Clusters de PCs

Gilberto Díaz [a] **Herbert Hoeger** [a, b] **Luis A. Nuñez** [a, c]

gilberto@cecalc.ula.ve hhoeger@ula.ve nunez@ula.ve

Centro Nacional de Cálculo Científico [a]
Universidad de Los Andes (CECALCULA)
Corporación Parque Tecnológico de Mérida
Mérida 5101, VENEZUELA

Centro de Simulación y Modelos [b]
Facultad de Ingeniería
Universidad de Los Andes
Mérida 5101, VENEZUELA

Centro de Astrofísica Teórica [c]
Departamento de Física
Facultad de Ciencias
Universidad de Los Andes
Mérida 5101, VENEZUELA

Julio 2002

RESUMEN

Los Clusters de PCs se hicieron populares con el proyecto *Beowulf*. La idea consiste en armar un máquina, con gran poder de computo, interconectando PCs y usando software gratuito como Linux, MPI o PVM, a un costo considerablemente menor que el de supercomputadores comparables. En este trabajo se presenta los distintos aspectos que se deben tomar en cuenta a la hora de implementar un cluster Linux.

1. INTRODUCCIÓN

Desde la aparición de las computadoras, el hombre constantemente ha mantenido una demanda por mayor poder de computo. No importa que tan potentes puedan ser las máquinas actuales, comparadas con sus predecesoras de solo un par de años atrás, la inquietud, el deseo y la necesidad por resolver problemas de mayor envergadura, con más

precisión, más realísticos y por lo tanto más complejos, conserva abierto este apetito por máquinas más rápidas, más precisas y con mayores capacidades de almacenamiento. Es decir, independientemente de la capacidad de computo que se tenga, siempre habrán aplicaciones que requieren de mayor poder computacional.

El modelado y la simulación numérica de problemas científicos e ingenieriles complejos de dinámica de fluidos, predicciones climáticas, diseño de circuitos electrónicos, reacciones químicas, modelos ambientales y procesos de manufacturación, tradicionalmente han impulsado el avance de los computadores. Hoy en día también esta siendo promovido por aplicaciones comerciales que requieren procesar grandes cantidades de datos. Entre ellas encontramos realidad virtual, vídeo conferencias, bases de datos paralelas, diagnóstico médico asistido por computadoras, procesamiento de imágenes y minería de datos entre otras.

Hay muchas aplicaciones que deben ser resultas en tiempos razonables. En la medida que estas aplicaciones se hacen más complejas, requieren de más tiempo de computo. La predicción climática es una de ellas. Obtener la preedición climática de mañana dentro de cuatro días hace que ésta sea inútil. La escasez de poder de computo por lo general fuerza a la simplificación de los modelos, limitando su provecho, con el fin de producir resultados en tiempos adecuados.

Exploremos un poco más el último ejemplo. La predicción climática numérica modela la atmósfera dividiéndola en regiones tridimensionales o celdas. Las condiciones de cada celda como temperatura, humedad, dirección del viento, presión, etc., son calculadas a intervalos, usando las condiciones existentes en el intervalo previo, mediante la aplicación de ecuaciones matemáticas complejas. Estos cálculos se repiten muchísimas veces para modelar el paso del tiempo. La característica principal de esta aplicación es el número de celdas necesarias. Dado que el estado de una zona es afectado por eventos distantes, se deben considerar áreas de gran tamaño. Si dividimos la atmósfera en celdas de tamaño 1 km. x 1 km. x 1 km. sobre un altura de 20 km. (20 celdas a lo alto), tendremos unas $25x10^8$ celdas. Suponiendo que el calculo de cada celda, para un paso, requiere de 200 flops[1], entonces todas las celdas requieren de $5x10^{11}$ flops. Si la predicción es por 10 días a intervalos de 7 minutos, estamos hablando de unas 10^{15} operaciones. Una máquina que pueda ejecutar 180 Mflops[2] (más o menos equivalente a un Pentium III de 700MHz), necesitaría unos 64 días para obtener la predicción

[1] flops = floting point operations (operaciones de punto flotante).

[2] Mflops = Mega flops por segundo = 1.000.000 de flops por segundo.

suponiendo un flujo continuo de datos hacia al CPU y que no existan retrasos de ningún tipo (accesos a memoria, disco, etc.).

La eficiencia de un computador depende directamente del tiempo requerido para ejecutar una instrucción básica y del número de instrucciones básicas que pueden ser ejecutadas concurrentemente. Esta eficiencia puede ser incrementada por avances en la arquitectura y por avances tecnológicos. Avances en la arquitectura incrementan la cantidad de trabajo que se puede realizar por ciclo de instrucción entre los que encontramos memoria bit-paralela[3], aritmética bit-paralela, memoria cache[4], canales, memoria intercalada, múltiples unidades funcionales, *lookahead*[5] de instrucciones, *pipelining*[6] de instrucciones, unidades funcionales *pipelined* y *pipelining* de datos. Una vez incorporados estos avances, mejorar la eficiencia de un procesador implica reducir el tiempo de los ciclos: avances tecnológicos.

Hace un par de décadas, los avances de arquitectura solo estaban presentes en los supercomputadores: los computadores más poderosos y rápidos en términos de eficiencia de CPU y capacidades de I/O[7]. Dada la evolución de la tecnología, el supercomputador de hoy puede ser el computador personal de mañana. Los supercomputadores comenzaron a aparecer a mediados de 1970 gracias al célebre ingeniero *Seymour Cray*, quien apropiadamente es llamado el padre de la supercomputación. Cray era capaz de diseñar una máquina desde cero si consideraba que había formas de optimizar su rendimiento y a él le debemos muchas de las capacidades presentes en los computadores actuales. Sus máquinas se convirtieron en estándares de la industria y una de sus contribuciones más importantes es el *procesamiento vectorial* en el cual se encadenan largas series de cálculos mediante circuitos especializados. Los sistemas diseñados por

---

[3] n bits (n>1) son procesados simultáneamente en oposición con bit-serial en donde solo un bit es procesado en un momento dado.

[4] La memoria cache es un buffer de alta velocidad que reduce el tiempo efectivo de acceso a un sistema de almacenamiento (memoria, disco, CD, etc.). El cache mantiene copia de algunos bloques de datos: los que tengan más alta probabilidad de ser accesados. Cuando hay una solicitud de un dato que esté presente en el cache, se dice que hay un *hit* y el cache retorna el dato requerido. Si el dato no esta presente en el cache, la solicitud es pasada al sistema de almacenamiento y la obtención del dato se hace más lenta.

[5] Consiste en buscar, decodificar y buscar los operadores de la siguiente instrucción mientras se está ejecutando la instrucción actual.

[6] *Pipelining* se puede ver como la división de una tarea en varias subtareas cada una de las cuales puede ser ejecutada independientemente como en una línea de producción.

[7] I/O = input/output (entrada/salida).

Cray fueron piezas maestras de tecnología y de estética. La elegancia física era tan importante como la eficiencia de la máquina.

En los últimos años la incorporación de avances de arquitectura en los microprocesadores ha sido significativa. Sin embargo, no podemos depender continuamente de procesadores más rápidos para obtener más eficiencia. Hay limites físicos, como la velocidad de la luz, que eventualmente van a desacelerar la reducción que se ha visto año tras año en el tiempo que dura un ciclo (tiempo para ejecutar la operación más básica) de CPU.

Dadas las dificultades en mejorar la eficiencia de un procesador, la convergencia en eficiencia entre microprocesadores y los supercomputadores tradicionales, y el relativo bajo costo de los microprocesadores[8], ha permitido el desarrollo de computadores paralelos viables comercialmente con decenas, cientos y hasta miles de microprosesadores. Un *computador paralelo* es un conjunto de procesadores capaces de cooperar en la solución de un problema. Esta definición incluye supercomputadores con cientos de procesadores, máquinas con múltiples procesadores, redes de estaciones de trabajo (NOWs[9]) y redes de PCs (*clusters* de PCs).

2. ¿POR QUÉ *CLUSTERS* DE PCs?

Los supercomputadores, por su propia definición, son las máquinas más costosas que se puedan encontrar en el mercado debido a que usan la tecnología más avanzada disponible. Para mantener un computador dentro de la definición de supercomputador se requiere de una inversión considerable en investigación y desarrollo, cosa que solo es posible dentro de grandes compañías sólidas. Es precisamente esta alta inversión que hace que los supercomputadores sean onerosos y estén fuera del alcance de la gran mayoría. Proyectos con altos requerimientos de ciclos de CPU, usualmente solicitan tiempo en centros de supercomputación o se corren en máquinas más lentas, lo que lleva a esperar por semanas y hasta meses por los resultados.

