A description...

Departamento de Arquitectura de

Computadores y Sistemas Operativos

Informe de Progreso del  
Trabajo de Investigación

Whole genome alignment in High Performance Computing environments

Autor: Julio César García Vizcaíno

NIE: Y1497678R

Director: Antonio Espinosa, Juan Carlos Moure

Línea de Investigación: Aplicaciones bioinformáticas.

Año de inscripción: 2011

Año previsto para la presentación: 2014

Régimen de dedicación: Completa

Bellaterra, 25 de mayo de 2012

[Resumen 5](#__RefHeading__666_1025365738)

[Introducción 5](#__RefHeading__668_1025365738)

[Hadoop Distributed File System – HDFS 7](#__RefHeading__670_1025365738)

[Gestión de trabajos en cluster Hadoop 8](#__RefHeading__672_1025365738)

[Objetivo de la Investigación 9](#__RefHeading__674_1025365738)

[El método de la Investigación 9](#__RefHeading__676_1025365738)

[Seguimiento 10](#__RefHeading__678_1025365738)

[Publicaciones 10](#__RefHeading__680_1025365738)

[Artículos Previstos para Envío 10](#__RefHeading__682_1025365738)

[Planificación 11](#__RefHeading__684_1025365738)

[Conclusiones 11](#__RefHeading__686_1025365738)

[Referencias Bibliográficas 11](#__RefHeading__688_1025365738)

[Introducción 4](#__RefHeading__666_1025365738)

[Hadoop Distributed File System – HDFS 4](#__RefHeading__666_1025365738)

[Gestión de trabajos en cluster Hadoop 4](#__RefHeading__666_1025365738)

[Objetivo de la Investigación 5](#__RefHeading__666_1025365738)

[El método de la Investigación 5](#__RefHeading__666_1025365738)

[Seguimiento 5](#__RefHeading__666_1025365738)

[Publicaciones 5](#__RefHeading__666_1025365738)

[Artículos Previstos para Envío 6](#__RefHeading__666_1025365738)

[Planificación 6](#__RefHeading__666_1025365738)

[Conclusiones 6](#__RefHeading__666_1025365738)

[Referencias Bibliográficas 6](#__RefHeading__666_1025365738)

1. Resumen

En las Universidades y Centros de Investigación es común la existencia de pequeños clusters no-dedicados que se utilizan en clases de práctica de laboratorio. Estos clusters se caracterizan por ejecutar carga local que, además de ser conocida, es demasiada controlada y que varía en el espacio y el tiempo. La infrautilización de estos clusters no-dedicados - utilizados solamente durante las clases de prácticas - y la capacidad de estos para combinar la ejecución de aplicaciones de usuarios locales con aplicaciones ejecutadas en paralelo, ha generado una tendencia importante de estudio, hacia el diseño de sistemas capaces de ejecutar, aplicaciones paralelas, junto con las cargas locales de cada nodo del cluster en particular.

En este trabajo estamos especialmente interesados en combinar la ejecución de carga local con aplicaciones paralelas desarrolladas bajo el paradigma MapReduce. Estas aplicaciones manejan un volumen intensivo de datos – algunas decenas de gigabytes – y la localidad de datos es un factor importante a considerar. Para ejecutar aplicaciones Mapreduce, el entorno paralelo ha de ser capaz de suportar estas aplicaciones. Un tipo de cluster capaz de hacerlo es el Cluster Hadoop. Proponemos una política de planificación de trabajos para un cluster Hadoop que saque provecho de la localidad de los dados, inherente a lo paradigma MapReduce, y que, utilizando reserva de recursos, combine la ejecución de cargas locales y cargas en paralelo. Esta combinación de cargas no puede comprometer el rendimiento de las aplicaciones locales o las prestaciones de las aplicaciones paralelas. La Figura 1 presenta un entorno multicluster Hadoop no-dedicado, con carga local y carga en paralelo inyectada al cluster por una capa global de planificación de trabajos.

