# Modelos y Algoritmos para Problemas de Procesamiento en Entornos de Big Data

Gabriel H. Tolosa[1], Pablo Lavallén[1], Tomás Delvechio[1], Agustín Marrone[1], Andrés Giordano[1] , Esteban A. Ríssola[1,3]
{tolosoft, plavallen, tdelvechio, eamarrone, agiordano}@unlu.edu.ar
esteban.andres.rissola@usi.ch

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

## Resumen

La idea del procesamiento de datos masivos (Big Data) se ha desarrollado sostenidamente en los últimos años, estableciéndose como un nuevo paradigma para resolver problemas. Por un lado, el crecimiento en el poder de cómputo y almacenamiento habilita la posibilidad de manejar volúmenes de datos de varios órdenes de magnitud. Por el otro, generan la necesidad de contar no solamente con plataformas que permitan distribuir el procesamiento sino, además, con algoritmos que lo realicen de forma eficiente.

Una de las primeras aplicaciones de Big Data son los motores de búsqueda de escala web, sistemas que procesan miles de millones de documentos y deben responder a los usuarios con estrictas restricciones de tiempo, típicamente, milisegundos. Análogamente, el procesamiento de grafos masivos provenientes del mapeo de la estructura de las redes sociales presenta desafíos de forma sostenida.

Estos escenarios se caracterizan por una complejidad creciente en espacio y requieren soluciones cada vez más sofisticadas ya que la cantidad de datos y de usuarios crece conforme evolucionan en el tiempo. Además, han aparecido requerimientos para ofrecer respuestas sobre flujos de datos que ocurren en tiempo real (*streaming*) por lo que es un requisito considerar modelos que puedan tomar decisiones *on-line* utilizando estos datos.

Este trabajo presenta las líneas de investigación que se proponen en el contexto de los datos masivos a partir del estudio, diseño y evaluación de estructuras de datos y algoritmos que operan eficientemente, ya sea sobre documentos, grafos sociales o interacciones de usuarios, entre otros.

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

## Contexto

Esta presentación se encuentra enmarcada en el proyecto de investigación "Algoritmos Eficientes y Minería Web para Recuperación de Información a Gran Escala" del Departamento de Ciencias Básicas (UNLu).

## Introducción

La gran asimilación de tecnologías de la información y la comunicación por parte de las sociedades en las últimas décadas genera un renovado interés sobre algoritmos que permitan almacenar y procesar grandes volúmenes de datos en límites de tiempo cada vez más precisos [28]. Además se observa un auge y maduración de tecnologías de cómputo en la nube, base sobre la cual se plantea la posibilidad de acceder a crecientes capacidades de procesamiento y almacenamiento de forma rápida y con esfuerzo reducido en el despliegue y gestión de recursos. Una de las áreas donde se dio la convergencia de ambos procesos generando nuevas oportunidades de investigación es el área de Big Data (Datos Masivos). Este enfoque ofrece esquemas de procesamiento y almacenamiento distribuido considerando prioritario permitir la escalabilidad horizontal y la tolerancia a fallas, como es el caso de la plataforma Spark [26].

De forma complementaria, surge la necesidad permanente de nuevos modelos y estrategias para resolver los problemas que aparecen continuamente

de forma eficiente. Un caso clásico son los motores de búsqueda web, que procesan miles de millones de documentos[1] y deben responder a los usuarios con estrictas restricciones de tiempo, típicamente, milisegundos [2]. Otro ejemplo común hoy en día es el procesamiento de grafos masivos provenientes, por ejemplo, de la estructura de una red social [18] o de interacciones entre usuarios. También son ejemplos los servicios de *streaming* [28, 22], la genómica [20] y la meteorología [15].

Estos problemas requieren, en general, procesamiento distribuido, paralelo y algoritmos altamente eficientes [5]. Para su abordaje, se utilizan plataformas de almacenamiento y procesamiento no tradicionales [17], como Hadoop [25] y Spark [26], sobre las cuales se situan capas de software que implementan algoritmos de búsquedas, aprendizaje automático y optimización. En la mayoría de los casos, la partición del problema y la distribución de la carga de trabajo son aspectos de las estrategias que requieren ser optimizados de acuerdo al problema [24].

