Estructuras de Datos y Algoritmos Eficientes para Búsquedas Web y Procesamiento de Grandes Datos

Gabriel H. Tolosa[1], Pablo Lavallén[1], Tomás Delvechio[1], Agustín Marrone[1], Francisco Tonin Monzon[1], Federico Radeljak[1], Esteban A. Ríssola[1,3] y Esteban Feuerstein[2]
{tolosoft, plavallen, tdelvechio, eamarrone, ftonin, fradeljak}@unlu.edu.ar
esteban.andres.rissola@usi.ch; efeurest@dc.uba.ar

[1]Departamento de Ciencias Básicas, Universidad Nacional de Luján
[2]Departamento de Computación, FCEyN, Universidad de Buenos Aires
[3]Facoltà di Scienze Informatiche, Università della Svizzera Italiana

Resumen

Desde hace ya un par de décadas, la cantidad de información, las aplicaciones y el número de usuarios digitales crece exponencialmente, formando un ecosistema en el cual se intenta explotar la masividad de los datos y presentan a la comunidad científico/tecnológica nuevos desafíos.

Por ejemplo, los motores de búsqueda para la web son aplicaciones que procesan miles de millones de documentos para responder consultas de los usuarios. Esto genera nuevas necesidades de almacenamiento, procesamiento y búsquedas, expandiendo los límites del trabajo en una sola máquina y unos pocos algoritmos. Las consultas deben responderse en milisegundos, con resultados relevantes, sobre un escenario altamente heterogéneo.

Además, el área de "Big Data", que se aplica a cúmulos de datos que no pueden ser procesados y/o analizados de forma eficaz y eficiente utilizando técnicas tradicionales, aporta nuevos enfoques que complementan la idea anterior.

En este proyecto se estudian, proponen, diseñan y evalúan estructuras de datos y algoritmos para soportar búsquedas de escala web y/o analizar datos masivos de forma eficiente.

Palabras clave: algoritmos eficientes, motores de búsqueda, estructuras de datos, grandes datos.

Contexto

Esta presentación se encuentra enmarcada en los proyectos de investigación "Algoritmos Eficientes y Minería Web para Recuperación de Información a Gran Escala" del Departamento de Ciencias Básicas (UNLu) y "Modelos y herramientas algorítmicas avanzadas para redes y datos masivos"del Departamento de Computación de la Facultad de Ciencias Exactas y Naturales (UBA).

Introducción

Desde hace ya un par de décadas, la información digital está creciendo exponencialmente. Algunas estimaciones indican que desde 2005 hacia 2020 el universo digital multiplicará por un factor de 300 su crecimiento, duplicando su tamaño aproximadamente cada 2 años [15]. A modo de ejemplo, en Twitter[1] se publican más de 250.000 tweets por segundo y en Facebook se generan más de 40.000 posts (también por segundo), que equivalen a más de 300 GB. El también creciente número de usuarios en el mundo digital, en particular, en redes sociales utilizando dispositivos móviles [10, 11], conforman un ecosistema en el cual se desarrollan nuevas actividades que intentan explotar la masividad de los datos y presentan a la comunidad científico/tecnológica nuevos desafíos.

Por esto, existe la necesidad permanente de nuevos enfoques, estrategias y técnicas que, haciendo uso de las herramientas computacionales adecuadas, ayuden a resolver de forma eficiente los problemas que aparecen continuamente. El ejemplo más saliente a mencionar es el caso de los motores de búsqueda para la web, herramientas que se basan en un necesidad de información del usuario e intentan satisfacerla. En esta área, se aplican técnicas de re-

[1]https://blog.hootsuite.com/twitter-statistics/

cuperación[2] sobre una porción del espacio web que han recorrido y procesado [2].

El tamaño y complejidad de la web[3] genera nuevas necesidades de almacenamiento, procesamiento y búsquedas, expandiendo los límites del trabajo en una sola máquina y unos pocos algoritmos. Se requiere hoy de procesamiento distribuido, paralelo y altamente eficiente. Las consultas deben ser respondidas en pequeñas fracciones de tiempo (milisegundos) y deben ofrecer resultados relevantes sobre un escenario altamente heterogéneo, en el cual aparecen oportunidades únicas para avances científico/tecnológicos en áreas como algoritmos, estructuras de datos, sistemas distribuidos y procesamiento de datos a gran escala [5].

