Revista Cubana de Ciencias Informáticas
Vol. 9, No. 3, Julio-Septiembre, 2015
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 49-62

Tipo de artículo: Artículo de revisión
Temática: Tecnologías de bases de datos
Recibido: 08/05/2015 | Aceptado: 01/06/2015

Acerca de la aplicación de MapReduce + Hadoop en el tratamiento de Big Data

About MapReduce + Hadoop application in the treatment of Big Data

Antonio Hernández Dominguez [1*], Adrian Hernández Yeja [1]

[1] Centro Telemática. Facultad 2. Universidad de las Ciencias Informáticas, Carretera a San Antonio de los Baños, km 2 ½, Torrens, Boyeros, La Habana, Cuba. CP.: 19370. ayeja@uci.cu

[*] Autor para correspondencia: ahdominguez@uci.cu

Resumen

MapReduce + Hadoop es un modelo de programación que es utilizado por disímiles empresas que se dedican al desarrollo de software en el mundo, entre ellas Google y Yahoo. Dicho modelo brinda soporte a la computación paralela sobre grandes colecciones de datos (Big Data) en grupos de computadoras. El presente artículo está enfocado en la evaluación de esta interesante técnica para la recuperación eficiente de información sobre grandes volúmenes de datos. Por su parte dicha técnica permite establecer las capacidades necesarias con las que debe contar una base de datos de información masiva, tanto desde la perspectiva de almacenamiento y técnicas de indexación, como de distribución de las consultas, escalabilidad y rendimiento en ambientes heterogéneos.

Palabras clave: MapReduce, Hadoop, Big Data, computación paralela.

Abstract

MapReduce+ Hadoop is a programming model that is used by dissimilar companies engaged in software development in the world, including Google and Yahoo. The model provides support for parallel computing on large data sets (Big Data) in groups of computers. This article focuses on the evaluation of this interesting technique for efficient retrieval over large volumes of data. Meanwhile this technique allows for the necessary capabilities that should tell a massive database of information, both from the perspective of storage and indexing techniques, and distribution of queries, scalability and performance in heterogeneous environments.

Keywords: *MapReduce, Hadoop, Big Data, parallel computing*

Revista Cubana de Ciencias Informáticas
Vol. 9, No. 3, Julio-Septiembre, 2015
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 49-62

Introducción

Desde sus orígenes el hombre ha tenido la necesidad de buscar una explicación a cada situación para poder dominar la naturaleza y ponerla a su servicio. La experiencia acumulada en su enfrentamiento al mundo exterior, ha sido el factor determinante en la mayor o menor posibilidad de lograr tal propósito. Cuando esta experiencia ha resultado insuficiente, se ha visto precisado a ir a la búsqueda de la información relevante a su necesidad de descifrar determinado problema. Debido a esto se puede afirmar que la información ha sido premisa y factor determinante para impulsar el desarrollo de la humanidad.

La relación (hombre-necesidad de información) ha transcurrido por diferentes etapas que han marcado cambios significativos en la vida del hombre, entre las que se encuentran la revolución que supuso el lenguaje, en un segundo momento la escritura, reconocida como la segunda gran revolución informativa, posteriormente la imprenta y finalmente la cuarta revolución informativa vino acompañada de las telecomunicaciones. En la actualidad, con el desarrollo vertiginoso de las Tecnologías de la Información y Comunicación, y específicamente Internet, se ha manifestado una tendencia hacia el crecimiento del desarrollo de aplicaciones para la web, que en dependencia del tipo de negocio al que estén asociadas, se inclinan o no al procesamiento de grandes volúmenes de datos.

La expansión de las redes sociales y el eficiente desarrollo de portales en general, dedicados a la gestión de información dentro de las empresas, mediante la personalización de sistemas de gestión de contenidos (CMS), son ejemplos tangibles del crecimiento desmedido en los volúmenes de información manejados (Big Data), sencillamente ya no es posible de hablar de gigabyte de información, actualmente es normal el orden de los petabyte y esto cada día que pasa va en ascenso. Como bien es sabido, el crecimiento de la información de forma exponencial es directamente proporcional a los requerimientos de hardware necesarios para soportar dicho aumento. Por tanto las empresas al iniciar cualquier proyecto de informatización, deben garantizar que se implemente primero, una adecuada infraestructura tecnológica basada, entre otros aspectos, en técnicas que posibiliten un correcto almacenamiento y posterior análisis de la información, tratando de utilizar, en los casos que no sea posible el uso de tecnología de punta, ambientes que no consuman muchos recursos, que sean más escalables y que provean una alta disponibilidad.

