Longitudinal Analytics on Web Archive Data: It's About Time!*

Gerhard Weikum[1], Nikos Ntarmos[2], Marc Spaniol[1], Peter Triantafillou[2], András Benczúr[3], Scott Kirkpatrick[4], Philippe Rigaux[5], and Mark Williamson[6]
Max Planck Institute for Informatics, Germany[1]
University of Patras, Greece[2]
Hungarian Academy of Sciences, Hungary[3]
Hebrew University, Israel[4]
Internet Memory Foundation, France[5]
Hanzo Archives Ltd., UK[6]
weikum@mpi-inf.mpg.de ntarmos@ceid.upatras.gr mspaniol@mpi-inf.mpg.de peter@ceid.upatras.gr
benczur@sztaki.hu kirk@cs.huji.ac.il philippe.rigaux@internetmemory.org markw@hanzoarchives.com

ABSTRACT

Organizations like the Internet Archive have been capturing Web contents over decades, building up huge repositories of time-versioned pages. The timestamp annotations and the sheer volume of multi-modal content constitutes a gold mine for analysts of all sorts, across different application areas, from political analysts and marketing agencies to academic researchers and product developers. In contrast to traditional data analytics on click logs, the focus is on longitudinal studies over very long horizons. This longitudinal aspect affects and concerns all data and metadata, from the content itself, to the indices and the statistical metadata maintained for it. Moreover, advanced analysts prefer to deal with semantically rich entities like people, places, organizations, and ideally relationships such as company acquisitions, instead of, say, Web pages containing such references. For example, tracking and analyzing a politician's public appearances over a decade is much harder than mining frequently used query words or frequently clicked URLs for the last month. The huge size of Web archives adds to the complexity of this daunting task. This paper discusses key challenges, that we intend to take up, which are posed by this kind of longitudinal analytics: time-travel indexing and querying, entity detection and tracking along the time axis, algorithms for advanced analyses and knowledge discovery, and scalability and platform issues.

1. MOTIVATION

Big-data analytics for the *Web of the Future* - Web 2.0 (communities, their behavior, etc.) and Web 3.0 (semantic annotations, linkeddata.org, etc.) - has been a hot topic for some time. However, the *Web of the Past* is an equally important topic for both academics and real-life applications. Academically, longitudinal data analytics is even more challenging and has not received due attention. The sheer size and content of such web archives lends itself to wide applicability for analysts in a great number of different domains.

National libraries and organizations like the Internet Archive (archive.org) and its European sibling (internetmemory.org) have been capturing Web contents over decades. These archives host a wealth of information, providing a gold mine for sociological, political, business, and media analysts. For example, one could track and analyze public statements made by representatives of companies such as Google or Tandem Computers, characterizing the evolution of patterns in their attitude towards energy efficiency. Another example could be tracking, over a long time horizon, a politician's public appearances: which cities has she/he visited, which other politicians or business leaders has she/he met, and so on. Analyses of this kind could also be carried out on large news archives, but this can be seen as variant of Web archive analytics; moreoever, the Web (and especially the recent Web 2.0) has a wider variety of coverage, potentially leading to the discovery of more interesting patterns and trends.

Web archives contain timestamped versions of Web sites over a long-term time horizon. This *longitudinal dimension* opens up great opportunities for analysts. For example, one could compare the notions of "online friends" and "social networks" as of today versus five or ten years back. Similar examples relevant for a business analyst or technology journalist could be about "tablet PC" or "online music". This requires finding all Web pages from certain eras that contain these and/or other related phrases. Unfortunately, this is beyond hope today. Web archives like the Internet Archive provide URL-based access only, via the Wayback Engine. For a given URL, you can retrieve all archived versions of the page; then you can navigate a version on a per-site basis and will be automatically connected to the proper version as of the same snapshot. But this kind of time-travel browsing does not work across sites. There is no support for keyword search along the time axis at all. The NutchWAX open-source software has been tried for archive search, but has not been deployed for public access. Searching for phrases (i.e., multiple keywords occuring contiguously in a page) is computationally much harder (requiring position indexes, etc.), and way beyond today's capabilities. The goal is to allow rich text queries with a temporal filter, such as {"*tablet*

*The authors are the main investigators of the LAWA project, a brand-new project funded by the European Commission, see `http://www.lawa-project.eu` for details.

