CTC, Cornell University, October 2004. This preliminary experience has been extended to the students Computer Science Department Windows PC network, approximately 70 PC's. This Windows cluster has been used to study performance and scalability, although it is clear that this is not a dedicated cluster (resources are shared with normal students computer work).

1 INTRODUCCION

En los últimos años el cómputo en sistemas distribuidos utilizado en Computación de Alta

Performance (HPC) ha permitido, en ámbitos académicos, la resolución de numerosos problemas que previamente resultaban inabordables ya sea por la excesiva cantidad de memoria que requerían o por que el tiempo de cómputo necesario para su resolución tornaba el costo de la simulación inadmisible. Sin embargo el acceso a esos sistemas en general estuvo restringido siempre a un sistema operativo como Unix o a plataformas especializadas. Del mismo modo el software relacionado estaba construido para ser utilizado en dichos entornos y a menudo limitado al entendimiento y uso de expertos. Este esquema de acercamiento por parte del usuario a los recursos de HPC no se modifica en el caso de tratarse de lo que comúnmente se denomina C*luster Computing*, que consiste, en el uso de una red de PCs o estaciones de trabajo que proveen un sistema relativamente económico de cómputo pero con capacidades similares a las que en el pasado estaban reservadas a las supercomputadoras.

Un sistema de esta naturaleza que se ha generalizado en el ambiente académico es el cluster Beowulf: un sistema de computación paralelo dedicado que consiste en un conjunto de PC's y una red de interconexión standard de modo tal que los nodos se emplean en una configuración stand-alone. Los clusters Beowulf son clusters con performance escalable, lo que significa que el diseñador puede aumentar proporcionalmente la performance con la incorporación de máquinas.

Si bien es cierto que la versión más generalizada del cluster Beowulf utiliza el sistema operativo Linux, han surgido desarrollos recientes de software que funcionan bajo el sistema operativo Windows dedicadas a la computación distribuida y al HPC por lo que hoy en día es posible construir clusters operando bajo Windows. También es posible construir sobre equipos de alto rendimiento utilizar virtualización, un concepto altamente eficiente para el desarrollo y testeo de prototipos de sistemas distribuidos o para ejecutar varios sistemas operativos, simultáneamente en una máquina física.

Además de sistemas dedicados al HPC o cluster de cómputo, en los diferentes departamentos de ciencia de las universidades, existen enormes salas de PC's dedicadas a estudiantes y docentes que operan sobre entornos gráficos, en general versiones de Windows o Linux. Para aprovechar esa enorme capacidad de cómputo subutilizada en gran parte de su tiempo, se han generado proyectos interesantes para el buen uso de esos "ciclos ociosos". Muchas de ellas requieren de un 'rebuteo' del sistema que convierte a una red de PC's que durante el día funcionan como máquinas de uso académico en un cluster dedicado por la noche. Otros proyectos proponen la instalación de un software que convierte a una PC en parte de un sistema de nodos interconectados para trabajar en paralelo. El proyecto Condor es uno de los más conocidos, aunque su versión mas completa corre para máquinas bajo Unix.

Este trabajo propone la ampliación de la utilización de la capacidad de cómputo existente, esto es, la conversión de una red de computadoras en una red de cómputo distribuido utilizando las PC's de los laboratorios del departamento de computación de la FCEyN de la UBA, bajo el SO Windows XP. El cluster así construido fue denominado Educativo por las características académicas del mismo: utilización no exclusiva por usuario y baja velocidad de interconexión. Para llevar a cabo esta conversión solo es necesario instalar sobre el software incluido en cada PC, las librerías de MPI (open source) y una herramienta

desarrollada para este propósito encargada del uso y administración del cluster. Esta última se compone de dos módulos principales que reparten la tarea de administración del servicio y manejo del cluster por parte del cliente.

En la sección 2 de este trabajo desarrollamos el concepto HPC y sus derivados en un entorno Windows. La sección 3 propone una revisión del software Virtual PC utilizado en el armado y puesta a punto de una cluster virtual sobre dos Workstation HP. La sección 4 es una visión de las implementaciones de librerías para cálculo paralelo como Mpich-NT y la naturaleza de la ventaja de trabajar con esta librería. En la sección 5 se presenta la herramienta *ClusterAdmin* y algunas de las ventajas que provienen de su utilización en un entorno académico. La última sección contiene algunos resultados de performance realizados sobre el cluster formado por los 70 nodos del departamento.

2 HPC BAJO WINDOWS

La computación de alta performance (HPC) se utiliza para resolver problemas computacionales que requieran de una cantidad de procesamiento significativo, o de acceso rápido a una gran cantidad de datos. Una reducción del tiempo necesario para la ejecución de una aplicación dada o el manejo de datos intensivo es el objetivo principal del HPC en muchas áreas de la ciencia.

Tradicionalmente HPC fue dominio de supercomputadoras con un conjunto muy costoso de soluciones propietarias, difíciles de instalar, mantener y utilizar. La mayoría de las aplicaciones que corren sobre esas arquitecturas fueron diseñadas por los propios investigadores del área que además de ser expertos en sus temas de investigación debían tener un conocimiento avanzado en ciencia computacional. El uso de plataformas mas amigables como las aportadas por Linux y más recientemente por Windows son una alternativa viable para expandir la HPC a un conjunto más amplio de usuarios. Al mismo tiempo los requerimientos para administrar una red en Windows necesaria para un entorno HPC son cada vez más simples, a lo que debe sumarse la plataforma gráfica con la posibilidad de utilizar nuevas tecnologías como .NET.

Dichas herramientas actualmente están extendiendo su alcance para permitir el trabajo en computación paralela, que hoy en día es la forma mas común de acercarse a HPC. Este paradigma puede llevarse a cabo con un multiprocesador en una computadora simple o por el uso de una serie de procesadores distribuidos en red (cluster).