Durante los 80, con la aparición de sistemas operativos distribuidos como Chorus y Amoeba, el desarrollo de mecanismos de pase de mensajes, y el fuerte incremento en la capacidad de computo de las estaciones de trabajo, se dan los primeros pasos hacia lo que hoy conocemos como *clusters* de PCs. Dos elementos retenían un desarrollo más

---

[8] Tienen una demanda sustancialmente mayor que la de otros procesadores que permite dividir los costos de diseño, producción y comercialización entre más unidades.

[9] Network of Workstations.

profundo en el área: el costo de los equipos y las licencias de software. El sistema operativo UNIX incorporo muchas de la ideas en desarrollo, pero su licencia era costosa, y las estaciones de trabajo de IBM, Sun, Digital, etc., estaban por las decenas de miles de dólares.

En los 90 se dan ciertos eventos muy favorables. Los PCs comienzan a exhibir la capacidad de las estaciones de trabajo, sus precios se hacen muy asequibles y los costos de los equipos de redes disminuyen significativamente. Por otro lado, el surgimiento de *LINUX*, un sistema operativo gratuito originalmente desarrollado por el Finlandés *Linus Torvalds* y luego mediante la colaboración de un sin número de voluntarios alrededor del mundo, compatible con *UNIX* y capaz de correr sobre PCs, permite finalmente satisfacer las demandas de computación a una fracción del costo asociado a los supercomputadores. En 1994 *Donald Becker* y su equipo en la *NASA*, logran conectar una serie de PCs mediante un software especial, creando un sistema que denominaron *Beowulf*, con una eficiencia comparable a los supercomputadores y que se convirtió en el modelo de los *clusters* de PCs.

Además de que los *clusters* de PCs tienen un rendimiento comparable a los supercomputadores a una fracción del costo de estos, existen otras ventajas:

- **Ensamblaje:** no se requiere tener un doctorado en computación y años de experiencia para ser capaz de construir un cluster. Hoy en día estudiantes de bachillerato son capaces de ensamblar PCs. Las partes se pueden comprar por separado: tarjeta madre, procesador, tarjeta de video, tarjeta de sonido, disco duro, lectrora/escritora de CDs, monitor, teclado, fuente de poder, etc., de acuerdo a los gustos y necesidades, y al acoplarlas tienen un PC hecho a la medida. Con ciertos conocimientos adicionales de redes pueden armar un cluster.
- **Mantenimiento y disponibilidad:** dado que los elementos que forman un cluster se encuentran fácilmente en el mercado (son componentes de producción masiva y por lo tanto de bajo costo), al fallar alguno de ellos se puede reemplazar sin mayores inconvenientes. Es más, los clusters están formados por PCs individuales interconectados por una red y en la gran mayoría de los casos no es necesario poner fuera de servicio todo el cluster para reemplazar un componente, sino solo el nodo (el PC o máquina) al que esta asociado el componente. Por lo general los supercomputadores constan de CPUs interconectados por redes especiales dentro de una misma caja. Reemplazar un componente implica apagar totalmente la máquina. Además hay que esperar que llegue el experto de la compañía con la pieza muy particular y costosa para reparar el supercomputador.

- **Hospedaje:** debido al alto costo de los supercomputadores, estos son albergarlos en centros especiales. Dentro de estos centros, ellos se encuentran en salas muy particulares con sistemas de aire acondicionado, filtrado de aire, ductos de enfriamiento, cableados y sistemas de protección eléctrica especiales, etc. Además de que deben contar con administradores, consultores, personal de mantenimiento, etc., con una preparación importante en el manejo de estas máquinas. Los *clusters* requieren de alojamientos mucho más modestos con el requerimiento principal de poseer un sistema eléctrico adecuado. En muchos casos ni siquiera hace falta poseer aire acondicionado.
- **Modernización y expansión:** por la esencia misma de lo que son los supercomputadores, cuando un centro recibe una de esta máquinas dentro de poco aparecen nuevos modelos. Actualizar los supercomputadores se traduce en comprar los nuevos modelos. Por lo particular que son los supercomputadores, expandir sus capacidades de memoria, almacenamiento en disco, numero de CPUs, etc., se traduce en inversiones sustanciales. Como los *clusters* están compuestos por elementos disponibles de múltiples fabricantes y debido a la compatibilidad que estos tratan de mantener con las diferentes generaciones de una misma familia de componentes, se hace sencillo modernizarlos. Actualizar el cluster con CPUs más potentes puede ser tan sencillo como sacar un CPU de la tarjeta madre e instalar otro, o quizás, reemplazar la tarjeta madre y el CPU conservando el resto de los componentes: memoria, tarjetas de video, etc. Expandir la capacidad de memoria y de almacenamiento en disco no requiere de inversiones sustanciales dado el bajo costo de estos componente. Añadir CPUs implica agregar PCs de fácil adquisición.

A pesar de que los clusters surgen como una alternativa de computación de alto rendimiento a bajo costo, existen otras aplicaciones para las cuales los clusters son convenientes. Entre ellas tenemos:

- **Servidores Web**: Con la explosión mundial de la Internet se hace necesario que sitios populares, como Yahoo y Google, tengan capacidad en exceso a fin de servir las solicitudes de sus clientes. No solo se requieren respuestas en tiempos razonables, también se hace necesario que éstos sitios que estén disponibles constantemente. Sitios como Yahoo dependen de propagandas para subsistir y si presentan fallas frecuentes simplemente sucumbirían. Microsoft e IBM pierden reputación si sus servidores se encuentran caídos. Las empresas que hacen comercio electrónico, por ejemplo Amazon.com, podrían ver su negocio seriamente afectado si no son accesibles. Los clusters ofrecen una solución a éstos dos problemas ya que por un

lado la agregación de máquinas permite hacer una distribución del trabajo y por otro lado la redundancia de elementos de computo ofrece una alta disponibilidad del servicio.

- **Servidores de archivos**: Los clusters también son ideales como servidores de archivos y por lo tanto para aplicaciones de bases de datos. Ellos permiten distribuir tanto las consultas a los datos como los datos mismos entre diferentes procesadores y diferentes unidades de disco respectivamente. Esto permite acelerar considerablemente las respuestas del sistema. Obsérvese que este tipo de aplicaciones esta estrechamente ligada con la anterior.
- **Aplicaciones inherentemente paralelas**: Hay numerosas aplicaciones que se caracterizan por ser intensivas computacionalmente e inherentemente paralelas; el trabajo se puede dividir en subtrabajos que son relativamente independientes uno del otro. Estas subtareas pueden ser un mismo algoritmo ejecutado sobre diferentes porciones de los datos del problema o diferentes cómputos que se pueden efectuar en paralelo. Mencionaremos algunas de ella. El *trazado de rayos* consiste en simular en una imagen las trayectorias de los rayos de luz que emanan de una fuente, haciendo que la imagen sea mucho más realística. *Simuladores de vuelos* usados para entrenamiento de pilotos en los cuales se debe responder de inmediato al ambiente de vuelo y los comandos del piloto. La *minería de datos* que analiza inmensas cantidades de datos con la intención de encontrar patrones o relaciones que son prácticamente imposibles de encontrar manualmente.

Hoy en día un buen porcentaje de las máquinas que aparecen listadas como las más poderosas en el sitio www.top500.com, son clusters tal como se puede apreciar en la figura 1.

3. CONCEPTOS RELACIONADOS CON CLUSTERS DE PCs

A continuación se definirán ciertos conceptos relacionados con clusters de PCs.

- **Paralelismo:** El paralelismo permite dividir una tarea en partes que pueden ser ejecutadas independientemente, con lo cual se logra obtener resultados en forma más expedita. El paralelismo se puede implementar a nivel de hardware y a nivel del software.