5th Biennial Conference on Innovative Data Systems Research (CIDR '11) January 9-12, 2011, Asilomar, California, USA..

PC" Europe market} *@ [2003-2007]*, rank the results by time-aware relevance, and aggregate them into a suitable form for subsequent analytics (e.g., into some form of temporally extended text cube).

Analysts are not interested in text or Web pages per se, even if the underlying sources are in text or multimodal form. Instead, they want to see, compare, and understand the behavior of (and trends about) entities like companies, products, politicians, music bands, songs, movies, etc., thus calling for *entity-level analytics* over Web archives. For the tablet-PC search, they would ideally obtain information grouped by named entities like Apple Corp., Microsoft Corp., etc. - combined with the time dimension, for example, by year. Likewise, the matches for "tablet PC" would ideally be based on product names rather than the literal text, to capture also related products such as e-book readers. This calls for lifting the entire archive contents, or at least the slices that are relevant for this analytic task, from the text level to the entity level: detecting named entities, resolving ambiguous names, tracking the same entity in its mentions over extended time periods. Obviously, this is a daunting task, regarding both semantics and scalability, already for current corpora in digital libraries (e.g., PubMed) or enterprises. As entities morph and get renamed over time, e.g., by company acquistions or mergers, this is a grand challenge for large-scale longitudinal analytics.

This paper elaborates on these challenges, identifies specific technical aspects and discusses the problem space. We address semantic and scalability issues. To appreciate the latter, let us merely point out that the Internet Archive currently holds more than 150 Billion versions of Web pages, captured during the timeframe from 1996 until now. Its coverage is getting sparser as Web contents has become so diverse, dynamic, and humongous. A high-coverage archive would have to be an order of magnitude larger.

2. CHALLENGE: TIME-TRAVEL INDEXING

One of the fundamental underpinnings of the envisioned kind of longitudinal analytics is indexing for time-travel queries. The general form of these queries is $t_1 t_2 \ldots t_m @ [T_{low}, T_{high}]$ where the t_i are text terms and T_{low} and T_{high} are the boundaries of a time interval of interest (time points are a special case). In the simplest case, terms are just single keywords, but analysts also need phrases (e.g., product names, campaign slogans, quotations) and may even use additional constructs like negation or distinguishing optional from mandatory terms (e.g., in analyzing intellectual-property issues). A good design for an index to support such queries is not obvious at all. It entails difficult issues regarding 1) the choice of data structures, for example, multidimensional index trees versus IR-style inverted lists versus hash-based synopses, 2) the challenge of efficiently building this huge index (for a given data structure) and incrementally maintaining it, and the associated consistency issues for concurrent analytics tasks, 3) the organization of the index on scale-out platforms so that time-travel queries can be run with high throughput, user-acceptable latency, high availability, and low cost (incl. energy-efficiency). Note that throughput is an issue, even if analysts are a rare species, compared to Facebook users. The reason is that complex analytic tasks may trigger, under the hood, a large number of simpler time-travel queries.

Data structures: The choice is widely open. A database person, at first thought, would most likely advocate a *multidimensional index structure* like an R-tree or perhaps a tailored time-key index like the Multiversion B-tree or the TSB-tree. However, mapping our data space onto one of these structures is full of problems. First, the high dimensionality of the text-term aspect prevents a straightforward mapping; there is no way to support a million text terms by a one-million-dimensional R-tree. And we do need to support multidimensional queries that consist of several terms. Second, the processing of multidimensional range queries over these indexes would result in non-sequential, traversal-style access patterns, thus hurting the effectiveness of the hardware data caches. With many-core processors and flash-based storage, access locality is absolutely crucial for performance. An alternative approach is to adopt the IR paradigm of *inverted lists*. There is usually, one list of postings (document identifiers and associated payload data such as precomputed scores) per term. Each list can be compressed extremely well (see IR literature and also the techniques used by major search engines), and this results in excellent sequential performance. However, with the temporal dimension folded into such an index, an inverted list would contain the postings for all document versions across the entire timespan of the archive. Consequently, lookups for time points or short intervals are penalized by this blow-up in the version space. This problem is aggravated for phrase search. Unless we constrain the flexibility of the analyst by pre-identifying interesting phrases, we need a position index. Here each per-word list contains postings for each occurrence of the word in each document. Blending this kind of index organization with the temporal dimension in an optimized way is a formidable challenge.

