Publication Date Prediction through Reverse Engineering of the Web

Liudmila Ostroumova Prokhorenkova, Petr Prokhorenkov,
Egor Samosvat, Pavel Serdyukov
Yandex
Leo Tolstoy st. 16, Moscow, Russia
{ostroumova-la, eeight, sameg, pavser} @yandex-team.ru

ABSTRACT

In this paper, we focus on one of the most challenging tasks in temporal information retrieval: detection of a web page *publication date*. The natural approach to this problem is to find the publication date in the HTML body of a page. However, there are two fundamental problems with this approach. First, not all web pages contain the publication dates in their texts. Second, it is hard to distinguish the publication date among all the dates found in the page's text.

The approach we suggest in this paper supplements methods of date extraction from the page's text with novel link-based methods of dating. Some of our link-based methods are based on a probabilistic model of the Web graph structure evolution, which relies on the publication dates of web pages as on its parameters. We use this model to estimate the publication dates of web pages: based only on the link structure currently observed, we perform a "reverse engineering" to reveal the whole process of the Web's evolution.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Publication dates; web pages; link-based method; likelihood optimization

1. INTRODUCTION

In recent years, time dimension has been gaining increasing importance for search engines, leading to a new research area known as temporal information retrieval [3]. In this paper, we focus on one of its most challenging tasks: detection of a document *publication date*. Web page publication dates constitute an extremely valuable piece of knowledge for a variety of search tasks. For example, this information is essential for computing features for recency-sensitive ranking of web documents [4, 5]. Page publication dates can also be used in crawling policies [15]: links found on recently created pages usually lead to recently created pages with up-to-date content, therefore the corresponding pages should be crawled in priority.

Unfortunately, the publication dates of a large share of web pages cannot be easily and reliably determined. The most common way to determine the publication date of a web page is *content-based*, i.e., to find this date in the HTML body of this page. For example, some specific categories of web pages, like news articles, usually contain their publication dates within their content. Yet, even in this case, there are several difficulties: such pages may contain several candidate dates to choose from, these dates can be written in different formats and for different time zones, etc. Needless to say, the content-based approaches cannot detect publication dates for web documents that do not contain any text, e.g., images or videos. In this paper we suggest an algorithm which estimates the dates of web pages regardless of the presence of the exact publication date in their HTML bodies.

In some other cases, a web page's publication date can be considered equal to the date of the first crawl of that page. However, due to resource constraints, not all web-sites are re-crawled frequently enough to make it possible to detect new pages immediately after their publication. In addition, we may consider the scenario when a search engine plans to start operating in a new country. Since that country has not been considered as the search engine's target market until this moment, it is very unlikely that it has been investing a lot of resources into crawling and re-crawling the pages interesting mostly to the users of that country, therefore we cannot count on the first crawl dates to recover the publication dates of most pages. Similar situation arises when a search engine plans to expand its index in the same country (e.g., by deeper crawls).

The algorithm we suggest in this paper combines content-based methods of date extraction with novel *link-based methods*. We mostly focus on the publication date detection for pages which are not likely to contain the correct publication date in their HTML bodies. For such pages, we use the link structure in order to estimate their publication dates. We propose a group of multi-step *date propagation* methods, which iteratively estimate the dates of pages using the already known dates of their neighbors. Our approach generalizes and extends several one-step link-based methods proposed in the literature [14, 16]. We also propose a theoretically grounded approach which is based on a realistic model of the Web [10]. It was shown in [11] that the popularity of almost all new pages decays exponentially with time. Based on this observation, a realistic model of link

WSDM'16, February 22–25, 2016, San Francisco, CA, USA.

DOI: http://dx.doi.org/10.1145/2835776.2835796.

evolution was proposed in [10]. In this model, the publication dates of web pages are used to predict the evolution of the Web link structure. Here we do the reverse operation, i.e., we use this model to estimate the publication dates of web pages. Given only the currently observed link structure, we apply "reverse engineering" to reveal the whole process of the Web's evolution. Namely, we find such publication dates which maximize the probability that the Web graph observed in reality is produced by this model. To the best of our knowledge, we are the first to propose a model-based probabilistic method for document dating. To sum up, the contributions of this paper are the following:

- We suggest an algorithm for publication date detection which combines content-based methods of date extraction with link-based methods.
- As link-based methods, we propose a general approach to date propagation as well as a more sophisticated algorithm based on a realistic model of the Web.
- We suggest a method which allows us to reduce the computational complexity of our algorithms. As a result, the proposed algorithms run in linear time.
- We also propose a novel data-driven parameter selection method which adapts the algorithms to each host individually. The motivation behind this is that different parts of the Web may evolve by different rules, therefore, we should adapt our methods and parameters to these rules.

We perform the experiments on two different datasets: one dataset consists of about 4M web pages located on 70 different hosts, another one is MemeTracker dataset [1].

The rest of the paper is organized as follows. In the next section, we discuss different existing approaches to web documents' dating. In Section 3, we present our approaches and the model on which some of them are based. Section 4 describes the data used to analyze the performance of our algorithm. The experimental results are presented in Section 5. Finally, we conclude the paper and outline directions for future research.

2. RELATED WORK

The problem of publication date detection was analyzed in a variety of studies. In this section, we briefly describe the state-of-the-art approaches.

Extraction of publication date. The most widely used approaches to dating web documents are based on the recognition of temporal expressions in their texts [2, 7]. However, the problem of choosing the correct publication date from among the recognized candidates, which is central to document dating, is not discussed in these papers. A more complete solution to content-based dating is suggested in [13]. Here all three necessary steps are made: date extraction, normalization of candidate dates, choosing the correct publication date. However, this method can be applied only to pages which contain the publication dates in their texts or URLs.

Link-based methods. Link structure of a graph can also be used in order to estimate publication dates. For example, in [14] a link-based method was used to solve a different but related problem: they estimate the last update time, i.e., the moment when the content of a page was updated, while we are interested in the moment when the page was initially created. The following assumption is used in [14]: connected web resources tend to have similar update patterns. In order to estimate a document's Last-Modified value, the authors consider several types of neighbors, like documents containing links to the given document or documents pointed to by the given document. They observed a positive correlation between a document's Last-Modified value and the Last-Modified average over its neighbors. The strongest correlation occurs with a document's out-links. We use these methods as our baselines in this paper.

Both the publication date and the last modified date are useful for a search engine. For some pages, like the main pages of news sites, the last modified date shows whether the news articles referred by them are up-to-date. On the other hand, for the majority of web pages, their publication date is the date when their main content was created, while the Last-Modified value often corresponds to some minor layout changes. For example, the main content of news pages usually stays stable, while surrounding ads and dynamic links are constantly changing. Therefore, publication dates are often used as features in web search ranking [4].

In [16] backlinks are used to estimate the publication date of URI. Authors take the earliest timestamp out of the following: the first time someone shortened the URI, the first time someone tweeted the URI, the first time it appeared in a public web archive, etc. The method based on the same principle (taking the earliest of the publication dates of the pages linking the given one) we also use as our baseline.

In this paper, we generalize the notion of aggregation and consider different functions for aggregating the information over the neighbors of a page. We also generalize the notion of propagation: in [14] and [16], only one-step propagation is considered (i.e., only the nearest neighbors are used to predict the date of a page), while we assume that the propagation can be done in multiple steps.

