Implementation of a Computational Cluster in a Non-dedicated Environment Using a Diskless Methodology

Implementación de un Cluster Computacional en un Ambiente no Dedicado Usando una Metodología Diskless

Cristian C. Ruiz Sanabria
camilo1729@gmail.com
Erick Menéses C.
erickmeneses@gmail.com
Universidad Industrial de Santander, Colombia

Abstract

The computational Clusters have reduced to a reasonable cost the demand of computing that current researches require. Although, its cost is lower than the cost of a supercomputer, the processing power is directly proportional to the number of computational resources dedicated to it. Commonly, organizations have a huge number of computational resources dedicated to different activities, many of them are only being used during his labor times and in the remain time they stay inactive. This work pretends to develop a solution that can take advantage of this idle time, based on ramfs technology used on ramdisks, which allow to store a root file system completely in RAM. This propose has the objective of creating a diskless environment that provides a good performance of parallel applications execution, as well as the capacity of doing a deploy without any change in the client`s computer and with few configuration required on behalf of the user.

1. Introducción

Los sistemas diskless son una interesante opción para la implementación de clusters computacionales por los siguientes motivos: disminuyen el costo total del sistema, reducen las posibilidades de fallas en disco, son más fáciles de administrar, además del hecho de que muchas aplicaciones de cálculo intenso no requieren ninguna unidad local de almacenamiento. Pero la ventaja más significativa de estos sistemas es la posibilidad de integrarlos a infraestructuras de cómputo existentes para la utilización de los recursos ociosos. En este artículo presentamos nuestro trabajo, el cual consistió en el desarrollo de un sistema diskless que se adaptara a nuestras necesidades, utilizando los recursos ociosos presentes en las salas de cómputo del campus universitario.

El presente artículo está organizado de la siguiente forma: En la siguiente sección se hace una revisión del estado del arte, seguidamente y a partir de esto se expone la propuesta planteada, su desarrollo, funcionamiento y finalmente en las dos últimas secciones se muestra los resultados experimentales obtenidos y las conclusiones de nuestro trabajo.

2. Estado del Arte

Por un lado la dificultad para encontrar recursos computacionales dedicados a procesamiento, y por otro la abundancia de los mismos dedicados a otros oficios donde solo se usa una fracción de su capacidad, han hecho que aumente el interés y estudio en infraestructuras cluster flexibles que puedan aumentar su poder computacional haciendo uso del tiempo de procesamiento ocioso con maquinas de la organización. Dicha propuesta ha sido abordada desde diferentes puntos de vista y el objetivo es claro llegar a usar la misma máquina para dos actividades distintas: las actividades rutinarias de la organización y como un nodo de procesamiento para un cluster. Con el fin de conservar la integridad de la configuración y ciertos componentes del cluster, no es factible que el usuario en su trabajo diario pueda utilizar el mismo espacio de trabajo y sistema, que utiliza el cluster. Además se ha demostrado en [1] que el sistema más utilizado es Windows con más de un 90 % de uso y el ambiente común en las implementaciones cluster es Linux. Esto conlleva a que se necesite dos sistemas completamente separados, uno para el cluster y otro para el usuario común, para lo cual han surgido las siguientes soluciones:

1. Uso de maquinas virtuales.
2. Alojamiento de los sistemas en dos particiones diferentes que se ejecutan a diferentes turnos.
3. Metodología diskless.

NDEE [2] utiliza una técnica basada en máquinas virtuales, que permite explotar los recursos incluso cuando estos están utilizándose, se basa en la premisa de que las maquinas desperdician ciclos de procesamiento, mientras están interactuando con el usuario en tareas como: digitar texto o esperar una descarga.

Otros sistemas como ICluster [3] utilizan una partición del equipo cliente, esta es creada en un archivo por un programa de instalación y sirve como espacio de almacenamiento para el sistema de archivos raíz. La maquina es monitoreada para poder reiniciarla en el momento debido.

