CAPÍTULO 1. Administración de Big Data en el cloud: *Retos, soluciones y herramientas*

Genoveva Vargas-Solar

Javier A. Espinosa-Oviedo

1.1 Introducción

Big Data es un término popular del que se habla en la prensa, la televisión, la academia y la industria. Tiene qué ver con la actividad humana que consiste en producir (grandes cantidades) de datos digitales y ha sido declarado tema prioritario por muchos gobiernos que ven en la ciencia de los datos un medio para activar la economía. Sin embargo, es aun difícil caracterizar el fenómeno Big Data porque diversos puntos de vista y disciplinas intentan definirlo. Es cierto que casi todo el mundo ve detrás de este término el perfil de un diluvio de datos que deben ser procesados y administrados (e.g., Peta 10^{15}, Exa 10^{18}, Zetta 10^{21}, Yotta 10^{24}, etc.). Algunas estimaciones aseguran que en 2020 habrá 35 Zeta bytes de datos digitales. La proliferación de datos introduce nuevos retos científicos, técnicos, éticos, jurídicos, y sociales. El consenso en vigor se materializa en el modelo 3Vs [1] para caracterizar "Big Data": *Volumen*,

Variedad (datos de diferentes tipos, estructurados o no) y *Velocidad* (flujos de datos producidos continuamente).

Big Data fuerza a ver los datos de manera matemática para luego poder asociarles un contexto. ¿Cómo pueden los investigadores usar herramientas estadísticas y tecnología computacional para identificar patrones de información comunes y representativos? ¿Cómo se pueden interpretar correlaciones significativas? ¿Cuál es el rol de formas tradicionales de modelos y teorías analíticas para accedes a los datos? Lo que realmente se quiere hacer es ver al conjunto de datos en maneras que le digan cosas y le respondan preguntas que no se le habían ocurrido [2][3]. Estas preguntas requieren infraestructuras bien adaptadas que puedan organizar, evaluar datos de forma eficiente y optimizar consultas y ejecutar algoritmos que requieren recursos de cálculo y de memoria importantes. La evolución de las arquitecturas hacia el cloud han causado que, los requerimientos de administración de datos deban ser revisitados [4][5]. En ese contexto es posible explotar el paralelismo para procesar datos e incrementar la disponibilidad y la fiabilidad del almacenamiento gracias a la duplicación. Organizar Big Data en soportes de persistencia (cache, memoria principal o disco), despachar procesos y producir y entregar resultados implica tener infraestructuras de administración de datos bien adaptadas que aun no están presentes en los sistemas existentes. Por ello es importante revisitar y proveer arquitecturas de sistemas e infraestructuras que estén adaptadas a Big Data. La clave es que estas infraestructuras y sistemas puedan esconder la complejidad de acceso y administración de Big Data y que provean las interfaces para configurarlas de acuerdo a los requerimientos aplicativos.

Concentrándose en los aspectos de administración de Big Data, el objetivo de este capítulo es primero, examinar los retos más importantes a través del modelo 3V y discutir sobre las arquitecturas de sistemas que ofrezcan soluciones conscientes del modelo 3V. Así, el resto del capítulo se organiza de la siguiente manera. La sección 1.2 caracteriza Big Data en términos del modelo 3V e insiste sobre los aspectos que dan lugar a nuevos retos en la administración de datos y en las características esperadas de los Big Data procesados. La sección 1.3 describe las plataformas de proceso de datos incluyendo sistemas paralelos, NoSQL y sistemas administradores de Big Data. La sección 1.3 introduce el ciclo de vida del proceso de Big Data. También describe posibles aplicaciones subrayando los requerimientos esperados cuando los datos son observados a grano fino. Finalmente la sección 1.5 concluye el capítulo y discute perspectivas en la administración de Big Data.

1.2 Las 3Vs de Big Data

Al mismo tiempo que se reconocen casos de éxito como el del Sloan Digital Sky Survey [6], las bases de datos sobre el genoma, la biblioteca del Congreso en Estados Unidos, quedan muchos retos técnicos que tienen que abordarse para explotar al máximo el potencial de Big Data. El *volumen* masivo de los datos es uno de los retos mayores, y es el que más fácilmente se reconoce. Sin embargo, hay otros retos impuestos por la *variedad* y la *velocidad* [7]. Por variedad se entiende heterogeneidad de tipos de datos y de sus representaciones e interpretaciones semánticas. Por velocidad se entiende la cadencia en la que los datos llegan y el tiempo en que estos flujos se deben procesar.

1.2.1 Variedad de los datos

La variedad de los datos ha sido un problema recurrente desde que ha sido posible digitalizar datos multimedia y desde que la producción de documentos se ha convertido en una práctica diaria en organizaciones y en el contexto doméstico. Continuamente ha habido trabajos para modelar datos y documentos digitales con diferentes formatos. Las representaciones de datos en bruto se han estandarizado (documentos PDF, JPEG, GIF para imágenes, MP3 para audio, etc.) y han sido asociados a modelos de datos para facilitar operaciones de manipulación y de recuperación de información.