Language models. Language models are also often applied to dating web documents [6, 8, 9]. Such models are useful when there is no explicit publication date in the body of a document. However, the problem is that the date detection granularity is too coarse for such approaches: their error rates are often measured in years, decades or even centuries.

In this paper, we suggest a new link-based method which, as any link-based method, has to be combined with a text-based date extraction approach: the reliable candidate publication dates extracted from documents' texts are used to detect the publication dates of documents that have no reliable (or any) candidate dates found in their content.

3. ALGORITHM

3.1 General scheme

In this section, we describe our algorithm for publication date detection. Obviously, at the beginning of the dating process no dates are known. In this situation, only a text (or metadata) based date extraction method can help to date some documents. Further, we can use link-based methods in order to approximate the publication dates of the rest of the documents, for which the content-based method could not determine their publication dates with sufficient certainty. It is important to note that the first stage of the dating process is unavoidable, since any link-based dating method requires some reliable publication dates ("anchor" dates) to be fixed for a fraction of documents, as it needs to be guided toward some specific time period the correct publication dates of the other documents most probably belong to. The general description of our algorithm is the following.

Stage 1: content-based extraction of anchor and seed dates. As we discussed, the first stage of our algorithm is to extract candidate dates from the URL and the HTML body of each page and choose the most probable publication date from among the candidates. Our content-based method of dates extraction is described in Section 3.2. For some pages it is possible to detect highly reliable *anchor dates*, which will be fixed for the rest of the algorithm. For some other pages, candidates dates can also be extracted, but they are less reliable and their estimates can be improved at Stage 3, such dates are called *seed dates*. For the rest of the pages, content-based date extraction is simply impossible.

Stage 2: anchor and seed dates propagation. Obviously, some pages may not have any dates in their texts. Therefore, at the second stage of our algorithm, we choose some approximations of dates for all such pages, i.e., for the pages without the seed or anchor dates. These estimates can be improved at Stage 3. We use and compare several *date propagation* methods. The description of these methods is given in Section 3.3. In our experiments, if some pages have no dates even after the date propagation, then we set some *constant* value for them.

Stage 3: likelihood optimization. Seed dates, propagated dates, and constant dates together form *initial dates* which can be improved at this stage by our *likelihood optimization* method. Our method is based on the model of the Web evolution, which is described in Section 3.4. We find the dates which maximize the probability that the observed graph is produced by this model. This stage is described in Section 3.5.

Let us sum up the notation. *Anchor dates* are trusted dates which are extracted from URLs or HTML bodies of pages and they cannot be changed during the link-based dating process. *Seed dates* are not such reliable dates extracted from the texts. *Propagated dates* are dates obtained as a result of the date propagation procedure. *Constant dates* are set for all other pages.

3.2 Anchor and seed dates extraction

Our research focuses on link-based dating methods, and such methods always work on the top of content-based approaches, though are not specific to any of them. In this paper, we use a state-of-the-art content-based method to extract dates. This method relies on several rules (regular expressions) in order to extract dates. The rules are similar to the ones described in [13], but they are adapted to Russian language. Several problems analyzed in [13] do not occur in our case: almost all pages are in one language, belong to one time zone, and most dates have only one interpretation (e.g., 03/04/05 usually means 3 April, 2005). In contrast to [13], we can extract dates not only from the URL, title, or snippet, but also from the HTML body of a page.

First, using our rules, we extract dates from the URL and the HTML body of a page. Then, as in [13], we remove apparently corrupted dates such as dates referring to a very old time or some time in the future. Then, in order to choose the best date, we look at where each candidate date was found: in the document's URL, title, main content, etc. We have a larger number of page components than [13], since we consider the entire page contents, not their summaries. We utilize the segmentator currently used by Yandex, the most popular search engine in Russia, which allows us to split document into zones/locations, such as title and main content of a document. We consider the following locations:

1. URL
2. Title
3. Just before main content
4. Just after main content
5. Main content of a document
6. All other locations

In Section 5, we analyze the reliability of dates found in different parts of a document. Based on these experimental results, we use the following method of choosing the anchor date. If there is a date found in the URL of a document, then we take this date. Otherwise, we take the date from the title. If there is no such date, then we take the date from before the main content of a document. If all three locations contain no candidate dates, then this page has no anchor date. The dates found at the other locations are less reliable, but they can be used as seed dates. In Section 5, we show that dates found after the main content are more reliable than the dates found in the main content, while the dates found in the main content are more reliable than the dates found in other locations (category 6 above).

3.3 Date propagation

Several date propagation methods were proposed in [14] and [16]. The idea of these approaches is to take the dates of a web page's immediate neighbors and use some function of them in order to estimate the date of the page. In [14], the authors propose estimating the last update time as the average of the last update times over either the documents containing links to the given document or the documents outlinked by the given document. In [16], the earliest date of the web pages containing links to a given page is used as the estimate for its publication date. We generalize and extend these approaches in two ways:

- We propose and compare various aggregation functions;
- We compare two propagation schemes: one-step propagation, as used in [14, 16], and multi-step propagation that we propose in this paper.

Let us first define several aggregation functions we consider in this paper. The three of them were already discussed: *in-avg* propagation takes the average over the dates of the documents containing links to the selected document, *out-avg* propagation takes the average over the dates of the documents pointed to by the selected document, *in-min* takes the minimum of the dates of the documents containing links to the selected document. We also propose *all-avg* propagation, which takes the average over all neighbors' dates, *out-max*, which takes the maximum over the out-neighbors dates, and *out-max-in-min*, which is the middle between *in-min* and *out-max*. The latter method is motivated by the assumption that a page p is usually created later than the pages pointed by p and earlier than the pages which contain links to p. Finally, one more aggregation method called *model-q* (q is a parameter, $0 \leq q \leq 1$) is discussed in Section 3.6.

We consider the seed and anchor date propagation not necessarily as a one-step, but also as a multi-step process. At each step, we consider all pages which have at least one already dated neighbor, i.e., a neighbor with an anchor date, a seed date, or a date assigned at the previous steps of the propagation process. For these pages we approximate their publication dates by taking a function of the dates of their neighbors (we consider both incoming and outgoing

links). Note that, at the first propagation step, as in [14, 16], we take a function of anchor and seed dates (obtained at Stage 1), while, at the next steps, we take a function of only the dates that had been obtained by the propagation at the previous steps. We continue this procedure until we cannot date more pages. Note that after applying any method or combination of methods from Sections 3.2 and 3.3 we may have pages that are not dated. In this case we set their dates to a predefined constant value (see Section 5.1).

3.4 Model description

In addition to the previously described propagation methods, we also propose a more sophisticated link-based approach which is motivated by the realistic probabilistic model of the Web [10]. The general approach from [10] models the behavior (in terms of links creation) of pages that may belong to different hosts. In this paper we mostly work with intra-site links, therefore here we describe the model with only one host[1]. If one needs to apply the method for many hosts and inter-links, it is just necessary to regard all hosts as one host in this case (and we do this for one of our datasets). Our model differs from the model [10] in several respects and further in this section we describe these differences and the reasons for them.

The model under consideration has several parameters. For each page p, we have the number of outgoing links m_p, its intrinsic quality q_p, and the publication time t_p. Besides these page-specific parameters, we also have the rate of attractiveness decay λ, an auxiliary constant c ($c > \lambda$), and the number of pages n. Note that the parameters n and m_p are completely defined by the link structure of the host. On the contrary, the parameters t_p (for pages without anchor dates), q_p, λ, and c are hidden, and will therefore be tuned during the optimization procedure (see Section 3.5).