Sistemas como Pelican[1] y computemode [2] utilizan espacio en RAM y almacenamiento en un servidor, el cual provee un ambiente que permite el despliegue de los sistemas sin la necesidad de modificar la computadora cliente. Se basan en el uso de un servidor que es el encargado de almacenar y proveer los sistemas de archivos que van a utilizar cada uno de los sistemas desplegados. Utilizan la tecnología PXE [4] permitiendo que las computadoras puedan arrancar por medio de la red. Pelican fue desarrollado con un propósito más académico y de pruebas. Computemode es una solución más robusta provista con el manejador de colas OAR [3] el cual hace que esta solución sea factible para trabajos de producción.

3. Propuesta

El trabajo que se presenta en este artículo consistió en el desarrollo de un nuevo sistema que se acoplara a nuestras necesidades, a través del uso de algunos componentes pertenecientes a infraestructuras existentes y el desarrollo de otros nuevos.

Actualmente las implementaciones cluster basadas en diskless presentan ciertos inconvenientes, principalmente porque el sistema de archivos raíz de los nodos clientes es cargado por medio un sistema de archivos de red, generando cierta latencia en la ejecución del sistema como se muestra en [5] y tráfico en la red [6] que aumenta con el número de nodos. Dado que una aplicación paralela hace uso intensivo de la red, provoca que infraestructuras de red que no presentan gran ancho de banda como pueden ser fast ethernet, no sean una muy buena alternativa para el montaje de un cluster diskless ya que degradaría seriamente el desempeño de este.

Teniendo en cuenta lo anterior se buscó una manera de ten

er un sistema Linux completamente almacenado en RAM, para esto el tamaño total del sistema debería ser pequeño con el fin de disponer de suficiente memoria, para la ejecución del sistema y sus respectivos procesos. La solución fue el uso del sistema de archivos ramfs [7], el cual utiliza el mecanismo de caching del kernel Linux como un sistema de archivos dinámico y liviano basado en RAM. Este sistema de archivos es utilizado en casi todas las distribuciones Linux, para implementar el *initramfs* [8], cuya misión es cargar módulos y realizar algunas configuraciones iníciales, que permitan el montaje del verdadero sistema de archivos raíz que normalmente estará en disco.

El uso de esta tecnología permitió tener un sistema completamente almacenado en memoria RAM, sin requerir almacenamientos externos como un servidor o alguna unidad local de almacenamiento.

4. Desarrollo del sistema

El Desarrollo del sistema se describirá en tres etapas las cuales se detallaran a continuación.

4.1. Implementación del Sistema Base

El sistema debía ser liviano con el fin de que fuera factible almacenarlo en la memoria RAM de la computadora, para esto se eliminaron módulos y servicios innecesarios sin perder funcionalidad en el sistema, incorporando los elementos esenciales para el funcionamiento y administración de un sistema Linux. Obteniéndose finalmente, un sistema completamente funcional, en un tamaño aproximado de 50 MB.

4.2. Optimización del despliegue

Dado que el servicio de DHCP el cual es utilizado por Computemode, presenta unos tiempos de petición y adquisición que hacen que el proceso de despliegue demore varios segundos. Se desarrolló una serie de elementos que permiten la asignación de la dirección ip por medio de la línea de comandos del kernel, la cual es suministrada por el cargador de arranque pxelinux.

---

[1] PelicanHPC, http://idea.uab.es/mcreel/ParallelKnoppix/

[2] Computemode Grid Manager, Deploying lightweight framework on intranets, http://computemode.imag.fr/old/

[3] OAR, Resource Management System for High Performance Computing, http://oar.imag.fr/

4.3. Mecanismo de Incorporación de Aplicaciones

Para incorporar las demás herramientas necesarias en un cluster, se desarrollo un conjunto de scripts que permite a petición del usuario, la integración de las aplicaciones requeridas en las imágenes desplegadas; permitiendo tener ambientes configurables para diferentes necesidades.

5. Funcionamiento del Sistema

En la siguiente grafica se describe el proceso de carga del sistema operativo en un nodo de cómputo.