Aplicaciones en paralelo pueden clasificarse como débilmente acopladas o fuertemente acopladas, dependiendo de la frecuencia de comunicación entre los diferentes nodos. Las fuertemente acopladas pueden correr en sistemas scale-up de multiproceso simétrico (SMP) o en sistemas scale-out de proceso masivamente paralelo (MPP). Las débilmente acopladas pueden correr sobre clusters scale-out con red estándar Ethernet (redes de estaciones de trabajo (NOW)), sistemas con balance de carga basados en Web service (WS-LB), sistemas de ciclos compartidos, o cómputo en GRID.

Scale-up es un término que se utiliza para describir la escalabilidad de un sistema. Cuando se incrementa la capacidad de cómputo, el sistema permite sumar más procesadores o más memoria incrementando el poder de resolución. La arquitectura SMP es la más utilizada para

escalado en HPC. En estos sistemas cada proceso tiene su propio cache dedicado y comparte una memoria principal común. Esta memoria permite una alta velocidad de comunicación entre los procesos. Algunos de los procesadores que se utilizan en arquitecturas SMP son computadoras de instrucciones reducidas (RISC) como DEC Alpha, Sun SPARC y IBM RS6000.

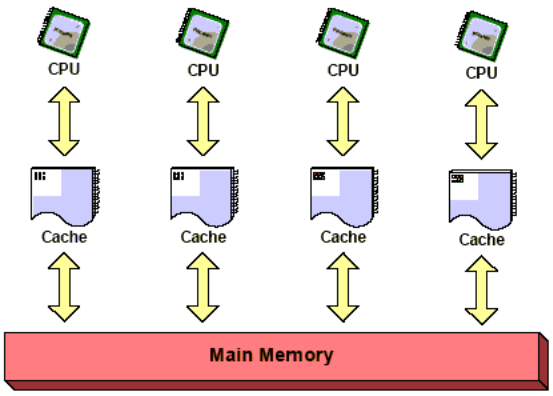

Figura 1. Muestra un arquitectura típica SMP.

Scale-out es el término utilizado para describir la idea de que una aplicación puede ser escalada efectuando particiones y distribuyendo los mismos en distintos procesadores que cooperan en un sistema completo. Si aumenta el tamaño del problema, se pueden agregar más computadoras para proveer mayor capacidad de cómputo. Los sistemas de multiproceso paralelo masivo (MPP) son los más populares en sistemas scale-out. Cada nodo posee sus propios recursos y le permite al usuario un uso efectivo del poder del cómputo distribuido.

Las aplicaciones en paralelo que corren en nodos diferentes sobre un sistema típico MPP poseen un alto grado de interacción entre nodos. Es fundamental entonces, la conexión a través de una red de alta velocidad para que el sistema sea operativo, con tiempos de latencia reducidos y un ancho de banda suficientemente alto para desarrollar buenas performances. Hoy en día se pueden construir sistemas MPP con servidores estándar utilizando servicios de red comunes. Algunos ejemplos de estos sistemas pueden ser los sistemas IBM SP2, los SGI Origin 2000, y los cluster Beowulf .

Los sistemas débilmente acoplados requieren una comunicación entre procesos mínima. Un modelo típico de esas aplicaciones requieren un master o controlador que mantiene las tareas a hacer y uno o mas nodos trabajadores que ejecutan las tareas en la red. Es tarea del master también recibir las repuesta de trabajos realizados por los diferentes nodos y armar la respuesta global. Este modelo es llamado en general inherentemente paralelo y no requiere de altas velocidad de comunicación de red.

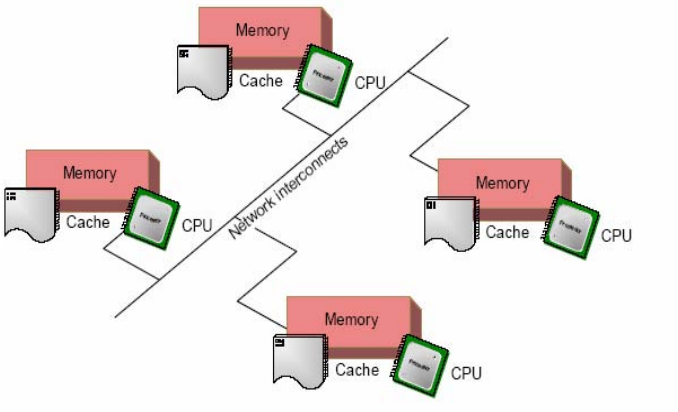

Figure 2. Arquitectura MPP.

Los sistemas NOW son similares a los sistemas MPP, la única diferencia es que se utiliza interconexiones en ethernet en vez de una red de alta velocidad. Con requerimientos de tráfico de redes bajos es posible construir un HPC cluster usando placas de bajo costo. Por otro lado existen placas de red de alta velocidad del orden del gigabits que incrementaran la performance del cluster. Esta alternativa, además de poseer una relación costo/performance óptima, facilita el uso de estaciones de trabajo de propósitos generales como herramientas HPC.

Por último hallamos las aplicaciones que pueden compartir ciclos de cómputo de máquinas multipropósito como las PC's de una departamento de cualquier facultad, pudiéndose obtener esa suma de ciclos inutilizados con propósitos de HPC. A pesar de que estos sistemas son muy atractivos en concepto, poseen muchos desafíos en la implementación. Cada estación de trabajo individual debe poseer la capacidad de no bajar la performance que posee como consola individual de trabajo para el usuario y al mismo tiempo proveer una performance aceptable para las actividades de HPC. La disponibilidad de estos sistemas es también muy impredecible, ya que por ejemplo puede ocurrir que el usuario de la consola decida reiniciar la máquina o apagarla y así cortar la red de comunicación. Esto seguramente afectará cualquier aplicación remota que se esté corriendo y los trabajos deben tener la capacidad de ser reprogramados. Otra problema es la seguridad ya que deben reprogramarse controles de firewalls necesarios para mantener una red segura.