The model can be described as follows. At the beginning we have n pages and no links between them. Each page p has its publication time t_p. Then, for each page p we generate m_p outgoing links. All links are modeled as mutually independent random variables that determine their target pages. The probability of a page r to be chosen as a target page for a link from a page p is proportional to the relative attractiveness of r according to p, which is a function of q_r (intrinsic quality of r) and the age difference $a_{p,r}$ for the pages p and r, that is, $t_p - t_r$. Note that in the model the difference $a_{p,r}$ can be negative, i.e., there is a possibility of an edge between p and r with $t_p < t_r$. In a real web graph such a link can be added at a moment $t > t_r$ if p was updated at t. The attractiveness function is defined as follows:

$$\mathrm{attr}(q_r, a_{p,r}) = \begin{cases} q_r \cdot e^{-\lambda a_{p,r}} \cdot \left(1 - \frac{e^{-c a_{p,r}}}{2}\right) & \text{if } a_{p,r} \geq 0\,, \\ q_r \cdot e^{-\lambda a_{p,r}} \cdot \frac{e^{c a_{p,r}}}{2} & \text{if } a_{p,r} < 0\,. \end{cases} \tag{1}$$

Let us discuss the case of $a_{p,r} \geq 0$. First of all, the attractiveness of r is proportional to its quality. Second, the attractiveness decreases with the age of r, i.e., older pages are less popular. These two multipliers are proposed and motivated in [10]. The third multiplier is explained further in this section. In the case of $a_{p,r} < 0$, as we also discuss further, the probability of a link is small and decreases exponentially with $a_{p,r}$.

We use this model in order to estimate the publication dates of web documents. Namely, we are given an oriented graph (nodes are the web documents and edges are the links between them) and we assume that this graph is constructed according to the procedure described above. We are given the observed values of some parameters (the numbers of outgoing links m_p and some anchor publication dates t_p) and want to find the rest of the unknown values to maximize the probability that the observed graph is constructed under the described model. The parameters with unknown values are: the rate of attractiveness decay λ, the constant c, the qualities of all pages q_p, the publication times for non-anchor pages t_p. The detailed description of our method of likelihood maximization is given in Section 3.5.

As we discussed earlier, the model we use in this paper slightly differs from the model described in [10]. The differences are the following.

- The number of outgoing links for each page is not a random variable, but a fixed number. We assume this since we know the number of outgoing links for each page from the real data.
- Pages appear at specific times t_p, which are not set randomly according to a Poisson process. In other words, in our case, t_p are the model parameters, this allows us to use a maximum-likelihood estimation method and estimate t_p from the real data. For the same reason, q_p are not random variables but parameters, also to be estimated from the real data.
- We change the attractiveness function a little. In [10] the attractiveness function is defined as $\mathrm{attr}(q_r, a_{p,r}) = q_r \cdot e^{-\lambda a_{p,r}} \cdot I(a_{p,r} > 0)$. As a result, edges going from older pages to newer ones are prohibited in [10]. We replace the indicator $I(x > 0)$ (the Heaviside step function) by the following sigmoid function:

$$f(x) = \begin{cases} 1 - \frac{e^{-cx}}{2} & \text{for } x \geq 0\,, \\ \frac{e^{cx}}{2} & \text{for } x < 0\,. \end{cases} \tag{2}$$

  Sigmoid functions are often used to smooth the step function. The larger the value c in (2), the closer the sigmoid function to the step function.

The latter modification helps us solve the following two problems.

1. In the original model [10], the probabilities of edges are not differentiable. On the other hand, our method of finding the optimum parameters requires continuously differentiable probabilities and here the sigmoid function helps.
2. Although, it was shown in [10] that in real data almost all links go from newer pages to older ones, there are always some edges which go conversely (e.g., a page can be updated and a link to some newer page can be added). In this case, according to the original model [10], the probabilities of some edges are equal to 0 and, regardless of the tuning of all the parameters, the probability of getting the observed graph is equal to 0. Therefore, we would not be able to find the optimum parameters in this case. We avoid this problem by using a sigmoid function, so that in our case this probability is always greater than 0.

Note that the above-mentioned problems can be solved by using any sigmoid function, for example, the well-known

[1]Although inter-site links can be successfully used in some applications, for example when a website mainly links to other websites rather than to its own pages, in our dateset (see Section 4.1) there are less than 1% of links which are inter-site (including those pointing outside the dataset).

logistic function. We choose the sigmoid function $f(x)$ defined in Equation (2) since it allows us to find the optimal parameters with an algorithm of lower complexity (see Appendix for details). We require $c > \lambda$ since we want to have $\mathrm{attr}(r) \to 0$ as $a_{p,r} \to -\infty$, i.e., we want the probabilities of edges which go from older pages to newer ones to be exponentially small. We do this since usually in real networks almost all links go from newer pages to older ones [10].

3.5 Likelihood optimization

Now we present our optimization algorithm which is based on the model described above. For each host, the algorithm consists of the following steps.

1. Fix c and λ.
2. Find initial approximations of t_i, and q_i for all pages.
3. Take several optimization steps for improving the initial estimates of t_i and q_i. While the estimates of q_i are improved for all pages, we do not change t_i for the pages with the anchor dates.
4. Select the appropriate stopping time.

Initial approximations. First, we fix c and λ. We choose the optimal values for each host according to the *data-driven parameter selection method* (see Section 3.7).

Then, we find the initial approximations of t_i, and q_i (for all pages i). Several methods of choosing the initial approximations of publication dates are described in Sections 3.3 and 3.6, namely, anchor dates, seed dates, propagated dates, and constant dates. Recall that anchor dates are reliable and we do not change them, while other dates can be corrected.

It was shown in [10] that a good model for the real data can be obtained if (1) the attractiveness of a page is proportional to its quality and decreases exponentially with its age, (2) the quality of a page is proportional to the number of its incoming links. Based on this observation, initially, we consider the quality q_i of a page i to be equal to the number of incoming links to this page.

Optimization steps. This is the most important part of the likelihood optimization algorithm. Recall that our primary goal is to improve the initial publication dates for the web pages without anchor dates. Since we assume that the model described in Section 3.4 is realistic, we want to find the values of the parameters which maximize the probability that a random graph G_{model} constructed according to the model coincides with the link graph G_{real} observed in the reality. We want to maximize the likelihood, that is (disregarding the constant multiplier $\prod_p m_p!$)

$$L(\bar{t}, \bar{q}) = \prod_{ij \in G_{real}} \mathrm{P}(ij \in G_{model}) \,. \qquad (3)$$

By i and j we denote nodes of a graph and the notation $ij \in G$ means that the oriented edge from i to j belongs to the graph G. Further in the paper we assume that the nodes are ordered according to their publication times, i.e., $i \le j$ if and only if $t_i \le t_j$. Edges are generated independently, therefore we can just consider their product in Equation (3). According to the model, the likelihood function $L(\bar{t}, \bar{q})$ is the following:

$$L(\bar{t}, \bar{q}) = \prod_{ij \in G_{real}, i \ge j} \frac{q_j e^{-\lambda(t_i - t_j)} \left(1 - \frac{e^{-c(t_i - t_j)}}{2}\right)}{W(t_i)} \cdot$$

$$\cdot \prod_{ij \in G_{real}, i < j} \frac{q_j e^{-\lambda(t_i - t_j)} \frac{e^{c(t_i - t_j)}}{2}}{W(t_i)},$$

where $W(t_i)$ is the total weight of all pages at time t_i:

$$W(t_i) = \sum_{j \le i} q_j e^{-\lambda(t_i - t_j)} \left(1 - \frac{e^{-c(t_i - t_j)}}{2}\right) + \\ + \sum_{j > i} q_j e^{-\lambda(t_i - t_j)} \frac{e^{c(t_i - t_j)}}{2} \,. \qquad (4)$$

The weight $W(t_i)$ appears in the denominator since we have to normalize attractiveness in order to obtain a probability.