En los años 1980's el modelo relacional (modelo de datos estructurado) se definió sobre una base teórica sólida basada en la noción de relación matemática y la lógica de primer orden. El modelo relacional distingue la noción de esquema (intención) y la extensión de la relación. La dicotomía esquema-datos es fundamental para la solución relacional. Las relaciones permiten la manipulación de datos estructurados independientemente de su representación física en una computadora. Dado que una relación es un conjunto de tuplas que sólo contienen valores atómicos, varias consecuencias deben ser consideradas. Primero, la relación en su carácter de conjunto, no puede contener datos repetidos; en una tupla ningún atributo puede tener un valor asociado de tipo conjunto, tabla u otra relación; no puede haber un valor sin identificador o faltante en una tupla. Estas restricciones dieron lugar a extensiones del modelo relacional puesto que se consideró poco expresivo. La primera aproximación fue relajar la primera forma normal de las relaciones y autorizar valores de atributos de tipo relación. Generalizando el uso de constructores de tipo producto cartesiano y conjunto, y luego agregando

listas y tablas se llega a la definición del modelo orientado a objetos que es más expresivo. Se hicieron intentos para definir DBMS orientados a objetos y estos sistemas fueron descritos en [8].

Los modelos estructurado y semi-estructurado (HTML, XML, JSON) son administrados por sistemas administradores de bases de datos y por motores de búsqueda que exploran el Web y en archivos locales de computadoras. La mayoría de los datos semi-estructurados son documentos electrónicos que surgieron con el Web. Nosotros consideramos que las bases de datos orientadas a objetos influenciaron el modelo JSON (JavaScript Object Notation), hoy usado como modelo de intercambio de datos en el Web. JSON es un formato general de datos de tipo texto derivado de la notación de objetos en JavaScript[1].

Más tarde, con la ola de sistemas NoSQL, han surgido otros modelos de datos que están siendo usados para tratar Big Data. El modelo de datos Llave-Valor (Key/value[2]) asocia a una llave con un valor de tipo simple o complejo. Los registros llave-valor son distribuidos por nodos instalados en redes de servidor usando por ejemplo una función de hash sobre la llave. El modelo llave-valor es simple y se usa para trabajar con datos simples como los de las bitácoras (logs), sesiones de usuarios, carritos de compra electrónica, que deben recuperarse rápidamente y donde hay poca manipulación de los elementos.

En el modelo orientado a documentos los datos son documentos semi-estructurados que corresponden a estructuras anidadas que pueden ser irregulares (similares a los lenguajes de marcado como XML)[3]. En el

---

[1] Diversos sistemas NoSQL como CouchDB [42] y MongoDB están basados en el modelo JSON (ver la sección NoSQL).

[2] Memcached, Redis y Riak son ejemplos de sistemas que usan este modelo.

[3] MongoDB y CouchDB son los sistemas más representativos que adoptan este modelo.

modelo orientado a columnas los datos son agrupados en columnas a diferencia de las relaciones tradicionales almacenadas en renglones. Cada elemento puede tener un número diferente de columnas (sin esquema fijo)[4]. Los modelos orientados a documentos y a columnas son usados para trabajar con bitácoras de eventos, blogs, análisis del Web (contadores en columnas). La manipulación de documentos se hace sobre el contenido con pocos requerimientos de atomicidad y aislamiento. Las familias orientadas a columnas son manipuladas en concurrencia y con alto desempeño.

Los modelos orientados a grafos ofrecen conceptos como nodo, arista y operaciones de navegación para representar respectivamente objetos y operaciones de consulta. Los nodos y las aristas pueden tener propiedades de la forma <key, value>[5]. Estos modelos son adaptados para datos altamente conectados y la recuperación de los datos se hace usando relaciones..

Las representaciones semánticas del contenido también aparecieron para apoyar el Web semántico (lenguajes de ontologías como OWL, modelos de anotaciones como RDF) y herramientas de búsqueda instaladas en computadoras y otros dispositivos. Para mejorar la escalabilidad la investigación actual [9] se están aplicando modelos paralelos a la ejecución de motores de razonamiento donde las ontologías son ligadas y los datos asociativos tienen miles de nodos y relaciones.

Esta diversidad es en cierto sentido el corazón de los retos (y de la integración de datos en general), dado que ya no es pertinente esperar trabajar con formatos de datos estandarizados y tener modelos de datos

---

[4] HBase, Cassandra e Hypertable son ejemplos de sistemas que adoptan este modelo.
[5] Neo4J es un ejemplo de sistema que adopta este modelo.

genéricos usados para representar contenido. En cambio, la tendencia es tener representaciones de datos que faciliten la manipulación eficaz, el almacenamiento y recuperación de datos distribuidos, heterogéneos y casi crudos. Los retos clave son (i) acoplar la construcción de bases de datos (limpieza de datos) con "time to market" reducidos a pesar del volumen y de la cadencia de producción de los datos; (ii) escoger el modelo de datos adecuado para una colección de datos dada, considerando sus características, el tipo de manipulación que se va a realizar y el tipo de operaciones que se aplicarán a los datos; (iii) las propiedades "no funcionales" garantizadas por el sistema como desempeño de las funciones de búsqueda dada la posibilidad de asociar índices simples y complejos a las colecciones de datos.