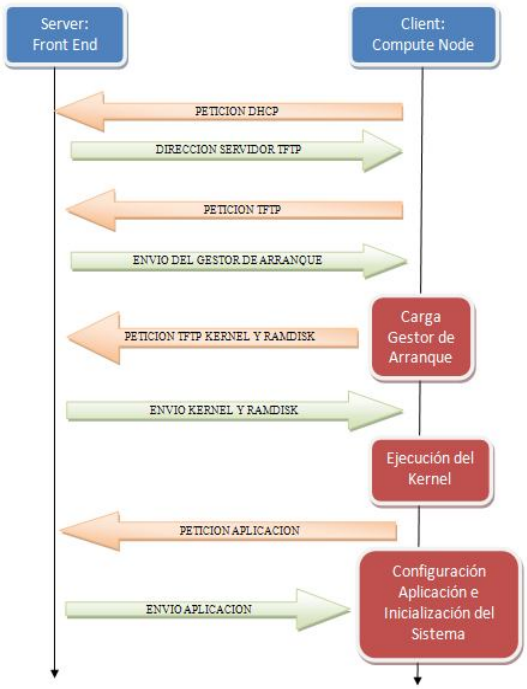

Figura 1. Proceso de carga del sistema operativo en un nodo.

El proceso es similar a las implementaciones diskless tradicionales. La gran diferencia presente, es la no necesidad de montar un sistema de archivos raíz por medio de NFS, además de contar con un mecanismo de incorporación de aplicaciones por demanda.

La grafica a continuación muestra el funcionamiento del mecanismo de despliegue de aplicaciones. Si el usuario desea un conjunto de recursos con determinada aplicación, lo primero es la reservación de estos recursos por medio de OAR, una vez obtenidos los recursos, el usuario a través de los comandos implementados en este trabajo, despliega la aplicación deseada en los nodos reservados. Finalmente cuando la reservación termina las maquinas vuelven a su estado inicial.

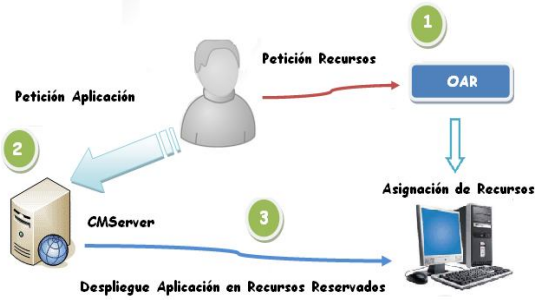

Figura 2. Despliegue de una Aplicación

Las aplicaciones pueden ser integradas en caliente con el procedimiento que anteriormente se describió, o pueden configurarse para que las aplicaciones se carguen cuando el sistema es desplegado a los nodos.

6. Pruebas y Resultados

La implementación fue evaluada en 40 computadores. Cada máquina incluido el servidor implementado, contaba con las siguientes características: CPU Intel Pentium 4 3.2 GHz, 2 GB de memoria RAM; todas las maquinas conectadas por una red Fast Ethernet de 100 Mbs.

6.1. Pruebas de despliegue

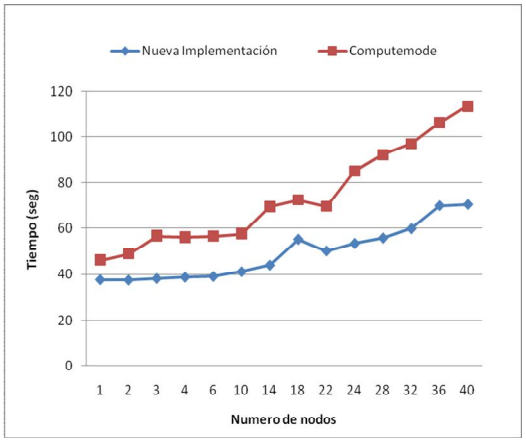

Figura 4. Tiempo de despliegue

En la grafica se observa el tiempo de despliegue, el cual comprende la transferencia del kernel, el ramdiks, el software de aplicación que en este caso fue MPICH y la completa carga del sistema. Estos resultados se contrastaron con los tiempos de despliegue de Computemode y como se puede apreciar, se obtienen mejores resultados.

