Identifying, Attributing and Describing Spatial Bursts

Michael Mathioudakis
Computer Science
University of Toronto
mathiou@cs.toronto.edu

Nilesh Bansal
Computer Science
University of Toronto
nilesh@cs.toronto.edu

Nick Koudas
Computer Science
University of Toronto
koudas@cs.toronto.edu

ABSTRACT

User generated content that appears on weblogs, wikis and social networks has been increasing at an unprecedented rate. The wealth of information produced by individuals from different geographical locations presents a challenging task of intelligent processing.

In this paper, we introduce a methodology to identify notable geographically focused events out of this collection of user generated information. At the heart of our proposal lie efficient algorithms that identify geographically focused information bursts, attribute them to demographic factors and identify sets of descriptive keywords. We present the results of a prototype evaluation of our algorithms on BlogScope, a large-scale social media warehousing platform. We demonstrate the scalability and practical utility of our proposal running on top of a multi-terabyte text collection.

1. INTRODUCTION

User generated content that appears on blogs, microblogging websites, wikis and social networks proliferates at profound rates. The number of active blogs, for instance, is estimated at the order of tens of millions, while several hundreds of millions of user profiles exist in the largest social networking sites. In this context, several attributes of interest can be captured automatically to a certain extent and can be utilized for the further analysis of the contributed information. In the blogosphere, for example, each blogger has a well defined physical location captured by a profile, while each piece of information contributed by bloggers (a blog post) is associated with a timestamp that declares the time the post was published. In addition, demographic information is disclosed such as age, gender, or profession, to name a few. Similar information is available for users of social networks or microblogging services.

In the context of the BlogScope[1] project [4], we have built the infrastructure to automatically collect information published online by individuals. The project aims to create a scalable data aggregation platform equipped with data mining and language processing primitives to support advanced information retrieval tasks. Currently, BlogScope tracks the Blogosphere, news sources, social networks, and several other online forums. In addition, it warehouses metadata about the content, including time of creation and demographic profile of the author, based on publicly available information. At the time of writing, the platform was indexing more than 500M text documents, adding over 1M new documents daily.

Given such a vast collection of information we are interested in automating the process of information discovery. A fundamental property of social media that is utilized by information aggregators, such as ours, is the following: when an event of interest to a certain segment of individuals takes place there is a surge in the volume of documents related to the event; as a result, such a surge of information (an information *burst*) can be utilized to identify events of interest. For example, there were many events associated with Barack Obama that took place within the last year. However, some of them prompted more individuals to write about them. For example the election of Barack Obama to the US presidency sparked a lot of online activity and had a precise temporal scope; it took place on November 4th 2008. As a result, for a few days in the vicinity of November 4th, there was an information burst related to that event.

Such information bursts are commonly associated with points or ranges in time and often have a clear geographical scope. Consider an event (say a scandal) associated with a public figure in London. Since the scope of the event is local it is expected that information will be contributed mainly by individuals from London. In this case, an information burst with a clearly identified geographical component (spatial information burst) can be identified. Taking this idea one step further, one can utilize additional information from the online profile of individuals in an attempt to *attribute* the information burst to specific demographic factors. For example one might identify that an information burst related to a concert is primarily attributed to males of age between 20 and 35.

As a final step in information discovery, after an information burst has been identified, it is possible to extract information from the related documents in order to *describe* the event that caused the burst. The idea is that since all documents associated with the information burst comment on the same event, some keywords are frequently used together and when identified, they shed light on the burst. Moreover, they offer suggestions for query expansion so that documents related to the specific event can be retrieved with high precision. Continuing the example for query 'Barack Obama', keywords correlated with the information burst pertaining to his election would be 'election', 'president', 'McCain', etc.

This paper proposes algorithmic techniques for the tasks defined above. In summary, we make the following contributions: (1) We introduce scalable algorithms to identify spatial information bursts. (2) We present efficient techniques to attribute bursts to specific demographic factors. (3) We present techniques to analyze bursts and effectively identify sets of keywords that describe the burst.

The paper is organized as follows. Related work is discussed

[1]`http://www.blogscope.net`

Proceedings of the VLDB Endowment, Vol. 3, No. 1
Copyright 2010 VLDB Endowment 2150-8097/10/09... $ 10.00.

in section 2. Our contributions are provided in sections 3 to 5: section 3 describes spatial burst detection, section 4 presents algorithms that attribute spatial bursts to demographic factors and section 5 presents techniques that produce keywords aiming to describe bursts. Section 6 contains a detailed experimental evaluation of the techniques proposed in the paper. The appendix contains background material on burst detection in time-series (Appendix A), improvements (Appendix B) over the basic techniques, as well as experimental results (Appendices D, E) that complement those of section 6. Pseudocode is provided in Appendix C.

2. RELATED WORK

Kleinberg [9] proposed a model for burst identification over document streams. Burst detection is performed using a multi-state automaton, where states correspond to different levels of burstiness. Our contribution for spatial burst detection is inspired by that model; we thus present it in detail in Appendix A. Another work by Kleinberg and Tardos [10] provides a 2-approximation linear programming algorithm to our spatial burst detection problem. Fung et. al. [8] used a model similar to [9] for the detection of bursty keywords in document streams. The paper showed how to utilize bursty keywords for document classification. In another work [15], Zhu and Shasha focused on the efficient detection of bursty windows in time series.

Statistical discrepancy functions are used to quantify the difference between distributions and are commonly used to identify regions where two spatial distributions differ significantly. Such regions can be interpreted as areas where one spatial distribution exhibits a burst in comparison with the other. There is a long line of work that deals with this problem [1, 2, 6, 11]. The work in [1], for example, provides both exact and approximate algorithms to identify the maximum discrepancy rectangle on a surface. The main drawback of such approaches is that they are high polynomial and in practice not scalable to large data collections. We included the maximum discrepancy methods in our experiments (Section 6) as a comparison baseline for the performance of our techniques.

In [3], Backstrom et. al. use a probabilistic framework to identify the centers of search activity related to search engine queries, as well as measure the dispersion of such activity around its center. The notion of 'search activity center' intuitively corresponds to the notion of 'spatial burst' as defined and used in our work. However, there are two important differences between the two approaches. The first is that [3] assumes prior specification of the number of search activity centers, while our spatial burst framework is not restricted to a predefined number of spatial bursts. Secondly, in contrast with our approach, parameter estimation in [3] is done over an unbounded parameter space, that can lead to poor performance, as demonstrated in the experimental section.

The problem of burst attribution, namely determining the demographic factors that primarily contribute to a burst, is related to the problem of explaining differences in a data cube. In [13], Sarawagi proposed techniques to help OLAP analysts navigate a data-cube, focusing on tuples that explained big differences at an aggregate level. The algorithmic techniques and developments herein are highly distinct as the problem formulation in our case is different than that in the case of OLAP cubes.

Fung et. al., [7], proposed an algorithm to construct a hierarchy of events related to a particular query. The basic idea is to first identify events associated with specific bursty keywords and then hierarchically create sub-events by clustering similar documents. Document clustering, however, requires significant processing. We seek to develop interactive and highly scalable solutions.

3. SPATIAL BURSTS

Consider a setting where geographically distributed individuals contribute information in the form of documents. Each document d is authored by an individual that resides in a specific geographical location g_d, declared in the individual's profile. Moreover, it is associated with a timestamp t_d that indicates the time d was created, as well as other metadata, such as the age or gender of the author.

In this setting, it is natural to seek *spatial bursts* of information, i.e. *geographic locations at which documents related to specific keywords exhibit a surge*. For example, one may look for locations that exhibit a surge in documents that contain the keywords "barack", "obama" and were created on November 4th, 2008 (day of US elections). In general, one specifies a set of keywords q and a time interval δt and identifies locations that exhibit a surge in documents that were created within δt and contain q. In what follows, we use the term 'query' to refer to the set of keywords q.