The idea of the optimization algorithm is the following. For each page p (without an anchor date) we want to improve our guess about its publication time t_p. It means that we want to find t_p and q_p that maximize $L(t_p, q_p)$ in the two-dimensional space. Our method of finding all such optimal points simultaneously is based on the gradient descent method[2]. This method requires the computation of all partial derivatives of the function $\log L(\bar{t}, \bar{q})$ with respect to t_i and q_i. These derivatives we compute in Appendix. For each page p we make one step in the quality dimension (in the right direction according to the corresponding derivative, see Equation (6)). And for all pages without the anchor dates we make one step in the time dimension (see Equation (5)). At each step we change quality and time by ± 1 (i.e., one day for the time dimension and one unit for the quality dimension). In Appendix, we describe an efficient algorithm for computing *all* the derivatives *in linear time*. Finally, the complexity of the likelihood optimization algorithm is $O(N\,e)$, where e is the number of edges and N is the number of optimization steps. Note that the complexity of one step of any date propagation method is $O(e)$.

Stopping criteria. From the beginning, we suspected that our optimization algorithm works better for some hosts, while for others it can be less useful. We apply a data-driven parameter selection algorithm (see Section 3.7) in order to find the optimal number of steps of the optimization procedure for each host and, in particular, to decide whether we should use this optimization for this host or not.

3.6 Simplification of likelihood optimization

In this section, we answer the following question: is there any similarity between the date propagation strategies and the likelihood optimization? In particular, we show that the likelihood optimization algorithm (implicitly and approximately) uses an aggregation function. At the end of this section, we propose new aggregation functions which are based on the probabilistic model described above.

It was shown in [10] that $W(t_i)$ is asymptotically a constant. So, we can replace $W(t_i)$ in the likelihood function $L(\bar{t}, \bar{q})$ by a constant W. Because of this approximation, the qualities of individual pages become just multipliers in the likelihood function $L(\bar{t}, \bar{q})$, so we further assume $q_i = 1$ for all i. These assumptions substantially simplify all the above reasonings (see Section 3.5). Let us now try to find the optimal date for a page p given the dates of all its neighbors. The function we want to maximize depends only on the probabilities of incoming and outgoing links for p:

$$\prod_{pi \in G_{real}} \mathrm{P}(pi \in G_{model}) \cdot \prod_{jp \in G_{real}} \mathrm{P}(jp \in G_{model})$$

$$= \prod_{pi \in G_{real}} \frac{e^{-\lambda(t_p - t_i)} f(t_p - t_i)}{W} \cdot \prod_{jp \in G_{real}} \frac{e^{-\lambda(t_j - t_p)} f(t_j - t_p)}{W} \,.$$

[2] *http://en.wikipedia.org/wiki/Gradient_descent*

Now, if we take a logarithm of this function and approximate $f(x)$ by

$$\tilde{f}(x) = \begin{cases} 1 & \text{for } x \geq 0\,, \\ \frac{e^{cx}}{2} & \text{for } x < 0\,, \end{cases}$$

then the function we have to maximize is

$$F(t_p) = -\lambda \sum_{pi \in G_{real}} (t_p - t_i) - \lambda \sum_{jp \in G_{real}} (t_j - t_p)$$

$$+c \sum_{pi \in G_{real}, t_p < t_i} (t_p - t_i) + c \sum_{jp \in G_{real}, t_j < t_p} (t_j - t_p)\,.$$

Let us find the solution to this maximization problem. Let $deg_{in}(p)$ be the number of incoming links to p, $deg_{out}(p)$ be the number of outgoing links from p, and $deg(p) = deg_{in}(p) + deg_{out}(p)$ be the total number of links. Consider all neighbors of p (both incoming and outgoing) and sort their publication times: $T_1 \leq \ldots \leq T_{deg(p)}$. Clearly, $F(t_p)$ is continuous on $(-\infty; +\infty)$ and differentiable on $(-\infty; +\infty)$ except at the points T_i, $1 \leq i \leq deg(p)$. The derivative $F'(t_p)$ is equal to

$$F'(t_p) = \lambda(\deg_{in}(p) - deg_{out}(p))$$

$$+c\left(|\{pi \in G_{real}, t_p < t_i\}| - |\{jp \in G_{real}, t_j < t_p\}|\right)\,.$$

Therefore, $\text{argmax}(F(t_p)) = T_k$, where

$$k = \lceil deg_{out}(p) - \frac{\lambda}{c}(deg_{out}(p) - deg_{in}(p)) \rceil\,.$$

However, if the value $deg_{out}(p) - \frac{\lambda}{c}(deg_{out}(p) - deg_{in}(p))$ is integer, then $F'(t_p) = 0$ on the interval $[T_k; T_{k+1}]$, and $\text{argmax}(F(t_p))$ is the whole interval $[T_k; T_{k+1}]$, so, in this case, we choose $t_p = \frac{T_k + T_{k+1}}{2}$.

Interestingly, the solution to this simplified problem depends only on the ratio λ/c, not on both of them. We denote this ratio by q, $0 \leq q \leq 1$. Note that if $c = 2\lambda$ ($q = 0.5$), then the optimal t_p is the median of its neighbors' dates.

The obtained result gives us a group of date propagation methods which we call *model-q* propagation, $0 \leq q \leq 1$. We use the obtained aggregation function with the multi-step propagation principle. The optimal q can be chosen by the data-driven parameter selection method, described in the next section.

3.7 Data-driven parameter selection

Our motivation for this approach is that different hosts obviously have different properties, therefore different methods/parameters can be optimal for them. We propose a data-driven parameter selection method in order to answer three questions:

- How many steps of the likelihood optimization should be applied?
- Which λ and c are optimal?
- Which date propagation method is the best for a host?

In order to find the best parameters we do the following. We take all host's pages with anchor dates and divide them into 5 equal disjoint subsets (buckets). Then, for each bucket B, we "forget" all anchor dates for the pages from this bucket. After this, we apply the set of algorithms we want to compare and measure the mean absolute error on pages in B (we compare the estimated dates with the actual anchor dates). Then we average the errors over all 5 buckets and choose the best combination of parameters, i.e., the one which gives the minimum average error.

4. DATA

In order to measure the performance of our algorithms we use two different datasets. The first dataset is a large sample of the web pages crawled by Yandex over a period of 17 months. Another dataset is the publicly available MemeTracker dataset [1].

4.1 Crawled dataset

Data sources. In this section, we describe the data sources we use in order to obtain the first dataset.

The first data source is the collection of documents crawled by Yandex. For each crawled page p we know the date when it was crawled t_p^{crawl}, the page's text, and its outgoing links. The second data source is the logs of Yandex.Metrica (*https://metrica.yandex.com/*) — a special tool for a site owner which allows to analyze user activity on the site (this tool is distributed by Yandex and is similar to, e.g., Google Analytics). In these logs, all anonymized user visits to the web pages of sites with the tool installed are recorded.