Figura 1 Entorno multicluster Hadoop no-dedicado

1. Introducción

MapReduce es un modelo de programación que permite el procesamiento y la gestión de grandes volúmenes de datos. Los datos de entrada son divididos en fragmentos de tamaños menores para que sean procesados de forma intermedia generando resultados parciales. Estos resultados parciales son procesados una vez más para generar el resultado final. El paradigma MapReduce es utilizado por empresas como Google para el procesamiento de archivos de log o búsqueda de textos en documentos, por ejemplo; y Facebook para detección de spam, data-mining y optimización.

Aunque no sea indicado para todos los tipos de problema, MapReduce simplifica el trabajo del programador porque abstrae cuestiones de la paralelización y permite al programador concentrarse en optimizar su código. La programación es dividida en dos funciones desarrolladas por el programador: Map y Redulce. El programador define una estructura de datos del tipo <clave, valor> y la función Map es aplicada, de forma paralela y distribuida, sobre cada fragmento en que los datos de entrada hayan sido divididos. El objetivo de la función Map es recorrer cada fragmento de datos buscando las coincidencias con la <clave> definida por el programador. Para cada coincidencia encontrada por la función Map es generada una tupla también en el formato <clave, valor> definida por el programador. Estas tuplas son almacenadas de forma temporal y serán procesadas a continuación por la función Reduce. Para cada fragmento de datos en que la entrada haya sido dividida es generada una lista de tuplas por la función MAP. Estas listas son entonces agrupadas y procesadas por la función Reduce.

La función Reduce actúa sobre los datos temporales que han sido generados por la función Map, haciendo el procesamiento de los valores que poseen la misma <clave>. La salida de la función Reduce también es una lista de <clave, valor> representando el resultado final de la aplicación. La Figura 2 presenta un esquema de MapReduce para una aplicación del tipo WordCount en un cluster Hadoop.

Hadoop es un framework desarrollado en el proyecto Apache y que implementa MapReduce inspirado en la propuesta de Google. Es un sistema de código abierto hecho en Java y después de su implementación original para clusters han sido implementadas versiones para varias arquitecturas incluso para Cell B.E, GPUs y procesadores multi-core.

Figura 2 – Esquema MapReduce en cluster Hadoop

El framework Hadoop hace de forma automática la fragmentación y distribución de los archivos de entrada, la planificación de los trabajos entre los nodos del entorno paralelo, realiza el control de fallos de los nodos y gestiona la necesidad de comunicación entre los nodos del cluster. Hadoop se ejecuta sobre un sistema de archivos distribuidos, Hadoop Distributed File System – HDFS, que es capaz de almacenar archivos de tamaños grandes a través de muchas máquinas. La fiabilidad es obtenida por la replicación de los dados a través de las máquinas del cluster sin necesidad de implementación de Raid. Son utilizados dos demonios para hacer la gestión de los datos: namenode y datanode.

El desarrollo de Hadoop está basado en el modelo master/worker donde es implementado el motor MapReduce: Job Tracker (el nodo master) y Task Tracker (los nodos Workers).

El planificador de trabajos está diseñado en módulos cargables que permite la sustitución de estos módulos de planificación de manera sencilla.

* 1. Hadoop Distributed File System – HDFS

Hadoop Distributed File System – HDFS es el sistema de archivos distribuidos implementado por Hadoop. Na verdad es una capa que se monta en el sistema de archivos de cada máquina del cluster. Está construido para permitir el suporte a diversos sistemas de ficheros. La gestión de los datos se realiza a través de dos demonios: namenode (en el nodo master) y datanode (en cada nodo worker).

Cuando se carga un archivo en el sistema, HDFS hace la división del archivo en bloques menores con tamaño definido por el gestor del sistema (por defecto de 65 MB). Bajo un factor de replicación, también definido por el gestor del sistema, los demonios hacen la distribución de los bloques de archivo por los nodos del cluster. Esta replicación de cada bloque de archivo además de permitir un controle de tolerancia a fallos, aumenta la posibilidad de garantizar la localidad de datos cuando Hadoop hace la distribución de tareas en el cluster. La Figura 3 presenta el sistema de ficheros distribuido de Hadoop.

Figura 3 – Sistema de archivos distribuidos de Hadoop - HDFS

* 1. Gestión de trabajos en cluster Hadoop

La gestión de los trabajos en un cluster Hadoop se hace por dos demonios: Job Tracker (el nodo master) y Task Tracker (los nodos Workers). Los trabajos MapReduce presentados al cluster son inyectados al sistema en una cola gestionada por el Job Tracker. La orden de ejecución de estos trabajos es definida por la política de planificación de trabajos de Hadoop. Por defecto esta política es basada en una cola del tipo FIFO – first in first out.