1.2.2 Volumen

Es cierto que lo que ve cualquiera de Big Data en primer lugar es el tamaño [10][11]. En la era del Internet, de las redes sociales, de los dispositivos móviles y de los sensores produciendo datos continuamente la noción de tamaño asociada a las colecciones de datos ha evolucionado rápidamente [7][12]. Hoy es normal para una persona de producir y administrar Terabytes de información en computadoras personales y en dispositivos con capacidad de almacenamiento [13]. Administrar grandes volúmenes de datos que crecen rápidamente ha sido un reto por muchas décadas [14][13][15]. En el pasado este reto era mitigado por procesadores cada vez más veloces, siguiendo la ley de Moore, para proveer recursos necesarios para tratar volúmenes de datos crecientes. Pero hay aquí un cambio fundamental en proceso: el volumen de datos está aumentando más rápido que los recursos de cálculo y las velocidades de los CPU no están evolucionado significativamente. El cómputo en la

nube integra cargas de trabajo múltiples con diferentes objetivos de desempeño (e.g. los servicios interactivos solicitan que los motores de procesamiento de datos regresen resultados con tiempos de respuesta específicos) [5]. Compartir recursos a este nivel en clúster grandes y caros requiere muevas formas de determinar cómo (i) correr y ejecutar tareas de procesamiento de datos para alcanzar metas efectivas en términos de costo para casa tarea; y (ii) tratar fallas del sistema que pueden ocurrir frecuentemente en clúster grandes (que deben enfrentar un crecimiento rápido de volúmenes de datos) [12][11].

1.2.3 Velocidad

El término *velocidad* (velocity) se refiere a la velocidad de generación de datos, o qué tan rápido los datos son generados y procesados para responde requerimientos específicos. Big Data es el resultado de una lectura continua y a grano fino del ambiente, la sociedad, las organizaciones, y los fenómenos naturales y sociales. Las observaciones se realizan en diferentes condiciones y con dispositivos diferentes. Por ello, Big Data son datos crudos producidos continuamente (i.e., flujos) que deben ser procesados para extraer información útil. (i.e., streams). Un flujo es una secuencia (a priori infinita) de tuplas (t_i, v_i) donde t_i es una estampilla y v_i es un valor simple o complejo. Hay dos posibles interpretaciones para un flujo y se puede considerar como un flujo de datos o un flujo de eventos. Hay dos aspectos a considerar: la movilidad de los datos y de los usuarios en un contexto espacio temporal; y la existencia de varios datos o flujos de eventos producidos continuamente que deben ser procesados bajo restricciones más o menos estrictas de tiempo real. Diversos trabajos han abordado el procesamiento de flujos y han propuesto Sistemas Administradores de Flujos (Stream Management

Systems – SMS) y Procesamiento de eventos complejos (Complex Event Processing CEP). Brevemente, la diferencia entre ambos radica en la semántica asociada a los datos. Mientras los SMS pueden calcular la temperatura media durante el día, los CEP prueban si la temperatura de un cuarto no es más alta que una cota y si es el caso, reaccionar en consecuencia.

Hay dos razones para considerar el procesamiento de flujos. La primera es cuando los datos de entrada llegan demasiado rápido como para almacenarlos completamente. Una estrategia pragmática es almacenar los necesarios para poder efectuar análisis con cierta precisión. No es solamente la velocidad de los datos de entrada, es posible recibir flujos e ir almacenando históricos para analizarlos en batch. Lo importante es mantener la velocidad entre la llegada de los datos y la producción de una retroalimentación que pueda ser transformada en decisiones. La segunda razón es considerar flujos para aplicaciones que desean tener respuestas inmediatas ante los datos de entrada. Gracias al incremento de aplicaciones móviles y los juegos en línea esta situación se presenta muy seguido.

1.3 Plataformas de procesamiento de Big Data

Esquemáticamente, las bases de datos (relacionales) y la administración de datos del Web han evolucionado en paralelo. Por un lado, está el éxito del relacional, de los Sistemas Administradores de Bases de Datos Paralelos con SQL. Por otro, están las bases de datos distribuidas, específicas para administrar los datos del Web desarrolladas usando el modelo Mal-Reduce (ver sistemas NoSQL). La solución clásica de bases de datos relacionales parecía poco adaptada para administrar los datos del Web porque requería fases sucesivas de diseño (ETL: Extract

Transform, Load) consideradas muy rígidas con respecto a (i) Fuentes de datos no estructuradas (e.g. documentos); (ii) aplicaciones Web específicas; (iii) alta disponibilidad y escalabilidad para un número creciente de usuarios. Además, los DBMS paralelos son sistemas caros con pocas propuestas libres como MySQL-C. Las plataformas de procesamiento de Big Data ofrecen estrategias con buen desempeño a pesar del volumen elevado de datos y los algoritmos de procesamiento glotones. Hoy estas estrategias empiezan a converger y a aprovechar lo mejor de los dos mundos. Desde el punto de vista de las aplicaciones, la pregunta es cómo combinarlas para explotar Big Data?

Tres arquitecturas han emergido para atacar el análisis de Big Data (analytics): los sistemas NoSQL, Map-Reduce/Hadoop, y DBMS relacionales extendidos. En un principio estas arquitecturas fueron implementadas como sistemas independientes. Hoy, la tendencia es desarrollar combinaciones innovadoras de las tres. CISCO y Oracle proponen hoy los primeros sistemas comerciales NoSQL instalados en Cloudera para alcanzar grande volúmenes de Big Data estructurados y no estructurados en tiempo real [16][17].