Algunas de las técnicas más utilizadas para este problema generalmente incluyen: balanceo de carga, replicación y distribución horizontal de la información (*sharding*), siendo esta última muy utilizada en estos momentos por las ventajas que ofrece. Para entender un poco mejor esto, se debe explicar que la idea principal consistiría en fragmentar de forma eficiente los datos, distribuyéndolos entre diferentes Bases de Datos, lo que implicaría complejizar

enormemente la gestión de la lógica del negocio, sin brindar soluciones contundentes al problema de la escalabilidad, debido a que el hecho de añadir más capacidad constituye aún un procedimiento bastante difícil. Es por ello que se hace necesario buscar métodos alternativos que permitan una simplificación en la escalabilidad del sistema. La utilización del paradigma MapReduce, de conjunto con su implementación más exitosa, el framework Hadoop, ha ganado en estos momentos un importante espacio en el mundo del tratamiento de grandes volúmenes de datos.

En este artículo se presenta una selección de las principales aplicaciones que ha tenido MapReduce + Hadoop a la gestión de Big Data, publicadas en las dos últimas décadas, haciendo énfasis en los algunas de sus ventajas y desventajas, así como los algoritmos utilizados.

Desarrollo

Computación paralela

La computación paralela es la utilización simultánea de múltiples recursos de computación para resolver problemas computacionales. Tradicionalmente, el software ha sido escrito para ser ejecutado en forma secuencial. Esto significa que para resolver un problema, el algoritmo que se implementa debe ser escrito como una secuencia de instrucciones. Como restricción se tiene que estas instrucciones deben ser ejecutadas una a la vez, y solamente en el orden especificado por el algoritmo.

Cada una de estas instrucciones es ejecutada por una CPU (*Central Processing Unit*). El tiempo que demora la resolución del problema depende de la eficiencia del algoritmo y de la rapidez con la que la CPU realiza los cálculos, haciendo que este último factor haya sido y siga siendo sujeto de estudio.

En la búsqueda de cómo continuar incrementando la capacidad de procesamiento, y tomando en cuenta otros factores limitantes, se ha considerado el paralelismo como buena solución. La computación paralela se ha usado históricamente, mostrando buenos resultados en distintas áreas, como en computación de alto rendimiento, servidores, aceleración gráfica, y muchos sistemas embebidos (Aragundi, y otros, 2009).

Paradigma MapReduce

Existen diversas definiciones para este modelo, por tanto se adoptará la posición que asume el Centro de Investigación Científica y Tecnológica de la Escuela Superior Politécnica del Litoral, Guayaquil, Ecuador, debido a que es una de las instituciones latinoamericanas que más ha publicado acerca de este tema en revistas científicas. En una de sus tesis de pregrado se plantea que MapReduce es un modelo de programación distribuida que permite el

procesamiento masivo de datos a gran escala de manera paralela. Desarrollado como alternativa escalable y tolerante a fallos para el procesamiento masivo de datos (Parrales Bravo, y otros, 2010). Dicho modelo se ha inspirado en las funciones map y reduce utilizadas comúnmente en los lenguajes de programación, aunque con propósitos totalmente distintos. Por otra parte es defendido por muchos como una alternativa muy simple para implementar concurrencia (Dean, y otros, 2004).

Historia

Este modelo surge, a partir de los principios conocidos en los años ochenta de la computación distribuida. Haciendo un poco de historia, las primeras implementaciones que se hicieron en Google requirieron la necesidad de realizar complejas operaciones de multiplicación de grandes matrices para calcular el *PageRank*, o lo que es lo mismo el ranking de páginas en una búsqueda. De ahí la emergente popularidad de *MapReduce* como un método de cálculo dentro del campo del álgebra lineal. Posteriormente la preocupación por tratar grandes colecciones de datos, llevó a crear algoritmos y frameworks capaces de poder procesar terabytes de información. Entre las primeras aplicaciones capaces de programar *MapReduce* emergió como favorito Hadoop, diseñado inicialmente por Doug Cutting, haciendo reverencia a su elefante de juguete (Vance, 2009). Fue desarrollado originalmente para apoyar la distribución del proyecto de motor de búsqueda Nutch.