Los trabajos encolados son divididos por Hadoop en un conjunto de tareas map y reduce. Para cada tarea se asocia un Split que mapea la ubicación del bloque de archivo que tocará a tarea.

Estas tareas son asignadas a cada uno de los Task Trackers. Para cada tarea recibida el demonio local crea una maquina virtual Java para ejecutarla. Cuando un el nodo local finaliza su tarea el Task Tracker encera la maquina virtual y informa, a través de un heartbeat, de su estado al Job Tracker que hace la asignación de una nueva tarea. La Figura 4 presenta la planificación de trabajos en Hadoop.

Figura 4 – Gestión de trabajos en Hadoop - HDFS

* 1. Objetivo de la Investigación

Se plantea como objetivo general de este proyecto de investigación: Diseñar políticas de planificación de trabajos MapReduce en clusters Hadoop no-dedicados para aplicaciones intensivas de datos.

Esta política de planificación deberá gestionar con reserva de recursos la mezcla eficiente de cargas locales controladas que varían en el espacio y tiempo y cargas paralelas de aplicaciones intensivas de datos o aplicaciones intensivas de datos y cómputo. También es un reto para esta política de planificación de trabajos, conciliar localidad de los datos y de uso eficiente de los recursos.

* 1. El método de la Investigación

A partir de la definición del marco de trabajo y teniendo las aplicaciones paralelas - desarrolladas bajo el paradigma MapReduce - y la carga local ya caracterizadas el método propuesto se basa en un desarrollo en espiral.

La política de planificación de trabajos MapReduce para clusters Hadoop no-dedicados objetivo de este trabajo doctoral será desarrollada a partir de la construcción de políticas que ganarán complejidad a medida que la investigación avance.

En cada etapa del desarrollo será diseñada una política de planificación de trabajos para el cluster Hadoop no-dedicado. Esta política tendrá en cuenta los parámetros del entorno definidos previamente y las métricas de prestaciones elegidas. Después del desarrollo de política se ejecutarán experimentos en el cluster Hadoop no-dedicado de investigación y se analizarán los resultados obtenidos por la política de planificación bajo las métricas definidas anteriormente.

A partir de la evaluación de los resultados obtenidos se puede repetir la fase de desarrollo o avanzar una vuelta en la espiral aumentando la complejidad de la política de planificación de trabajos. La Figura 5 presenta el método propuesto.

Para el primer año de investigación ha sido propuesto el diseño de una política de planificación de trabajos para clusters Hadoop no-dedicados que considere más de una aplicación paralela MapReduce y cargas locales que no varían. Tener en cuenta cuestiones de localidad de los datos y uso eficiente de los recursos (nivel local en un entorno multicluster).

Figura 5 – El método utilizado.

1. Seguimiento

Hemos analizado el framework Hadoop y sus parámetros de sintonización. Hemos caracterizado una aplicación bioinformática del tipo Read Mapping desarrollada bajo el paradigma MapReduce. Esta aplicación se caracteriza por buscar similitudes entre dos archivos con secuencia genética - un archivo de referencia y un archivo de consulta.

También evaluamos técnicas de virtualización de máquinas paralelas y hemos elegido trabajar con *containers* implementados en entornos Linux bajo nos *containers groups*. Esto permitirá hacer reservas de recursos para el uso compartido de cluster Hadoop por carga local y carga paralela MapReduce.

Hemos implementado una política de planificación de trabajos para clusters Hadoop no-dedicados basada en dos puntos:

* La reutilización de la máquina virtual java creada por el demonio local (Task tracker) de cada nodo del cluster donde se ejecutan las tareas Map y Reduce.
* La afinidad de tareas de trabajos distintos que hacen uso del mismo bloque de archivo de referencia. Esto permitió aprovechar aun más la lacolidad de los datos inherente al entorno Hadoop.

1. Publicaciones
   1. Artículos Previstos para Envío

Jornada: XXII Jornadas de Paralelismo. Envío de trabajos en Junio de 2011.