Documents d are collected and stored inside a data warehousing system. Specifically, consider a system that provides the following functionality [2] : given a query q and a time range δt, the system returns the following sets of documents: (i) the set $D_{\delta t}$ of *all* documents d with a timestamp t_d within δt, (ii) the set $R_{\delta t}$ of documents d that contain all keywords q *and* whose timestamp t_d belongs to δt. In what follows, documents $d \in R_{\delta t}$ will be referred to as 'relevant' to query q.

Let G be a grid; for a suitable choice of granularity, geographical entities of interest (e.g., cities) correspond to a cell in this grid[3]. In light of G, each document can be mapped to a particular cell of the grid. Also, given query q and time range δt, the sets $R_{\delta t}$ and $D_{\delta t}$ of documents returned by the system are associated with two spatial distributions: (a) $R^S = R^S_{\delta t}$, the spatial distribution of the relevant documents published within δt, (b) $D^S = D^S_{\delta t}$, the spatial distribution of all documents published within δt.

Spatial bursts are identified as cells for which the value of $R^S_{\delta t}$ is large in comparison with that of $D^S_{\delta t}$. One simple way to identify spatial bursts, then, is to threshold the fraction of relevant documents r_g over the total number of documents d_g that correspond to cell g. This approach however, might lead to the identification of 'spurious bursts', i.e. identification of 'bursty' cells that either are associated with a small number of total documents d_g (but large ratio r_g/d_g) or are geographically isolated. This is not desirable in cases when one looks for bursty cells g associated with large d_g or in the vicinity of other bursty cells. The spatial burst model described in the paragraphs that follow allows to avoid 'spurious bursts' but also contains the simple approach that was just described as a special case.

In order to identify cells with $R^S_{\delta t}$ values that are large in comparison with their $D^S_{\delta t}$ values, we model the state of cells of G using two binomial distributions $BIN(p_0)$ and $BIN(p_1)$, $0 \leq p_0 < p_1 \leq 1$. This corresponds to a generative model where each document in a cell satisfies query q with a smaller (p_0) or larger (p_1) probability, leading to a smaller (p_0) or larger (p_1) expected fraction r_g/d_g of relevant documents. Then, we characterize as 'bursty' those cells whose $R^S_{\delta t}$ and $D^S_{\delta t}$ distribution values are 'closer' to $BIN(p_1)$ rather than $BIN(p_0)$. We thus recognize two fundamental states for cells $g \in G$, namely 'bursty' ($s_g = 1$) and 'non bursty' ($s_g = 0$). Following the intuition behind using binomial distributions in the model, parameters p_0 and p_1 are set to represent what we consider to be a low or high fraction r_g/d_g of

[2]For a description of the technology used to provide such functionality the interested reader is referred to [12].

[3]In BlogScope, we use a grid with a granularity of nearly 2.5 degrees of latitude and longitude per cell.

relevant documents in a cell, based on distributions $R^S_{\delta t}$ and $D^S_{\delta t}$.

Characterization of cells as bursty or non-bursty is formulated as a minimization problem of an additive cost function C. For each cell there is a term that penalizes the 'distance' of the cell's $R^S_{\delta t}$ and $D^S_{\delta t}$ values from the binomial distribution that corresponds to its state. As a penalty measure we use the negative log-likelihood. Thus, for a cell g that contains r_g relevant documents, d_g total documents and is either bursty ($s_g = 1$) or non-bursty ($s_g = 0$), the cost function contains a term

$$c(s_g, r_g, d_g) = -\log\left[\binom{d_g}{r_g} p_{s_g}^{r_g}(1-p_{s_g})^{d_g - r_g}\right].$$

In addition, to avoid 'spurious' bursts, a cost term that penalizes neighboring cells of different state is included. Specifically, for neighboring cells with states x and y, the cost function includes a term $\tau(x, y)$ defined as

$$\tau(1,0) = \tau(0,1) = \gamma \geq 0, \quad \tau(0,0) = \tau(1,1) = 0.$$

What value we set for γ depends on how strict we are against 'spurious bursts'. In what follows, when two neighboring cells are assigned different states we say that there is a *transition* between them and that a *transitional cost* is imposed. Transitions between cells form a graph (with cells corresponding to nodes and transitions to edges between nodes) to which we refer as the *transition graph*.

Summing all terms up, cost function C takes the form

$$C = \sum_{g_{i,j} \in G} c(s_{g_{i,j}}, r_{g_{i,j}}, d_{g_{i,j}}) + \tau(s_{g_{i,j-1}}, s_{g_{i,j}}) + \tau(s_{g_{i-1,j}}, s_{g_{i,j}})$$

where $s_{g_{i,j}}$ is the state of cell $g_{i,j}$ with coordinates i and j on the grid. The problem of identifying spatial bursts is defined as follows.

PROBLEM 3.1 (SPATIAL BURSTS). *Given a spatial distribution R^S of documents related to a query q and a spatial distribution D^S, both over a grid G, assign bursty and non-bursty states to cells in G so that cost function C is minimized.*

In the special case of $\gamma = 0$, all transitional costs are equal to zero (0). Then, the cost function takes the form

$$C = C_1 = \sum_{g \in G} c(s_g, r_g, d_g)$$

and is minimized by states s_g that independently minimize the corresponding cost term $c(s_g, r_g, d_g)$. Specifically, a cell g is assigned state s_1 iff

$$c(1, r_g, d_g) < c(0, r_g, d_g) \Leftrightarrow$$

$$-\log\left[\binom{d_g}{r_g} p_1^{r_g}(1-p_1)^{d_g - r_g}\right] < -\log\left[\binom{d_g}{r_g} p_0^{r_g}(1-p_0)^{d_g - r_g}\right]$$

$$\Leftrightarrow \frac{r_g}{d_g} > \frac{\log\left((1-p_0)/(1-p_1)\right)}{\log\left((1-p_0)/(1-p_1)\right) + \log(p_1/p_0)}.$$

Therefore, in the special case where transitional costs are not taken into account, states are determined simply by the fraction of relevant documents r_g/d_g in a cell.

In the general case when $\gamma \geq 0$, problem 3.1 is NP-hard [10] and is solved in a straightforward manner by a dynamic programming algorithm. Consider the grid shown in figure 1. Cells in the grid are grouped by diagonals — in figure 1, for instance, cells of the $(t-1)$-th or t-th diagonal are grouped together. States of cells in the same diagonal are represented by a vector; for example, states in the $(t-1)$-th and t-th diagonal are denoted by vectors L and J, respectively. Arrows between cells depict possible transitions. Let $C^J(t)$ be the optimal cost corresponding to the part of the grid between the 1-st and the t-th diagonal, when the states of cells in the t-th diagonal are given by vector J. Similarly, let $C^L(t)$ denote the optimal cost for the part of the grid between the 1-st and the $(t-1)$-th diagonal, when the states of cells in the $(t-1)$-th diagonal are given by vector L. Then, $C^J(t)$ can be written in terms of $C^L(t)$ and the transition costs between the $(t-1)$-th and t-th diagonal

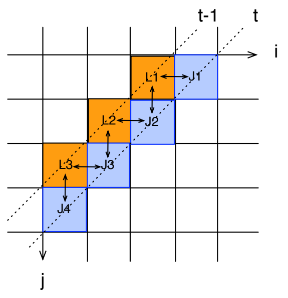


Figure 1: A dynamic programming algorithm solves Prob. 3.1.

$$C^J(t) = C_1^J(t) + \min_L(C^L(t-1) + \tau(J, L)) \tag{1}$$

where $C_1^J(t)$ is the sum of $c()$ terms along the t-th diagonal when the burst states of its cells are given by a vector J. Notice that for a particular diagonal with d cells, there are 2^d possible ways to set the associated vector of (burst) states. Therefore, the dynamic programming algorithm that follows equation 1 is of exponential complexity with respect to the grid size and is thus prohibitively costly for practical scenarios.