1.3.1 Sistemas NoSql

Al final de los años 1990's con el desarrollo del Web y de compañías como Google, Yahoo!, Facebook o Amazon, ofrecer soluciones de administración de datos se hizo crucial. Las bases de datos SQL monolíticas construidas para sistemas transaccionales (On Line Transaction Processing OLTP) y sistemas de análisis (On Line Analysis Processing OLAP) fueron rechazadas por ser demasiado caras, complejas y/o no suficientemente rápidas. El movimiento « Not Only

SQL »[6] nacía [18][19]. Por ejemplo, Google y Amazon desarrollaron sus propias propuestas (BigTable y Dynamo, respectivamente) para responder a requerimientos y luego la comunidad de software libre Apache open-source creó clones correspondientes como HBase y Cassandra, dos de las sistemas llave – valor más populares hoy en día. Han surgido muchos sistemas con sus propias implementaciones de diferentes modelos de datos y funciones de manipulación específicas. El de bases da datos NoSQL[7] provee una lista exhaustiva de los sistemas existentes.

Los objetivos principales de los sistemas NoSQL son incrementar la escalabilidad y la extensibilidad usando hardware clásico. También asegurar fiabilidad, tolerancia a fallas y buen desempeño, a pesar de que el número de consultas crezca. La característica común de estos sistemas es que permiten la manipulación de datos sin tener que definir un esquema, por lo tanto los datos son principalmente semi-estructurados. De este modo, evitan la definición de esquema y las fases de carga (loading) y limpieza de datos (cleaning). Las características de la arquitectura de estos sistemas son: extensibilidad horizontal para ejecutar operaciones simple sobre una multitud de servidores; particionamiento y duplicación de datos en varios servidores (data sharding); interfaces de bajo nivel para acceder a estos sistemas; modelo de acceso concurrente de datos más flexible que el de transacciones ACID (atomicity, consistency, isolation and durability); indexación distribuida y almacenamiento en memoria; evolución simple de estructuras de datos [5]. Estos sistemas ofrecen funciones Create Read Update Delete

[7] El sitio http://nosql-database.org/ reúne 150 sistemas con diferentes modelos y funcionalidades.

(CRUD) adaptadas para manipular eficazmente estructuras de datos de acuerdo al modelo de datos subyacente.

1.3.2 Procesamiento de datos paralelo usando map-reduce

Map-Reduce es un modelo de programación (MR [20]) para desarrollar procesamiento de datos definiendo funciones Map y Reduce inspiradas de la programación funcional. Es posible paralelizar el procesamiento de datos particionando y duplicando los datos en N máquinas, asumiendo que los datos son administrados por un sistema de archivos distribuidos. La función Map implementa una operación basada en la estrategia "divide y vencerás". La idea es dividir un conjunto de datos modelado como tuplas <key, value> en conjuntos más pequeños que pueden ser procesados de manera independiente. Los elementos llave valor pueden ser de tipo atómico o complejo. La función Reduce combina los resultados de las funciones Map y los reduce aplicando funciones de agregación o de sumatoria. Un programador debe escribir estas funciones pero pueden apoyarse en una plataforma que ofrece herramientas para coordinar las tareas que ejecutan estas funciones.

Las plataformas de ejecución MR consisten en general de un sistema distribuido de archivos para almacenar los datos de entrada y salida y un administrador de procesos que coordina la ejecución paralela de procesos. La primera propuesta, GFS-Google File System [21] apareció en 2003. La arquitectura consiste en millones de máquinas interconectadas bajo un modelo "share nothing" para incrementar la disponibilidad de datos y reducir el tiempo de respuesta; asegurar la fiabilidad del sistema duplicando los servidores y los datos, explotando el paralelismo inherente de la arquitectura. Este sistema procesa URL, datos del usuario (los resultados de consultas más recientes) e

información geográfica (la localización de puntos de interés de una región, las boutiques predilectas, hoteles e imágenes satélite de la calle). El factor de escala es alto con millones de URL, diferentes versiones de páginas, millones de usuarios, millones de consultas por Segundo, cientos de TB para las fotos satelitales. GFS ofrece una vista de archivos de datos conocida: "byte-stream-based". Esta vista es particionada sobre cientos y aun miles de nodos en un clúster [22]. GFS es asociado al modelo de programación Map-Reduce para permitir a los programadores procesar Big Data escribiendo dos funciones definidas por el usuario: map y reduce. Google construye aplicaciones distribuidos con las funciones MR que son ejecutadas por Hadoop [23] una plataforma abierta de ejecución de map reduce.

Con respecto a la arquitectura, los nodos Hadoop son de tipo diferente y todos tienen HDFS (Highly Distributed File System versión abierta de GFS [23]) y capas MR. HDFS está basados en una arquitectura maestro – esclavo que consiste en un maestro, el "name node" (una instancia por clúster), que administra metadatos para los datos de cada nodo; el nodo de respaldo del "name node"; el nodo de datos, una instancia instalada en cada máquina del clúster administra el almacenamiento en disco. De manera similar, los componentes MR son distribuidos y administran "trabajos": con el "name node" hay un "job tracker", y con el nodo de datos, un , " task tracker" ejecutando las funciones map y reduce en datos locales. Esta arquitectura escala a millones de nodos y puede administrar PB de datos. Los bloques de datos son suficientemente grandes para evitar sobrecargar el acceso a datos por disco. Los datos están almacenados en cada bloque en tres copias para asegurar la tolerancia a fallas, en lugar de usar discos espejo (e.g. RAID). La

localización de datos es transparente para las capas superiores y esto da la posibilidad de hacer optimizaciones como las implementadas por los RDBMS.