Sistemas de Imagen Simple (Single System Image. SSI)

Los sistemas SSI posibilitan el acceso a un sistema distribuido pero no pierden desde el punto de vista del usuario, la imagen de un recurso de cómputo simple unificado. De este modo el recurso es mas dúctil en el uso para el usuario y esconde la complejidad del sistema distribuido. SSI puede implementarse en diversos niveles de abstracción dentro de la arquitectura del cluster. Puede hacerse a nivel de hardware, sistemas operativo, middleware o aplicaciones.

El mayor desafío para un SSI consiste en obtener una completa transparencia del manejo de los recursos, una escalabilidad adecuada y la posibilidad de soportar las aplicaciones de los usuarios. Si bien existen SO como MOSIX, Solaris o UnixWare que proveen SSI, el modo mas común de implementarlos es a través de Middleware: utilizando un entorno de programación para el desarrollo y ejecución de aplicaciones, como PVM, o instalando un manejador de recursos (RMS) como Condor, LoadLeveler, Load Share facility (LSF), OpenPBS, Sun Grid Engine (SGE), LIBRA. Por último puede generarse una aplicación (de mayor orden de abstracción) que provea una interfaz especifica al usuario. Ejemplos de esto último son: PARMON, Linux Virtual Server, o Problem Solving Enviroments. Es dentro de este último grupo en donde ingresa la ClusterAdmin 1.0.

A nivel sistema operativo, un SSI provee, en cada nodo el soporte fundamental del sistema para la operación combinada del cluster. Servicios como protección, coordinación de procesos/threads, comunicación inter-nodos y manejadores de recursos para permitir que el usuario corra sus aplicaciones de mayor nivel. Los más comunes son los mencionados LINUX y MOSIX en software libre, o los comerciales como IBM's AIX, Windows NT y Windows Server Family.

La implementación de SSI mediante un manejo de recursos (RMS middleware) permite al usuario ejecutar trabajos sin la necesidad de entender la complejidad que se encuentra debajo de la arquitectura. En RMS manipula el cluster a través de cuatro aplicaciones importantes: 1) un manejador de recursos, 2) un encolador de trabajos (Job queuing), 3) una agenda de trabajos (jobs scheduling), 4) un manejador de trabajos (Job management). Un RMS manipula los recursos como procesadores, capacidad de disco en el cluster, mantiene la información de dichos recursos para conocer la cantidades disponibles y asigna los trabajos a cada máquina utilizando colas de trabajos para distribuirlos en cuanto se liberen recursos del cluster.

En la Figura 3 se presenta un esquema de dicho funcionamiento en ClusterAdmin. El manejo de los recursos es realizado por el RMS del sistema operativo, mientras que Cluster Admin se encarga de encolar y organizar los trabajos de los usuarios.

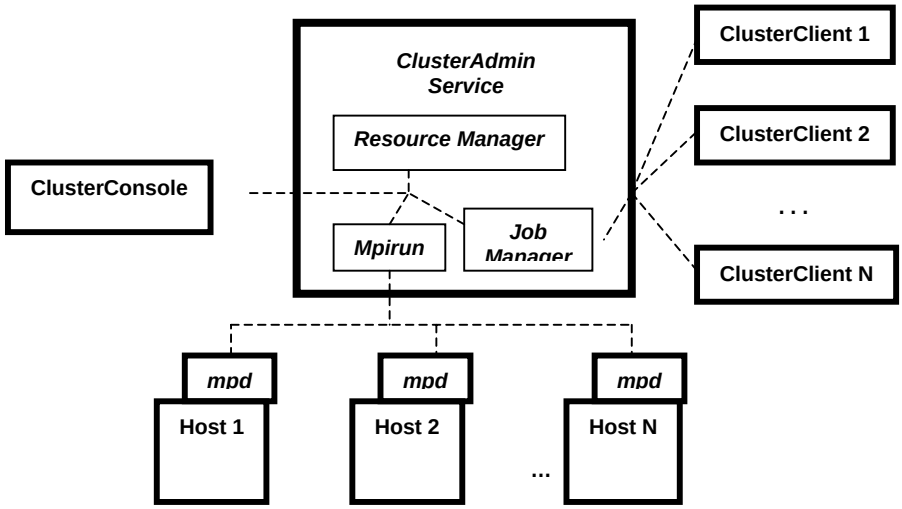

Figura 3. Manejo de Recursos mediante ClusterAdmin 1.0 .

3 VIRTUAL PC

Un sistema de máquinas virtuales (MV) provee un entorno completo en donde multiples sistemas operativos y procesos pueden coexistir. Utilizando MV, un simple host puede soportar múltiples sistemas operativos simultáneamene o una serie de nodos independientes que funcionen como un cluster virtual. En este contexto el sistema es particionado en un pequeño multiprocesador distribuyendo los recursos originales y obteniendo una partición física y un aislamiento tal, que un sistema virtual no es afectado por el otro.

La virtualización se constituye en un modo económico, cómodo y seguro de testear sistemas operativos, software de redes, de clusters y análisis de performace relativa para después migrar a un sistema real.

Sobre la base de dos máquinas físicas: 2 Workstation HP xw4200 con 1 GB de Memoria RAM, se crearon 8 nodos virtuales, distribuidos cuatro en cada máquina física. Cada nodo virtual simula una PC Windows Server 2003 con 128 Mega de Memoria RAM y comparte el uso del procesador de la workstation con otros tres nodos.

Uno de los nodos virtuales del cluster cumple la función de Domain Controler, el cual se encarga de proveer el servicio de **DNS**, **DHCP** y poseer un registro denominado **Active Directory** que cumple la función de registrar usuarios y los nodos del cluster. Otro de los nodos funciona exclusivamente como **File Server** que provee la lectura y la escritura de archivos en disco. Desde este nodo se ejecuta el proceso que correrá en paralelo. Los seis nodos restantes son nodos de cómputo que acceden a los recursos del cluster a través de la red virtual establecida por Virtual PC.