We provide a *practical* solution to Problem 3.1 by solving a similar but easier cost minimization problem, namely Problem 3.2. Problem 3.2 is similar to Problem 3.1 in that its cost function also penalizes (a) the 'distance' of a cell's $R^S_{\delta t}$ and $D^S_{\delta t}$ values from the binomial distribution that corresponds to its state and (b) transitions between cells. However, Problem 3.2 is defined so that it is easier to solve. Therefore, solutions to Problem 3.2 are much more efficient to obtain and, as we show experimentally in section 6, they can be used as good heuristic approximations to solutions of Problem 3.1.

Unlike Problem 3.1, in Problem 3.2 only a subset of all possible transitions between grid cells are taken into account in the cost function. Let g_{ij} be a cell and consider $g_{i,j-1}$ and $g_{i-1,j}$, two of its neighboring cells. Problem 3.2 considers the transition cost between g_{ij} and *only one* of cells $g_{i,j-1}$ or $g_{i-1,j}$, namely the one that under Problem 3.1 would be more 'likely' to impose a transitional cost. Specifically, if values of $R^S_{\delta t}$ and $D^S_{\delta t}$ for cell g_{ij} are closer to $BIN(p_1)$, then Problem 3.2 considers the transition cost only between g_{ij} and the cell ($g_{i,j-1}$ or $g_{i-1,j}$) that is closer to $BIN(p_0)$. Inversely, if cell g_{ij} is closer to $BIN(p_0)$, then a transition is considered only between g_{ij} and the cell that is closer to $BIN(p_1)$. Ties are resolved by always considering a transition from $g_{i,j-1}$. The graph of transitions considered by Problem 3.2 will be similar to the example of figure 2 and is in fact a set of binary trees.

Consequently, the cost function for Problem 3.2 is

$$C = C_1 + C_2 = \sum_{g_{i,j}} \left(c(s_{g_{ij}}, r_{ij}, d_{ij}) + \tau(s_g, s_{g_{ij}})\right) \tag{2}$$

where, for every $g_{i,j}$, we define g as

$$g = \arg\max_{g' \in \{g_{i,j-1}, g_{i-1,j}\}} \{c(0, r_{g'}, r_{g'}) - c(1, r_{g'}, r_{g'})\}.$$

when

$$c(0, r_{g_{ij}}, r_{g_{ij}}) < c(1, r_{g_{ij}}, r_{g_{ij}})$$

and as

$$g = \arg\max_{g' \in \{g_{i,j-1}, g_{i-1,j}\}} \{c(1, r_{g'}, r_{g'}) - c(0, r_{g'}, r_{g'})\}.$$

when

$$c(0, r_{g_{ij}}, r_{g_{ij}}) > c(1, r_{g_{ij}}, r_{g_{ij}}).$$

The formal definition of Problem 3.2 is the same with that of Problem 3.1, except that cost function C is given by formula 2.

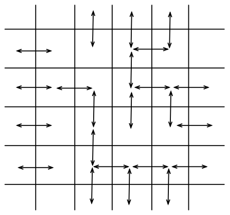


Figure 2: The graph of transitions considered by Problem 3.2 form a set of trees.

Problem 3.2 is also solved by a dynamic programming algorithm, however much more efficiently than Problem 3.1. Let $root$ be a cell of the grid and $left$, $right$ be its children on a tree of transitions. Moreover, let $C^s(root)$ be the optimal cost for the cells of the binary tree rooted at cell $root$, when $root$ is assigned state s (similarly for $C^{s_l}(left)$, $C^{s_r}(right)$). Then, $C^s(root)$ can be expressed in terms of $C^{s_l}(left)$ and $C^{s_r}(right)$:

$$C^s_{(root)} = C^s_1(root) + \min_{(s_1,s_2)} (C^{s_1}_{left} + \tau(s, s_1) + C^{s_2}_{right} + \tau(s, s_2)) \quad (3)$$

where $C^s_1(root)$ is the cost term $c()$ evaluated for cell $root$ with state s. The dynamic programming algorithm that follows equation 3 is linear with respect to the number of cells it considers.

PROPOSITION 1. *Problem 3.2 over a $n \times n$ grid is solved in $O(n^2)$ time.*

Proof For every node in a tree we solve the minimization problem for $b = 2$ different states, considering every time $O(b^2) = O(4)$ different cases for its children. The total number of nodes of all trees is $O(n^2)$. Thus, the total complexity of the algorithm is $O(n^2 b^3) = O(8n^2) = O(n^2)$. $\square$

Let *s-Spatial* be the algorithm that produces the transition graph and decides the states of grid cells according to equation 3 (pseudocode provided in Appendix C, Alg. 1). As we demonstrate experimentally in section 6, algorithm *s-Spatial* produces solutions that are excellent approximations to the solutions of Problem 3.1 and achieves fast computation of the bursts. In appendix B, we present *i-Spatial*, an algorithm that improves over the performance of *s-Spatial* in the case of largely uniform spatial distributions.

4. BURST ATTRIBUTION

Spatial bursts locate geographically focused surges in online activity related to a query. Since each individual is associated with demographic information, a natural question that arises is whether it is possible to attribute the burst to demographic factors. For example, a burst in the city of Toronto in June 2009 related to singer 'Britney Spears', might be primarily attributed to teenage females.

There are many ways to approach and formalize the problem of attributing a burst to profile features. One approach is to focus on a specific set of bursty cells I (call it a *region*) and ask what are the demographic factors in the absence of which no burst would have been detected. For example, consider the query 'Toronto Film Festival' and the burst associated with the city of Toronto in September 2009 and suppose that all ages 20 to 50 contributed significantly to it, while individuals of other ages did not. If it wasn't for individuals of ages 20 to 50, then, no burst would appear in Toronto and therefore the burst can be attributed to them. Another approach is to compare a bursty region I with a non-bursty region J and ask what are the demographic factors that make the difference wrt the 'burstiness' of the two regions. Continuing the example for the query 'Toronto Film Festival', suppose that in New York, where only young people 20-30 years old wrote about the event, no burst was captured. Comparing Toronto with New York, then, it is ages 30 to 50 that made the difference in burstiness. In the rest of the section, we provide a formalization of these approaches.

Consider a maximal set of neighboring bursty cells I (two cells are neighbors if exactly one of their coordinates differs by 1). We call I a *bursty region* and define its burst weight to be

$$W_I = \sum_{g \in I} \big(c(0, r_g, d_g) - c(1, r_g, d_g)\big) > 0. \quad (4)$$

Weight W_I captures how better spatial distributions R^S and D^S fit a binomial distribution with parameter p_1 over region I, instead of that with parameter p_0.