## Líneas de I+D

En este trabajo se describen líneas de I+D del grupo, las cuales se basan, principalmente, en mejorar la eficiencia en la recuperación de información de gran escala y el procesamiento de grandes volúmenes de datos, en general, sobre *commodity hardware* [9]. De forma continua aparecen nuevos desafíos en estas áreas y 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

#### a.1. Algoritmos para *Top-k*

En la actualidad, lo común es que los motores de búsqueda respondan en cuestión de milisegundos a las consultas de los usuarios (*query*), con un conjunto de los $k$ documentos más relevantes. Para ello, deben revisar una colección que se compone de miles de millones de elementos, por lo que es crucial mejorar y explorar nuevas técnicas que permitan "atravesar"de forma eficiente las estructuras de datos internas. A los algoritmos que toman un *query* y revisan un índice invertido en búsqueda de los documentos más relevantes se los pueden agrupar en dos categorías: i) TAAT (term-at-a-time), procesan de a un término del query a la vez, y ii) DAAT (document-at-a-time), en los que se procesa de a un documento a la vez. Para ello, en la familia de algoritmos DAAT, MaxScore [23] y WAND [13] almacenan un *score* global máximo (*upperbound*) y lo utilizan para saber si un documento puede formar parte del resultado final en los *top-k*. Como extensión a estos, existen los algoritmos Block-Max WAND [10], Variable Block-Max WAND [19] y Block-Max MaxScore [6], los cuales almacenan no sólo uno, sino un *upperbound* local a un bloque de cierto tamaño, con el fin de procesar aún una cantidad de entradas aún menor.

En esta línea se investigan los algoritmos DAAT que forman parte del estado del arte y, adicionalmente, se propone uno nuevo como extensión de MaxScore. En éste, en vez de almacenar sólo un *upperbound* global, se almacena un conjunto de ellos en una estructura similar a una *skip list* [7], dotando al algoritmo de más información para mejorar la eficiencia del procesamiento.

#### a.2. Búsquedas Selectivas

Complementando los algoritmos de búsqueda mencionados en la línea anterior, otra estrategia para acelerar el proceso está basada en la partición de la colección de documentos en porciones, $P$, (*shards*) de acuerdo a algún criterio (por ejemplo, temático) de manera tal de enviar las consultas solamente a un número reducido $n$ de nodos ($n << P$) que contengan particiones de la colección que potencialmente pueden satisfacer la consulta. Este problema se lo conoce como "búsquedas selectivas"(*selective search*) [16] e incluye métodos que permiten seleccionar los recursos adecuados, algoritmos de fusión de resultados parciales y estrategias adaptadas de *caching*.

Dichas estrategias son desafiadas cuando los documentos aparecen en flujos en tiempo real como, por ejemplo, las publicaciones en las redes sociales. Un caso paradigmático son las publicaciones en Twitter, en la cual millones de usuarios alrededor del mundo publican "documentos cortos"(*tweets*) desde diferentes tipos de dispositivos (generalmente, móviles), los cuales deben estar disponibles casi de inmediato (segundos) por lo que las estructuras de datos deben soportar un alto dinamismo [4].

[1]Según http://www.worldwidewebsize.com el índice de Google contiene aproximadamente unos 55.000 millones de documentos (Marzo/2019).

En esta linea de investigación se propone combinar la problemática de las búsquedas en tiempo real utilizando una arquitectura basada en búsquedas selectivas donde los criterios de actualización del índice invertidos por partición, las estrategias de caché a implementar y el algoritmo de búsqueda final impactan en la performance que se pretende optimizar (eficiencia y/o efectividad). Los objetivos generales perseguidos son el desarrollo de técnicas y algoritmos para la asignación de flujos de documentos a diferentes particiones de un índice para luego soportar búsquedas sobre un subconjunto de éstas, maximizando la performance (tanto eficiencia como efectividad).