6.2. Pruebas de rendimiento

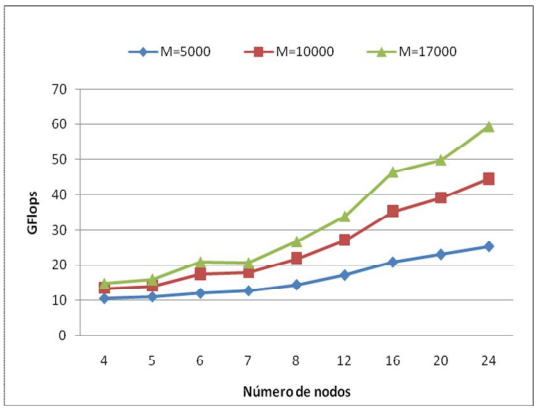

Figura 5. Desempeño del Sistema

Se ejecuto el benchmark linpack[4] con el fin de medir la capacidad de procesamiento con determinado número de nodos, se realizaron pruebas con diferentes tamaños de sistemas de ecuaciones a solucionar, los cuales están representados en la grafica por la letra M. El máximo rendimiento obtenido fue de 59.49 Gflops que corresponden a 24 nodos.

Este rendimiento puede ser comparado con el rendimiento teórico de una computadora súper escalar con la siguiente ecuación [9]:

$$R_{peak} = n_{cores} \cdot n_{FPU} \cdot f$$

Lo cual resulta para los procesadores utilizados en estas pruebas en un rendimiento teórico de 77.46 Gflops. Con esto podemos calcular la eficiencia, dividiendo el rendimiento obtenido entre el teórico dando como resultado una eficiencia de 76.80%.

7. Conclusiones

La implementación presentada en este trabajo, permite aumentar la capacidad de cómputo de un cluster sin incurrir en la adquisición de equipos dedicados.

El sistema implementado presenta un buen rendimiento en la ejecución de software en paralelo, como lo comprueba la eficiencia obtenida en las pruebas experimentales.

Aunque el sistema es almacenado en memoria RAM, se pudo observar que es posible ejecutar procesos demandantes en memoria, como el benchmark linpack.

[4] Netlib,HPL – *A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computer*, http://www.netlib.org/linpack/

8. Trabajo Futuro

Ya que estos sistemas están diseñados para ejecutarse en ambientes no dedicados, una funcionalidad que debe incorporarse es un mecanismo de checkpointing, el cual permita almacenar el estado de un trabajo y su posterior reanudación.

9. Bibliografia

[1] Gillen A, Kusnetzky D, Sayana A, Dayal H, Hingley M. Windows Operating Environment Forecast and Analysis. IDC publisher, document 24827, 2001.

[2]Reynaldo Navaes, Paulo roisenberg, Roquer Scheer, Non dedicated distributed enviroment: an Solution for Safe and Continous Explotation of Idle Cycles.

[3]Bruno Richard, Philippe Augerat. I-Cluster: Intense computing with untapped resources. HP Laboratories Grenoble April 4 2002.

[4] Pxe specification, The Preboot Execution Environment specification v2.1 published by Intel y Systemsoft. September 20 1999.

[5] James H, Laros III, Lee H ward. Implementing Scalable Disk-less Clusters using the Network File System (NFS). Sandia National labs. October 30 2003.

[6] Baris guler, Munira Hussain, Tau Leng, Victor Mashayekhi. The advantages of Diskless HPC Clusters Using NAS. Dell power solutions November 2002.

[7] Kernel org Home page. What is ramfs?. Available on: http://www.kernel.org/doc/Documentation/filesystems/ramfs-rootfs-initramfs.txt. Consulted in 26 july 2009.

[8] Landley, Rob (15 March 2005), Introducing initramfs, a new model for initial RAM disk, http://www.linuxfordevices.com/c/a/Linux-For-Devices-Articles/Introducing-initramfs-a-new-model-for-initial-RAM-disks/

[9] TobiasWittwer, An Introduction to Parallel Programming, VSSD 2006.