For our experiments we need both data sources. Further on we explain how we use them in order to get the ground truth dates.

Dataset. First, we sampled 10 random hosts H_i from one week (in December 2012) logs of Yandex.Metrica. We consider the set of all pages P_i on the host H_i which were crawled by the search engine during the period from January 2013 to May 2014. This dataset (D_1) is used for our preliminary experiments (see Section 5.1).

We also collected the additional data for 60 random hosts sampled in the same way. We use this dataset (D_2) for the final validation of our algorithms on the crawled data.

Overall, our crawled dataset consists of 4M pages from 70 hosts, the smallest host is of size 5K pages, the largest one is of size 1.2M pages. For each page p in the dataset we collected the following data: URL, document's text, the time of the first crawl t_p^{crawl}, outgoing links, and the first user visit recorded in the logs of Yandex.Metrica t_p^{visit}.

Ground truth dates. In order to compare our algorithm with the baseline strategies, we need to obtain the true publication dates for a set of web pages. It is a typical problem for link-based methods and methods based on language models. If some pages have real publication dates in their URL or HTML body, then some sample can be manually annotated. If there is no publication date anywhere on a web page, then it is impossible to even manually annotate a date. However, in this paper we mostly focus on the web pages without explicit dates in their body. Therefore, we have to somehow determine the actual publication dates for many such web pages somehow.

In order to determine dates we use two sources of information: the date t_p^{crawl} when a page p was first crawled by the search engine and the date t_p^{visit} when p was first visited by a user. The dates t_p^{crawl} and t_p^{visit} may coincide with the date of creation or they may be later. Finally, we use the following approximation of the ground truth: if for a page p we have $|t_p^{crawl} - t_p^{visit}| \leq 24$ hours, then this page is created at $t_p^{credible} = min\{t_p^{crawl}, t_p^{visit}\}$. We choose a 24 hours threshold since we measure error in days in this paper. Only 93K pages (∼3%) in our dataset D_1 have $t_p^{credible}$ time.

Admittedly, $t_p^{credible}$ is not 100% correct publication date, but we argue that this is the best approximation that a search engine can get for such pages. In order to verify the quality of the obtained dataset we take a random sample of

1K pages (100 per host for dataset D_1) which have $t_p^{credible}$ times and manually annotate dates for them. There are three possible situations for each such page:

- If the publication date can be found in the text of a document and it coincides with $t_p^{credible}$ date, then we say that $t_p^{credible}$ is correct.
- If the publication date can be found in the text of a document and it differs from $t_p^{credible}$ date, then we say that $t_p^{credible}$ is incorrect.
- If the publication date cannot be found in the text of a document, then we cannot say anything about the correctness of $t_p^{credible}$.

We get the following result. Out of 1K manually annotated dates only 19 have incorrect seed dates. On the other hand, 398 pages have no publication dates in their texts or URLs. These pages are not uniformly distributed over the hosts, namely, there are 3 hosts where almost all pages have unknown dates. This means that any method which extracts dates from the text is useless for such hosts. Remaining 583 pages have publication dates in their texts or URLs and these dates coincide with $t_p^{credible}$ date. Based on the above observations, further we validate all the algorithms on pages with $t_p^{credible}$ date.

4.2 Memetracker dataset

We also use MemeTracker public dataset [1], which covers about 96M blog posts and news articles published during 9 months from August, 2008 to April, 2009, and 418M links between them. See [12] for the details on how this data was collected. Our primary reason for using this additional dataset is to put aside the problem of date extraction from the page's text and to focus on the main problem — link-based date estimation. The pages in this dataset have publication timestamps. For our experiments we kept only links pointing to the documents also in the dataset, i.e., links with known timestamps both for the source and the destination. We finally obtained a dataset of about 29M links and 12M documents that we use in the following experiments. Since we know the time when each document was posted on the Web, in this dateset we have the ground truth dates for all the pages. For this dataset we do not analyze date extraction algorithms and focus only on link-based methods.

Our dataset consists of the pages belonging to about 250K different hosts. This means that a lot of hosts are represented by a very small number of pages: e.g., about 65K hosts are represented by only one page. Therefore, analyzing the hosts individually will not be effective. Further, we treat MemeTracker dataset as one host.

5. EXPERIMENTS

In this section, we compare several algorithms of publication dates estimation. All algorithms assign publication dates to all the pages in a dataset and then we compare their precision. As the measures of precision we use the mean absolute error, the median of the absolute errors, which is more resistant to outliers, and the *mean weighted absolute error*: $\sum_p \frac{deg_{in}(p)\cdot \mathrm{AbsErr}(p)}{\sum_i deg_{in}(i)}$, i.e., for each page p the error $\mathrm{AbsErr}(p)$ is weighted by the normalized number of incoming links $deg_{in}(p)$. The latter measure reflects how precisely we can estimate dates for more important pages. All metrics are measured in days.

We compare the following algorithms. First, we consider the 1-step propagation algorithms: *1-step in-avg*, *1-step out-avg*, *1-step all-avg*, *1-step in-min*, *1-step out-max*, *1-step out-max-in-min* (see Section 3.3). Note that our baselines are *1-step in-avg*, *1-step out-avg* from [14], and *1-step in-min* from [16]. Second, we consider the multi-step propagation algorithms: *in-avg*, *out-avg*, *all-avg*, *in-min*, *out-max*, *out-max-in-min*, *model-q* (the last one is defined in Section 3.6). Finally, we consider the likelihood optimization algorithm (see Section 3.5).

Location	mean / median / weighted err.	coverage (by non-constant)
Constant date	124 / 113 / 159	0%
+ URL	95.5 / 94.2 / 145	19.6%
+ before main content	87.3 / 92.1 / 103	20.1%
+ title	80.0 / 76.3 / 95.7	20.6%
+ after main content	79.1 / 75.4 / 95.2	20.8%
+ main content	78.8 / 74.0 / 87.7	21.3%
+ other	59.1 / 35.2 / 72.7	47.2%

Table 1: Errors for different seed dates on D_1

5.1 Experiments on crawled dataset D_1

In this section, we analyze the reliability of dates extracted from different parts of a document (see Section 3.2). Based on the results of these experiments, we choose seed dates for our optimization algorithm. All the experiments here are performed on dataset D_1.

We consider all pages in dataset D_1 and the dates extracted from their URLs and HTML bodies. First, we measure the precision of each location separately in order to find the most reliable locations. We noticed that the most reliable location is *URL* (the mean absolute error is 1.0 day) followed by *before main content* location (4.3 days) and *title* (4.8 days). If a date was found in one of these three locations on the page, then we say that this page has an anchor date (see Section 3.1). The other locations are less reliable: *after main content* (34 days), *main content* (74 days), *other* (88 days). If a date was found in one of these three locations, then we say that this date is the seed date. Finally, about a half of the pages in dataset D_1 have dates in their texts (i.e., they have a seed or an anchor date assigned) and 20% of the pages have an anchor date.