Como fue explicado inicialmente MapReduce divide el procesamiento en dos funciones: Map y Reduce. Cada una de estas fases utiliza pares <clave - valor> como entradas y salidas (Aragundi, y otros, 2009). A continuación se explican algunos elementos de cada una:

Función map()

La función map() posee la característica de trabajar sobre grandes volúmenes de datos. Estos datos son divididos en dos o más partes. Cada una de estas partes contiene colecciones de registros o líneas de texto. Una función map() es ejecutada para cada porción de datos por separado, con la finalidad de calcular un conjunto de valores intermedios basados en el procesamiento de cada registro. MapReduce agrupa los valores de acuerdo a la clave intermedia y posteriormente los envía a la función reduce () (Aragundi, y otros, 2009).

Función reduce ()

La función reduce () se ejecuta para cada elemento de cada lista de valores intermedios que recibe. El resultado final se obtiene mediante la recopilación e interpretación de los resultados de todos los procesos que se ejecutaron (Aragundi, y otros, 2009).

Revista Cubana de Ciencias Informáticas
Vol. 9, No. 3, Julio-Septiembre, 2015
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 49-62

MapReduce opera exclusivamente con pares clave/valor, es decir, el framework ve el input del trabajo como un set de pares clave/valor y produce sets de pares clave/valor como salida del trabajo, posiblemente de tipos distintos. Las clases clave y valores deben ser serializables por el framework, y por ende necesitan implementar la interfaz *Writable*. A continuación, se detalla en las figura 1 el flujo lógico del proceso MapReduce; por ejemplo para el caso en que se deseara contar cuantas veces aparece repetido un animal en una lista determinada, se haría lo siguiente:

Figura 1: Flujo Lógico de procesos MapReduce

Por tanto una simple implementación de dicho proceso sería:

```
1  map(function, list) {
2   foreach element in list {
3     value = function(element)
4     intermediateResult.add(value)
5   }
6  }
7
8  reduce(function, list, init) {
9   result = init
10  foreach value in list {
11    result = function(result, value)
12  }
13  outputResult.add(result)
14 }
```

Figura 2: Una muestra de cómo implementar MapReduce

Aplicaciones generales MapReduce

Actualmente es ampliamente utilizado por diferentes empresas encargadas de desarrollo de estos tipos de software. Las principales áreas de aplicación están relacionadas con:

- Tratamiento de Big Data.
- Grep distribuido: Se encarga de crear pequeños programas altamente especializados en una sola tarea, para luego utilizarlos combinados a través de los enlaces que pudieran existir entre ellos. Grep es una utilidad de línea de comandos escrita originalmente para ser usada con el sistema operativo Unix. Usualmente, grep toma una expresión regular de la línea de comandos, lee la entrada estándar o una lista de archivos, e imprime las líneas que contengan coincidencias para la expresión regular
- Ordenamiento distribuido.
- Construcción de índices invertidos.
- Sistemas de recomendación.
- Análisis de logs.
- Problemas de aprendizaje de máquina (*machine learning*) (Aragundi, y otros, 2009).

En la figura 3 se muestran las principales implementaciones de MapReduce, relacionando cada una con los lenguajes de programación utilizados, de ahí que se llegue a la conclusión acerca de la gran diversificación e importancia que ha alcanzado este modelo.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. 3, Julio-Septiembre, 2015
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 49-62