Como parte también de esta realidad, uno de los conceptos que ha ganado espacio en los últimos años es el de "Big Data". En términos generales, representa la idea de la masividad de datos, su velocidad de aparición y variedad de fuentes (estructurados, semi-estructurados y no estructurados) que se generan y requieren ser procesados para diferentes tareas. En particular, no son viables para el almacenamiento en sistemas de gestión de bases de datos tradicionales y, menos aún, en un único equipo de cómputo. Por lo tanto, esta idea aplica a cúmulos de datos que no pueden ser procesados y/o analizados de forma eficaz y eficiente utilizando técnicas y herramientas tradicionales [19], por lo que se requieren nuevos enfoques. En la actualidad, muchas aplicaciones importantes requieren de estrategias derivadas del concepto de Big Data para el procesamiento de sus datos. Por ejemplo, las redes sociales [20], los servicios de *streaming* [34], la genómica [24], la meteorología [16], entre otros.

Para el análisis de grandes volúmenes de datos, consumidos por millones de usuarios en casi todos los ámbitos de la actividad humana se emplean, entre otras, técnicas del área de *machine learning* como reconocimiento de patrones en textos, análisis estadístico, visualización, agrupamientos, redes neuronales, etc. [25], las cuales permiten convertir datos en información útil. En algunos casos, los problemas aumentan su complejidad ya que en sus procesos ingestan grandes volúmenes de datos de fuentes diversas en tiempo real [26].

[2]En sentido amplio, ya que se trata de múltiples fases de procesamiento como búsquedas y ranking, entre otras.

[3]Por ejemplo, Google indexa unos 45.000 millones de documentos de acuerdo a http://www.worldwidewebsize.com

Líneas de I+D

Este proyecto sigue con las líneas de I+D del grupo, las cuales se dividen en dos grupos principales, aunque con temas en la intersección de ambos. En el primer caso, se trata de mejorar la eficiencia en la recuperación de información de gran escala, rediseñando los algoritmos internos y las estructuras de datos usadas. En el segundo, se proponen y evalúan arquitecturas de procesamiento de grandes datos para problemas diversos. En ambos casos, existen oportunidades de investigación en temas poco explorados por la comunidad científica. En particular, las líneas de I+D principales son:

a. Estructuras de Datos y Algoritmos para Búsquedas

Los sistemas de búsqueda en texto utilizan como estructura de datos básica el índice invertido, formado por un vocabulario (V) con todos los posibles términos y un conjunto de *posting lists* (L) con información acerca de los documentos donde aparece cada término junto con información usada para el ranking (por ejemplo, la frecuencia de aparición de un término t_i en un documento d_j).

1. Caching: Una de la técnicas más utilizadas para aumentar la performance en los sistemas de búsqueda a gran escala es el *caching*, que se basa en la idea fundamental de almacenar en una memoria de rápido acceso los ítems que volverán a aparecer en un futuro cercano. En un motor de búsquedas esto se realiza en varios niveles, por ejemplo: resultados [23], posting lists [33], intersecciones [18, 14] y documentos [27].

Actualmente, nuestro grupo se enfoca principalmente en el problema de caching de resultados, posting lists e intersecciones. En el primero de los casos, se está trabajando con políticas de admisión para cachés de resultados utilizando aproximaciones basadas en *streaming*, ya que las consultas arriban en flujo y a altas tasas. La política de admisión se modela como un problema de clasificación y se lo aborda usando árboles de decisión dinámicos (Hoefding Adaptive Trees [3]) que permiten adaptarse a los patrones de consultas que evolucionan en el tiempo. Complementariamente, se está estudiando el almacenamiento de cantidades variables de resultados en caché, de acuerdo a la consulta.

Por otro lado, en cuanto al problema del caching de intersecciones, se está trabajando en el diseño

de políticas de reemplazo dinámicas para estructuras de datos que originalmente estaban definidas de forma estática como el caché Integrado de Listas + Intersecciones [28]. De esta manera, se pretende adaptar la estructura de datos y sus algoritmos de gestión para soportar reemplazo dinámico. Se evalúa tanto la tasa de aciertos (*Hit Rate*) como el costo que permite ahorrar este tipo de caché de forma que impacte positivamente en las prestaciones.

En cuanto al caché de posting lists, se intenta mejorar el uso del espacio en memoria. A partir de que los índices invertidos se almacenan en bloques comprimidos se está diseñando un esquema que permite cachear solo "algunos"bloques, de acuerdo a diferentes criterios.