Las dos máquinas físicas se interconectan con un cable cruzado permitiéndonos una velocidad de conexión de 1 Gigabit .

Instalación del Controlador de Dominio, Servidor de archivos y Nodos de Cómputo.

El Domain Controller requiere una dirección IP estática con lo que se permitiría mediante los servicios de DNS y DHCP resolver nombres y asignación de direcciones IP dinámicas a los nodos del cluster. Posteriormente en el Active Directory se da de alta al usuario *clusteruser* con los permisos para utilizar el cluster. Cada usuario tendrá su cuenta y un espacio de almacenamiento propio para poder ejecutar sus programas lo cual permite disponer de un esquema de seguridad adecuado.

La Arquitectura del Cluster requiere un nodo dedicado a funcionar como Servidor de Archivos que no tiene carga de procesamiento y se dedica a leer y/o escribir archivos al disco. En la Figura 4 se muestra una imagen de esta funcionalidad operando en uno de los nodos virtuales del cluster.

Una vez que el Domain Controller y el File Server fueron creados, se procedió a la creación de los 6 nodos virtuales restantes sobre los que se ejecutará el servicio Mpich NT. Cada nodo se registra en el dominio en el Active Directory, desde el cual se podrán configurar. Los nodos virtuales disponibles en el dominio son DCVNodo (Domain Controller), un LFSVNodo (File Server), y seis nodos denominados VCNodo0 - VCNodo5 (Workers) .

Una vez armado el cluster virtual se procedió a instalar Mpich NT 1.2.5. Esta es una distribución libre de MPI que requiere plataforma Windows NT o superior y un compilador Visual C++ 6.0 o superior. El desarrollo Mpich NT posee una aplicación ***MPD*** (*Multi-purpose daemon*) que es la encargada de inicializar los procesos en cada nodo.

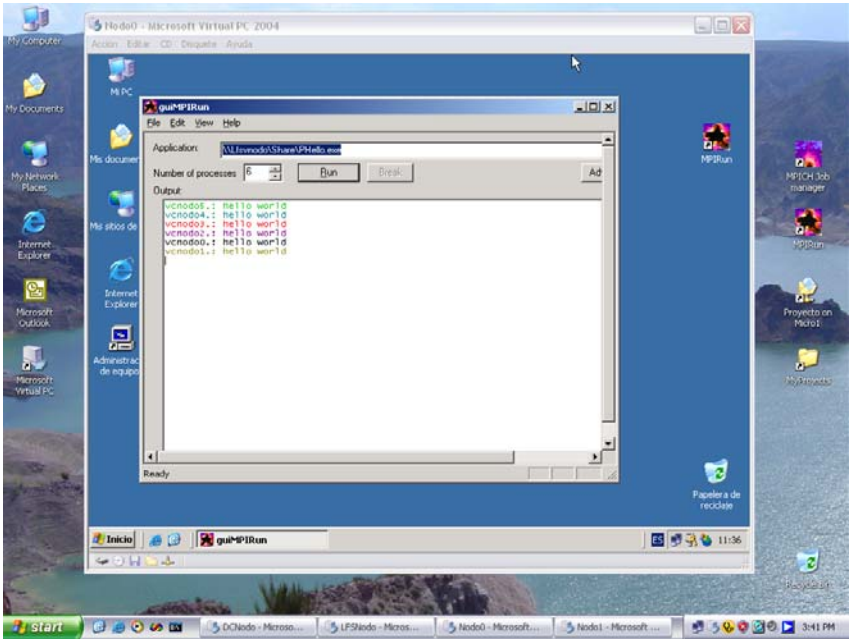

Figura 4. Desde Nodo0 se dispara el pedido de procesamiento en paralelo en todos los nodos virtuales.

4 CLUSTER EDUCATIVO

Tomando como base la experiencia acumulada en el cluster virtual se procedió a implementar un cluster para uso educativo sobre un laboratorio de PC's con sistema operativo Windows XP en el Departamento de Computación de la FCEyN, UBA. En primer lugar se debieron sortear los requerimientos de seguridad del laboratorio y proponer un uso del cluster que no atentara contra el mismo. Uno de los problemas principales a resolver consistió en las prioridades de uso. Utilizando Mpich-NT y MPIRun cada nodo puede funcionar como master. Para el propósito de generar una red que funcione como un cluster real, no es posible permitir ese modo de operación por el riesgo de saturar los procesadores y la interconexión de la red. De modo que se diseñó una herramienta denominada ClusterAdmin la cual proporciona un entorno Cliente/Servidor. El servidor se instala en un nodo que funciona como tal y desde allí distribuye los procesos requeridos desde cada cliente. ClusterAdmin encola los procesos y administra el tiempo dedicado a cada trabajo.

El Servicio ClusterAdmin es básicamente un manejador de recursos a nivel de software que convierte para el usuario, todo el cluster en una simple ventana de comunicación: el *Client-ClusterAdmin,* el cual Ofrece una interfase amigable para encolar procesos de computo paralelo al tiempo que permite ver los procesos encolados. También se le permite al usuario monitorear los nodos disponibles de la red a su disposición para realizar el calculo.

Este esquema permite que cada *proceso mpi* que se ejecute en paralelo sea autenticado con un único usuario clusteruser del Domain Controller en cada nodo.