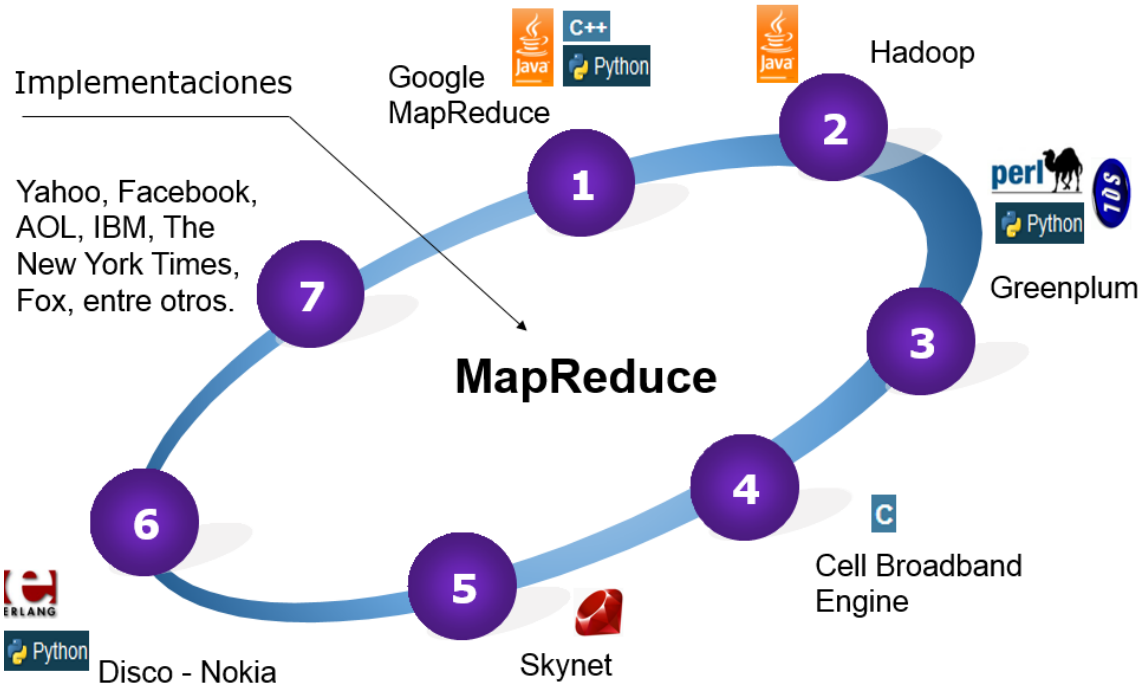

Figura 3: Implementaciones existentes de MapReduce

Hadoop MapReduce

Hadoop es una implementación *Open Source* de MapReduce para el procesamiento de grandes clústeres de datos; actualmente tiene el liderazgo en términos de popularidad para analizar enormes cantidades de información. El mismo ha sido utilizado exitosamente por grandes compañías para el almacenamiento y procesamiento de conjuntos de datos inmensos (Borthakur, 2011) (Abdullah, 2010) y (Sarkar, 2013).

Hadoop está inspirado en el proyecto de Google File System (Ghemawat, y otros, 2003) y en el paradigma de programación MapReduce. Hadoop está compuesto de tres piezas fundamentales: Hadoop Distributed File System (HDFS), Hadoop MapReduce y Hadoop Common.

El trabajo de Hadoop MapReduce consiste principalmente en dos funciones definidas por el usuario: map y reduce. La entrada de un trabajo Hadoop MapReduce es un conjunto de pares clave-valor (k, v) y la función map se llama para cada uno de estos pares. La función map produce cero o más pares de clave-valor (k', v') intermedios. El framework agrupa estos valores por medio de k' y llama la función reduce para cada grupo. Por último, la función reduce produce cero o más resultados agregados. La belleza de Hadoop MapReduce es que los usuarios por lo general solo tienen que

Revista Cubana de Ciencias Informáticas
Vol. 9, No. 3, Julio-Septiembre, 2015
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 49-62

definir las funciones map y reduce. El framework se encarga de todo lo demás, como es la paralelización y fallos (Dittrich, y otros, 2012).

Hadoop MapReduce utiliza un sistema de archivos distribuido para leer y escribir los datos. Típicamente se utiliza el sistema de archivos *Hadoop Distributed* (HDFS), que es el equivalente de código abierto de Google Sistema de Archivos (Ghemawat, y otros, 2003). El rendimiento de E/S de Hadoop MapReduce depende en gran medida de HDFS.

En la figura 4 se presenta la arquitectura de Hadoop MapReduce (Lee, y otros, 2012):