2. Procesamiento de Top-k:

A través de los años, se han desarrollado múltiples estrategias para procesar *queries* de la forma mas eficiente posible. En general, se las agrupan en dos categorías: i) estrategias TAAT (*term-at-a-time*), en las que se procesa de a un término del query a la vez, y ii) estrategias DAAT (*document-at-a-time*), en las que el procesamiento se realiza de a un documento por vez. Un caso práctico es retornar los k documentos más relevantes, de acuerdo a un *score* (top-k).

Realizar esto de forma exhaustiva puede resultar costoso, por lo que se han propuesto diferentes algoritmos que buscan evitar procesar aquellos documentos que probablemente no formen parte de los k primeros resultados. Dos de los más conocidos, y que forman parte del estado del arte, son *MaxScore* [29, 13] y WAND [12], los cuales almacenan de forma global para cada *posting list* su *score* mas alto (*upper bound*), y utilizan esta información para determinar si un documento candidato tiene la posibilidad de ingresar al conjunto de respuestas. Por otro lado, recientemente se ha desarrollado *Waves* [7], un algoritmo que se basa en el uso de un índice multicapa, en el cual cada capa contiene las entradas con mayor impacto de cada término.

En este trabajo se propone una extensión a MaxScore, utilizando no sólo uno, sino una serie de *upper bounds* para decidir si un documento puede formar parte de los top-k resultados. En esta nueva versión se debe mantener una estructura adicional similar a una *skip list* con su correspondiente *upper bound*. De esta forma, se le provee al algoritmo de más información que puede utilizar en cada iteración para evitar procesar documentos innecesariamente.

3. Estructuras Escalables: Los sistemas de búsqueda que ingestan información en tiempo real (por ejemplo, sitios de microblogging como Twitter) y la ponen a disposición (indexan) en un intervalo muy corto de tiempo requieren de enfoques diferentes a los tradicionales para la gestión de sus estructuras de datos [4, 1]. Para poder mantener la eficiencia conforme se incrementa la cantidad de información que consumen, resulta indispensable mantener el índice invertido en memoria principal y gestionarlo racionalmente, manteniendo solamente aquella información que permita alcanzar prestaciones de efectividad aceptables [6]. En este sentido, el grupo trabaja en técnicas de control del crecimiento de las estructuras de datos [26] a partir de estrategias de invalidación de entradas en el vocabulario, entre otras. Siguiendo esta línea, se propone el desarrollo de una nueva familia de algoritmos de invalidación y poda selectiva [21] de las estructuras de datos a partir de la evolución y dinámica del vocabulario.

4. Estructuras Comprimidas: La compresión de datos es una técnica bien establecida en el ámbito de los sistemas de búsqueda de gran escala y ha sido extensamente estudiada. En esta línea se trata, básicamente, de la compresión de números enteros (identificadores de documentos y frecuencias de los términos). En trabajos recientes [22], se ha mostrado que combinando diferentes técnicas se puede alcanzar mejores tasas de compresión sin afectar el tiempo de procesamiento. En nuestro grupo, se investiga el concepto de representación y compresión híbrida de una posting list, es decir, se combinan arreglos de bits (*bitvectors*) con listas de enteros (representación tradicional) a partir de explotar las repeticiones de patrones que aparecen en las posting lists. Luego, tanto los bitvectors como las posting lists resultantes se comprimen con las técnicas que mejor *tradeoff* (tasa de compresión/tiempo de descompresión) ofrecen para el problema planteado [33].

b. Procesamiento y Análisis de Grandes Datos

Las plataformas de procesamiento distribuido para Grandes Datos llevan varios años de desarrollo y utilización. Éstas proporcionan interfaces que ofrecen un nivel de abstracción considerable en la utilización de un cluster a cambio de agregar capas de software que gestionan los recursos. Ejemplos de esto son plataformas como Hadoop [30], Spark [31] y Storm. El grupo investiga cómo utilizarlas eficien-

temente en los problemas antes mencionados, principalmente bajo restricción de recursos (*commodity hardware*) [8].

Un caso puntual, es la construcción de índices mediante procesos distribuidos en un cluster, que suele ser un requerimiento para las máquinas de búsqueda. Esto se debe a que los documentos pueden ser procesados en paralelo por diferentes nodos y realizarse posteriormente un proceso de integración de los datos para la construcción del índice invertido final, el cual puede estar particionado o no. Por otro lado, aparecen estructuras de datos para índices que ofrecen un mejor rendimiento para la recuperación (bajo ciertas condiciones) como es el caso de Block-Max [9] y que son de interés cuando se requiere soportar búsquedas a escala masiva.