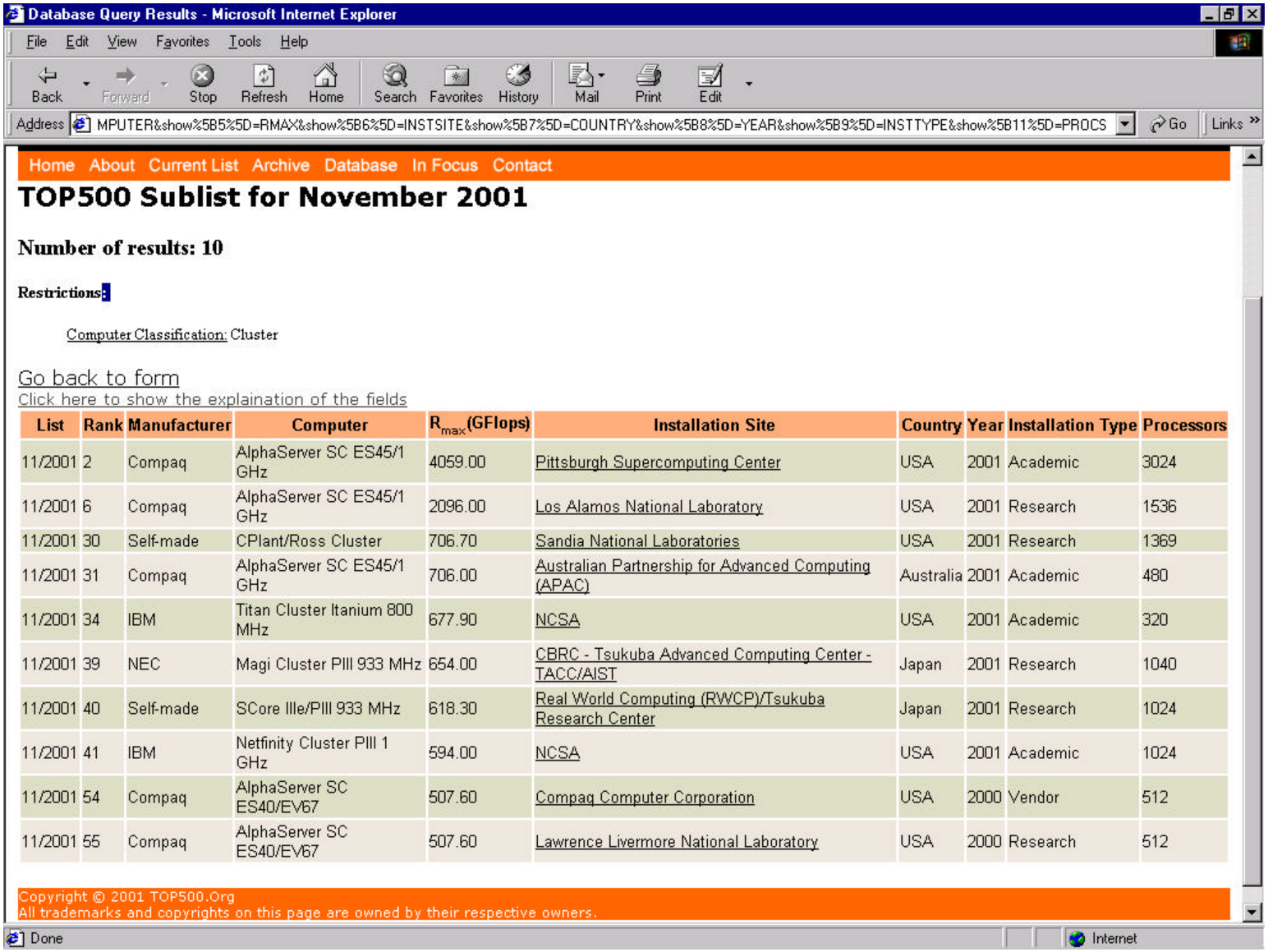

Figura 1. Top 500

A nivel del hardware se implementa mediante el uso de múltiples unidades funcionales, pipeline, caches, etc. Este paralelismo es básicamente transparente, sin embargo, conocer como opera puede permitirle al usuario optimizar su código de forma de sacarle el máximo provecho a este paralelismo. Nótese que estas optimizaciones son muy especificas a la arquitectura de la máquina. El paralelismo de software es aquel en el que dividimos una tarea en subpartes que serán distribuidas y correrán en distintos procesadores. La labor de particionamiento puede resultar fácil o ser un verdadero reto.

- **Optimización:** Optimizar un código consiste en escribirlo o generarlo (cuando la optimización proviene de un compilador) de forma tal de tome en cuenta las características de la máquina (la arquitectura) y que el número de instrucciones y de bifurcaciones sea el menor posible. Tanto la paralelización como la optimización buscan acortar el tiempo de obtención de la solución de un programa, sin embargo, usan procedimientos distintos.
- **Memoria compartida y memoria distribuida:** Cuando hablamos de paralelismo, y por lo tanto asumimos que tenemos disponibles múltiples procesadores, existen dos paradigmas de programación fundamentales que están basados en la visión que los procesos (o tareas) tienen de la memoria. Cuando la memoria es vista por todos los procesos como un solo bloque y cualquier proceso tiene acceso a cualquier región de la memoria, hablamos de *memoria compartida*. En este caso la comunicación entre los procesos se hace compartiendo datos que están en la memoria. Cuando los procesadores tienen asociados memorias privadas no accesibles a otros procesadores, se dice que la *memoria es distribuida*. La comunicación entre los procesos es a través de mensajes. Las dos librerías de pase de mensaje más usadas son: *MPI* (*Message Passing Interface*) y *PVM* (*Parallel Virtual Machine*). Los clusters de PCs caen en la categoría de máquinas de memoria distribuida.
- **Dependencia:** La dependencia se da cuando cierta parte del código no puede proceder si no se tienen los resultados de otros fragmentos del código.
- **Sincronización:** Sincronizar consiste en poner a la par dos o más procesos o subtareas. Cuando un proceso va a correr un código dependiente de resultados calculados por otro proceso, ejecuta una instrucción de sincronización la cual lo hace esperar por los resultados necesarios. Una vez recibidos los resultados, resume su labor.

- **Latencia:** Se refiere al tiempo que transcurre entre el momento en que se da una solicitud de transferencia de datos y el momento en que la transferencia efectivamente comienza. Esto se debe principalmente a la inicialización de dispositivos y la preparación de los datos. Se da principlamente cuando hay acceso a la memoria, al los discos y a la red.
- **Granularidad:** La granularidad esta relacionada con la cantidad de trabajo que se puede efectuar antes de ser necesario cierto nivel de sincronización debido a las dependencias entre las subtareas. Si el monto del trabajo es considerable, decimos que la *granularidad es gruesa*. Si es poco hablamos de *granularidad fina*. En los clusters la comunicación es a través de pase de mensajes. Por lo tanto hay consumo de tiempo para ensamblar el mensaje, enviarlo por la red, recibirlo del otro lado y finalmente desensamblarlo. Este tiempo es mucho mayor que el tiempo que requiere un acceso a memoria en las máquinas de memoria compartida. Los clusters son adecuados cuando el problema presenta una granularidad gruesa. Si la granularidad es fina, el tiempo de comunicación o sincronización predomina haciendo que la solución paralela del problema sea menos eficiente que su solución secuencial. Uno de los problemas principales de los computadores de memoria compartida es que el acceso a memoria se satura rápidamente a medida que se incrementa el número de procesadores en la máquina, mientras que en las máquinas de memoria distribuida el número de procesadores puede crecer significativamente. La eficiencia promedio real de las aplicaciones en máquinas de memoria compartida esta entre un 30-50% de la eficiencia pico anunciada y entre 5-15% para los clusters. Estas eficiencia pico anunciadas se dan para ciertas aplicaciones muy particulares y no representan las aplicaciones promedio (Gordon y Gray, 2001).
- **Red:** La red esta formada por elementos (interfaces, switches, cables, etc.) que permiten interconectar distintos procesadores, bien sea dentro de una misma caja (como algunos supercomputadores) o en cajas diferentes (como los PCs), para que estos puedan comunicarse entre si. Cuando hablamos de redes hay también un conjunto de conceptos que se deben manejar. Estos conceptos serán enfocados hacia Clusters de PCs. Entre ellos tenemos:
    - *Medio de transmisión*: al nivel más bajo las comunicación entre computadores requiere convertir los datos en alguna forma de energía y enviarla a través del medio de transmisión. Por ejemplo, corriente eléctrica para enviar a través de cable (par trenzado), luz a través de fibra óptica y ondas de radio a través del aire.