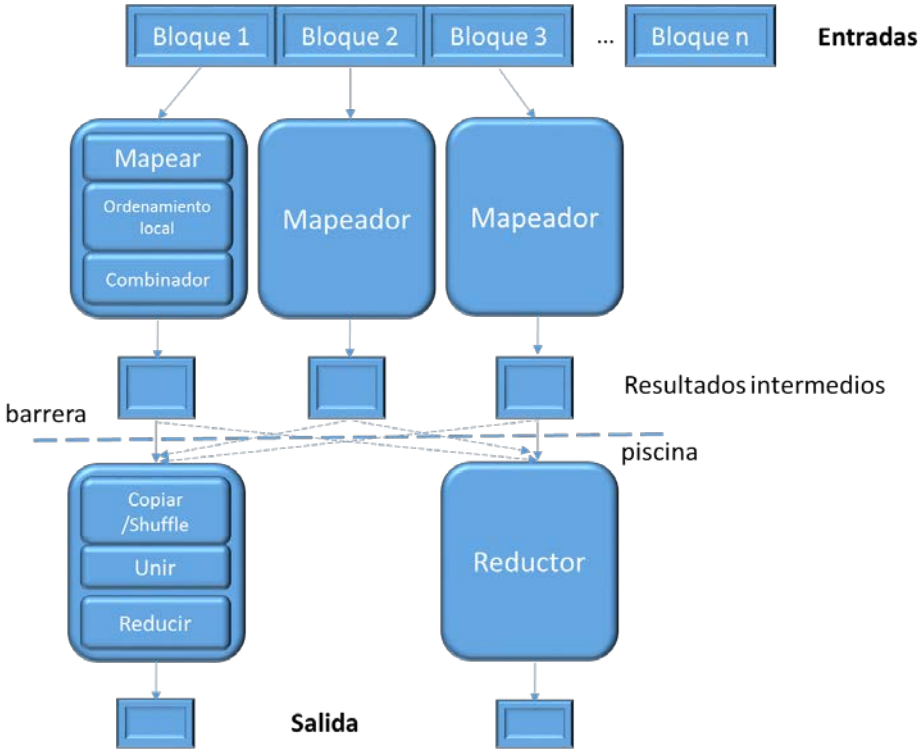

Figura 4: Arquitectura Hadoop MapReduce

Big Data

El concepto Big Data aplica para toda aquella información que no puede ser procesada o analizada utilizando procesos o herramientas tradicionales. Según la consultora Gartner (Gartner, INC, 2013) Big Data son activos de información de alto volumen, alta velocidad y variedad que demandan rentabilidad, formas innovadoras de procesamiento de la información para mejorar la comprensión y toma de decisiones. La emocionante promesa de Big

Data es que la información puede ser coleccionada y luego analizada para obtener conocimientos, patrones y conexiones sin saber de antemano lo que se debe buscar (INTEL CORPORATION, 2014).

Según la Compañía de Gestión del Conocimiento McKinsey los datos no-estructurados están creciendo en internet a una razón aproximada del 72% anual. Entiéndase por datos no-estructurados aquellos que son subidos a la red, con poca información adicional. A esto se le suman aquellos que son generados por infinidad de sensores y automatismos, debido a la actual tendencia del no tan futuro: "*Internet de las cosas*".

Por otra parte una investigación hecha por Intel, reveló el nivel de crecimiento a nivel de información que es dispersada a través del mundo (ya sea por internet, mail, diarios, libros o cualquier forma de esparcimiento de información posible), este cálculo estimado ha mostrado que el ser humano desde el origen de los tiempos hasta el 2003 ha generado aproximadamente 5 exabytes de información. Mientras que solamente en el 2012, la cantidad de información crecería a 2,7 zettabytes, representando un aumento de más de 500 veces la cantidad de información generada antes del 2003, pero esto no es todo, se espera que para el año 2015, esta cantidad de información crezca en un factor de 3 (Jara, 2012).

Esta gran cantidad de información que se genera constantemente es necesario procesarla eficientemente. Muchas organizaciones, compañías e investigadores se han enfocado en este sentido; algunos ejemplos incluyen aplicaciones analíticas de la web, aplicaciones científicas y redes sociales (Dittrich, y otros, 2012).

Hadoop MapReduce y Big Data

Se han creado gran variedad de frameworks de computación distribuida para realizar este procesamiento (Herodotou, y otros, 2011) (Dean, y otros, 2004) (Isard, y otros, 2007) (Murray, y otros, 2011) (Olston, y otros, 2008) (Thusoo, y otros, 2010) (Yu, y otros, 2008) (Zaharia, y otros, 2012) que permiten al usuario usar un conjunto de operaciones de alto nivel para manejar aspectos complejos de la computación distribuida. Uno de los más conocidos y exitosos lo representa Hadoop (White, 2009).

Las aplicaciones prácticas de Hadoop MapReduce son disímiles; es posible encontrarlo en la construcción del indexado de búsqueda de Google, Amazon, y Yahoo; aplicaciones de minería de datos, análisis de logs de usuarios, etc. (Padhy, 2012).