Now we want to compare several strategies which assign dates to *all* web pages based only on their URLs and HTML bodies. The first very basic strategy assigns constant dates to all web pages. As expected, the best constant date is the middle of the considered time interval, i.e., the middle of September, 2013. If we assigned this date to all the pages in our dataset, then the mean absolute error would be equal to 124 days (see Table 1)[3]. Then, we order the locations according to their mean errors and analyze different combinations of locations. First, we add the dates found in URLs, i.e., if a document has a date in its URL, then we assign this date, otherwise we assign the constant date. We further add more and more locations and measure the mean, the median and the mean weighted absolute errors (see Table 1). The last column in this table shows the fraction of non-constant dates for a given combination of locations. However recall that we average errors over *all* web pages.

[3]In practice, we may not have any "considered time interval", especially if a search engine comes to a new market. In these cases, a practitioner may choose to assign no date instead of a constant date or assign the date to be the date of the first crawl. We use the constant dates in order to fairly compare the precision of all algorithms given the full coverage.

	Method	mean/median/ weighted err	coverage
baselines	seed and anchor dates	57.1/30.2/70.0	47.7%
	1-step in-avg	55.9/29.8/56.9	49.3%
	1-step out-avg	56.3/27.6/68.2	64.1%
	1-step in-min	58.0/29.8/66.0	49.3%
proposed methods	1-step all-avg	55.5/27.7/57.0	64.4%
	1-step out-max	56.9/28.1/67.5	64.1%
	1-step out-max-in-min	56.7/30.3/70.0	49.0%
	in-avg	55.5/29.7/56.5	49.7%
	out-avg	58.0/25.0/70.1	68.6%
	all-avg	52.4/22.3/55.6	69.7%
	in-min	57.8/29.7/65.8	49.7%
	out-max	57.1/22.3/72.7	68.6%
	out-max-in-min	56.6/30.0/70.2	49.4%
	model-0.5	51.9/19.7/53.9	69.7%
	model-0.6	**51.2**/19.7/**53.6**	69.7%
	model-0.7	51.4/**19.5**/53.7	69.7%
	likelihood optimization	**49.9/19.1/51.2**	69.7%

Table 2: Comparison of the algorithms on D_2

	Method	mean/median/ weighted err	coverage
baselines	anchor dates	77.6/79.7/89.6	50.0%
	1-step in-avg	71.0/73.3/39.7	57.9%
	1-step out-avg	66.8/66.6/41.7	65.5%
	1-step in-min	71.5/73.6/77.8	57.9%
proposed methods	1-step all-avg	64.1/62.3/38.6	68.6%
	1-step out-max	67.4/67.0/50.0	65.5%
	1-step out-max-in-min	73.5/75.8/58.0	54.8%
	in-avg	70.2/72.5/39.0	58.9%
	out-avg	64.5/62.4/39.6	69.0%
	in-min	70.9/73.0/77.2	58.9%
	all-avg	**58.0**/52.6/37.9	75.3%
	out-max	65.5/63.1/48.1	69.0%
	out-max-in-min	72.7/75.2/56.3	55.8%
	model-0.3	58.3/52.3/43.0	75.3%
	model-0.4	**58.0**/**52.0**/38.1	75.3%
	model-0.5	58.2/52.2/**35.7**	75.3%
	likelihood optimization	**57.4/51.3/33.4**	75.3%

Table 3: Comparison of the algorithms on MemeTracker

5.2 Experiments on crawled dataset D_2

In this section, we analyze the performance of our methods and compare them with the baseline approaches on dataset D_2.

Date propagation. We analyze and compare different date propagation methods. First, we extract seed and anchor dates as described in the previous section. Then, we compare several date propagation functions (see Sections 3.3 and 3.6). The results of this comparison are presented in Table 2. We compared several values of q in model-q propagation ($q = 0.0, 0.1 \ldots, 1.0$) on dataset D_1 also. The best results were obtained with $q = 0.6$. In Table 2 we present the results for $q \in \{0.5, 0.6, 0.7\}$ on dataset D_2. Note that the optimal value for each host can be tuned by our data-driven parameter selection method. In Table 2, we are mostly interested in the second column, which shows the mean/median/weighted errors measured on all pages. Model-q propagation shows the best results according to all metrics. Note that we set constant dates to the pages which are not dated, so the third column demonstrates the coverage of the dating method (i.e., the fraction of non-constant dates after the dating procedure) and the second column shows the average errors over all pages, including those which were assigned with the constant date. Usually, the multi-step propagation has a smaller error than one-step propagation because of the higher coverage.

We also applied our data-driven parameter selection approach in order to choose the best propagation strategy for each host. This allowed us to improve the mean absolute error up to 50.5 days, although, median absolute error and mean weighted absolute error slightly increased to 19.9 and 54.0 respectively. Note that the parameter selection method minimizes mean absolute error, so there is no guarantee that any other measures will improve (if one were to consider another measure as the most important, that measure should be used as the target of minimization).

Likelihood optimization. First, for each host we choose the best method among all propagation methods with our data-driven parameter selection approach. The dates obtained by this propagation are used as initial dates for the likelihood optimization. Second, we use the parameter selection method to find out if the likelihood method is better (according to the mean absolute error) with any settings of its parameters (we fix the maximum number of steps $N = 100$). We noticed that our data-driven parameter selection approach chooses zero number of steps to about 40% of hosts, which means that for other 60% the likelihood optimization is predicted to be useful with some values λ, c, $N_0 > 0$. Likelihood optimization allowed us to get better results according to all measures. Finally, we get 11%, 31%, and 10% improvements of mean, median, and weighted errors respectively over the best (according to the corresponding metric) of the baselines.

Note that, for instance, in the case of web search ranking [4], even small improvements in the publication date estimates may affect the quality of ranking. Since the quality is of the highest priority in such applications (used by hundreds of millions of people daily), it should be preferred to periodically apply the best method, i.e., the likelihood optimization. For some other applications, where the complexity can be an issue, it might be preferred to use our multi-step date propagation methods of lower complexity, which demonstrate slightly worse but still sufficiently good results (see Appendix for the discussion of complexity).

5.3 Experiments on MemeTracker

Recall that this dataset consists of about 12 million web pages with known dates. In order to measure the performance of the algorithms we take x% random web pages and "forget" their dates. We consider all dates that are left as anchor ones. Then, we find initial approximations for the forgotten dates by using the date propagation methods. After that, we apply the likelihood optimization procedure. In this section, we measure the average errors only using the pages with forgotten dates. Mean errors measured over *all* pages would be two times smaller (since half of the pages have zero errors). We take 15th of December as the constant date since it is the middle of the considered time interval.

The results for $x = 50\%$ are presented in Table 3. The value $x = 50\%$ is chosen according to our observations on the crawled dataset, where we observe that about half of the pages have no seed or anchor dates (see the coverage for seed dates in Table 2). We applied our data-driven parameter selection algorithm to choose the propagation method and it chose *model-0.4*. Then, we apply the likelihood optimization algorithm. The data-driven parameter selection method predicts the likelihood optimization to be useful in this case (with 80 steps). Note that the obtained likelihood optimization algorithm is the best according to all three measures.