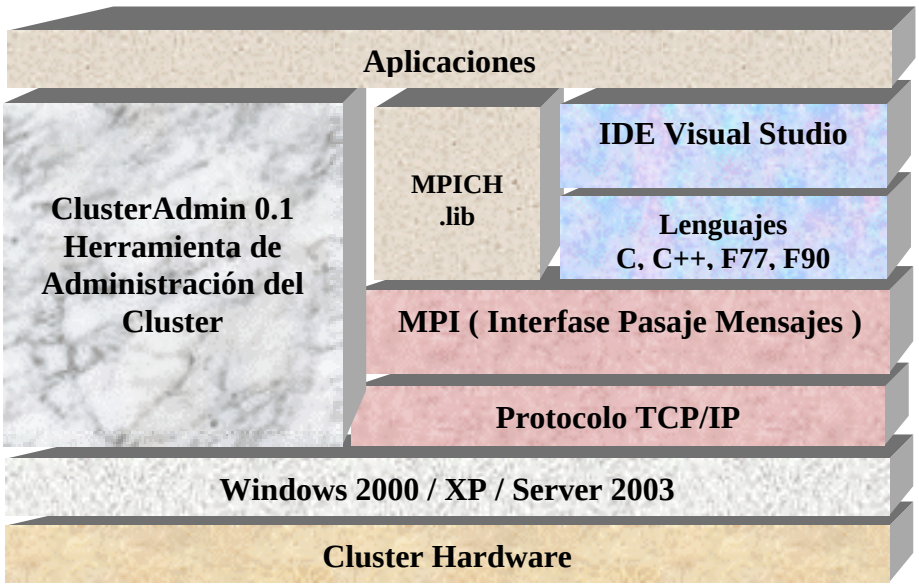

Figura 5. Arquitectura del Cluster en SO windows y el modo de funcionamiento de ClusterAdmin.

El *Servicio ClusterAdmin e*jecuta los procesos en paralelo encolados por los usuarios del cluster. Se encarga de autentificar los procesos con un único usuario, simplificando la

asignación de privilegios de usuario del Dominio. También monitorea los nodos disponibles de la red y permite administrar los usuarios que en cada momento se encuentran operando sobre el cluster. Otra tarea del administrador es crear una carpeta para cada usaurio en un espacio compartido para ser utilizada por los servicios de *mpd* y *mpirun*, necesarios para el trabajo en paralelo. Por último, *ClusterAdmin* mantiene un log diario de los programas ejecutados.

La *Consola de Administración* administra la configuración del servicio. Tiene acceso a los logs de los programas ejecutados por los usuarios y manipular los trabajos encolados en el cluster por todos los usuarios conectados. A través de ella es posible encender, detener y reiniciar el servicio. En la Figura 6 se observa una imagen de la consola ClusterAdmin.

El *ClusterAdmin 0.1* fue desarrollado Visual C++ 7 .Net y se adaptará para utilizar la nueva versión de Mpich2 .Net del laboratorio Argonne National Laboratory.

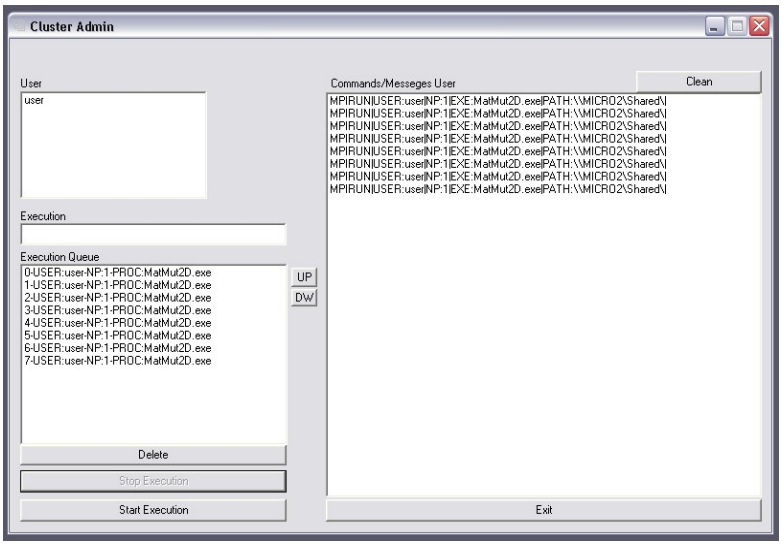

Figura 6. Cluster Admin. 0.1 .

Al mismo tiempo el usuario posee una consola de trabajo que opera en cada máquina. (Figura 7) Esa es la puerta de entrada al cluster y la única herramienta que debe aprender a utilizar. Después de adutentificar su usuario, debe elegir el comando que va a ejecutar sobre la cantidad de procesos a eleción y determinar la ubicación del ejecutable. La aplicación también presenta una cola de los trabajos en ejecuación en ese momento sobre el cluster.

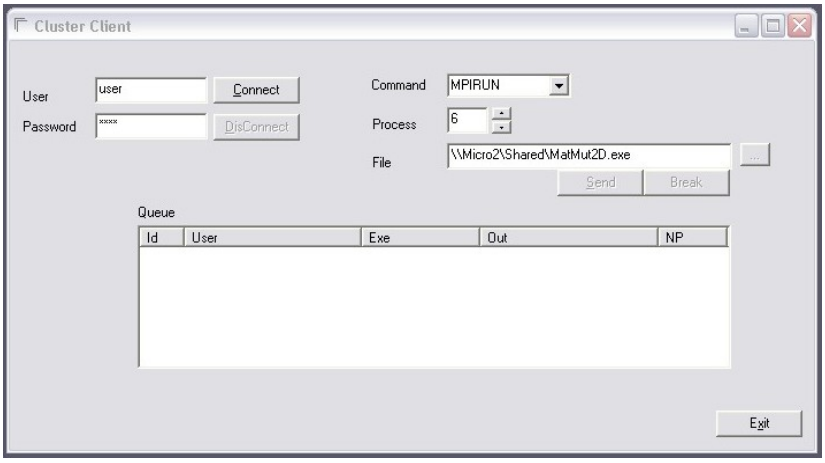

Fig 7. Cluster Cliente. 0.1 .