Without loss of generality assume that all documents can be characterized by an attribute A with domain $Dom(A) = \{a_1, \ldots, a_k\}$. Formula 4 can be re-written as follows

$$W_I = \sum_{g \in I} (c(0, r_g, d_g) - c(1, r_g, d_g)) =$$

$$= \sum_{g \in I} (r_g \log \frac{p_1(1-p_0)}{p_0(1-p_1)} + d_g \log \frac{1-p_1}{1-p_0}) = \sum_{g \in I} (r_g \alpha + d_g \beta) =$$

$$= \sum_{g \in I} \sum_{c} (r_{gc} \alpha + d_{gc} \beta) = \sum_{c} \sum_{g \in I} (r_{gc} \alpha + d_{gc} \beta) =$$

$$= \sum_{c} \sum_{g \in I} (c(0, r_{gc}, d_{gc}) - c(1, r_{gc}, d_{gc})) = \sum_{c} W^c_I$$

with $\alpha = \log \frac{p_1(1-p_0)}{p_0(1-p_1)} > 0$, $\beta = \log \frac{1-p_1}{1-p_0} < 0$ and r_{gc} (d_{gc}) denoting the number of relevant (all) documents from cell g with attribute A value equal to a_c. Therefore, the weight of a spatial burst is a sum of individual weights, each corresponding to two R^S and D^S spatial distributions exclusively made of documents with some attribute value a_c.

Consider the following problem.

PROBLEM 4.1. *Identify the largest set of attribute values C s.t.*

$$\sum_{c \in C} W^c_I < 0.$$

Set C has the following property: if two spatial distributions R^S and D^S consisted only of documents with attribute values in C (i.e. we had $r_{gc} = d_{gc} = 0$ for $c \notin C$), then we would have

$$W_I = \sum_{c} W^c_I = \sum_{c \in C} W^c_I + \sum_{c \notin C} W^c_I = \sum_{c \in C} W^c_I + 0 < 0,$$

contrary to formula 4 and therefore, I would not have been identified as a bursty region. Therefore, it is intuitive to say that the burst is attributed to those attribute values a_c s.t. $c \notin C$.

The intuition behind the problem is illustrated with a simple qualitative example. Suppose spatial burst detection is performed with parameters p_0, p_1 for query q and time interval δt and let R, D denote the sums of values of R^S, D^S respectively that correspond to a bursty region I. Suppose also that all documents are characterized by an attribute A with domain $Dom(A) = \{a_1 = 1, a_2 = 2, a_3 = 3, a_4 = 4\}$. In figure 3 we plot the sum of R values that correspond to attribute values a_c, $1 \leq c \leq 4$ and suppose for simplicity that D values are uniformly distributed over different a_c's. In the plot, two horizontal lines indicate the thresholds implied by the parameters p_0 and p_1. For all $A \neq 3$, the corresponding sums of the R values are below the threshold implied by p_0. Therefore, the burst of region I is attributed to the documents with $A = 3$ without which the region I would not be bursty.

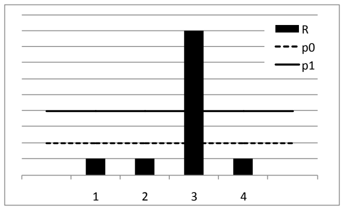


Figure 3: Number of relevant documents inside region I that have $A = a_c$, $1 \leq c \leq 4$.

THEOREM 1. *There exists an optimal algorithm to solve Problem 4.1 in $O(k \log k)$ time, with $k = |Dom(A)|$.*

The problem is solved by sorting the weights W_I^c and picking greedily the smallest of them as far as their sum does not exceed 0 [5].

We now turn to the case we compare two geographic regions with respect to their bursty behavior. Again, spatial burst detection is performed with parameters p_0 and p_1 for a query q and time interval δt and we identify a non-bursty region I_0 and a bursty region I_1. Let R_0 and R_1 denote the values of R^S that correspond to regions I_0 and I_1 respectively and as before, assume that all documents are characterized with an attribute A that takes values from the set $\{a_1 = 1, a_2 = 2, a_3 = 3, a_4 = 4\}$. For simplicity, assume that all cells g of the grid share a common value d_g and that for any cell g and $1 \leq c, c' \leq 4$ we have $d_{gc} = d_{gc'}$ (all attribute values share the same total number of documents inside a cell).

For each attribute value a_c, we calculate the number of relative documents inside regions I_0 and I_1 that have $A = a_c$ (i.e., we calculate $\sum_{g \in I_0} r_{gc}$ and $\sum_{g \in I_1} r_{gc}$) and we plot it in figure 4. Since we have assumed a uniform distribution D^S and the attribute values, the parameters p_0 and p_1 imply two thresholds in the values of the distributions R_0 and R_1, which we demonstrate with horizontal lines in figure 4. As shown in plot of R_1, the number of documents with attribute values $A = 2$ or $A = 3$ exceeds the threshold set by p_1 and therefore these two attribute values have a positive burst weight for region I_1 – while the other two attribute values have a negative burst weight, since they do not exceed the threshold set by p_0. Consequently, by solving Problem 4.1 for I_1 we would attribute the burst of I_1 to the two attribute values $A = 2$ and $A = 3$.

However, if we compared region I_1 with region I_0 in terms of burstiness, then we would notice that the plots for region I_0 and region I_1 are nearly identical for all attribute values except for $A = 3$. Then, since region I_1 is bursty while I_0 is not, we should attribute the *difference in burstiness* between the two regions to $A = 3$.

The problem of attributing differences in burstiness to particular attribute values is stated as follows.

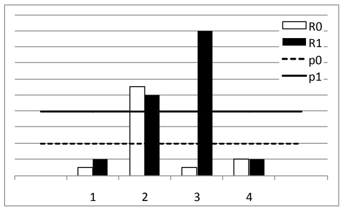


Figure 4: Number of relevant documents inside the regions I_0 and I_1 that have $A = a_c$, $1 \leq c \leq 4$.

PROBLEM 4.2. *Let I_1 and I_0 be two regions on a grid with $W_{I_1} > 0$ and $W_{I_0} < 0$. Find the largest set of attribute values $C \subseteq A$ s.t. $\sum_{c \in C}(W_{I_1}^c - W_{I_0}^c) < -W_{I_0}$.*

To see the intuition behind the problem, notice that W_{I_0} is the burst weight of region I_0 (assumed negative) and therefore $-W_{I_0}$ is the additional burst weight that we would need to 'add' to region I_0 so that its burst weight would turn positive. On the other hand, $(W_{I_1}^c - W_{I_0}^c)$ is the difference in burstiness that is due only to attribute value c. It is now easy to observe that the set C that solves the second problem has the following property: even if I_0 was assigned the same weights with I_1 for attributes in C, then I_0 would still not have positive burst weight (would not be bursty). Thus, we interpret documents with attribute values in the set $(A - C)$ as the ones primarily responsible for the difference in the bursty behavior between the two regions.

The problem is solved with the same optimal algorithm as in the case of the previous problem in $O(k \log k)$ time, with $k = |Dom(A)|$. The model extends directly to more than one attributes $A_1, \ldots, A_d$ with

$$W_I = \sum_{c_1 \in A_1} \sum_{c_2 \in A_2} \cdots \sum_{c_d \in A_d} W_I^{c_1 c_2 \ldots c_d}$$

for a region I and $C \subseteq A_1 \times A_2 \ldots \times A_d$ for Problems 1 and 2. The complexity of the problem increases with more dimensions. However, for the small property domains we consider, complexity is not a big issue.

5. KEYWORD BASED DESCRIPTION OF BURSTS

In this section, we turn to the identification of keywords that describe a burst. Assume we have identified temporal (using background techniques surveyed in Appendix A) and/or spatial bursts for a query q. The following subsections present techniques that identify keywords that aim to describe the burst.

5.1 Query Expansion

A first attempt could utilize generic techniques for query expansion to identify keywords highly related to q. For instance, to quantify how related a keyword w is to query q, we can use the measure of *mutual information* [14]. A simple iterative algorithm can then be employed to expand query q: retrieve the documents that contain query q, compute the k most related keywords, say $w_1, w_2, \ldots w_k$, then form k queries by pairing q with each $w_i, 1 \leq i \leq k$; provided that a burst is also identified for a newly formed query $q \cup w_i$, $1 \leq i \leq k$, the process is repeated until a maximum expansion depth has been reached. All keyword expansions to q can be reported or, since for each keyword 'related' to q we have a score, we can report only the highest scoring paths starting from q if desired.