Index build: MapReduce is a popular paradigm for building huge indexes on distributed storage. For standard text indexes, this is indeed a great way of harnessing scale-out architectures (and algorithmically not that different from parallel-database techniques). We are studying how to exploit and extend this paradigm in order to build the combined text-time indexes that we need. A key option pertains to the creation of combined text-and-time indices vs separate time and text indices.

Assume each document has a lifespan defined by its last update to current time. At the next update, the lifespan of the document expires and a new one emerges. Thus, the same document, as it evolves through time, is essentially viewed as a series of different documents. In this setting, one could create a MapReduce job (actually a series of MapReduce jobs) that builds text indices, per term, per preset time interval. This is logically equivalent to first creating text indices with posting lists where each posting (docID, term, score) is annotated with a lifespan and then partitioning the posting lists by time intervals. The approach embeds lifespan information with each posting, merging the time and text indices. However, building such an index creates issues such as how to handle documents whose lifespans overlap the indexed time intervals? Replication of relevant posting list entries is a solution. But this exacerbates an already difficult issue: space. An even more formidable task is deciding which should be the preset time intervals for which to build the indices. At one end, large preset time intervals defeat the purpose. At the other, small index time intervals will be inefficient for queries with large time horizons, since many time-interval indices would need to be merged at query time.

A promising solution might be to build several overlapping time-interval indices, at different time-interval granules and employ the best selection of time indices at query time. These time-interval indices can be hierarchically organized, creating in essence an index of indices, which can be traversed to discover the optimal individual indices to use. Deciding this is a difficult optimization task on its own right.

Another approach would be to build separate indices for time and text. The interesting issues here pertain how to best utilize the MapReduce framework to build time-interval structures and which structures lend themselves to better buildup with MapReduce. For more elaborate structures, which are candidates for time indices (such as interval trees and segment trees), it is not clear how to utilize MapReduce for building and efficiently accessing them at query time.

3. CHALLENGE: QUERYING & RANKING

Query processing: With a rich suite of indexes and synopses, the query processor faces many choices in combining the system's data structures. Querying text and time in an integrated manner is a largely unexplored territory. Prior work on news mining typically assumes that timestamps are high-quality metadata. This is not the case at all in Web archives, and this in turn implies that many queries are explorative with wide time-range conditions and complex search conditions about phrases (contiguous words) or soft phrases (nearly contiguous words within a proximity window).

Ranking models: For conventional text search, the ranking of query results is based on word-occurrence statistics, including the idf measure (inverse document frequency) for the specificity of terms. For time-travel queries, the situation is more complicated. A once rare word may now be used in an inflationary manner, so it would now have low weight in the score aggregation for a multi-keyword query. But when a temporal query travels back to that former period, the idf value back then matters. For example, in the query "online friend"@August2002, "friend" would now have low weight but should have high weight as of 2002. Similar issues arise with "static" authority measures such as PageRank, as they are no longer that static anymore in the context of longitudinal archives. Statistical values like idf change continuously with every new update, as they are corpus-dependent (and not confined to a single document). Approximation techniques at lower cost are presumably sufficient, but finding good trade-offs with (probabilistic) guarantees on the deviation error is difficult. Hence, the time dimension affects everything: not just data, but also its indices and their maintenance, and the related statistics and metadata that govern ranking models.

With phrases as query conditions, there are further difficulties. Ideally, we would like to consider the idf value of an entire phrase rather than merely aggregating the scores of the constituting words. But this cannot be precomputed, as it may be only now that we realize the interestingness of a phrase like "online friend" and would now like to pinpoint the onset of this emerging phrase years ago. Finally, bursty-ness in time could be an important ingredient in the ranking of search results. For example, an analyst may look for interesting time points in the longitudinal answers to a query "tablet PC". This should ideally return pages on the 2002 edition of Windows for tablet PCs, the launching of the iPad in 2010, the revival of e-books, and also salient points or periods for more specific results such as intensive press coverage of specific products in certain regions of the world. Here, interesting points could be found by considering the "first derivative" of measures like idf, PageRank, etc.