- *Protocolo*: es un acuerdo o conjunto de reglas que definen el formato, el significado y la manera en que los mensajes son enviados entre computadores. Entre la información que contiene un mensaje, además de los datos mismos que se quieren comunicar, se encuentra la dirección del destinatario y del remitente.
- *Interfase*: es el hardware (una tarjeta que se agrega al PCs) que toma la información empaquetada y la convierte a un formato que puede ser transmitido por el medio físico o medio de transmisión.
- *Latencia*: tiempo que transcurre entre el momento en que un procesador solicita una transferencia de datos y el momento en que efectivamente comienza la transmisión.
- *Ancho de banda*: se define como la tasa a la cual se puede enviar datos entre procesadores y viene expresada en bits por segundo.
- *Switch*: dispositivo electrónico, con varios canales de entrada y de salida, que hace uso de la dirección del destinatario de un mensaje para decidir por que canal enviarlo.
- *Tecnologías de redes*: definen como se usa el medio de transmisión y el tipo de medio. Entre las más conocidas están: *ATM* (*Asynchronous Transfer Mode*), *Ethernet*, *Token Ring*, *FDDI* (*Fiber Distributed Data Interconnect*) y *Frame Relay*. Ethernet es una de las más populares y usa un cable o bus, compartido por todas las máquinas, como medio de transmisión. Cuando un computador quiere enviar un mensaje, chequea si el medio esta desocupado y de estarlo procede con el envío. Si el medio esta siendo usado, espera un tiempo aleatorio antes de volver a tratar. Ethernet viene en tres versiones que difieren en el ancho de banda: Ethernet original con 10 Mb/s (mega bit por segundo), Fast Ethernet con 100 MB/s y Giga Ethernet con 1000 Mb/s. Las tres versiones exhiben una latencia menor a los 90 microsegundos (debido al protocolo la latencia no es fija). Existen otras tecnologías con latencias menores pero más costosas.
- *MPI*: es una especificación estándar para librerías de pase de mensajes para sistemas homogéneos (hay compatibilidad entre los PCs) basadas en primitivas *send* (envío) y *receive* (recepción). Existen múltiples implementaciones.
- *PVM*: es otra librería de pase de mensajes que permite colecciones heterogéneas de máquinas UNIX.

En los Clusters de PCs tenemos un ambiente de memoria distribuida. Cada PC es propietario de su memoria local. Los PC se comunican mediante el envío de mensajes y por lo general se usan implementaciones de MPI o PVM como librerías de comunicación. Se debe tener una granularidad gruesa para poder obtener beneficio de la paralelización de un problema. La sincronización de procesos se logra bloqueando un proceso que ejecuta la primitiva *receive* hasta recibir un mensaje de uno o más procesos con los cuales desea sincronizarse. El medio de trasmisión más usado para conectar clusters es el cable (par trenzado) usando la tecnología Ethernet (Fast o Giga). Como con Ethernet todos los PCs comparten el medio, pares de PCs distintos no pueden comunicarse simultáneamente. Con el fin de permitir las comunicaciones simultaneas y por lo tanto reducir la latencia, los *switches* ofrecen una alternativa a expensas de una mayor inversión.

4. HARDWARE DE UN CLUSTER

A la hora de construir un cluster Linux es necesario considerar diversos aspectos de diseño para tomar decisiones que contribuyan al mejor desenvolvimiento de la máquina basandose en los requerimientos iniciales. Es recomendable realizar una revisión constante de las tendencias actuales antes de emprender un nuevo proyecto, y así contar con nueva tecnología que satisfaga nuestras expectativas.

La mayoría de los proveedores de PCs acostumbran a vender maquinas con componentes que no son necesarios dentro de un cluster, por ejemplo, tarjetas de video sofisticadas, tarjetas de audio, etc. Con un poco de información extra, se puede obtener el hardware apropiado por un costo mucho más bajo, simplemente evitando la adquisición de elementos innecesarios.

Esta sección trata sobre los diferentes aspectos relacionados con el hardware que se deben ser considerados en el diseño de un cluster.

4.1. HARDWARE DE LOS NODOS

Un cluster Linux es una red de nodos, donde cada uno de ellos es un computador personal común. Por esto, los nodos constituyen el elemento principal del cluster, los cuales son responsables de todas las actividades asociadas con la ejecución de los programas de aplicación y de dar soporte al software especializado presente en los clusters. Según la función que cumplen los nodos pueden ser ubicados dentro de las siguientes categorías.

- Ejecución de instrucciones.
- Almacenamiento rápido de información temporal.
- Alta capacidad de almacenamiento de información persistente.
- Comunicación con ambientes externos incluyendo otros nodos.

Uno de los principales inconvenientes impuesto por la tecnología es el llamado *processor - memory gap*. Esto es, la diferencia de velocidades entre el procesador y la memoria. El rendimiento de los procesadores es duplicado cada 18 meses (ley de *Moore*), alrededor de un 60 % por año, mientras que la memoria solo mejora un 9 % por año. Por esta razón, no es posible almacenar datos en la memoria tan rápido como el procesador puede manejar esos datos, y por eso a menudo el procesador debe esperar por la memoria. Esta diferencia de velocidades se incrementa un 50 % por año. La figura 2 muestra la evolucion de estos dos componentes.

Actualmente se puede elegir cada componente de los nodos dentro de una gran variedad, por ejemplo, hay más de una familia de procesadores y dentro de cada familia existe más de una alternativa. Seleccionar la configuración apropiada para los nodos de un cluster puede parecer algo difícil de realizar debido a la gran diversidad de componentes presentes en el mercado. Sin embargo, existe un conjunto de parámetros críticos que caracterizan primordialmente a un nodo.

- **Frecuencia de reloj del procesador:** Esta es la principal señal dentro del procesador que determina la tasa de procesamiento de instrucciones.
- **Rendimiento punto flotante pico:** Es la combinación de la frecuencia de reloj y el numero de operaciones punto flotante que pueden ser procesadas.
- **Tamaño de la memoria cache:** Es la capacidad de almacenamiento del buffer de memoria de alta velocidad entre la memoria principal y el procesador.
- **Capacidad de la memoria principal:** Es la capacidad de almacenamiento del sistema principal de memoria del nodo donde reside el conjunto de datos globales de las aplicaciones.
- **Capacidad de disco:** Es la capacidad de los dispositivos de almacenamiento secundario.
- **Ancho de banda pico de la tarjeta de red:** Es el ancho de banda teórico proporcionado por la interfaz de red.

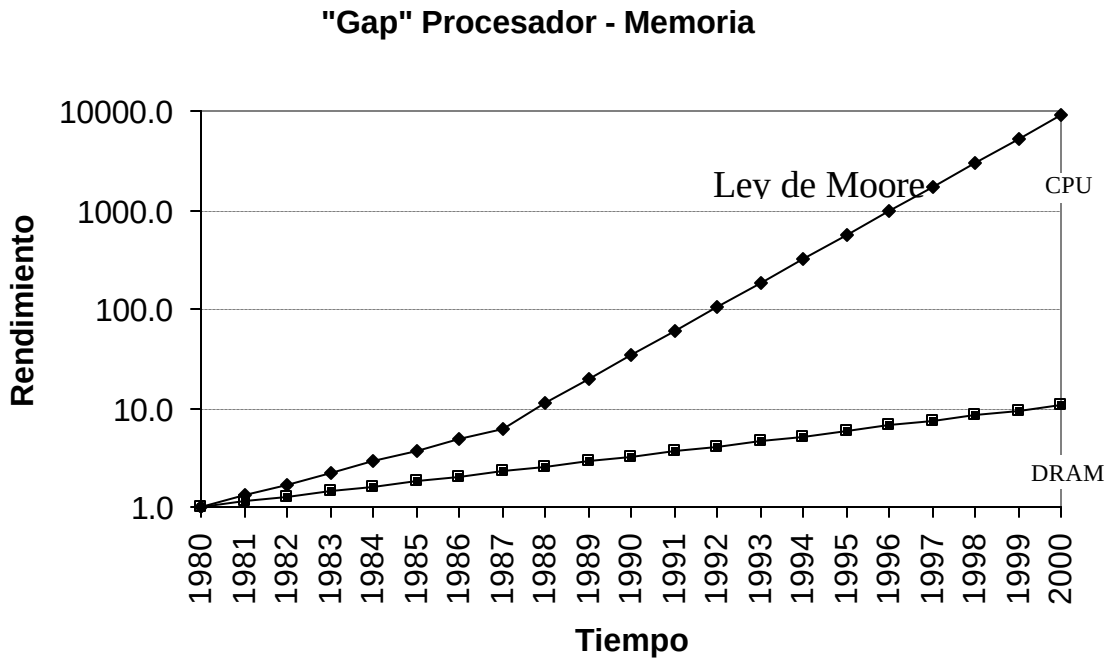

Figura 2 “Gap” Procesador - Memoria

4.1.1. Procesador

El procesador constituye toda la lógica requerida para la ejecución del conjunto de instrucciones, gestión de la memoria, operaciones enteras y punto flotante, y el manejo dela memoria cache.