The approach is reasonable, but suffers in terms of performance as for a query q, in order to retrieve k related keywords for an expansion depth d we should form k^d queries in the worst case. The associated costs can be prohibitive for online scenarios primarily due to the number of document retrievals involved and the costs associated with identifying related keywords. We refer to this algorithm as *ContentBased* in the sequel.

5.2 Expansion Based on Curve Similarity

We aim to reduce the costs associated with query expansion and in particular the overhead associated with identifying related keywords for a specific query. The approach we describe is based on the following observation: when a query q is bursty over a time interval b_q, then keywords w that occur frequently together with q often exhibit a burst themselves over the same interval. Therefore, such keywords w are good candidates for keywords related to q.

BlogScope has an inverted index of 13M keywords. For each keyword, the index contains a list of document-id's, that correspond to the documents that contain it at each time step (e.g., day). Therefore, for each keyword w in the index we can construct a time series of the number of documents containing the keyword for each time step. For each such time series it is possible to identify all segments b_w that exhibit bursty behavior (e.g., using techniques such as those of Appendix A) and insert all such bursty segments into a suitable indexing structure (e.g., a one dimensional R-tree).

Assume that q exhibits bursty behavior over segment $b(q)$. We then retrieve all segments in the R-tree that partially overlap $b(q)$ (say above $\theta\%$). As described in the previous paragraph, such segments generally correspond to different keywords w_i. We order the keywords by assessing the Pearson correlation coefficient ρ between time series segments $b(q)$ and $b(w_i)$ for each returned keyword w_i. We maintain the top-k keywords (according to ρ) and rank them by their $tf - idf$ values [14]. For each keyword $w_i, 1 \leq i \leq k$ we expand q with w_i thus forming query $q \cup w_i$ and repeat the process until the desired expansion depth is reached. Notice however that at each step we need the corresponding time-series segment for the newly formed queries $q \cup w_i$. This information is not materialized and to compute it one has to retrieve and intersect the document-id lists of q and w_i. We refer to this algorithm as *CurveCorr*.

We may speedup the computation even more. Instead of retrieving the curve of an expanded query $q' = q \cup w_i$, we estimate it, thus avoiding querying again the index. Estimation is based on the correlation coefficient ρ between the two curves. When $\rho = -1$ the curves are anti-correlated (they behave in exactly opposite ways). For this reason, we heuristically estimate that the corresponding queries q and w_i do not occur together in documents and therefore the curve of $q' = q \cup w_i$ will have only zero values. On the other hand, when $\rho = 1$ the curves are perfectly correlated. When this happens, we estimate that all the occurrences of the query (q or w_i) with the smallest curve value happen together with the other query. Estimation for other values of ρ is done by using ρ to scale between the two extreme cases we just described.

$$q'[t]_{est} = (1 + \rho) \cdot \min\{b(q)[t], b(w_i)[t]\} \qquad (5)$$

We refer to this algorithm as *CurveEstim* in the sequel.

6. EXPERIMENTS

We provide quantitative and qualitative experimental results and establish the scalability and practical utility of the techniques presented. In this section, we evaluate spatial burst detection. For experiments on burst attribution and keyword-based description of bursts, we refer the reader to appendices D and E. All experiments are performed over the vast data collection of BlogScope, which currently includes an excess of 3 TB of social media data.

Query Sets	Queries	Description
HotKeywords	airbus, bhutto, former, gunman, hessbolah, israel, mccain, mulroney, premier, romney	Hot Keywords on Blogscope for Feb 18th 2008
ZeitgeistDec	digestive system, traductor, neitman marcus, NBA, telus.ca, banque scotia, flow 93.5, mastermind, chemistry, rush hour 3	Top 10 Google Zeitgeist queries for Canada, Dec 2007
ZeitgeistJan	chateleine, peinture, u2, synonyms, croatia, carmen electra, tiny pic, rabbits, encyclopedia, Mila Kunis	Top 10 Google Zeitgeist queries for Canada, Jan 2008

Figure 5: The sets of queries.

6.1 Spatial Burst Detection

6.1.1 Scalability

For the purposes of scalability experiments, we implemented algorithm *s-Spatial*, described earlier in section 3, as well as algorithm *i-Spatial*, an improved version of `s-Spatial`, detailed in Appendix B. Algorithm *i-Spatial* differs from *s-Spatial* in that it produces and operates upon a smaller transition graph than *s-Spatial*, thus allowing for faster computation of the burst states of cells. This improvement is based on the observation that *s-Spatial* often produces transition graphs with long branches over grid cells of the same burst state and that such graphs can be 'condensed' to ones of smaller size. This is achieved by substituting successive cells of the same state with one cell, so that we end up with the same cost function but a smaller transition graph. The observation is formally stated and proved in Appendix B.

In addition, we implemented techniques presented in [1], which address the problem of maximizing the Kulldorf [11] statistical discrepancy between two distributions over a grid. Although of high complexity, such techniques have been previously used to identify 'bursty' regions of activity and we thus include them as a baseline comparison to our approach. We also implemented the algorithm described in [3] for the case it looks for a single 'activity center'. We include this algorithm in our experiments because the notion of 'activity center' intuitively corresponds to the notion of a 'spatial burst point' defined in our work.

In summary, we implemented the following algorithms: (1) `s-Spatial`, described in Section 3 (2) `i-Spatial`, described in Appendix B (3) `sp-Variation`, described in [3] (3) `Exact` [1], an optimal algorithm that identifies a rectangle of maximum discrepancy on a grid (4) `Approx` [1], an approximate algorithm that identifies a rectangle of maximum discrepancy on a grid.

Data was collected by issuing queries to BlogScope that returned pairs of R^S and D^S distributions over a grid that represents a world map. We generated three sets of queries named, *HotKeywords*, *ZeitgeistDec* and *ZeitgeistJan*, shown in figure 5. We model grids of various sizes, producing R^S and D^S distributions over grids of varying granularity, with dimensions $(901 \cdot f) \times (459 \cdot f)$ for $f \in \{1, 1/3, 1/10\}$ and sizes 413559, 46053, 4186 respectively (we'll refer to f as the *granularity parameter*) and for each set of queries and granularity, we report the average running time of the algorithms. The grid of size $901 \times 459 = 413559$ (for $f = 1$) has a granularity that allows major cities to be covered by a single cell.

Results are shown in figure 6 for one of the datasets (the plots are similar for the other two and are omitted due to space constraints). We have plotted the running time of the algorithms in ms (y axis in log scale), as a function of the grid size. We observe that the `s-Spatial` algorithm scales much better with the size of the grid not only compared to the `Exact` and `Approx` algorithms but also compared to `sp-Variation`, since the latter performs its param-

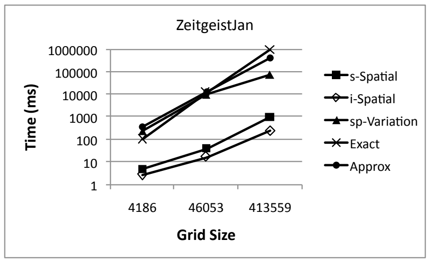


Figure 6: Average running time of the algorithms in milliseconds (log-scale)

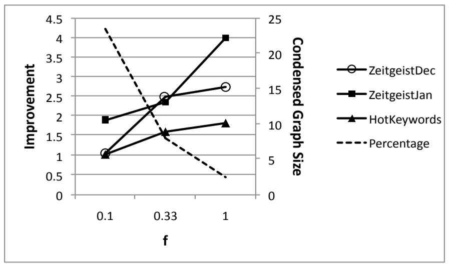


Figure 7: The size of the condensed transition graph as a percentage of the initial graph and improvement of `i-Spatial` over `s-Spatial`, in terms of *how many times faster* it runs for the three datasets , versus granularity parameter f.

eter estimation over an unbounded space of values and for each guess it needs to compute a likelihood value over the entire grid.