4. CHALLENGE: ENTITY TRACKING

Entity detection: Detecting named entities in Web pages and thus lifting the entire analytics to a semantic rather than keywords level is a grand challenge already for standard text mining. The difficulties arise from name ambiguities, thus requiring a disambiguation mapping of mentions (noun phrases in the text that can denote one or more entities) onto entities. For example, the mention "Bill Clinton" can be the former US president William Jefferson Clinton, but Wikipedia alone knows five or so other William Clintons. If the text says only "Clinton", the number of choices increases, and phrases like "the US president" or "the president" have a wide variety of potential denotations. For established kinds of data cleaning and text mining, methods for entity resolution (aka. record linkage) have made reasonable progress (e.g. by using statistical learning for collective labeling), and could handle a good fraction of such cases.

Entities in time: In the Web archive case, some additional aspects are assets while others pose major obstacles. The timestamp of an archived Web page can help to narrow down the disambiguation candidates for phrases like "the US president". Similarly, the connection with previous and successive versions of the same page can help to identify changes at specific timepoints, which may in turn be cues for entity resolution. Cases where the temporal dimension introduces new complexity are when names of entities have changed over time. Examples are people's name changes after getting married or divorced (or simply out of some mood), or organizations that undergo restructuring in their identities. Bell Labs is a notorious example; a simpler one is Tandem Computers, a leading company on highly available, scalable systems in the 1980s. Suppose a technology-and-business analyst wants to track companies that used products of Tandem Computers, over the last 30 years (Web archiving does not go back that long, but there are digitized news archives from this era). Tandem was acquired by Compaq, which was later acquired by HP; the NonStop product line (incl. NonStopSQL) has many instances with all kinds of naming variations and still exists today. So we need to identify, from the site captures of Web archives, all mentions of this business entity, its products, and also the enterprises that employed one these products over the years. This would allow us to construct an entire timeline of how the company and its products were doing over several decades: business tracking at the entity-relationship level, automatically inferred from Web history. Such entity tracking should be combinable with filters and aggregations on keywords or phrases. For example, we could restrict the entire analysis to input sources that contain "zero downtime" or "24x7" or "ultra-high availability". We emphasize again that this could be an ad-hoc interest of some analyst; so hardly anything could be precomputed.

A benchmark proposal: Generalizing the example, a conceivable but currently still elusive benchmark could be the following. For all page versions in a Web archive, with 100 billions of files, identify all entities that are known to Wikipedia at some point in the Wikipedia history. That is, map each mention to the Wikipedia article as of the proper timepoint. For example, when Carla Bruni is mentioned on an entertainment site of July 2005, we should map her name to the former model and singer Carla Bruni Tedeschi,

as reflected in Wikipedia as of this time. But the same name seen in August 2009 should be mapped to the French first lady Carla Bruni-Sarkozy (her offical name now). Here different time periods pose different ambiguity challenges. The timestamps of the archived pages are only of partial help, because the page contents can be older; crawl-based dating is not reliable at all. This is a big issue in dealing with locations (e.g., Mumbai vs. Bombay) and organizations (e.g., Bell Labs vs. AT&T Bell Labs vs. Alcatel-Lucent Bell Labs). The challenge lies in the enormous scale and temporal depth, and, of course, the goal of accomplishing this benchmark task with very high precision and recall.

5. CHALLENGE: EFFICIENT ANALYTICS

Interesting phrases and entities: The envisioned system should support a wide spectrum of analytical tasks, spanning the text, entity, and time dimensions. We want to address different tasks of increasing complexity: frequent phrases, interesting phrases, comparative slicing, time pivoting, and entity-entity as well as entity-phrase co-occurrences. Here an interesting phrase could be one that is salient for a particular time period. This could be modeled by information-theoretic measures like relative entropy. Intuitively, a phrase is interesting if it is frequent in the period of interest and infrequent otherwise. An example could be "yes we can" for the period January 2008 through June 2009. Similar analyses should be possible for interesting entities, returning, for example, Deepwater Horizon for the period April through July 2010. This kind of analytics is algorithmically well understood, but carrying it out on a 100-billion-pages archive is a very ambitious goal.