Los nodos generalmente contienen procesadores *Alpha*, *Intel x86* o *AMD*. La utilización de otro tipo de procesador es permitido, sin embargo, no se consideran de uso común, ya que se elimina una de las principales características de cluster Linux (uso de componentes comunes), la cual permite reemplazar de forma fácil y con bajos costos cualquier componente del sistema.

Las máquinas con más de un procesador (*Simetric MultiProcessor* o *SMP*) son utilizadas comúnmente en clusters debido a la gran capacidad de prestaciones que proporcionan. Sin embargo, la velocidad de los buses de las tarjetas madres no tienen la capacidad necesaria para dar apoyo a arquitecturas SMP, lo que representa un cuello de botella entre los diferentes medios de almacenamiento y el procesador.

4.1.2. Memoria

La memoria de un computador personal es el sistema de almacenamiento más cercano al procesador. Las características deseables de la memoria son: rapidez, bajo costo y gran capacidad. Desafortunadamente, los componentes disponibles hasta ahora, solo poseen una combinación de cualquiera dos de estas características. Los sistemas de memoria modernos utilizan una jerarquía de componentes implementados con diferentes tecnologías que juntos, y en condiciones favorables, logran obtener las tres características. A pesar de todo esto, la capacidad de almacenamiento de memoria se ha incrementado considerablemente, cuadriplicándose cada tres años aproximadamente, mientras que su costo ha sufrido un constante decremento.

Las memorias constituidas por semiconductores dieron un cambio significativo a la predominancia de los medios de almacenamiento magnéticos de los años 70. Actualmente hay dos tipos de memoria de semiconductores: memoria estática de acceso aleatorio (SRAM[10]), la cual se caracteriza por ser muy rápida pero de capacidad moderada, y la memoria dinámica de acceso aleatorio (DRAM) cuya capacidad de almacenamiento es considerable pero opera de forma más lenta.

La memoria estática es implementada con celdas de bits fabricadas con circuitos *flip-flop* de transistores múltiples. Estos circuitos activos pueden cambiar su estado y ser accesados rápidamente, sin embargo, su consumo de energia es significativo. Este tipo de memoria es empleada en aquellas partes del sistema donde se requieren medios de almacenamiento rápidos tales como memorias cache L1 y L2.

La memoria dinámica de celdas de bits es fabricada con capacitores y transistores de puentes simples. Estos capacitores almacenan una carga en forma pasiva y las operaciones de acceso a cada celda consume esta carga, además, el aislamiento de los capacitores no es perfecta y la carga se pierde con el tiempo auque no sea accedida. Por esto, debe ser restablecida con cierta frecuencia la carga de los condensadores, lo cual implica tiempos de accesos mayores. Dentro de esta categoría podemos encontrar algunas variaciones como memoria dinámica tipo *Extended Data Ouput* (*EDO DRAM*) que proporciona un esquema de buffer interno modificado que mantiene los datos en la salida más tiempo que las DRAM convencionales. El otro tipo de memoria dinámica es la memoria dinámica sincrona, que implementa un modo de cauce por etapas que permite iniciar un segundo ciclo de acceso antes de ser completado el ciclo anterior.

---

[10] Más detalles de este y otros términos se pueden encontrar en *http://www.webopedia.com*

Las diferentes necesidades de velocidad de almacenamiento y las diferentes implementaciones da origen a una jerarquía de memoria, ilustrada en la figura 3.

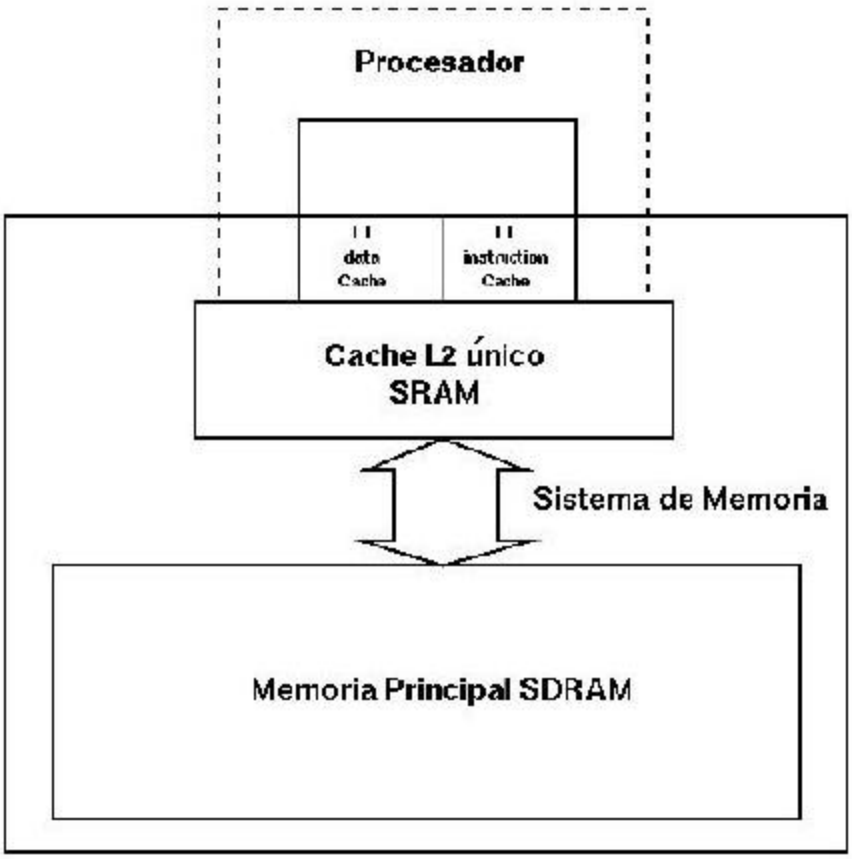

Figura 3 Jerarquía de Memoria

4.1.3. Disco

Toda la información presente en los sistemas de memoria se pierde una vez que el computador se apaga o se reinicia. Es por eso que son necesarios los sistemas de almacenamiento secundarios tales como CD roms, floppies, discos duros, etc. De éstos el único medio realmente necesario es el disco duro ya que el resto de los dispositivos por lo general no se usan en ambientes de calculo intensivo.

Los discos duros mantienen copia del sistema operativo, programas y datos, así, se cuenta con un medio de almacenamiento para mantener grandes cantidades de información. Existen dos interfaces principales utilizadas para manejar discos duros: *IDE* y *SCSI*. Originalmente, las interfaces IDE, de menor rendimiento, tenían predominancia

en el mercado de PCs debido al bajo costo en relación a los discos SCSI. Sin embargo, el actual abaratamiento de los costos de las interfaces SCSI han permitido incorporarlas en las nuevas tarjetas principales, aunque todavía los precios no son comparables con los discos IDE. Por lo general, se recomienda utilizar discos SCSI en situaciones de altas tasas de operaciones de lectura y escritura, por ejemplo, directorios hogares, y utilizar discos IDE en situaciones de bajo número de accesos como por ejemplo espacios dedicados al sistema.

Existen varios métodos para configurar los medios de almacenamiento en un cluster Linux, los cuales difieren en rendimiento, precio y facilidades en la administración.

- **Clientes sin disco (Disk-less)**

  Los nodos esclavos o clientes no poseen disco duro interno y toman todos los sistemas de archivos a través de la red. Es el nodo maestro el que proporciona, usualmente a través de NFS, los sistemas de archivos para los nodos esclavos.

  La principal ventaja de esta configuración es la facilidad en la administración del cluster ya que al agregar un nuevo nodo solo hay que modificar ciertos archivos en el servidor.

  La desventaja de tener clientes o esclavos sin disco es que el trafico a través de la red se incrementa. Dependiendo de la red instalada, esta puede ser una configuración poco adecuada para el cluster.

- **Instalación Local Completa en los Clientes**

  Todo el software, tanto el sistema operativo como las aplicaciones, son instaladas en los discos internos de cada nodo cliente. Esta configuración reduce a cero el trafico NFS para obtener el sistema operativo o cualquier otra aplicación por parte de los nodos esclavos.