Additionally, we observe significant improvement in the running time achieved by `i-Spatial` (the times reported for this algorithm include the time needed to transform the initial graph of transitions to its 'condensed' version). The improvement is due to the sparseness of the spatial distributions that is larger for smaller granularity and results in condensed transition graphs much smaller in size than the initial ones (Fig. 7).

6.1.2 Qualitative Results

Queries q were submitted to BlogScope, with temporal interval δt set as the first 10 days of March 2008; we used keywords or sets of keywords of local interest to specific regions of the world (Figure 8). Retrieving distributions R^S and D^S for a query, we set the parameter p_0 equal to the weighted average of the values of R^S normalized by the corresponding values of D^S and set p_1 to three standard deviations higher than p_0. We deliberately set γ to 0 to exclude the effect of the transition costs.

In figure 8, the set of keywords 'northern rock' appear to have spatial bursts in the UK, where a bank with the same name was in the center of attention in the local economic and political news. Furthermore, the keyword 'cricket' appears to be bursty in India and Australia; both countries were involved in an important cricket game on March 4th. The name of the Spanish prime minister 'zapatero' appeared to be bursty in Spain for the first days of March, due to the elections that took place that time. Finally, notice that the keyword 'snowstorm' has a burst across all regions of northeastern US that were under a snowstorm from the 7th to the 9th of March.

We now demonstrate the result quality obtained solving Problem 3.2. As detailed in section 3, we use solutions to Problem 3.2 – produced by algorithms *s-Spatial* or *i-Spatial* – as approximate solutions to Problem 3.1. The quality of these solutions can be ex-

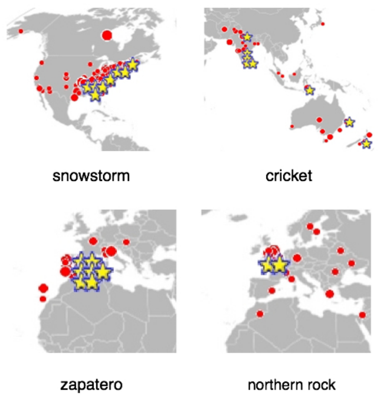


Figure 8: Examples of spatial bursts (denoted with stars).

pressed formally in terms of the ratio

$$r = C^1(S_2)/C^1(S_1)$$

where S_1, S_2 are solutions to Problem 3.1 and Problem 3.2, respectively, over the same grid G and $C^1(S_1)$, $C^1(S_2)$ are the corresponding costs of the solutions according to the cost function C of Problem 3.1. Obviously, the smaller the approximation ratio r, the better the quality of the approximation. For $\gamma = 0$ the two problems are equivalent and therefore we have $r = 1$. For $\gamma \geq 0$, we define the *transition cost ratio* as

$$t = trC^1(S_2)/C^1(S_2)$$

where $trC^1(S_2) \leq C^1(S_2)$ is the sum of the transition costs τ for approximate solution S_2, i.e.

$$trC(S) = \sum_{g_{i,j} \in G} \tau(s_{g_{i,j-1}}, s_{g_{i,j}}) + \tau(s_{g_{i-1,j}}, s_{g_{i,j}}) \qquad (6)$$

where $s_{g_{i,j}}$ is the state of cell $g_{i,j} \in G$ according to solution S. Intuitively, t expresses what part of solution S_2 cost is due to transition costs and, thus, parameter γ. To demonstrate the effect of $\gamma \geq 0$ on the quality of the solution, we conducted experiments on a grid of size 10x10 populated with synthetic data – the size of the grid was chosen to be small so that we could efficiently acquire a solution to Problem 3.1. Specifically, we measured the ratio r as a function of the transition cost ratio t [4]. The grid was populated with a constant spatial distribution D and a binomial spatial distribution R, $BIN(p_0)$. We subsequently solved the two problems for a range of values of γ. The results, shown in figure 9 for parameters $p_0 = 0.5$, $p_1 = 0.55$, demonstrate that when the transition costs do *not* dominate the cost of the approximate solution ($t < 0.8$) a small approximation ratio is achieved ($r < 6$). These cases correspond to settings that are of practical interest, since for larger values of parameter γ and ratio t, none or few cells are characterized as 'bursty', due to large transition costs. Very similar results were obtained for a wide range of values of p_0, p_1.

6.1.3 Parameter Sensitivity

We conducted experiments on synthetic data to demonstrate the effect of parameters on burst detection. The experiments were performed over a grid of size 500×500. All cells g of the grid were assigned a total of $d_g = 100$ documents. The number of relevant

[4]We present our experimental results in terms of t instead of γ, since t is more informative, while, for a particular solution S, there is a linear relationship between t and γ.

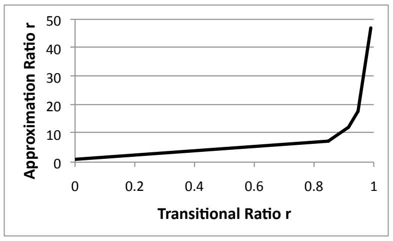


Figure 9: Approximation ratio r vs. transition cost ratio t.

documents r_g per cell g was set as follows. Each cell was randomly set as 'bursty' with probability 10% and 'non-bursty' otherwise. 'Non-bursty' and 'bursty' cells g were assigned a number r_g of relevant documents according to binary distribution $BIN(p'_0)$ and $BIN(p'_1)$, respectively. After the synthetic data were generated, algorithm `s-Spatial` was employed with parameters p_0, p_1, γ, with p_0 and p_1 possibly different from the respective p'_0, p'_1 that were used to generate the data. For every execution of `s-Spatial`, we measured the fraction of cells that is identified as 'bursty'.

The first set of experiments demonstrates the effect of parameter γ. To populate the grid with synthetic data, we used $p'_0 = 0.01$ and $p'_1 = 0.050/0.075/0.100$. Subsequently, we employed algorithm `s-Spatial` with the same parameters $p_0 = p'_0$, $p_1 = p'_1$ and ranging values of $\gamma \in [0, 100]$. Figure 10(a) reports the fraction of bursty cells for different values of γ and $p_1 = p'_1$. We observe that the fraction of bursty cells decreases smoothly as γ increases. Moreover, to demonstrate the effect of γ on spurious bursts, we considered the transitions adjacent to at least one bursty cell in the full transition graph and we report as 'burst concentration' the fraction of those transitions that connect two bursty cells (Figure 10(b) – the plot is almost identical for all three values of p_1). The results show that for larger γ the fraction of isolated, spurious bursts tends to decrease.

The second set of experiments isolates the effect of parameter p_1 on burst detection (the situation for p_0 is symmetric). To generate the synthetic data, we used $p'_0 = 0.01$ and $p'_1 = 0.05$. Subsequently, we employed `s-Spatial` with the same $p_0 = p'_0 = 0.01$, $\gamma = 0$ and ranging p_1. Figure 10(c) reports the fraction ζ of bursty cells for different values of p_1 and, as we can see, it shows little sensitivity for values around $p_1 = 0.05 = p'_1$. Bigger changes in fraction ζ's value are exhibited when p_1 becomes too large or too small relative to p'_1.

7. CONCLUSIONS

We presented algorithms and techniques for spatial burst detection, attribution and keyword description of bursts over large collections of user generated text. We plan to make these features available in the public version of our social media platform.