Congreso: 23TH International Symposium on Computer Architecture and High Performance Computing. Envío de trabajos en Junio de 2011.

Congreso: IEEE International Parallel & Distributed Processing Symposium. Envío de trabajos en Septiembre de 2011.

Congreso: Eurosys 2012. Envío de trabajos en Septiembre de 2011 (previsión).

1. Planificación

De nuestra propuesta original de trabajo para el primer año, aún tendremos que analizar los resultados obtenidos de los experimentos.

También, tendremos que trabajar con las cargas locales. Ha de caracterizarse la carga local e implementar el uso de containers. El uso de containers permitirá compartir los recursos entre la carga local y la carga paralela MapReduce.

Además, tendremos que evaluar la política de planificación de trabajos para clusters Hadoop no-dedicados utilizando otras métricas para la carga paralela, como por ejemplo el throughput.

1. Conclusiones

Aun estamos en la primera vuelta de la espiral de desarrollo de la política de planificación de trabajos para clusters Hadoop no-dedicados. La política propuesta está basada en dos puntos: la planificación de trabajos paralelos MapReduce y la reserva de recursos utilizando containers para permitir la mezcla de cargas locales con cargas paralelas.

Hasta ahora, ha sido implementado la planificación de trabajos paralelos MapReduce. Esta implentación está basada en dos puntos: la afinidad de tareas de trabajos distintos que compartan el mismo bloque de archivo de entrada y la reutilización de máquinas virtuales. Los resultados de los primeros experimentos utilizando la política propuesta, demuestran una disminución en el tiempo de makespan de trabajos MapReduce encolados en el cluster Hadoop, comparados con los valores de la política de planificación por defecto. Todavía aun faltanos hacer una análisis más detallada de estos resultados.

1. Referencias Bibliográficas

Fischer, M.J.; Su, X.; Yin, Y. Assigning tasks for efficiency in Hadoop: extend abstract. 2010, in ACM Sysmposium on Parallelism in algorithms and architectures. (ACM-SPAA’10).

Ibrahim, S.; Jin, H.; Cheng, B.; Cao, H.; Wu, S; Qi, L.; Cloudlet: towards mapreduce implementation on virtual machines. 2009, ACM International Symposium onn High performance distributed computing. (ACM-HPDC’09).

Isard, M.; Prabhakaran, V.; Currey, J.; Wieder, U.; Talwar, K.; Goldberg, A. Quincy: fair scheduling for distributed computing clusters. 2009 , in Symposium on Operating systems principles (ACM -SOSP '09).

Kim, S.; Han, H.; Jung, H.; Eom, ,H.; Yeom, H. Harnessing input redundancy in a MapReduce framework. 2010, in Symposium on Applied Computing. (ACM – SAC’10).

Luo, Y.; Guo, Z.; Sun, Y.; Qiu, J.; Li, W.W. A Hierarchical Framework for Cross-Domain MapReduce Execution. 2011, In High Performance Distributed Computing (ACM – HPDC’11).

Qin, A.; Tu, D.; Shu, C.; Gao, C. Xconveryer: guarantee hadoop Throughput via Lightweight OS-Level Virtualization. 2009, Eighth International Conference on Grid and Cooperative Computing. (IEEE-GCC’09).

Shafer, J.; Rixner, S.; Cox, A.L. The Hadoop distributed filesystem: Balancing portability and performance. 2010, IEEE International Symposium on Performance Analysis of Systems & Software. (IEEE –ISPASS’10).

Stephen S.; Pötzl, H.; Fiuczynski, M. E.; Bavier, A.; Peterson, L. Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors. SIGOPS Oper. Syst. Rev. 41, 3 (March 2007).

Xie, J.; Yin, S.; Ruan, X. Ding, Z.; Tian, Y.; Majors, J.; Manzanares, A.; Qin, X. Improving MapReduce performance through data placement in heterogeneous Hadoop clusters. 2010, IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum. (IEEE-IPDPSW’10).

Zaharia, M.; Konwinski, A.; Joseph, A.D.; Katz, R.; Stoica, I.Improving MapReduce Performance in Heterogeneous Environments*.* 2008, 8th USENIX Symposium on Operating Systems Design and Implementation. (OSDI'08).