A continuación se presentan casos de éxito de Hadoop en el procesamiento de grandes volúmenes de datos en diferentes áreas del conocimiento humano.

En (Taylor, 2010) se presenta una implementación de Hadoop en investigaciones de bioinformática, en donde se hizo necesario procesar petabytes de datos en un análisis paralelizado tolerante a fallos. De igual forma, en (O'Driscoll, y

otros, 2013) se presentó la utilidad de Hadoop en el estudio del Genoma Humano, en donde el procesamiento distribuido y paralelizado de datos se hace necesario.

El procesamiento de grandes cantidades de información se hace vital en las redes por el tráfico que se transporta en las mismas; esta información representa por ejemplo la entrada en los Sistemas Detectores de Intrusos (NIDS), por lo que es necesario un procesamiento eficiente, como se presentó (Veetil, y otros, 2014), donde Hadoop se utiliza para la detección y emisión de alertas ante intrusiones en la red en tiempo real. De forma similar se describió en (Qiao, y otros, 2013) un sistema de análisis de tráfico desconectado usando Hadoop en donde se concluyó la eficiencia del mismo para el procesamiento de flujo de datos y su versatilidad para la detección de nodos fallidos. Un estudio similar se realizó en (Qiu, y otros, 2011), donde se diseñó e implementó una plataforma basada en Hadoop para la administración de sistemas distribuidos de flujo de datos, mejorando los sistemas tradicionales en cuanto a eficiencia, escalabilidad y confiabilidad. También se profundizó en (Ventrapragada, y otros, 2012) la utilización de Hadoop en la detección de fallos de hardware sospechosos fallidos, en donde se permitió llevar a cabo decisiones en cuanto a balanceo de carga y replicación de datos cuando se detectan nodos fallidos.

La detección temprana de tráfico anómalo puede evitar ataques en redes que pueden causar serios daños. En (Jeong, y otros, 2012), se realiza un estudio de cómo Hadoop puede procesar gran cantidad de tráfico que se genera en las redes para el procesamiento eficiente de los incidentes de seguridad que se producen.

En un esfuerzo por entender la distribución y comportamiento de eventos sísmicos correlacionados, se obtuvo en (Addair, y otros, 2014) 300 millones de sismogramas, para lo cual se necesitaron 42 días para realizar el procesamiento de forma convencional. Sin embargo, con la utilización de Hadoop se logró un rendimiento 19 veces mejor.

Facebook ha hecho uso también de Hadoop, como se constata en (Borthakur, y otros, 2011), donde se describe las razones por las que esta gran compañía seleccionó Hadoop sobre otros sistema como Apache Cassandra (Cassandra, 2013) y Voldemort (Feinberg, 2011).

Se presentó en (Glaser, y otros, 2013) un experimento para el análisis de energía física que requirió el procesamiento de 15 PB de información. Fue inconveniente almacenar, acceder y procesar esta información usando el hardware y software actuales, mientras que con Hadoop se obtuvieron resultados satisfactorios evaluándolos en un entorno de computación en la nube.

En (Lee, y otros, 2013) se describe una aplicación en tiempo real que resume posts de blogs por relevancia considerando el tiempo que los mismos son escritos. Se utiliza Hadoop para hacer el software escalable y tolerante a fallos. Se valora la calidad de los resultados obtenidos con respecto a métodos tradicionales.

Por otro lado, en (Li, y otros, 2012) se aprovecha la administración y almacenamiento distribuido en computadoras de bajas prestaciones de Hadoop el modelo procesamiento de datos en cuanto a la identificación Braille. Las pruebas mostraron que el modelo provee eficiencia en el procesamiento de datos en grandes clústeres.

En la publicación (Gao, y otros, 2014) se escribe la utilidad de Hadoop en los Sistemas de Información Geográfica, donde se construye una plataforma escalable y de alto rendimiento para el procesamiento de la información, demostrando cómo se reduce el tiempo de procesamiento comparado con operaciones de la misma magnitud en sistemas de escritorio tradicionales.

Conclusiones

El uso de nuevas herramientas gratuitas de procesamiento masivo y distribuido de datos como Hadoop a través del paradigma MapReduce, permiten a las empresas acceder a la posibilidad de escalar de manera veloz en función de sus requerimientos, sin tener que depender de la capacidad de procesamiento de un solo equipo de trabajo o de cantidades pequeñas de información sobre las cuales realizar análisis.