- **Instalación NFS Estándar**

  Esta configuración es el punto medio de las dos anteriores. El sistema operativo se encuentra en los discos internos de los nodos esclavos y estos obtienen los directorios hogar de los usuarios y los sistemas de archivos que contienen las aplicaciones, a través de NFS, desde el nodo maestro.

- **Sistemas de Archivos Distribuidos**

  Los sistemas de archivos son aquellos que son compartidos por todos los nodos, es decir, cada nodo posee un pedazo del sistema de archivos lo cual incrementa la velocidad en los accesos a la información debido a la presencia de más de un

dispositivo físico para el manejo de los datos. Sin embargo, esta configuración esta en fase experimental y por esta razón no es recomendada.

4.2. RED

La red de interconexión convierte a un conjunto de computadores personales en un solo sistema. Además, proporciona el acceso remoto al cluster y a sus servicios. Originalmente, fue posible crear clusters Linux debido a la disponibilidad de tecnología de red debajo costo y ancho de banda moderado. Ethernet fue el protocolo por excelencia utilizado en los inicios de los cluster, pero en la actualidad existe una gran variedad de tecnologías que pueden ser utilizadas para construir clusters Linux. Sin embargo, la relación costo-rendimiento de Fast-Ethernet proporciona la mejor opción para implementar la red de un cluster. Otra razón para seleccionar esta topología de red es la facilidad para proporcionar escalabilidad a la hora de agregar nuevos nodos al cluster. El rápido surgimiento de Gigabit-Ethernet podría proporcionar un medio alternativo para la red del cluster, sin embargo, su alto costo y latencia similar a Fast-Ethernet aún no la hacen una tecnología atractiva.

4.2.1. Hubs y Switches

Las interfaces de red proporcionan la conexión entre el procesador y la red del sistema (System Area Network, SAN). La efectividad de la red del sistema y su escalabilidad depende de los medios mediante los cuales los nodos son interconectados. Estos medios incluyen cables coaxiales pasivos, repetidores activos y switches inteligentes.

Una de las grandes ventajas de Ethernet fue la utilización de los cables coaxiales que no necesitaban de dispositivos adicionales de lógica costosa para la interconexión de los nodos. Irónicamente, los Hubs y repetidores baratos junto con cables par trenzado remplazaron el cable coaxial. Estos dispositivos todavía utilizan el protocolo CSMA/CD y todos los nodos se ven unos a otros.

Los switches son dispositivos más sofisticados que también aceptan paquetes sobre cable par trenzado, sin embargo, estas señales no son repetidas hacia todos los nodos sino enviadas solo al nodo destino. Esto permite dedicar todo el ancho de banda a la comunicación entre cualquier par de nodos. En topologías de tipo arbol la utilización de switches puede convertir al nodo raíz en un cuello de botella debido al alto trafico que manejaría.

La configuración de la red es el punto más importante en la construcción de clusters Linux ya que le da características especiales: proporciona un nivel de seguridad al implementar una red privada, hace lucir al cluster como una sola máquina y aísla el trafico entre los nodos proporcionándoles un ancho de banda dedicado. La configuración más común se ilustra en la figura 4.

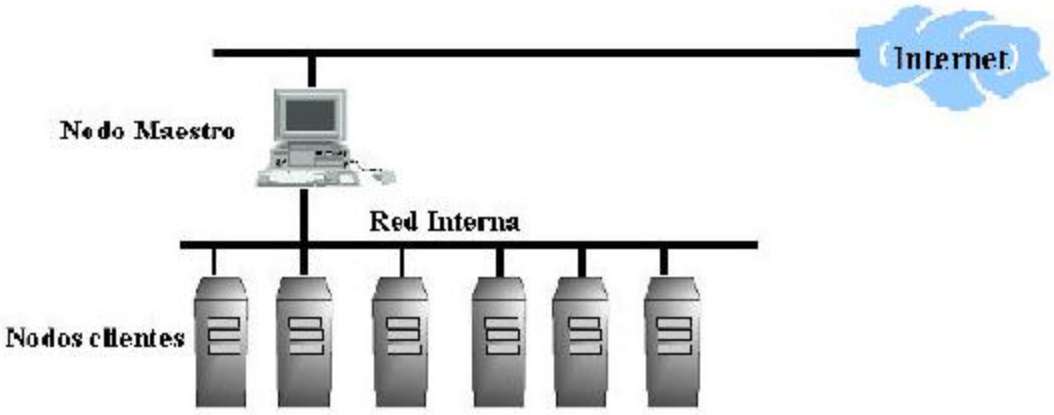

Figura 4 Configuración de red simple

El problema con esta configuración es que puede haber mucho trafico que interrumpa la comunicación de los nodos. Este trafico generalmente proviene del compartimiento de sistema de archivos de red (usando NFS). Para evitar esto, se puede agregar una segunda red que atienda los sistemas de archivos remotos. Ver figura 5.

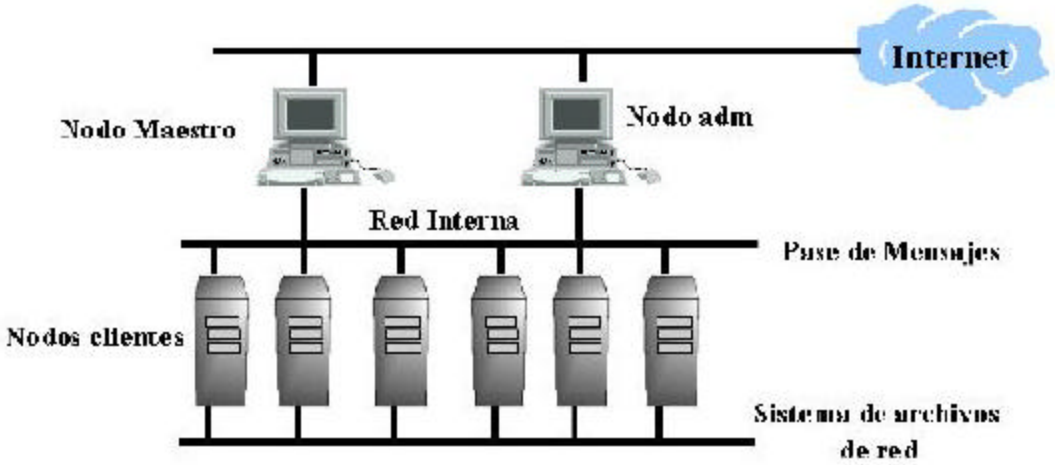

Figura 5 Configuración con dos redes

Otra de las soluciones utilizadas es una técnica llamada channel bonding que consiste en unir dos interfaces de red y hacerlas lucir como una sola con ayuda del sistema

operativo. Cada nodo posee dos interfaces de red y cada una esta conectada a un switch diferente. Ver figura 6.

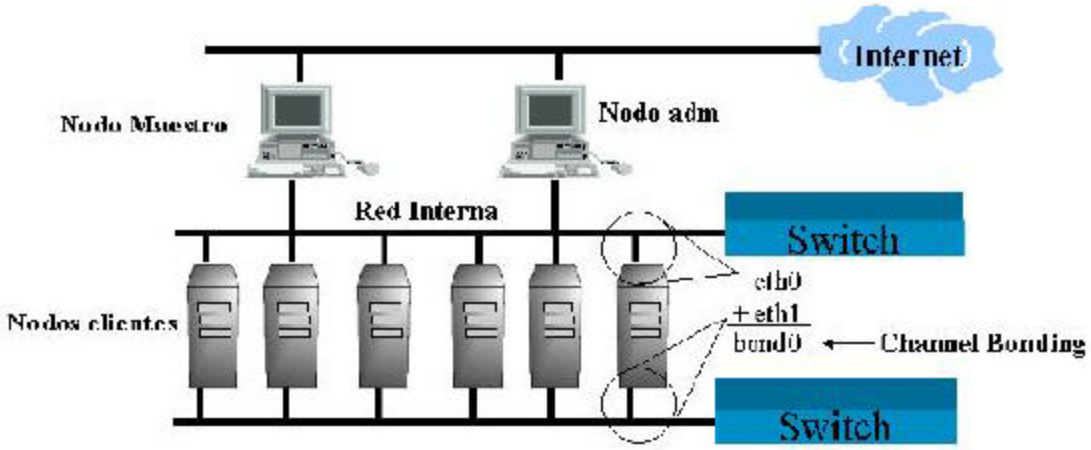

Figura 6 Configuración "channel bonding"

5. SOFTWARE DE UN CLUSTER

5.1. SISTEMA OPERATIVO

Linux es una variante del sistema operativo Unix. Desde su liberación en 1991 por su creador Linuz Torvalds se ha convertido en uno de los sistemas operativos más populares en la actualidad. Estrictamente hablando Linux es solo el kernel o núcleo de un sistema operativo, es decir, el proceso principal encargado de administrar todos los recursos de la maquina. Comúnmente se le denomina Linux al kernel más todo el conjunto de aplicaciones como ambientes de ventanas, navegadores, etc. En realidad, el núcleo empaquetado junto con el resto del software es denominado distribución. Las diferentes distribuciones de Linux proporcionan la infraestructura necesaria para proveer el resto de funcionalidades del sistema. Los servicios principales de una distribución de Linux son: facilitar el proceso de instalación y proveer una fuente de software. Los clusters Linux pueden utilizar cualquier distribución como sistema operativo. En sus inicios, los cluster para computación de alto rendimiento empleaban la distribución de Linux Slackware, ahora la mayoría de los clusters han migrado a la distribución de RedHat por su fácil administración y herramientas disponibles.