Text-entity-time analytics: Comparative slicing goes beyond the previous stage by trying to identify salient phrases or entities for different subsets of an archive, where the subsets are determined by ad-hoc filters on phrases, entities, and time. For example, we may be interested in a discriminative analysis of public quotations by key people of Google, for the year 2005 versus the year 2010, and perhaps with focus on European Web sites. Here we need to identify temporal slices of the archive, but also select only those page versions that contain mentions of the entity Google, the word "quotation" or some paraphrase for people's statements, and geographic names indicating Europe. The result could be phrases like "do no evil" (the company motto) for 2005 and "nobody was harmed" (Eric Schmidt's reaction to concerns about privacy) for 2010. This line of analyses is a form of co-occurrence mining in the joint space of text, entities, and time. Another analytic task could be time pivoting: for a given entity, find the most interesting timepoints or periods along with a digest of most salient phrases or entities. One can view this as a generalization of tag-cloud timelines. Supporting all this in a truly ad-hoc manner - without any pre-selected phrases or entities and precomputed statistics - is a challenge.

Efficiency: A good part of such tasks could be addressed by MapReduce-based algorithms in a scalable manner. However, these algorithms also need to be efficient in the sense that they use resources - processors, memory consumption, interconnect bandwidth, and energy - in a cost-beneficial manner. If an efficient algorithm can reduce resource consumption, this pays off directly in a lower electricity bill or the ability to run a higher throughput of independent tasks by a larger number of analysts. Thus, we envision a smart combination of scale-out-oriented scanning, hashing, and merging techniques with index lookups, complex execution plans, and statistical approximation methods. The optimization space has an enormous number of degrees of freedom. Here the notion of an execution plan goes way beyond the traditional kind of query execution plan, as it would also involve phrase-mining, entity-resolution, temporal-aggregation, and statistical-computation steps.

6. CHALLENGE: SCALABLE PLATFORM

Figure 1 shows a system architecture, addressing how control and data flow is envisaged within the LAWA project. The base layer consists of storing Web pages and updating the collection with data from new Web crawls. Data at the base layer is processed using MapReduce to produce so-called primary indices (currently .warc files) for the documents in the collection. We envisage a scalable row-store platform (such as HBase) and HDFS as the storage systems for these.

The primary indices contain the essential data, needed to build richer indices, such as text and time-text indices (of the forms discussed earlier), and other statistical structures, such as Bloom Filters and/or histograms needed to estimate join sizes (e.g. of text and time index files), sketches used to estimate cardinalities of sets, etc.

The top layer, the Analytics Engine, is responsible for processing analytics tasks. In turn, these may be expressed as a workflow of complex queries (e.g., involving joins, range selections, top-K operations, etc.) which are handed to the layer underneath, the Complex Query Processing Engine, which in turn may (or may not) utilize (or even build) additional query-specific indices.

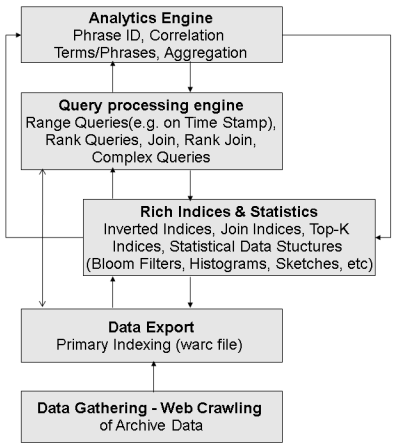

Figure 1: LAWA system architecture

We have discussed some of the key challenges associated with longitudinal analytics over massive data collections of Web archive data. Numerous open research problems are revealed and initial thoughts, trade-offs, and a system organization are presented. As a whole, this area presents new opportunities for our community to design, develop, and deploy solutions, which will help us learn from the past and anticipate the future. It's about time!

Acknowledgements

This work is supported by the 7^{th} Framework IST programme of the European Union through the focused research project (STREP) on Longitudinal Analytics of Web Archive data (LAWA) under contract no. 258105.