8. REFERENCES

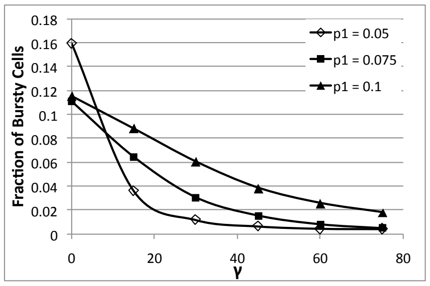
[1] D. Agarwal, A. McGregor, J. M. Phillips, S. Venkatasubramanian, and Z. Zhu. Spatial scan statistics: approximations and performance study. In *KDD*, 2006.

[2] D. Agarwal, J. M. Phillips, and S. Venkatasubramanian. The hunting of the bump: on maximizing statistical discrepancy. In *SODA 2006*, 2006.

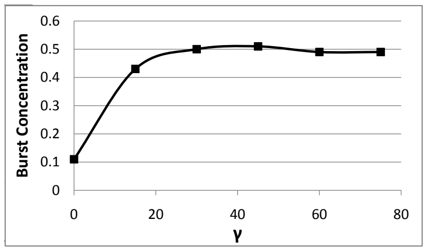
[3] L. Backstrom, J. M. Kleinberg, R. Kumar, and J. Novak. Spatial variation in search engine queries. In *WWW*, 2008.

[4] N. Bansal and N. Koudas. Blogscope: A system for online analysis of high volume text streams. In *WebDb*, 2007.

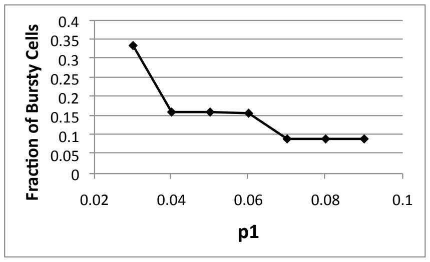
[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press and McGraw-Hill Book Company, 2001.



(a) Fraction of Bursty Cells vs γ



(b) Burst Concentration vs γ



(c) Fraction of Bursty Cells vs p_1

Figure 10: Parameter sensitivity

.

[6] D. P. Dobkin, D. Gunopulos, and W. Maass. Computing the maximum bichromatic discrepancy with applications to computer graphics and machine learning. In *Journal of Computer and System Sciences*, 1996.

[7] G. P. C. Fung, J. X. Yu, H. Liu, and P. S. Yu. Time-dependent event hierarchy construction. In *KDD*, 2007.

[8] G. P. C. Fung, J. X. Yu, P. S. Yu, and H. Lu. Parameter free bursty events detection in text streams. In *VLDB*, 2005.

[9] J. Kleinberg. Bursty and hierarchical structure in streams. In *KDD*, 2002.

[10] J. M. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *J. ACM*, 2002.

[11] M. Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods 26(6), 1481-1496*, 1997.

[12] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. 2008.

[13] S. Sarawagi. Explaining differences in multidimensional aggregates. *The VLDB Journal*, 1999.

[14] A. Siddharthan, C. D. Manning, and H. Schutze. *Foundations of Statistical Natural Language Processing.* Cambridge University Press, 1999.

[15] Y. Zhu and D. Shasha. Efficient elastic burst detection in data streams. In *KDD*, 2003.

APPENDIX

A. BURST DETECTION OVER TIME SERIES

Following [9], a query q to an online social activity tracking system will return a set of documents D ordered (for the purposes of our discussion) by time. Assume documents arrive in discrete time steps, the time duration of each step being a constant. For example a time step could be a suitably chosen time duration such as a minute, hour, day etc. Among the documents in the same time step (e.g., arriving in the same hour), some are considered as *relevant* (e.g. documents that constitute valid answers to q) and the rest are considered to be *irrelevant*. Let S and $\hat{S}$ be two integer valued arrays, each element of which corresponds to a time step. In $\hat{S}$ we store the total number of documents in each time step, while in S we store the number of the corresponding relevant documents in that time step.

A simple instantiation of the model considers the case of only two levels of "burstiness". That is, every time step t in the data stream is characterized either as 'bursty' (state $s_t = 1$) or 'non-bursty' (state $s_t = 0$), depending on whether it contains a "large" or a "small" fraction of relevant documents. What constitutes "large" and "small" numbers of relevant documents is left as parameters in the model. Thus, the model introduces parameters p_0 and p_1 corresponding to thresholds for small and large numbers of relevant documents. Moreover the model introduced a cost parameter γ, that signifies the 'cost' incurred whenever one moves from a non-bursty time step to a bursty one, or vice versa. With these parameters specified, the assignment of states to each time step is conducted by minimizing a cost function $C(S, \hat{S})$ that depends on the number of relevant and non-relevant documents in each time step and the transitions.

Let $s_t \in \{0, 1\}$ denote the burst state assigned to the t-th time step. The total cost of these assignments is given by the equation

$$C(S,\hat{S}) = C_1(S,\hat{S}) + C_2(S,\hat{S}) = \sum_{t=1}^{n} c(s_t, r_t, d_t) + \sum_{t=1}^{n-1} \tau(s_t, s_{t+1}) \tag{7}$$

with $r_t = S[t]$ and $d_t = \hat{S}[t]$. In equation 7, the first term of the cost is defined as

$$c(s_t, r_t, d_t) = -\log\left[\binom{d_t}{r_t} p_{s_t}^{r_t} (1 - p_{s_t})^{d-r}\right]. \tag{8}$$

Furthermore, the term $\tau(s_t, s_{t+1})$ in equation 7 is defined by the formulas

$$\begin{aligned} \tau(1,0) &= \tau(0,1) = \gamma \\ \tau(0,0) &= \tau(1,1) = 0 \end{aligned}$$

and denotes a cost incurred whenever we move from a non-bursty state to a bursty state, i.e. when we move from state $s_t = 0$ to state $s_t = 1$.

Typically, p_0 is set to $\frac{\sum_{t=1}^{n} r_t}{\sum_{t=1}^{n} d_t}$, the weighted average of the normalized values of S and p_1 is set to some larger value. On the other hand, parameter γ introduces an additional difficulty in characterizing as 'bursty', elements of the array that follow non-bursty ones. Given the parameters p_0, p_1 and γ, we can decide the optimal assignment of states by running a standard forward dynamic programming algorithm on S and $\hat{S}$.

B. IMPROVEMENTS

In this section, we present algorithm *i-Spatial*, that improves upon the performance of algorithm *s-Spatial*.

Recall that the context in which we study spatial bursts involves data arriving from locations distributed across the globe which are represented by a grid on a world map. In this context, it is evident that the R^S and D^S distributions(of section 3) omitting the time range for simplicity, are *sparse*, i.e. in a particular time step there are parts of the grid that produce a large amount of data, while there are many cells for which there are no related data.

Secondly, as we observe in figure 2 the binary trees that constitute a transition graph under our model are not full (there are internal nodes of the trees that do not 'branch', i.e. have only one child). In addition, when the R^S and D^S distributions are sparse and under the convention that we always choose a specific dimension (for the purposes of our discussion the vertical) to resolve ties there will be long paths in the transition graph consisting of nodes with only one child (*non-branching paths*). Cells on these paths will all have the same state and we demonstrate how to take advantage of such conditions to speed up the process of spatial burst detection.

To become more specific, as we have seen so far the cost function consists of two terms C_1 and C_2 (see equation 2); the first depends only on the distributions R^S and D^S and the second contains the transition costs. Suppose for a moment that we do not take the term C_2 into account (i.e., set $\gamma = 0$) and we identify that there is a non-branching path in a transition tree consisting only of cells that have the same state s, bursty or non-bursty (see figure 11). In proposition 2, we prove that for $\gamma > 0$ the states of this path will have the same state s' – which is not necessarily equal to s.