## b. Procesamiento de Grafos Masivos

### b.1. Cálculo Distribuido en Grafos Evolutivos

Como se ha dicho anteriormente, uno de los desafíos surgidos con la aparición de las aplicaciones que procesan datos masivos es la de generar representaciones adecuadas para los datos recolectados. Una de las estructuras más longevas y versátiles de las ciencias de la computación para estas representaciones son los grafos, formados por un conjunto de nodos y relaciones entre éstos (aristas) [7]. Diversos algoritmos que se aplican sobre grafos son ampliamente utilizados en la industria y la academia para abordar un gran abanico de soluciones. Algunos de estos algoritmos son los de camino mas corto (Single Source Shortest Path o SSSP y All-Pairs Shortest Path o APSP), y suelen constituir la base de muchos procesos en ámbitos como la recuperación de información [3] o análisis de redes sociales [1], por nombrar un par de ejemplos.

Este tipo de procesos suelen tener soluciones aceptadas desde hace mucho tiempo, las cuales no escalan cuando el tamaño de los datos supera un umbral, dando lugar a soluciones en tiempos no triviales [8]. Por otro lado, es todo un desafío la implementación de estrategias que aprovechen las características distribuidas de las plataformas existentes [12]. Otro problema que presentan los enfoques secuenciales tradicionales y no escalables es que muchas aplicaciones exigen de forma creciente respuestas cada vez mas instantáneas, lo que introduce la posibilidad de construir soluciones aproximadas en el resultado pero eficientes en el tiempo insumido. Otro punto a considerar es que los grafos, al representar relaciones, pueden cambiar a lo largo del tiempo, que se conoce como grafos dinámicos [14], y admiten soluciones que tengan en cuenta procesamientos previos para no tener que recalcular completamente las métricas ante cada modificación del mismo.

Dentro del marco del proyecto, el objetivo de esta linea de trabajo es diseñar, analizar e implementar algoritmos eficientes para el cálculo de las distancias mínimas (*shortest path*) para grafos que pueden o no evolucionar en el tiempo. Estas soluciones se abordan utilizando estrategias de cómputo distribuido sobre plataformas de Big Data (por ejemplo, Spark) sobre hardware commodity y soportando tolerancia a fallas.

### b.2. Estimación de Distancia entre Nodos

El problema de la distancia entre dos nodos tiene múltiple aplicaciones prácticas, por ejemplo, para el ranking en búsquedas[2]. Se lo define como la longitud del camino más corto entre ellos y se vuelve inviable si se requiere responder en pocos milisegundos. Esta métrica es usada en numerosos algoritmos que apuntan a resolver problemas como la recomendación de links [27] y agrupamiento de usuarios [11], entre otros.

El cálculo exacto de esta métrica entre dos nodos arbitrarios se ve afectado en cuanto a la eficiencia del proceso debido al tamaño de los grafos actuales resultando prohibitivo para aplicaciones prácticas. Por ejemplo, en redes sociales digitales el número de relaciones (aristas) supera ampliamente la cantidad de usuarios (nodos)[3]

La estimación de este valor es una alternativa válida y la reducción del error asociado (a un costo computacional bajo) conforma una de las líneas de investigación de este proyecto. En este enfoque se utiliza un conjunto de nodos, llamados *landmarks* [21], que se toman como referencia para estimar luego la distancia entre dos nodos arbitrarios. El problema de la selección de *buenos landmarks*, es decir, aquellos que permitan minimizar el error de estimación, es una pregunta abierta ya que existen diversos criterios a aplicar que consideran grafos de diferente tamaño, densidad y dinámica. Algunos resultados preliminares sobre nuevos métodos de selección

[2]Un caso es la red de contactos profesionales Linkedin.

[3]Twitter: 321 millones de usuarios activos, 226.947 millones de vínculos. `http://investor.twitterinc.com/home/default.aspx`

de *landmarks* indican que usar métricas tradicionales en conjunto (*Closeness Centrality*, grado, entre otras) , sobre ciertos grafos, potencian la estimación de la distancia logrando mejores resultados respecto del uso individual de éstas.

El desarrollo de métodos de corrección de la distancia estimada no ha sido abordado hasta el momento y es otra de las lineas de investigación de este proyecto. La idea consiste en lograr un algoritmo que, basado en características generales del grafo (como la densidad, diámetro, tamaño de comunidades, entre otras) permita corregir el valor de la estimación. Algunos análisis preliminares sobre grafos sintéticos indican que es posible reducir el error mediante esta técnica.