5.1.1. Características

Linux es un Unix completo desarrollado con las contribuciones de la comunidad internacional. Dentro de las características más resaltantes de este sistema operativo se tiene:

- **Paginamiento por demanda**: Linux proporciona soporte para el manejo de memoria virtual incluyendo traducción de direcciones y paginamiento por demanda. Estas paginas son bloques de memoria continuos, generalmente de 4KB de longitud.
- **Memoria Virtual:** Linux ofrece soporte para la gestión de paginas de memoria entre el disco duro y la memoria física.
- **Multitarea:** Linux es un sistema operativo que maneja concurrencia real en la ejecución de procesos. Esto se hace a través de la técnica de tiempo compartido empleando un programador de tareas que reconoce y gestiona prioridades.
- **Soporte para redes:** Una amplia variedad de topologías y protocolos son soportados por Linux. Además, ofrece vías alternas para extender las capacidades de red de tecnologías comunes, por ejemplo, channel bonding que consiste de una técnica para hacer trabajar dos interfaces de redes como una sola.
- **Robustes:** Linux es una plataforma extremadamente confiable proporcionando sistemas que pueden correr por meses o años sin bloquearse.

5.2. SOFTWARE DE APLICACION

Un cluster Linux es simplemente un conjunto de maquinas conectadas a través de un medio de interconexión. Existen diversas herramientas para hacer que este conjunto de máquinas colaboren entre si para ejecutar tareas.

- **MPI (Message Passing Interface):** Es uno de los paradigmas de programación paralela más comunes. El calculo paralelo en un cluster se realiza dividiendo un trabajo en varias porciones y distribuyéndolos en los nodos de tal manera que cada uno de ellos ejecute una de estas porciones. Cualquier información que necesiten intercambiar alguno de estos subprocesos es realizada a través de mensajes. MPI cuenta con varias implementaciones como por ejemplo: mpich y lam. Todas las implementaciones de MPI son bibliotecas que proporcionan rutinas para el manejo de pases de mensajes. El objetivo principal de MPI es lograr la portatibilidad a través de diferentes maquinas, tratando de obtener un grado de portabilidad comparable al de

un lenguaje de programación que permita ejecutar de manera transparente, aplicaciones sobre sistemas heterogéneos. (http://www.netlib.org)

- **PVM (Parallel Virtual Machine):** Es un conjunto integrado de herramientas de software y bibliotecas que emulan un marco de computación paralela de propósito general, flexible y heterogéneo, sobre un grupo de computadoras, de arquitectura variada, interconectadas. En sus inicios PVM fue bastante utilizado pero su utilización ha sido mermada por MPI.

- **Bibliotecas Matemáticas:** Una gran variedad de bibliotecas matemáticas están disponibles a los usuarios para ahorrar tiempo en la resolución de problemas. Muchas de ellas están basadas en MPI y dentro de las más conocidas se encuentran: BLAS, LAPACK, SCALAPAK, etc. (http://www.netlib.org)

- **HPF (High Performance Fortran):** es un conjunto de extensiones para Fortran 90 que permite a los programadores especificar como los datos son distribuidos a través de múltiples procesadores en un ambiente de programación paralela. La construcción del HPF permite a los programadores utilizar el potencial de paralelismo a un nivel relativamente alto, sin entrar en los detalles de bajo nivel del pase de mensajes y sincronización. Cuando un programa en HPF es compilado, el compilador asume la responsabilidad de organizar las operaciones paralelas en una maquina física, reduciendo enormemente el tiempo y esfuerzo para el desarrollo de programas paralelos. Para aplicaciones, los programas paralelos pueden ejecutarse significativamente más rápido que los programas Fortran secuenciales. (http://www.pgroup.com)

- **Mosix:** Mosix es una herramienta desarrollada para sistemas tipo UNIX, cuya característica resaltante es el uso de algoritmos compartidos, los cuales están diseñados para responder al instante a las variaciones en los recursos disponibles, realizando el balanceo efectivo de la carga en el cluster mediante la migración automática de procesos o programas de un nodo a otro en forma sencilla y transparente.

  El uso de Mosix en un cluster de PC's hace que este trabaje de manera tal, que los nodos funcionan como partes de un solo computador. El principal objetivo de esta herramienta es distribuir la carga generada por aplicaciones secuenciales o paralelizadas.

  Las bibliotecas paralelas como PVM y MPI disponen de herramientas para la asignación inicial de procesos a cada nodo, sin embargo, no consideran la carga

existente de los nodos ni la disponibilidad de la memoria libre. Estos paquetes corren a nivel de usuario como aplicaciones ordinarias, es decir son incapaces de activar otros re- cursos o de distribuir la carga de trabajo en el cluster dinámicamente. La mayoría de las veces es el propio usuario el responsable del manejo de los recursos en los nodos y de la ejecución manual de la distribución o migración de programas.

A diferencia de estos paquetes, Mosix realiza la localización automática de los recursos globales disponibles y ejecuta la migración dinámica "on line" de procesos o programas para asegurar el aprovechamiento al máximo de cada nodo.

- **PBS:** Es un sistema que proporciona una serie de herramientas para la gestión de trabajos batch, utilizando una unidad de programación de tareas. Además, permite el enrutamiento de estos trabajos a través de diferentes computadores. PBS cuenta con capacidades para definir e implementar políticas sobre la utilización de los recursos disponibles.

  Actualmente el hardware disponible para los sistemas Unix proporciona recursos de gran poder de procesamiento. Esto ha generado la necesidad de contar con mecanismos permitan planificar la ejecución de tareas sobre la base de los recursos disponibles. PBS ha sido creado para satisfacer esta necesidad utilizando tres herramientas principales.

  - *Servidor de Trabajos (Job Server, pbs_server)*: Este conforma el elemento principal del sistema y su función es proporcionar los servicios básicos para recibir, crear, ejecutar y modificar trabajos batch
  - *Ejecutor de Trabajos (Job Executor, pbs_mon)*: Es conformado por un demonio que se encarga de la ejecución real de los trabajos. Una vez, que este recibe un trabajo desde el servidor, crea una copia con una sesión del usuario solicitante. También, es responsable de enviarle al servidor la salida del trabajo ejecutado.
  - *Planificador (Job Scheduler, pbs_sched)*: Es un demonio que maneja las políticas de ejecución de trabajos, es decir, decide donde y cuando colocar un trabajo en ejecución. Las políticas de ejecución pueden ser definidas por el administrador del sistema.
  - Un conjunto de comandos y una interfaz grafica que pueden utilizar los usuarios para gestionar la ejecución de sus trabajos.