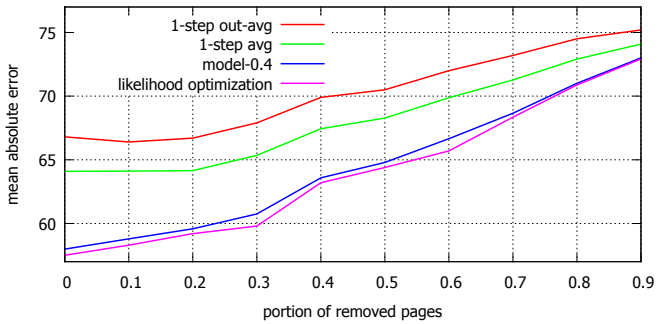

Figure 1: Influence of sparsity in data: mean absolute error

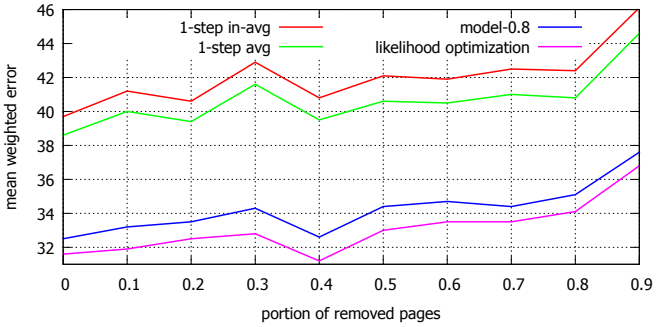

Figure 2: Influence of sparsity in data: mean weighted error

Performance on reachable pages only. Recall that we average errors over all web pages with forgotten dates, although about a half of these pages do not have any date even after the multi-step date propagation; these pages are assigned the constant date and all link-based methods are useless for such pages (they are not connected by any path to any dated page). If we exclude these pages from the evaluation and measure the error on the remaining pages, then we obtain more distinguishable results: e.g., the mean error of the best baseline *1-step out-avg* is **56.3** days, of *model-0.4* is **38.9** days (**-31%**), of the likelihood optimization is **37.7** days (**-33%**). On the crawled dataset we can make similar observations.

Dataset size influence. In addition, we analyze the influence of the dataset size on the quality of our algorithms. In order to do this, we thin out the data: we progressively remove a bigger fraction of web pages, and then, as before, we assume that for the half of the remaining pages the dates are not known and we estimate these dates with our methods. The obtained mean absolute error is shown on Figure 1. Here we present the following methods: the best of the baselines *1-step out-avg*, the best one-step method *1-step avg*, the best multi-step method *model-0.4*, and the likelihood optimization method. As expected, a sparser link structure leads to worse performance of the algorithms. We also demonstrate that it is possible to improve the weighted error measure much more if our parameter selection approach directly optimizes this measure. Figure 2 is analogous to Figure 1, but it presents the mean weighted error for all the algorithms. Here the best baseline is *1-step in-avg* and the best multi-step algorithm is *model-0.8*. The parameters of the likelihood optimization algorithm are chosen according to the data-driven parameter selection approach which optimizes the mean weighted error. It turns out that the weighted error is much less sensitive to the sparsity in data than the mean absolute error.

Finally, it is worth mentioning that all the improvements obtained on both datasets are statistically significant according to the paired t-test at less than 0.001 level.

6. CONCLUSION AND FUTURE WORK

In this paper, we suggested and compared several methods for publication dates estimation. We primarily focused on link-based methods: we systematized and generalized several simple date propagation methods previously proposed in the literature and we also suggested a more sophisticated likelihood optimization based method. In addition, we proposed the data-driven parameter selection approach which allows to tune our algorithms individually for each host. We demonstrated the improvements over the baseline approaches on two large Web samples: the first one consists of pages crawled by a major commercial search engine and the second one is MemeTracker public dataset. For example, the mean error is improved by 10% and 14% on the crawled and MemeTracker datasets respectively. If we exclude isolated pages from the evaluation, we get up to 33% improvements.

The most promising direction for the future research is to deeper analyze the hosts for which the likelihood optimization is predicted not to be useful. The link structure can evolve by some other rules in such cases, therefore it is reasonable to develop and apply a more suitable model for these hosts.

7. REFERENCES

[1] http://www.memetracker.org/data.html.
[2] D. Ahn, J. Rantwijk, and M. Rijke. A cascaded machine learning approach to interpreting temporal expressions. *Proc. of NAACL-HLT Conference*, 2007.
[3] R. Campos, G. Dias, A. M. Jorge, and A. Jatowt. Survey of temporal information retrieval and related applications. *ACM Computing Surveys*, 47, 2014.
[4] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz. Towards recency ranking in web search. *Proc. WSDM*, pages 11–20, 2010.
[5] A. Dong, R. Zhang, P. Kolari, and J. Bai. Time is of the essence: Improving recency ranking using twitter data. *Proc. WWW*, pages 331–340, 2010.
[6] N. Kanhabua and K. Nørvåg. Using temporal language models for document dating. *Proc ECML PKDD*, 2009.
[7] O. Kolomiyets and M.-F. Moens. Comparing two approaches for the recognition of temporal expressions. *KI 2009: Advances in Artificial Intelligence*, 2009.
[8] A. Kumar, J. Baldridge, M. Lease, and J. Ghosh. Dating texts without explicit temporal cues. *arXiv preprint arXiv:1211.2290*, 2012.
[9] A. Kumar, M. Lease, and J. Baldridge. Supervised language modeling for temporal resolution of texts. *Proc. of CIKM*, pages 2069–2072, 2011.
[10] D. Lefortier, L. Ostroumova, and E. Samosvat. Evolution of the media web. *Proc. of WAW, LNCS*, 8305:80–92, 2013.
[11] D. Lefortier, L. Ostroumova, and E. Samosvat. Timely crawling of high-quality ephemeral new content. *Proc. CIKM*, pages 745–750, 2013.
[12] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. *KDD*, 2009.
[13] Y. Lu, W. Meng, W. Zhang, K.-L. Liu, and C. Yu. Automatic extraction of publication time from news search results. *Proc. ICDE Workshops*, 2006.
[14] S. Nunes, C. Ribeiro, and G. David. Using neighbors to date web documents. *Proc. WIDM*, 2007.
[15] C. Olston and M. Najork. Web crawling. *Foundations and Trends in Information Retrieval*, 4(3):175–246, 2010.
[16] H. M. Salah and M. L. Nelson. Carbon dating the web: estimating the age of web resources. *Proc. WWW*, 2013.

APPENDIX

Complexity reduction. Let us discuss the complexity of the likelihood optimization algorithm which optimizes q_p for all pages and t_p for all pages without anchor dates. Let n and e be the number of nodes and the number of edges in the graph G_{real}, respectively. The naïve straightforward method allows to compute $W(t_i)$ for all i in $O(n^2)$. After this, $L(\bar{t},\bar{q})$ can be computed in $O(n^2+ne)$. If we want to use the gradient descent method for each node, then one step of the gradient descent for only one node can be made in $O(n^2+ne)$. Making one optimization step for all nodes would take $O(n^3+n^2e)$. Since we have hundreds of thousands pages on one host, this complexity is far too high. That is why we propose an efficient method which allows to simultaneously optimize qualities and dates for *all* pages and allows to make one optimization step for all pages with $O(n+e)$ complexity.