El trabajo con este entorno presenta varias ventajas: permite ampliar la capacidad de cómputo de una red de computadoras en principio disconexas, sin costo adicional y utilizando recursos existentes y subutilizados. Por otro lado el software ClusterAdmin es una herramienta de acceso al Cluster por los alumnos independiente de la administración del mismo. Al mismo tiempo las máquinas del departamento pueden ser utilizadas libremente por los usuarios y el cluster no requiere de rebuteos de las PC compartiendo los ciclos libres de sus procesadores con el usuario del cluster. Una ventaja adicional es que desde cualquier sala de computadoras se tiene acceso efectivo a los procesadores distribuidos por todo el departamento. Una posible extensión de esta red a todas las máquinas existentes en una institución, requiere tan solo el acuerdo con los usuarios particulares respectivos pudiendo asi potencialmente unificar la capacidad de cómputo de toda una facultad.

Las desvantajas de este cluster con respecto a uno dedicado se dan en dos niveles fundamentales. En primer lugar y el mas importante en referencia con la performance, por la baja velocidad de comunicación que poseen las redes de PC’s del departamento. La mayoría poseen placas de red de 100 Mbits que están muy por debajo de los usuales modos de comunicación de un cluster dedicado. Por otro lado los laboratorios cuentan con un firewall en cada máquina, y un firewall interlaboratorios (cuyos puertos debieron habilitarse para permitir la comunciación) y requiere de la colaboración fuerte de los administradores de red para la implementación. Por otro lado al no ser un cluster dedicado y ser compartido por los alumnos, es impredecible el momento en que un nodo deje de funcionar lo que vuelve inestable el cálculo.

5 PERFORMANCE

Para analizar la performance del Cluster Educativo se corrieron un conjunto de programas especialmente diseñados para analizar diversos factores que hacen al tiempo de cómputo: cálculo, pasaje de mensajes, e intercomunicación entre nodos.

Los mismos son analizados en el cluster en tiempo compartido con las actividades del laboratorio con el resto del alumnado de la facultad a los que no se les proporcionó información de lo que se estaba realizando en sus consolas individuales.

Los programas evalúan costos de cómputo y comunicación entre nodos para después evaluar el tiempo de procesamiento (T) en función del número de operaciones involucradas (n) y el número de procesos utilizados (p). Para caracterizar la performance de un sistema paralelo se definen dos funciones adicionales.

La primera es Speedup, una relación entre el tiempo serial sobre el tiempo paralelo

$$S(n,p) = \frac{T_\sigma(n)}{T_\pi(n,p)}$$

Para un valor de p filo, se comprueba que $0<S<=p$. Si $S(n,p)=p$ se dice que el sistema presenta speedup lineal. En general esto no se verifica en sistemas reales. Lo que ocurre en la mayoría es 'slowdown', lo que significa que un programa paralelo corre en p procesos mas despacio que en el caso serial.

La segunda es la Eficiencia: una medida de la utilización de los procesos en un programa paralelo relativo a un programa serial.

$$S(n,p) = \frac{T_\sigma(n)}{pT_\pi(n,p)} \qquad 0 \le E(n,p) \le 1$$

Si la eficiencia es 1, el sistema presenta speedup lineal. En general $E(n,p)<=1/p$.

Para el cálculo de estas funciones no evaluamos los tiempos de la IO, que se extraen de la medición dentro del código.

Este análisis se llevo a cabo con un programa que evalua la integral por dos métodos bien diferenciados. El método determinístico de Simpson, que requiere de una división del dominio de integración entre procesos, y el método de estocástico de Montecarlo, que por el contrario, ejecuta una integración en cada nodo sobre el dominio completo.

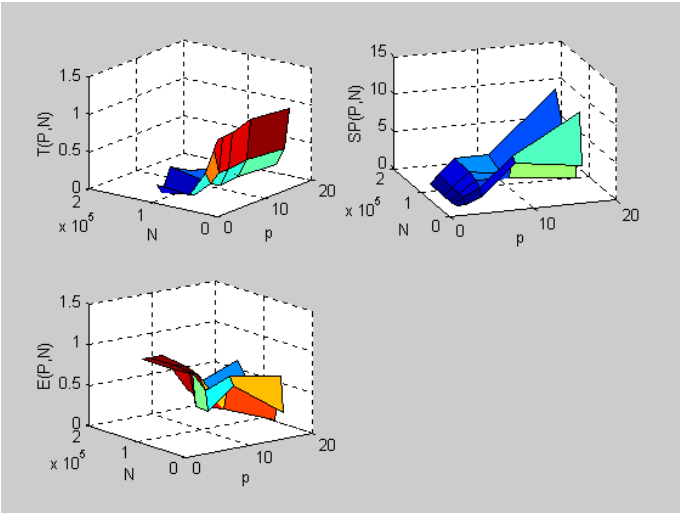

Figura 8. Función Tiempo , Speed Up y Eficiencia del programa de integración por el método de MonteCarlo.