Muchos trabajos se han concentrado en la implementación de operadores relacionales usando MR [24][25][26]. Por ejemplo, la selección corresponde a un filtro para Map. La proyección se expresa también con una simple función Map. Para el join, Map [26][27][28] provee tuplas con el atributo para join como clave y el reduce realiza el join. Por lo tanto, una consulta relacional puede implementarse en un conjunto de trabajos MR. Este tipo de solución es ofrecido por Hive [29] o Pig [30] construidos sobre Hadoop y que proveen lenguajes de tipo SQL para programar el procesamiento de datos en sus ambientes de ejecución asociados.

1.3.3 Sistemas administradores de Big Data

Los RDBMS paralelos usan arquitecturas "shared-nothing" [31] donde los datos son distribuidos en un clúster usando una estrategia de particionamiento, en general basada en hash pero también usando particionamiento por rangos o de manera aleatoria (range o random partitioning). Las consultas son procesadas usando técnicas paralelas, o divide y vencerás basadas en hash [32]. Las arquitecturas paralelas atacan dos factores esperando alcanzar un comportamiento lineal: (1) aceleración lineal (linear speed up) usar el doble de recursos para que el programa corra dos veces más rápido, por ejemplo procesar 10TB usando 8 nodos en lugar de 4 y así dividir el tiempo de ejecución entre dos; (2) usar el doble de recursos para procesar una base de datos dos veces más grande en el mismo tiempo de respuestas, por ejemplos procesar 10TB en 4 nodos y 4 discos y procesar 10TB usando 8 nodos y

8 discos. En realidad la aceleración linear es una visión teórica porque otros factores deben ser considerados: el tiempo de interconexión entre procesadores, balance de carga entre servidores distribuidos.

[33] compara RDBMS con las plataformas Hadoop para realizar tareas de análisis de datos. RDBMS parecen alcanzar buen desempeño a condición de hacer una configuración adecuada, similar al de las soluciones Hadoop o a veces mejor en algunos casos. Estos sistemas son robustos y estables pero caros (no hay sistemas abiertos disponibles). In cambio, las plataformas Hadoop son accesibles y requieren menos esfuerzo de configuración pero no ofrecen transparencia y requieren gran esfuerzo de diseño de algoritmos para implementar operaciones binarias como los join. El desarrollo de aplicaciones es siempre ad-hoc cuando se usa Hadoop porque no es una solución general como los RDBMS. Por ello, [33] cree que hay necesidad de diseñar plataformas altamente escalables para la nueva generación de almacenamiento de información, búsqueda y análisis: un Big Data Management System (BDMS). Esto puede hacerse combinando y extendiendo ideas de la administración de datos semi-estructurados, bases de datos paralelas, y la primera generación de plataformas para cálculo intensivo de datos (en particular Hadoop/ HDFS), ASTERIX[8] tiene como objetivo permitir el acceso, carga, almacenamiento, indexación, consulta y análisis y publicación de grandes cantidades de datos semi-estructurados. El diseño del BDMS ASTERIX está bien adaptado para manipular casos de uso que van de colecciones de datos rígidas como las relacionales –cuyas estructuras son bien comprendidas y poco variables- hasta datos

[8] http://asterix.ics.uci.edu

complejos y más flexibles donde se planea poco por anticipado y las instancias son variables y auto descritas.

1.3.4 Discusión

Se han hecho diferentes comparaciones entre los RDBMS, NoSQL y Hadoop [34][35][36]. Los sistemas NoSQL parecen ofrecer indexaciones demasiado simple in comparación con los sistemas RDBMS y animan la programación de consultas usando el modelo map reduce en contraste con la solución declarativa y optimizada de los RDBMS. En cambio, Hadoop puede procesar datos directamente sin definir un esquema. Hadoop y NoSQL no proveen soluciones generales, y esto requiere mucha experiencia y esfuerzo de programación para implementar soluciones. En todos los casos, la extensibilidad y la tolerancia a fallas son aspectos abordados por todos los sistemas de acuerdo a sus características. En los últimos meses se ha propuesto Google Cloud Dataflow[9] para hacer accesible datos y análisis a todo mundo [37] a través de un procesamiento de datos en pipeline basado en un modelo centrado en datos accesible a todos. La solución permite observar la ejecución, obtener información sobre los datos, y sin tener que instalar clúster, configurar parámetros y optimizar el uso de recursos.

1.4 Ciclo de vida de Big Data y aplicaciones

"*Big Data es un cambio de paradigma en la manera en la que pensamos sobre los datos, dónde los recolectamos, cómo los analizamos y cómo monetizamos la interpretación de su análisis*," dice Kimball [38]. Por ello, es importante analizar el ciclo de vida de Big Data y subrayar los retos en cada fase. No solamente importan los resultados del análisis, en el caso de Big Data

[9] http://googlecloudplatform.blogspot.fr/2014/06/sneak-peek-google-cloud-dataflow-a-cloud-native-data-processing-service.html

todas las fases del proceso introducen retos: ¿qué guardar y qué tirar? ¿En qué condiciones se adquieren y se limpian los datos? ¿Qué es volátil y qué es persistencia, y por cuánto tiempo? Los resultados pueden y deben usarse para futuros análisis?