Otro caso abordado es el procesamiento de flujos de *streams* de vídeo provenientes de cámaras de vigilancia. Se trabaja en estudiar la escalabilidad y eficiencia de un cluster Spark [32] para el problema de monitoreo en tiempo real y la detección de figuras humanas en las secuencias de imágenes (*frames*). Esto último se modela como un problema de clasificación. La carga de trabajo se distribuye entre los nodos bajo diferentes arquitecturas y se analizan los requerimientos de hardware para cumplir con las restricciones temporales. Resultados preliminares muestran que los requerimientos para procesar el video en tiempo real utilizando las técnicas mencionadas son altos y se requiere de la optimización tanto del cluster (selección de parámetros, lo que no es trivial con este tipo de herramientas [17]) y de las piezas de software. También se trabaja con esquemas de muestreo para evitar procesar todos los *frames* del video, pero manteniendo las prestaciones.

En esta línea de investigación se utilizan plataformas y técnicas del ámbito de Grandes Datos (Por ejemplo, Hadoop/MapReduce) para estudiar, diseñar y evaluar algoritmos para diferentes problemas como los mencionados y tratar de establecer los parámetros que determinan la eficiencia del proceso, tales como propiedades de los datos de entrada o las características de un cluster particular.

Resultados y objetivos

El objetivo principal del proyecto es estudiar, desarrollar, aplicar, validar y transferir modelos, algoritmos y técnicas que permitan construir herramientas y/o arquitecturas para abordar algunas de las problemáticas relacionadas con las búsquedas a gran escala y el procesamiento de grandes datos.

Se estudian problemas relacionados con estructuras de datos y algoritmos, combinándolas con técnicas de aprendizaje automático y optimización para aplicaciones de búsqueda. Se proponen mejoras que apuntan a la eficiencia en una tarea. En particular, se espera alcanzar los siguientes objetivos:

- Definir y evaluar estructuras de datos híbridas que permitan ahorrar espacio mientras mantienen la performace del sistema, amortiguando el impacto del crecimiento en la cantidad de información que se debe manejar.
- Mejorar técnicas de caching específicas en motores de búsqueda, tanto políticas de reemplazo como de admisión (tema que no ha tenido suficiente desarrollo aún). En especial, se incorpora el análisis del flujo de consultas en *streaming* y se utilizan algoritmos apropiados.
- Diseñar y evaluar versiones optimizadas de algoritmos de procsamientos de *queries*, que permitan mejorar las prestaciones de los servicios de búsqueda.
- Diseñar arquitecturas para aplicaciones específicas del área de Big Data, orientadas, por un lado, a la indexación masiva distribuida y, por el otro, al procesamiento de flujos de datos en *streaming* (como los flujos de video en tiempo real).

Formación de Recursos Humanos

Este proyecto brinda un marco para que docentes auxiliares y estudiantes lleven a cabo tareas de investigación y se desarrollen en el ámbito académico. Recientemente, se ha finalizado una tesis de la maestría en "Exploración de Datos y Descubrimiento de Conocimiento", DC, FCEyN, UBA y tres de Licenciatura en Sistemas de Información (UNLu).

Actualmente, se están dirigiendo tres trabajos finales de la Lic. en Sistemas de Información (UNLu), hay dos pasantes alumnos y un becario CIN (Beca Estímulo a las Vocaciones Científicas). Se espera dirigir al menos dos estudiantes más por año y presentar dos candidatos a becas de investigación.

Referencias

[1] N. Asadi, J. Lin, and M. Busch. Dynamic memory allocation policies for postings in real-time twitter search. *CoRR*, abs/1302.5302, 2013.

[2] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval - The concepts and technology behind search, 2nd ed.* Pearson Education Ltd., 2011.

[3] A. Bifet and R. Gavaldà. Adaptive learning from evolving data streams. In N. M. Adams, C. Robardet, A. Siebes, and J.-F. Boulicaut, editors, *Advances in Intelligent Data Analysis VIII*, pages 249–260, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[4] M. Busch, K. Gade, B. Larson, P. Lok, S. Luckenbill, and J. Lin. Earlybird: Real-time search at twitter. In *Proc. of the 28th International Conference on Data Engineering*, ICDE '12. IEEE Computer Society, 2012.

[5] B. B. Cambazoglu and R. A. Baeza-Yates. Scalability and efficiency challenges in large-scale web search engines. In *Proc. of the Eighth ACM International Conference on Web Search and Data Mining, WSDM*, 2015.