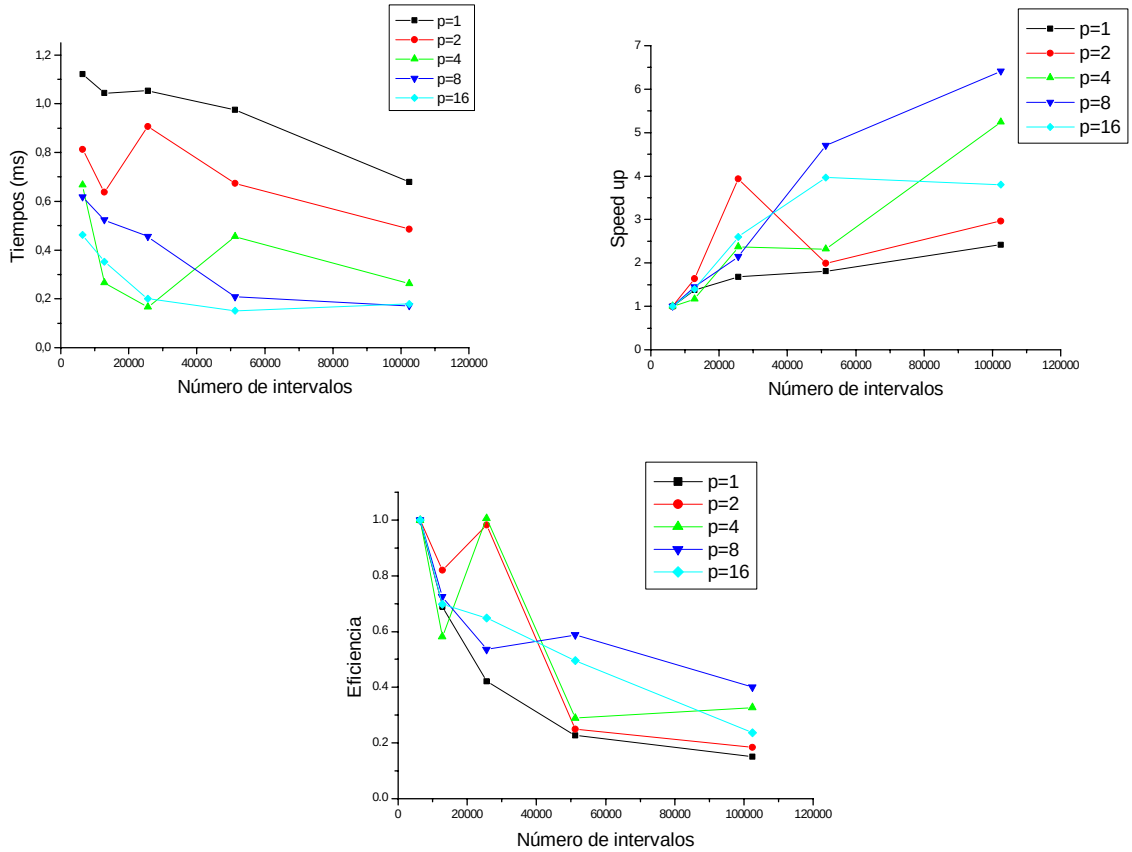

Figura 9. Performance del CE utilizando un programa de integración por el método de Simpson.

Las Figuras 8 y 9 presentan los resultados de las mediciones de tiempos, speedup y eficiencia en el CE. Puede observarse que los resultados en principios si bien presentan ventajas con respecto al cálculo serial, no son óptimos. De hecho la eficiencia alcanza entre un 20 y un 50 por ciento para las configuraciones de nodos analizadas. Pueden compararse esos resultados con los obtenidos mediante los mismos programas medidos en un cluster Linux dedicado en donde la eficiencia se mantiene en el rango [0.8,1] (figura 12).

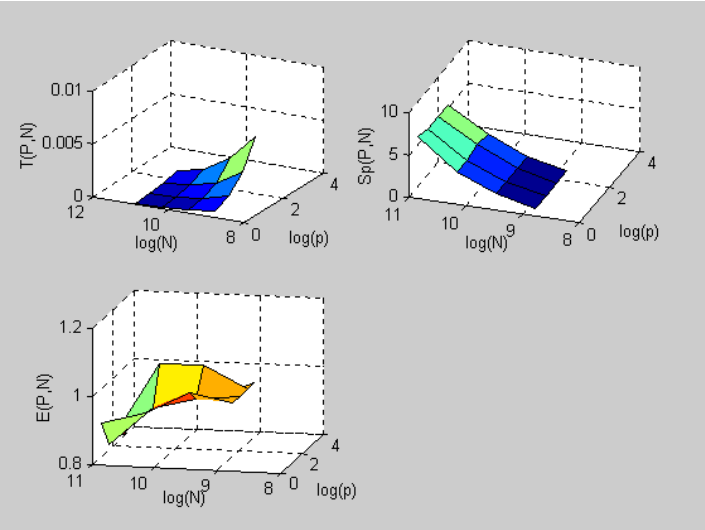

Figura 10. Funcion Tiempo, Speed Up y Eficiencia para un cluter dedidado Linux sobre 8 PC PIV de diversas velocidades de procesador.

Las razones más importantes que explican este comportamiento son: 1) el costo comunicacional críticamente lento. 2) El costo de comunicación interlaboratorios que genera un nuevo cuello de botella comunicacional. 3) El uso de la red compartido. Sobre este punto puede notarse la falta de homogeneidad de las mediciones en cada caso. Mientras en el cluster dedicado las curvas son suaves, en el CE se observan picos asimétricos debido a que en una dado cálculo la red esta más o menos congestionada y las máquinas más o menos utilizadas.

En una segunda instancia se diseño un programa que contempla el pasaje de mensaje entre nodos para estudiar los tiempos de comunicación entre procesos. El programa *comunica* envía una serie creciente de datos a la cantidad de procesos involucrados en el cálculo. El programa mide el tiempo de comunicación utilizando las sentencias Send y Recv clásicas y utilizando Brodcast. El objetivo de estas mediciones es determinar el tiempo de latencia y el bandwith de la comunicaión entre nodos.

Dato una cierta “unidad” de mensaje, el costo de enviar ese mensaje se puede estimar como:

$$tiempo = T_{latencia} + cteT_{bandwidth}$$

En la Figura 11 se grafican los tiempos de comunicación, pudiéndose apreciar la fluctuación de las mediciones que señala la heterogeneidad de la red. La Figura 12 presenta el aumento del tiempo de latencia en función de la cantidad de procesos entre los que se realiza la comunicación. Se grafican ambos modos de comunicaciones con el mensaje de tamaño mínimo. De ambos gráficos se puede concluir que el mayor costo de la comunicación esta dado por el tiempo de latencia.