1.4.1 Adquisición de datos

Los datos son recolectados a partir de una fuente que los genera. La mayoría de estos datos no son de interés, y pueden ser filtrados y comprimidos en diferentes órdenes de magnitud [12][10][39][11]. Un reto es definir estos filtros de manera que no se descarte información útil. La adquisición de datos requiere investigación in ciencia de reducción de datos que puedan procesar inteligentemente datos en bruto hacia un tamaños que sus usuarios puedan manejar sin perderse. Además, se requieren técnicas de análisis en línea para procesar flujos de datos de manera dinámica, dado que no es posible ni viable almacenarlos primero y luego reducirlos y consolidarlos.

Además es necesario generar automáticamente los buenos metadatos que describan qué datos son archivados y cómo son archivados y medidos. Otro aspecto importante es la proveniencia de los datos. Archivar información a cerca de los datos y su nacimiento no es útil a menos que la información pueda ser interpretada y realizada a través de un pipeline de análisis de datos. Entonces, se requiere hacer investigación para generar metadatos adecuados y para diseñar sistemas de datos que puedan administrar la proveniencia de los datos y des sus metadatos con pipelines de análisis.

1.4.2 Limpieza de Datos

Dada la heterogeneidad de los datos, no es suficiente archivarlos. Se requiere expresar sus diferencias de estructura y semántica en formas que

sean comprensibles por una computadora, y que puedan ser resueltas automáticamente. Hay mucho trabajo a hacer para integrar los datos que pueden proveer alguna respuesta; y se requiere más trabajo para resolver diferencias de manera automática y libre de errores. Usualmente, hay diferentes maneras de almacenar la misma información, cada una con sus ventajas y desventajas. Debemos empoderar a otros profesionales, como científicos de diferentes disciplinas, para diseñar bases de datos efectivas, a través de herramientas que les ayuden en el proceso de diseño o predecir el proceso de diseño desarrollando técnicas que permiten usar bases de datos a pesar de no haber sido diseñadas a través de un proceso "inteligente".

1.4.3 Análisis y minería de datos

Los métodos para consultar y hacer minería de Big Data son fundamentalmente diferentes del análisis estadístico tradicional en muestras pequeñas. Frecuentemente, Big Data tiene ruido, dinámico y heterogéneo, inter-relacionado y poco confiable. Sin embargo, aun Big Data con ruido podría ser más valioso que pequeñas muestras porque la estadística general obtenida de patrones frecuentes y el análisis de correlaciones sobre valúan fluctuaciones individuales y pasan por alto patrones y conocimiento más fiables.

Big Data está habilitando la nueva generación de análisis de datos con respuestas en tiempo real. En el futuro, las consultas sobre Big Data serán generadas automáticamente para crear contenido en sitios Web, para poblar listas y recomendaciones, y para proveer análisis ad hoc de valor sobre datos para saber si se descartan o permanecen [40]. Escalar el proceso de consultas complejas a terabytes habilitando respuestas

interactivas en tiempos de respuesta razonables es un problema abierto de investigación hoy en día.

Los pipelines de análisis implican múltiples pasos e hipótesis inherentes. Estudiando cuál es la mejor manera de capturar, almacenar e interrogar la proveniencia, junto con técnicas para capturar metadatos adecuados hacen posible crear una infraestructura que dote a los usuarios de posibilidades para interpretar resultados de análisis, repetirlos con diferentes hipótesis y parámetros o colecciones de datos. Frecuentemente, es la visualización de datos que permite comprender el impacto de Big Data. La visualización puede ayudar a producir y comprender Big Data. Algunas herramientas que se usan para descubrir cosas en grandes masas de datos son Visual.ly, Tableau, Vizify, D3.js, R.

1.4.4 Aplicaciones para Big Data

Actualmente, las organizaciones y los investigadores de diferentes áreas ven un tremendo potencial en el valor que puede ser generado al almacenar información digital, referida popularmente como "Big Data," y poniéndola a disposición para hacer análisis, consulta y otras actividades [41]: incrementar la efectividad de sus esfuerzos de mercadotecnia y servicio a cliente; análisis de sentimientos; seguir el progreso de epidemias; estudiar los tweets y las redes sociales para entender la manera en que la información de diferentes tipos se disemina y/o cómo puede ser usada para el bien de las personas. En una amplia gama de aplicaciones, los datos están siendo recolectados de manera nunca antes vista. Las decisiones que antes se tomaban casi adivinando, o creando modelos con mucha dificultad, pueden hacer basados en los mismos datos.