[6] C. Chen, F. Li, B. C. Ooi, and S. Wu. Ti: An efficient indexing mechanism for real-time search on tweets. In *Proc. of the 2011 ACM SIGMOD International Conference on Management of Data*. ACM, 2011.

[7] C. Daoud, E. Moura, D. Fernandes, A. Silva, C. Rossi, and A. Carvalho. Waves: a fast multi-tier top-k query processing algorithm. 20:1–25, 02 2017.

[8] T. Delvechio and G. H. Tolosa. Indexación distribuida con restricción de recursos. In *Simposio Argentino de GRANdes DAtos (AGRANDA)-JAIIO 46 (Córdoba, 2017)*, 2017.

[9] S. Ding and T. Suel. Faster top-k document retrieval using block-max indexes. In *Proc. of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11. ACM, 2011.

[10] P. A. Dreyer Jr. and F. S. Roberts. Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion. *Discrete Applied Mathematics*, 2009.

[11] D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, New York, NY, USA, 2010.

[12] A. B. et al. Efficient query evaluation using a two-level retrieval process. *Proc. Proceedings of the twelfth international conference on Information and knowledge management, ACM*, pages 426–434, 2003.

[13] T. S. et al. Optimization strategies for complex queries. *Proc. Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 219–225, 2005.

[14] E. Feuerstein and G. Tolosa. Cost-aware intersection caching and processing strategies for in-memory inverted indexes. In *In Proc. of 11th Workshop on Large-scale and Distributed Systems for Information Retrieval*, LSDS-IR'14, 2014.

[15] J. Gantz and D. Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future.*, pages 1–16,, 2012.

[16] X. Guo. Application of meteorological big data. In *Communications and Information Technologies (ISCIT), 2016 16th International Symposium on*, pages 273–279. IEEE, 2016.

[17] H. Karau and R. Warren. *High Performance Spark: Best Practices for Scaling and Optimizing Apache Spark*. ."O'Reilly Media, Inc.", 2017.

[18] X. Long and T. Suel. Three-level caching for efficient query processing in large web search engines. In *Proc. of the 14th international conference on World Wide Web*. ACM, 2005.

[19] S. Madden. From databases to big data. *IEEE Internet Computing*, 16(3):4–6, 2012.

[20] J. Magnusson. Social network analysis utilizing big data technology, 2012.

[21] A. Ntoulas and J. Cho. Pruning policies for two-tiered inverted index with correctness guarantee. In *Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[22] G. Ottaviano, N. Tonellotto, and R. Venturini. Optimal space-time tradeoffs for inverted indexes. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 47–56, New York, NY, USA, 2015. ACM.

[23] R. Ozcan, I. S. Altingovde, and O. Ulusoy. Cost-aware strategies for query result caching in web search engines. *ACM Trans. Web*, 5(2), May 2011.

[24] A. O'Driscoll, J. Daugelaite, and R. D. Sleator. 'big data', hadoop and cloud computing in genomics. *Journal of biomedical informatics*, 46(5):774–781, 2013.

[25] A. Rajaraman and J. D. Ullman. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011.

[26] E. Ríssola and G. Tolosa. Improving real time search performance using inverted index entries invalidation strategies. *Journal of Computer Science & Technology*, 16(1), 2016. ISSN: 1666-6038.

[27] T. Strohman and W. B. Croft. Efficient document retrieval in main memory. In *SIGIR 2007: Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[28] G. Tolosa, L. Becchetti, E. Feuerstein, and A. Marchetti-Spaccamela. Performance improvements for search systems using an integrated cache of lists+intersections. In E. Moura and M. Crochemore, editors, *String Processing and Information Retrieval*, pages 227–235, Cham, 2014. Springer International Publishing.

[29] H. R. Turtle and J. Flood. Query evaluation: Strategies and optimizations. *Inf. Process. Manage.*, 31(6):831–850, 1995.

[30] T. White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009.

[31] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark : Cluster Computing with Working Sets. *HotCloud'10 Proc. of the 2nd USENIX conference on Hot topics in cloud computing*, page 10, 2010.

[32] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. *HotCloud*, 10(10-10):95, 2010.

[33] J. Zhang, X. Long, and T. Suel. Performance of compressed inverted list caching in search engines. In *Proc. of the 17th international conference on World Wide Web*, WWW '08. ACM, 2008.

[34] P. Zikopoulos, C. Eaton, et al. *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media, 2011.