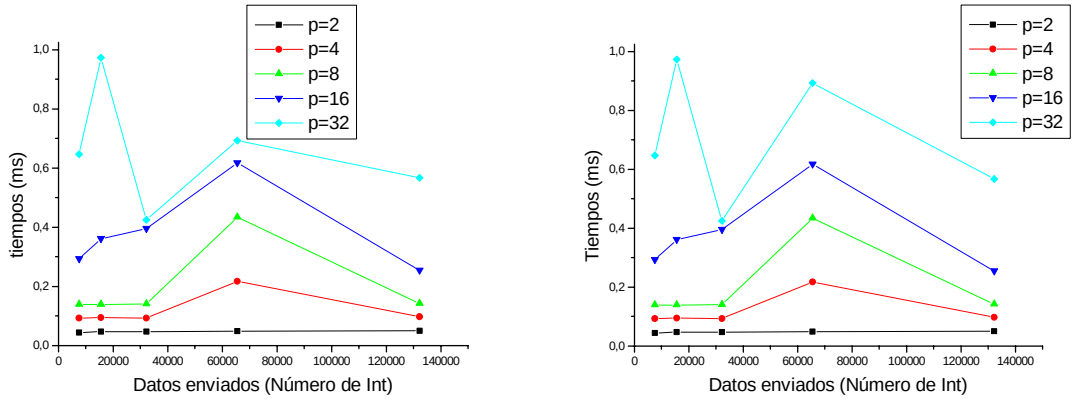

Figura 11: Tiempos de envío utilizando Send-Recv y Broadcast.

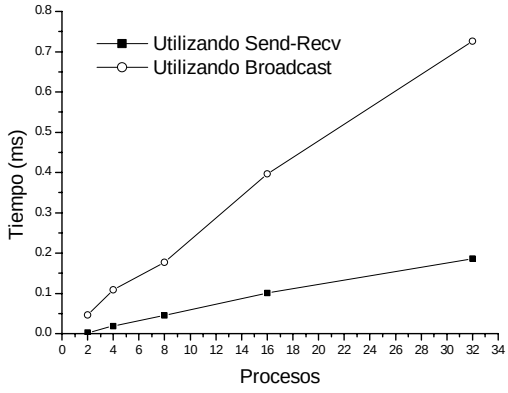

Figura 12: Latencia en función del número de procesos.

6 CONCLUSIONES

La utilización de la herramienta Virtual PC facilita notablemente el desarrollo del cluster Windows, debido a que permite emular el mismo, tanto en lo que hace a su construcción como a su operación, sobre una única máquina física. Esta característica, a su vez, facilita la detección de errores y/o posibles fallas en la configuración en vistas a la migración hacia un sistema real sobre una red extensa. Por otro lado se corroboró que un cluster virtual puede operar emulando a un cluster real en el cómputo paralelo con el Mpich-NT utilizado.

La experiencia llevada a cabo sobre el laboratorio al crear un cluster de uso educativo ha dado resultados satisfactorios. No sólo por la ductilidad de la administración sino por la simplicidad que ofrece su uso permitiendo a los alumnos introducirse en el cálculo paralelo de una forma sencilla. Claramente la performance obtenida como era esperable es sumamente baja. Se han detallado en la sección 5 las razones de estos resultados. Naturalmente, el costo de cómputo en los sencillos programas diseñados es mucho menor que el de comunicación inter nodos. La pérdida de eficiencia aumenta considerablemente si el análisis se realiza en horas de trabajo intensivo de los alumnos en el laboratorio. Sin embargo la idea fundamental que debemos rescatar de esta experiencia es que utilizando recursos preexistentes en los laboratorio de la facultad, con un software también preexistente y el agregado mínimo de software open source y de diseño propio, una red de PC de baja comunicación es factible de convertirse en una herramienta de calculo de HPC con costo cero y un trabajo mínimo. También es necesario remarcar que una migración a mejores performances no inplicaría mas que mejorar las comunicaciones de red de las PC o pensar en adquirir en futuras renovaciones del parque de hardware del departamento máquinas que posean los requerimientos necesarios para obtener mejor rendimiento en HPC.

Por último esta experiencia alienta el desarrollo de un cluster dedicado en un sistema operativo Windows debido a la ductilidad que presenta para su utilización, además de la enorme cantidad de software Open Source que se obtienen en referencia al mismo. Una comparación entre sistemas operativos puede resultar interesante, pero mas interesante aún es pensar en unir los esfuerzos de cómputo de cluster dedicados operando bajo diversos sistemas operativos lo cual ingresa dentro del paradigma de cómputo en GRID, en donde los futuros nodos trabajadores conectados ya no hara falta distinguirlos.

7 REFERENCIAS

[1] Building a Windows High-Performance compute cluster with Virtual PC. Cornell Theory Center, Octubre 2004.

[2] http://www-unix.mcs.anl.gov/mpi/mpich/mpich-nt/

[3] Installation and User's Guide to MPICH. A portable Implementation of MPI Version 1.2.6, David Ashton, Williams Gropp, and Ewing Lusk.

[4] Solution Guide for Migrating High Performance Computing (HPC) Applications from UNIX to Windows. Version 1.0. MICROSOFT

[5] Cluster computing: High-Performance,High-Availability, and High-Throughput Processing on a Network of Computers. Chee Shin Yeo, Rajkumar Buyra, Hossein Pourreza, Rasit Eskicioglu, Peter Graham, Frank Sommers. Handbook of Innovative Computing, Albert Zomaya (editor), Springer Verlag, 2005.

[6]Scalable Cluster Computing with MOSIX for LINUX, Ammon Barak, Oren La'adan, Amnon Shiloh. In Proc. 5-th Annual Linux Expo, pages 95--100, May 1999. http://citeseer.ist.psu.edu/barak99scalable.html

[7]The architecture of Virtual Machines. James E Smith, Ravi Nair.Computer. IEEEComputer Society. Pag 32-38. mAy 2005