El Sloan Digital Sky Survey [6] se ha convertido en un recurso importante para los astrónomos del mundo. Le dominio de la astronomía ha pasado de tomar fotografías del cielo por los astrónomos a bases de datos de imágenes listas para que ser analizadas y encontrar objetos interesantes e identificar fenómenos. En las ciencias biológicas, hay una tradición bien establecida de depositar datos científicos en repositorios públicos, y de crear base de datos públicas para uso de otros científicos. De hecho, hay una disciplina entera, la bioinformática que se ocupa al análisis de estos datos. A medida que la tecnología avanza, particularmente con la llegada de la nueva generación de secuenciamiento, el tamaño y el número de colecciones de datos experimentales disponibles se ha incrementado de manera exponencial. En el contexto de la educación imaginemos un mundo donde se tenga acceso a bases de datos con medidas detalladas del desempeño académico de cada estudiante. Estos datos podrían usarse para diseñar estrategias de educación más eficaces, empezando por la lectura, la escritura, las matemáticas hasta cursos avanzados universitarios. Estamos lejos de tener acceso a estos datos, pero hay tendencias en esta dirección. En particular hay una tendencia para instalar de manera masiva actividades educativas en el Web para generar datos detallados del desempeño de los estudiantes.

Las compañías puede cuantificar aspectos de comportamiento humano que no eran accesibles antes: redes sociales, flujos de noticias, SmartGrid, son maneras de medir "conversación", interés y actividad. Herramientas de Big Data basadas en aprendizaje automático pueden identificar a quien seguir para entender la manera en que los eventos y las historias en las noticias resuenan [9].

1.5 Conclusiones y perspectivas

La ola Big Data tiene impacto en las estrategias existentes para administrar datos. Primero, los datos son instalados en arquitecturas distribuidas y paralelas. Segundo, afecta el tipo y la precisión de los modelas que pueden ser derivados. Big Data no es solamente la talla, pero implica la producción continua de información incluyendo la recolección, limpieza, almacenamiento y análisis. Cada fase del procesamiento implica el uso de algoritmos glotones, estadística, modelos que debe escalar y ser realizados de manera eficaz. Hacer escalar la administración de los datos depende del tipo de aplicaciones que usan parcial o totalmente los resultados del análisis: tareas críticas requieren procesamiento en línea, mientras que otras tareas más analíticas pueden aceptar tiempos de producción más largos, aplicar diferentes algoritmos producir resultados con diferente precisión y fiabilidad, etc. (i.e., veracity).

Este proceso complejo hace evidente la necesidad de tener técnicas de administración de datos más eficientes que puedan escalar y adaptarse a la cadena de procesamiento tradicional: almacenamiento, administración de memoria (caching), filtrado y limpieza. Nuevas tecnologías de almacenamiento no tienen el mismo desempeño entre I/O secuencial o random. Esto requiere pensar de nuevo en cómo diseñar sistemas de almacenamiento y todo aspecto del procesamiento de datos, incluyendo el proceso de consultas, la calendarización de consultas, el diseño de bases de datos, métodos de control de concurrencia y métodos de recuperación (recovery). También es importante seguir el tipo de procesos aplicados a los datos y las condiciones en que se realizaron, porque los procesos deben poder ser reproducidos por ejemplo en el

caso de aplicaciones científicas. Estas técnicas deben guiarse por las características del hardware, memoria, recursos de almacenamiento y de cálculo usados y la manera en que son consumidos. Particularmente, en las arquitecturas cloud guiadas por modelos económicos de tipo "pay as you go" (paga lo que consume).

Los futuros BDMS deberán tomar en cuenta los requerimientos de análisis de las aplicaciones, las características de los datos (cadencia de producción, formatos, qué tan críticos son, tamaño, intervalo de validez), los recursos requeridos y el costo económico para proveer modelos analíticos útiles que pueden apoyar mejor la toma de decisiones, la recomendación, el descubrimiento de conocimiento, y las tareas de ciencia de datos.

Referencias

1. D. Laney, "3D Data Management: Controlling Data Volume, Velocity & Variety," *META Gr.*, no. February 2001, 2001.
2. B. Grinter, "A Big Data confession," *Interactions*, vol. 20, no. 4, Jul. 2013.
3. A. Halevy, P. Norvig, and F. Pereira, "The Unreasonable Effectiveness of Data," *IEEE Intell. Syst.*, vol. 24, no. 2, Mar. 2009.
4. S. Chaudhuri, "What next?: a half-dozen data management research goals for Big Data and the cloud," in *Proc. of the 31st PODS Symposium on Principles of Database Systems (PODS '12)*, 2012.
5. S. Mohammad, S. Breß, and E. Schallehn, "Cloud Data Management : A Short Overview and Comparison of Current Approaches," in *24th GI-Workshop on Foundations of Databases*, 2012.
6. A. S. Szalay, J. Gray, A. R. Thakar, P. Z. Kunszt, T. Malik, J. Raddick, C. Stoughton, and J. VandenBerg, "The SDSS skyserver: public access to the sloan digital sky server data," in *Proc. of the 2002 ACM SIGMOD Int. Conference on Management of data (SIGMOD'02)*, 2002, vol. 1, no. 1.
7. H. A.K and Madam Prabhu D., "No problem with Big Data. What do you mean by Big?," *Informatics*, no. October, pp. 30–32, 2012.