We assume that nodes are sorted according to the current estimates of their publication times, i.e., $t_1 \le \ldots \le t_n$. First, we compute $W(t_i)$ for all i in $O(n)$:

$$W(t_i) = \sum_{j\le i} q_j e^{-\lambda(t_i-t_j)}\left(1-\frac{e^{-c(t_i-t_j)}}{2}\right) + \sum_{j>i} q_j e^{-\lambda(t_i-t_j)}\frac{e^{c(t_i-t_j)}}{2} = e^{-\lambda t_i}\sum_{j\le i} q_j e^{\lambda t_j} - \frac{e^{-(\lambda+c)t_i}}{2}\sum_{j\le i} q_j e^{(\lambda+c)t_j} + \frac{e^{(c-\lambda)t_i}}{2}\sum_{j>i} q_j e^{-(c-\lambda)t_j}\,.$$

We can precompute all partial sums in the last equality in $O(n)$. After that, the values $W(t_i)$ for all i can be computed in $O(n)$. And the value $L(\bar{t},\bar{q})$ can be computed in $O(n+e)$.

Now we can compute the partial derivatives of $L(\bar{t},\bar{q})$ with respect to t_i for all i in $O(n+e)$. Let us start with some auxiliary observations

$$g(x) = \frac{\partial}{\partial x} f(x) = \begin{cases} \frac{ce^{-cx}}{2} & \text{for } x \ge 0\,,\\ \frac{ce^{cx}}{2} & \text{for } x < 0\,,\end{cases}$$

$$\frac{\partial}{\partial t_k} f(t_i-t_j) = (\delta_{ik}-\delta_{jk})g(t_i-t_j)\,,$$

$$\frac{\partial}{\partial t_k} e^{-\lambda(t_i-t_j)} = -\lambda(\delta_{ik}-\delta_{jk})e^{-\lambda(t_i-t_j)}\,.$$

Recall that $W(t_i) = \sum_j q_j e^{-\lambda(t_i-t_j)} f(t_i-t_j)$. Now we calculate partial derivatives of $W(t_i)$ with respect to t_k:

$$\begin{aligned}\frac{\partial W(t_i)}{\partial t_k} &= \sum_j q_j e^{-\lambda(t_i-t_j)}(\delta_{ik}-\delta_{jk})\left(g(t_i-t_j)-\lambda f(t_i-t_j)\right)\\ &= \delta_{ik}\sum_j q_j e^{-\lambda(t_i-t_j)}\left(-\lambda f(t_i-t_j)+g(t_i-t_j)\right)\\ &\quad -\sum_j q_j e^{-\lambda(t_i-t_j)}\delta_{jk}\left(-\lambda f(t_i-t_j)+g(t_i-t_j)\right)\\ &= \delta_{ik}\left(V(t_i)-\lambda W(t_i)\right) - q_k e^{-\lambda(t_i-t_k)}\left(g(t_i-t_k)-\lambda f(t_i-t_k)\right)\\ &= \delta_{ik}\left(-\lambda W(t_i)+V(t_i)\right) - q_k h(t_i-t_k)\,,\end{aligned}$$

where $h(x) = e^{-\lambda x}\left(g(x)-\lambda f(x)\right)$ and

$$V(t_i) = \sum_j q_j e^{-\lambda(t_i-t_j)} g(t_i-t_j)\,.$$

Similar to $W(t_i)$, all $V(t_i)$ can be computed in $O(n)$.

Recall that

$$L(\bar{t},\bar{q}) = \prod_{ij\in G_{real}} \frac{q_j e^{-\lambda(t_i-t_j)} f(t_i-t_j)}{W(t_i)}\,.$$

Now we calculate

$$\begin{aligned}&\frac{\partial}{\partial t_k}\log L(\bar{t},\bar{q})\\ &= \frac{\partial}{\partial t_k}\sum_{ij\in G_{real}}\left(\log q_j - \lambda(t_i-t_j) + \log f(t_i-t_j) - \log W(t_i)\right)\\ &= \sum_{ij\in G_{real}} -\lambda(\delta_{ik}-\delta_{jk}) + \sum_{ij\in G_{real}}\frac{(\delta_{ik}-\delta_{jk})g(t_i-t_j)}{f(t_i-t_j)}\\ &\quad -\sum_{ij\in G_{real}}\frac{\delta_{ik}\left(-\lambda W(t_i)+V(t_i)\right)-q_k h(t_i-t_k)}{W(t_i)}\\ &= -\lambda(deg_{out}(k)-deg_{in}(k)) + \sum_{kj\in G_{real}}\frac{g(t_k-t_j)}{f(t_k-t_j)}\\ &\quad -\sum_{ik\in G_{real}}\frac{g(t_i-t_k)}{f(t_i-t_k)} - \left(\frac{deg_{out}(k)\left(-\lambda W(t_k)+V(t_k)\right)}{W(t_k)}\right)\\ &\quad + q_k\sum_i \frac{deg_{out}(i)h(t_i-t_k)}{W(t_i)}\\ &= -\frac{V(t_k)}{W(t_k)}deg_{out}(k) + \lambda\, deg_{in}(k) + \sum_{kj\in G_{real}}\frac{g(t_k-t_j)}{f(t_k-t_j)}\\ &\quad -\sum_{ik\in G_{real}}\frac{g(t_i-t_k)}{f(t_i-t_k)} + q_k\sum_i\frac{deg_{out}(i)h(t_i-t_k)}{W(t_i)}\,.\end{aligned} \quad (5)$$

The first four terms in the last equation can be computed in $O(n+e)$. It remains to compute

$$q_k\sum_i\frac{deg_{out}(i)h(t_i-t_k)}{W(t_i)} = q_k\sum_{i\ge k}\frac{deg_{out}(i)h(t_i-t_k)}{W(t_i)}$$

$$+q_k\sum_{i<k}\frac{deg_{out}(i)h(t_i-t_k)}{W(t_i)} = -\lambda q_k e^{\lambda t_k}\sum_{i\ge k}\frac{deg_{out}(i)e^{-\lambda t_i}}{W(t_i)}$$

$$+\frac{(\lambda+c)q_k e^{(\lambda+c)t_k}}{2}\sum_{i\ge k}\frac{deg_{out}(i)e^{-(\lambda+c)t_i}}{W(t_i)}$$

$$+\frac{(c-\lambda)q_k e^{(\lambda-c)t_k}}{2}\sum_{i<k}\frac{deg_{out}(i)e^{(c-\lambda)t_i}}{W(t_i)}\,.$$

This expression for all k can be computed in $O(n)$.

The partial derivatives with respect to q_k can be computed in a similar way:

$$\begin{aligned}\frac{\partial}{\partial q_k}\log L(\bar{t},\bar{q}) &= \frac{deg_{in}(k)}{q_k} - e^{\lambda t_k}\sum_{i\ge k}\frac{deg_{out}(i)e^{-\lambda t_i}}{W(t_i)}\\ &\quad -\frac{e^{(\lambda+c)t_k}}{2}\sum_{i\ge k}\frac{deg_{out}(i)e^{-(\lambda+c)t_i}}{W(t_i)}\\ &\quad -\frac{e^{(\lambda-c)t_k}}{2}\sum_{i<k}\frac{deg_{out}(i)e^{(c-\lambda)t_i}}{W(t_i)}\,.\end{aligned} \quad (6)$$

Finally, the complexity of the likelihood optimization algorithm is $O(Ne)$, where N is the number of optimization steps. Recall that the complexity of one step of any date propagation method is $O(e)$.

Note that during the computation of the partial derivatives we have to calculate the sum of exponents e^{t_i} several times and a lot of such exponents are too large to fit the double or long double types what makes the computation infeasible. The solution to this problem is to operate with the logarithms of numbers (see *http://blog.smola.org/post/987977550/log-probabilities-semirings-and-floating-point*).