Es oportuno destacar que con la aplicación de este paradigma:

1. En vez de tener un número reducido de servidores con grandes prestaciones, se propone utilizar cientos/miles (o millones) de pequeños servidores distribuidos por el mundo.
2. En vez de realizar constantes copias de seguridad a la información almacenada, auxiliándose para ello en gigantescos *backups*, se plantea mantener sincronizada la misma, a través de un conjunto de “n” réplicas distribuidas en determinados servidores.
3. En vez de procesar los datos desde un lugar central, se tendría que distribuir los programas hacia dichos servidores, de esta forma los mismos serían ejecutados en forma paralela utilizando las ventajas de Map() y finalmente se consolidarían los resultados a través de la función Reduce().

Por lo tanto, MapReduce + Hadoop en el procesamiento de Big Data, puede ser una técnica extremadamente novedosa a la hora de desarrollar Aplicaciones Web, para negocios meramente complejos y cambiantes. Si bien es una tecnología relativamente incipiente, sus usos van en ascenso y estos pueden ser apreciados en el lanzamiento de

diversos productos desarrollados por empresas líderes en el negocio de Internet, como son Google y Amazon, solo por citar dos ejemplos, donde la “oferta" que brindan son netamente especializadas a los gustos de cada persona.

Referencias

- ABDULLAH, IBRAHIM BIN. 2010. *Incremental pagerank for twitter data using hadoop. Diss. Master’s thesis.* Edinburgh : s.n., 2010.
- ADDAIR, T. G., et al. 2014. *Large-scale seismic signal analysis with Hadoop. Computers & Geosciences.* s.l. : Computers & Geosciences, 2014. pp. 145-154. Vol. 66.
- ARAGUNDI, RIVAS, et al. 2009. *Tesis de pregrado: "Búsquedas avanzadas tipo Grep de tesis realizadas en la ESPOL utilizando el paradigma Map-Reduce".* [PDF] Guayaquil, Ecuador : s.n., 2009.
- BORTHAKUR, D., et al. 2011. *Apache Hadoop goes realtime at Facebook.* [ed.] ACM. s.l. : Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, 2011. pp. 1071-1080.
- BORTHAKUR, DHRUBA. 2011. *Apache Hadoop goes realtime at Facebook.* s.l. : Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, 2011.
- CASSANDRA, A. 2013. The Apache Software Foundation. *The Apache Cassandra project.* [Online] 2013. [Cited: mayo 4, 2014.] http://cassandra.apache.org/.
- DEAN, JEFFREY AND GHEMAWAT, SANJAY. 2004. *MapReduce: Simplified Data Processing on Large Clusters.* [PDF] San Francisco : OSDI'04: Sixth Symposium on Operating System Design and Implementation, 2004.
- DITTRICH, J. AND QUIANÉ-RUIZ, J. A. 2012. *Efficient big data processing in Hadoop MapReduce.* s.l. : VLDB, 2012.
- FEINBERG, A. 2011. *Project Voldemort: Reliable distributed storage.* s.l. : Proceedings of the 10th IEEE International Conference on Data Engineering, 2011.
- GAO, S., et al. 2014. *Constructing gazetteers from volunteered Big Geo-Data based on Hadoop.* s.l. : Computers, Environment and Urban Systems., 2014.
- GARTNER, INC. 2013. Big Data. *IT Glossary.* [Online] 2013. [Cited: abril 27, 2014.] https://www.gartner.com/it-glossary/big-data/.
- GHEMAWAT, SANJAY, HOWARD, GOBIOFF AND SHUN-TAK, LEUNG. 2003. *The Google file system.* s.l. : ACM SIGOPS Operating Systems Review, 2003. Vol. 37.