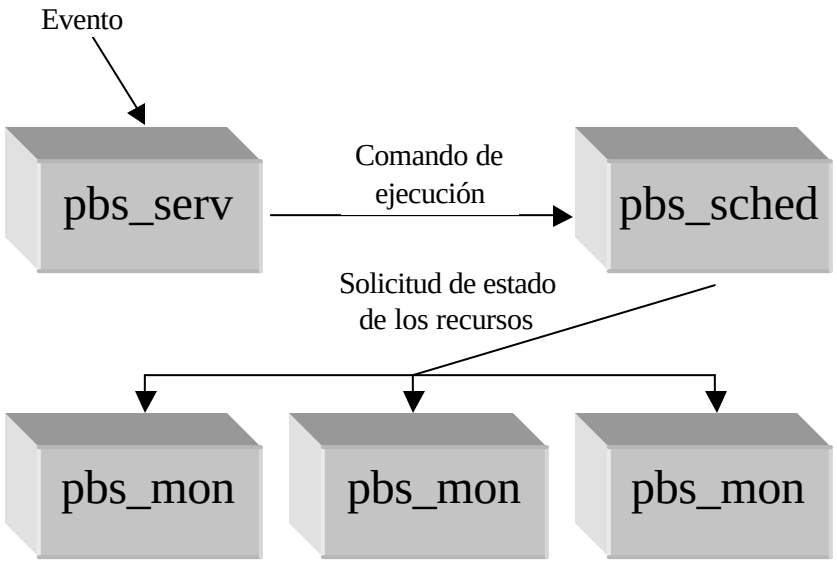

Figura 7 Ciclo de Ejecución (1)

Un ciclo de planificación de un trabajo se inicia con una solicitud de ejecución (evento) al servidor (pbs_serv). Este recibe la solicitud y envía al planificador (pbs_sched) un comando de ejecución. Este último pide a los ejecutores (pbs_mon) el estado de los recursos disponibles (Figura 7).

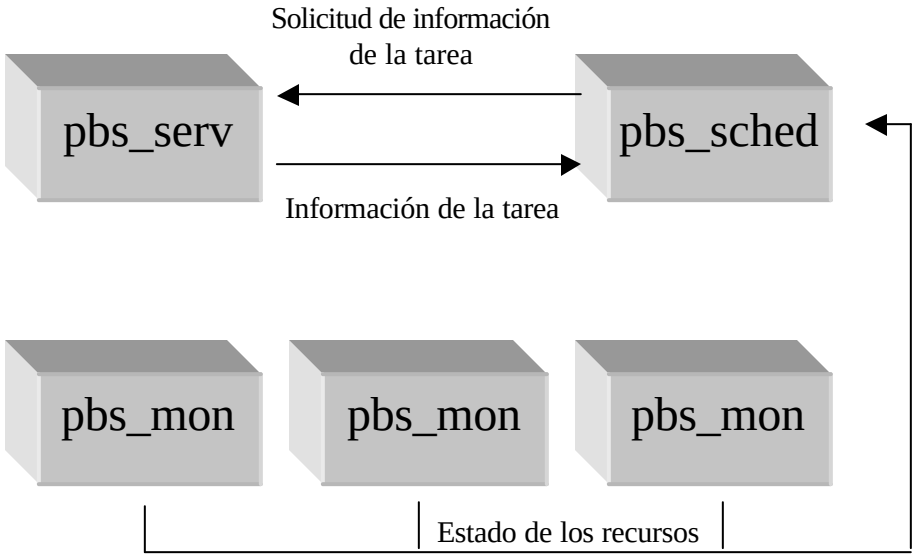

Figura 8 Ciclo de Ejecución (2)

Los ejecutores retornan los datos solicitados al planificador. Seguidamente, pbs_sched pide al servidor la información correspondiente al trabajo, y luego toma una decisión según las políticas implementadas y la información obtenida de los ejecutores y el servidor (Figura 8). Si el trabajo puede ser ejecutado, el planificador envía el resultado al servidor y este le envía la tarea al ejecutor seleccionado por el planificador (Figura 9).

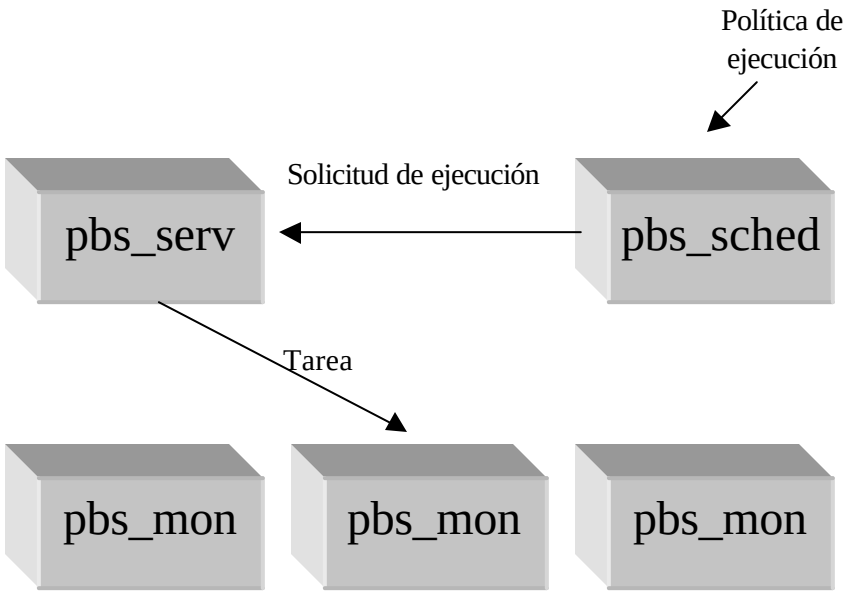

Figura 9 Ciclo de Ejecución (3)

5.3. SOFTWARE DE INSTALACION Y GESTION

- **SCE: (Scalable Cluster Environment)** Es un conjunto de herramientas de software abierto que permite a los usuarios desarrollar facilmente clusters de cálculo (Beowulf). Dentro de las herramientas se cuenta con aplicaciones para construir, gestionar y supervisar un cluster, además, posee una interfaz grafica bastante intuitiva y de fácil manejo. SCE es desarrollado en la universidad de Kasetsart en Tailandia. (http://sce.sourceforge.net)
- **Oscar:** Es un conjunto de herramientas para instalar, programar y utilizar clusters. Consiste de software abierto totalmente integrado y diseñado para manejar clusters Linux de tamaño mediano. Una de las desventajas de Oscar es la ausencia de aplicaciones para supervisión. (http://oscar.sourceforge.net)

- **Scyld Beowulf:** Es un sistema operativo para clusters que proporciona a los usuarios una imagen única de instalación. Utiliza Bproc para el manejo de los procesos. Bproc es un ambiente que permite gestionar los procesos en un cluster de forma centralizada. El programa de instalación de Scyld presenta limitaciones ante arquitecturas no homogéneas. (http://www.scyld.com)
- **ROCKS:** Es una colección de herramientas de software abierto, técnicas de administración e infraestructura de supervisión para la construcción de clusters. (http://rocks.npaci.edu)
- **LVS: (Linux Virtual Server)** Es una modificación al kernel de Linux para hacer que un grupo de máquinas trabaje en conjunto de forma transparente para ofrecer servicios distribuidos, con balanceo de carga implícito, como ftp, web, etc. (http://linuxvirtualserver.org)

BIBLIOGRAFIA

1) Spector, D. *Building LINUX Clusters*, O'Reilly, Sebastopol, California, 2002.
2) Wilkinson, B., y Allen, M. *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers*, Prentice Hall, Upper Saddle River, New Jersey, 1999.
3) Gordon, B., y Gray J. *High Performance Computing: Crays, Clusters, and Centers. What Next?* Technical Report: MSR-TR-2001-76. Microsoft Corporation. 2001. http://reserach.microsoft.com/pubs
4) Foster, I. *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*, Addison-Wesley, New York, 1995.
5) Geist, A., Suderam, V., y otros. *PVM: Parallel Virtual Machine. A User's Guide and Tutorial for Networked Parallel Computing*, MIT Press, Massachusetts, 1994.
6) Cormer, D. *Computer Networks and Internet*. 2da. Edicion. Prentice Hall, Upper Saddle River, New Jersey, 1999.
7) Hoeger, H. *Introducción a la Computación Paralela*. Reporte Interno. Centro Nacional de Cálculo Científico Universidad de Los Andes. Mérida, Venezuela, 1997. http://www.cecalc.ula.ve/documentacion/manuales_tutoriales.html
8) Snir, M., Dongarra, J., y otors. *MPI: The Complete Reference*. MIT Press, Massachusetts, 1996.

SITIOS DE INTERES EN INTERNET

1) *CeCalCULA*:
   http://www.cecalc.ula.ve/
2) Número 4 de la bibliografia:
   http://www.mcs.anl.gov/dbpp/
3) Número 5 de la bibliografia:
   http://www.netlib.org/pvm3/book/pvm-book.html/
4) Información sobre *MPI: Message Passing Interfae*:
   http://www.mcs.an.gov/mpi/index.html
5) Sitio con los 500 supercomputadores más poderosos:
   http://www.top500.com
6) National HPCC Software Exchange:
   http://nhse.npac.syr.edu/
7) Enciclopedia electrónica de términos computacionales:
   http://www.webopedia.com