8. M. Atkinson, D. Dewitt, D. Maier, K. Dittrich, and S. Zdonik, "The Object-Oriented Database System Manifesto r," pp. 1–17.
9. J. Langford, "Parallel machine learning on Big Data," *XRDS Crossroads, ACM Mag. Students*, vol. 19, no. 1, Sep. 2012.
10. P. Lyman, H. R. Varian, J. Dunn, A. Strygin, and K. Swearingen, "How Much Information?," *Count. Numbers*, vol. 6, no. 2, 2000.
11. P. C. Zikopoulos, C. Eaton, D. DeRoos, T. Deutsch, and G. Lapis, *Understanding Big Data*. McGraw-Hill, 2011.
12. R. Apps and R. Scale, *Big Data Sourcebook*. 2014.
13. M. L. Kersten, S. Idreos, S. Manegold, and E. Liarou, "The Researcher's Guide to the Data Deluge : Querying a Scientific Database in Just a Few Seconds," *Proc. VLDB Endow.*, vol. 4, no. 12, 2011.
14. L. Hoffmann, "Looking back at Big Data," *Commun. ACM*, vol. 56, no. 4, Apr. 2013.
15. A. Kleiner, M. Jordan, T. Ameet, and S. Purnamrita, "The Big Data Bootstrap," in *Proc. of the 29th Int. Conference on Machine Learning*, 2012.
16. Oracle, "Oracle NoSQL Database," *White Pap.*, no. September, 2011.
17. P. Zikopoulos, D. DeRoos, K. Parasuraman, T. Deutsch, J. Giles, and D. Corrigan, *Harness the Power of Big Data*. McGraw-Hill, 2013.
18. R. Cattell, "Scalable SQL and NoSQL data stores," *SIGMOD Rec.*, vol. 39, no. 4, May 2011.
19. P. J. Sadalage and M. Fowler, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot*. 2012.
20. J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, Jan. 2008.
21. S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. of the 19th ACM SOSP Symposium on Operating Systems Principles (SOSP'03)*, 2003, vol. 37, no. 5.
22. M. Cafarella, A. Halevy, W. Hsieh, S. Muthukrishnan, R. Bayardo, O. Benjelloun, V. Ganapathy, Y. Matias, R. Pike, and R. Srikant, "Data Management Projects at Google," *SIGMOD Rec.*, vol. 37, no. 1, 2008.
23. D. Borthakur, "HDFS Architecture Guide," *Apache Rep.*, pp. 1–13, 2010.
24. J. Dittrich and J.-A. Quiané-Ruiz, "Efficient Big Data processing in Hadoop MapReduce," *Proc. VLDB Endow.*, vol. 5, no. 12, Aug. 2012.
25. F. Li, B. C. Ooi, M. T. Özsu, and S. Wu, "Distributed data management using MapReduce," *ACM Comput. Surv.*, vol. 46, no. 3, Feb. 2014.
26. A. Okcan and M. Riedewald, "Processing theta-joins using MapReduce," in *Proc. of the 2011 ACM SIGMOD Int. Conference on Management of Data (SIGMOD '11)*, 2011.

27. J. D. Ullman, "Designing good MapReduce algorithms," *XRDS Crossroads, ACM Mag. Students*, vol. 19, no. 1, Sep. 2012.
28. J. Chandar, "Join Algorithms using Map / Reduce," *Slides*, 2010.
29. A. Thusoo, J. Sen Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, and R. Murthy, "Hive - a petabyte scale data warehouse using Hadoop," in *Proc. of the 26th ICDE Int. Conference on Data Engineering (ICDE'10)*, 2010.
30. C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, "Pig latin: a not-so-foreign language for data processing," in *Proc. of the 2008 ACM SIGMOD Int. Conference on Management of data (SIGMOD'08)*, 2008.
31. P. Valduriez, "Parallel Techniques for Big Data Outline of the Talk," *Slides*, 2013.
32. V. R. Borkar, M. J. Carey, and C. Li, "Big Data platforms: What's next?," *XRDS Crossroads, ACM Mag. Students*, vol. 19, no. 1, Sep. 2012.
33. M. Stonebraker, D. Abadi, and D. DeWitt, "MapReduce and parallel DBMSs: friends or foes?," *Commun. …*, 2010.
34. 451 Research, "Mysql vs. nosql and newsql: 2011-2015," *Report*, no. May 2012, pp. 2011–2015, 2015.
35. C. Mohan, "History repeats itself: sensible and NonsenSQL aspects of the NoSQL hoopla," in *Proc. of the 16th EDBT Int. Conference on Extending Database Technology (EDBT'13)*, 2013.
36. V. Borkar, M. J. Carey, and C. Li, "Inside ' Big Data Management ': Ogres , Onions , or Parfaits ?," in *EDBT '12 Proceedings of the 15th International Conference on Extending Database Technology*, 2012, pp. 3–14.
37. K. Ren, Y. Kwon, M. Balazinska, and B. Howe, "Hadoop's adolescence: an analysis of Hadoop usage in scientific workloads," *Proc. VLDB Endow.*, vol. 6, no. 10, Aug. 2013.
38. T. C. Economics, "For Big Data Analytics There ' s No Such Thing as Too Big," no. March, pp. 1–20, 2012.
39. Tableau, "What'ss the Big Deal About Big Data ?," *report*, no. January, 2013.
40. S. Idreos, I. Alagiannis, R. Johnson, and A. Ailamaki, "Here are my Data Files. Here are my Queries. Where are my Results?," in *Proc. of the 5th CIDR Biennial Conference on Innovative Data Systems Research (CIDR'11)*, 2011.
41. K. Michael and K. Miller, "Big Data: New opportunities and new challenges," *Computer (Long. Beach. Calif).*, vol. 46, no. 6, 2013.
42. CouchBase, "NoSQL Database Technology," *Report*.