- GLASER, F., et al. 2013. *Using MapReduce for High Energy Physics Data Analysis.* 2013.
- HERODOTOU, H., et al. 2011. *Starfish: A Self-tuning System for Big Data Analytics.* s.l. : CIDR, Enero 2011. Vol. 11.
- INTEL CORPORATION. 2014. What is Big Data? *Big Data Intelligence Begins with Intel.* [Online] 2014. [Cited: abril 27, 2014.] http://www.intel.com/content/www/us/en/big-data/big-data-what-is-big-data-landing.html.
- ISARD, M, et al. 2007. *Dryad: distributed data-parallel programs from sequential building blocks.* s.l. : EuroSys, Proc. ACM European Conf, Marzo 2007.
- JARA, JORGE. 2012. *Big Data & Web Intelligence.* [PDF] Paraguay : Universidad Católica "Nuestra Señora de la Asunción", 2012.
- JEONG, H. J., et al. 2012. *Anomaly teletraffic intrusion detection systems on hadoop-based platforms: A survey of some problems and solutions.* [ed.] IEEE. s.l. : 15th International Conference Network-Based Information Systems (NBiS), 2012. pp. 766-770.
- LEE, K., et al. 2012. *Parallel data processing with MapReduce: a survey.* s.l. : AcM sIGMoD Record, 2012. pp. 11-20.
- LEE, S., et al. 2013. *Real Time Micro-Blog Summarization based on Hadoop/HBase.* [ed.] IEEE. s.l. : 2013 IEEE/WIC/ACM International Joint Conferences Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013. pp. 46-49. Vol. 3.
- LI, N. F., et al. 2012. *Hadoop based data processing method and its application on Braille identification.* [ed.] IEEE. s.l. : 2012 Fifth International Conference Intelligent Networks and Intelligent Systems (ICINIS), 2012. pp. 329-332.
- MURRAY, D.G., et al. 2011. *CIEL: a universal execution engine for distributed data-flow computing.* s.l. : NSDI, Proc. USENIX Symp, Marzo 2011.
- O'DRISCOLL, AISLING, JURATE, DAUGELAITE AND ROY D., SLEATOR. 2013. *'Big data', Hadoop and cloud computing in genomics.* s.l. : Journal of biomedical informatics, 2013. pp. 774-781.
- OLSTON, C., et al. 2008. *Tomkins, Pig Latin: a not-so-foreign language for data processing.* [trans.] SIGMOD. s.l. : Proc. ACM Int'l Conf, Junio 2008.
- PADHY, R. P. 2012. *Big Data Processing with Hadoop-MapReduce in Cloud Systems.* s.l. : International Journal of Cloud Computing and Services Science (IJ-CLOSER), 2012. pp. 16-27. Vol. 2.

- PARRALES BRAVO, FRANKLIN RICARDO AND CALLE JARAMILLO, MARCO GENARO. 2010. *Tesis de pregrado: "Evaluación de MapReduce, Pig y Hive, sobre la plataforma Hadoop"*. [PDF] Guayaquil, Ecuador : s.n., 2010.
- QIAO, Y. Y., et al. 2013. *Offline traffic analysis system based on Hadoop*. s.l. : The Journal of China Universities of Posts and Telecommunications, 2013. pp. 97-103. Vol. 5.
- QIU, Z., LIN, Z. W. AND MA, Y. 2011. *Research of Hadoop-based data flow management system.* China : The Journal of China Universities of Posts and Telecommunications, 2011. pp. 164-168. Vol. 18.
- SARKAR, DEBARCHAN. 2013. *Microsoft SQL Server 2012 with Hadoop.* s.l. : Packt Publishing Ltd, 2013.
- TAYLOR, RONALD C. 2010. *An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics.* [PDF] s.l. : BMC bioinformatics 11, 2010. Vol. 12.
- THUSOO, A., et al. 2010. *Hive-a petabyte scale data warehouse using Hadoop.* s.l. : ICDE, Proc. IEEE Int'l Conf, Marzo 2010.
- VANCE, ASHLEE. 2009. *Hadoop, a Free Software Program, Finds Uses Beyond Search.* New York : New York Times, 2009.
- VEETIL, S. AND GAO, Q. 2014. *A Real-time Intrusion Detection System by Integrating Hadoop and Naive Bayes Classification.* 2014.
- VENTRAPRAGADA, A., et al. 2012. *Hadoop Compatible Framework for Discovering Network Topology and Detecting Hardware Failures.* [ed.] Third International Conference. IEEE. s.l. : Services in Emerging Markets (ICSEM), 2012. pp. 58-64.
- WHITE, TOM. 2009. *Hadoop: The Definitive Guide: The Definitive Guide.* [PDF] s.l. : O'Reilly Media, 2009.
- YU, Y., et al. 2008. *DryadLINQ: a system for general-purpose distributed data-parallel computing using a high-level language.* s.l. : OSDI, Proc. USENIX Symp, Diciembre 2008.
- ZAHARIA, M., et al. 2012. *Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing.* s.l. : NSDI, Proc. USENIX Symp, Abril 2012.