## Resultados y objetivos

El objetivo principal del proyecto es desarrollar, evaluar y transferir modelos y algoritmos para abordar algunas de las problemáticas relacionadas con las búsquedas a gran escala y el procesamiento de grandes datos, en este caso, grafos masivos. En general, se proponen mejoras que apuntan a la eficiencia y la escalabilidad. En particular, se espera alcanzar los siguientes objetivos:

- Diseñar versiones optimizadas de algoritmos de procesamiento de *queries* dotando a los algoritmos para $top-k$ de información accesoria que les permita recorrer las estructuras de datos eficientemente.
- Desarrollar de técnicas y algoritmos para la asignación de flujos de documentos a diferentes particiones de un índice para luego soportar búsquedas selectivas sobre un subconjunto de éstas, maximizando la performance (tanto eficiencia como efectividad) y considerando parámetros de la arquitectura (por ejemplo, número de procesadores, núcleos, etc.).
- Definir y evaluar estructuras y modelos de cómputo distribuido sobre hardware commodity para problemas de cálculo de métricas en grafos masivos, en particular en grafos evolutivos.
- Diseñar y evaluar estrategias de estimación de distancias entre nodos de un grafo masivo para problemas de búsqueda, junto con métodos de corrección de la estimación.

## Formación de Recursos Humanos

En el marco de estas líneas de investigación se están dirigiendo dos tesis de Licenciatura en Sistemas de Información (UNLu). Además, asociados al proyecto de investigación hay una estancia de investigación de la Secretaría de CyT (UNLu), una Beca Estímulo a las Vocaciones Científicas (CIN) y dos pasantías internas UNLu.

## Referencias


[1] D. A. Bader, S. Kintali, K. Madduri, and M. Mihail. Approximating betweenness centrality. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 124–137. Springer, 2007.

[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] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[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] K. Chakrabarti, S. Chaudhuri, and V. Ganti. Interval-based pruning for top-k processing over compressed lists. In *Proc. of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, pages 709–720, USA, 2011. IEEE Computer Society.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT press, 2009.

[8] A. Crauser, K. Mehlhorn, U. Meyer, and P. Sanders. A parallelization of dijkstra's shor-

test path algorithm. In *International Symposium on Mathematical Foundations of Computer Science*, pages 722–731. Springer, 1998.

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

[10] 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.

[11] J. Edachery, A. Sen, and F. J. Brandenburg. Graph clustering using distance-k cliques. In J. Kratochvíyl, editor, *Graph Drawing*, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[12] N. Edmonds, T. Hoefler, and A. Lumsdaine. A space-efficient parallel algorithm for computing betweenness centrality in distributed memory. In *2010 International Conference on High Performance Computing*, pages 1–10. IEEE, 2010.

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

[14] D. Ferone, P. Festa, A. Napoletano, and T. Pastore. Shortest paths on dynamic graphs: A survey. *Pesquisa Operacional*, 37(3):487–508, 2017.

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

[16] A. Kulkarni and J. Callan. Selective search: Efficient and effective search of large textual collections. *ACM Transactions on Information Systems (TOIS)*, 33(4):17, 2015.

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

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

[19] A. Mallia, G. Ottaviano, E. Porciani, N. Tonellotto, and R. Venturini. Faster blockmax wand with variable-sized blocks. In *Proc. of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 625–634, New York, NY, USA, 2017. ACM.

[20] 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.

[21] M. Potamias, F. Bonchi, C. Castillo, and A. Gionis. Fast shortest path distance estimation in large networks. In *Proc. of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 867–876, New York, NY, USA, 2009. ACM.

[22] 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.

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

[24] Y. Wang, L. Wu, L. Luo, Y. Zhang, and G. Dong. Short-term internet search using makes people rely on search engines when facing unknown issues. *PloS one*, 12(4):e0176325, 2017.

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

[26] 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.

[27] Y. Zhang and J. Pang. Distance and friendship: A distance-based model for link prediction in social networks. In R. Cheng, B. Cui, Z. Zhang, R. Cai, and J. Xu, editors, *Web Technologies and Applications*, Cham, 2015. Springer International Publishing.

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