PROPOSITION 2. *Let T be a transition tree and let $I \subseteq T$ be a non-branching path of cells that have the same burst state $s \in \{0,1\}$ when $\gamma = 0$. Then the points in I will have the same state $s' \in \{0,1\}$ when $\gamma > 0$ (in general $s \neq s'$).*

Proof Suppose that all cells g in I have the same state $s = 1$ when $\gamma = 0$. This means that

$$c(1, r_g, d_g) < c(0, r_g, d_g)$$

for all $g \in I$, (see figure 12, in which we've numbered the cells from 1 to n and depict the transitions). Suppose also that the cells at positions 0 and $n + 1$ have a fixed state 0 (if they had state 1 we could prove the lemma for a bigger interval). Then, we need only consider three choices for the states in the path I, when $\gamma \neq 0$ (shown in figure 12).

1. All cells are assigned state 1. In this case, the cost for the path I is equal to $\sum_{g=1}^{n} c(1, r_g, d_g) + 2\gamma$.
2. All cells are assigned state 0. In this case, the cost for the path I is equal to $\sum_{g=1}^{n} c(0, r_g, d_g)$.
3. At least one cell is assigned state 1 and at least one cell j is assigned state 0. In this case, the cost of the path I is at least $\sum_{g\in[1,n]-\{j\}} c(1, r_g, d_g) + c(0, r_j, d_j) + 4\gamma$.

It is easy to observe that the third choice is always worse than the first. Thus, we have that either all cells in I will be bursty or all will be non-bursty. By repeating the above process, we prove the lemma for the case where the cells in the non-branching path I have state 0 for $\gamma = 0$. □

Proposition 2 allows for improvement over the performance of `s-Spatial`. Since the costs $c(s, r_g, d_g)$ are additive w.r.t. r_g and d_g, we to observe (bullets 1 and 2 in the proof of the lemma) such n consecutive cells (of the same state when $\gamma = 0$) can be substituted with a single cell of values $r' = \sum_{g\in I} r_g$ and $d' = \sum_{g\in I} d_g$ before applying dynamic programming. We refer to the new transition graph as the *condensed* transition graph.

If n' is the number of points in the time-series after the substitution, then algorithm *spDynProg* (Appendix C, Alg 3) will be

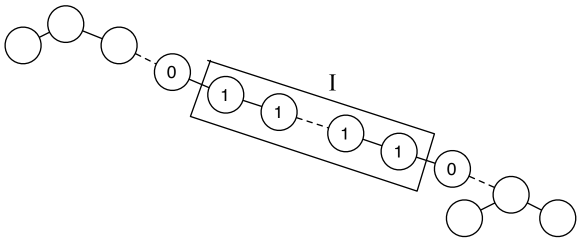


Figure 11: Cells with indexes 1 to n would be bursty if $\gamma = 0$. For $\gamma > 0$ there are three cases to consider.

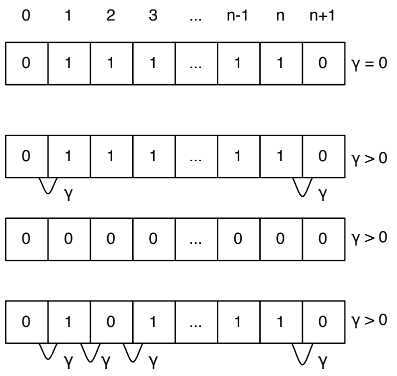


Figure 12: Cells in I would be bursty ($s = 1$) if $\gamma = 0$. For $\gamma > 0$ they will have the same state s', just as in the 1d case.

applied for a much smaller problem (in time $O(n'^2 b^3)$), and will have better running time in the average case. The worst case happens when for $\gamma = 0$ the states in every non-branching path follow the pattern $\ldots 0101010 \ldots$ (0 for non-bursty, 1 for bursty), in which $n' = n$ and there is no improvement in the running time of the dynamic programming algorithm. We refer to the improved version of algorithm *s-Spatial* as *i-Spatial* (pseudocode provided in Appendix C, Alg. 2).

B.1 Accounting for More States

So far, we have considered only the simple case of using only two states to signify a burst condition (bursty and non bursty). However, there might be cases where we want to model various degrees of burstiness and thus require $K > 2$ states. This might be especially desirable when the distributions R^S and D^S vary significantly and take a large range of values across the grid G.

Generalizing, we employ several binomial distributions that aid modelling multiple burst states. We wish to characterize with burst state $k \in \{0, 1, \ldots, K-1\}$ those cells with R^S and D^S values that are 'close' to the binomial distribution $BIN(p_k)$, for some $p_k \in [0, 1]$, $p_0 < p_1 < \ldots < p_{K-1}$. Therefore, we consider a cost function where for a cell $g \in G$ with r_g relevant documents and d_g total number of documents that has burst state $k \in \{0, 1, \ldots, K-1\}$, there is a term

$$c(k, r_g, d_g) = -\log\left[\binom{d_g}{r_g} p_k^{r_g} (1 - p_k)^{d_g - r_g}\right].$$

We create the transition graph in the same way as in section 3, but we modify the transition cost function as follows.

$$\begin{aligned} \tau(h, l) &= \tau(l, h) = (h - l) \cdot \gamma \\ \tau(h, h) &= \tau(l, l) = 0. \end{aligned}$$

for burst states $l < h$.

Furthermore, the results of proposition 2, can be generalized, as stated in proposition 3 (proof omitted due to space constraints).

PROPOSITION 3. *Let T be a transition tree and let $I \subseteq T$ be a non-branching path of cells that have the same burst state $s \in \{0, \ldots, k-1\}$ when $\gamma = 0$. Then, the points in I will have the same state $s' \in \{0, 1\}$ when $\gamma > 0$ (in general $s \neq s'$).*

C. PSEUDOCODE

This section provides pseudocode for algorithms *s-Spatial* and *i-Spatial*, described in section 3 and appendix B, respectively.

Algorithm 1 s-Spatial

```
Input: Grid $G$ // The $R^S$ and $D^S$ distributions are also contained in the grid

Produce the transition graph $X$ of $G$
for every tree $T$ in $X$ do
    Use spDynProg to calculate the optimal states of $T$
end for
```

Algorithm 2 i-Spatial

```
Input: Grid $G$ // The $R^S$ and $D^S$ distributions are also contained in the grid

Produce transition graph $X$ on $G$
Use Proposition 2 to compute $X'$, the condensed graph of $X$
for every tree $T$ in $X'$ do
    Use spDynProg to calculate the optimal states of $T$
end for
```

Algorithm 3 spDynProg

```
Input: Cell $root$, State $s_{root}$
Output: Cost $C$ // optimal cost for subtree under $root$

$C_1$ = $c(s_{root}, root.r, root.d)$
Cost $C$ = $\infty$

for State $s_{left}$ = 0 to 1 do
    for State $s_{right}$ = 0 to 1 do
        $C = \min\{C, C_1 + spDynProg(root.left, s_{left}) + \tau(s_{root}, s_{left}) + spDynProg(root.right, s_{right}) + \tau(s_{root}, s_{right})\}$
    end for
end for
```

D. EXPERIMENTS: BURST ATTRIBUTION

We submitted the query 'cricket' to BlogScope with a temporal restriction on the first 10 days of March 2008. We then assessed a solution to the burst attribution problem 4.2 for the regions of India and USA, for the `Age` attribute of blog posts, which takes values from 1 to 80. We varied the number of buckets in which we partition the domain of the attribute and we measure the time required to solve problem 4.2 in order to observe performance trends. We repeated the same experiment, this time with attributes `Age` and `Gender`, the latter assuming only two values $\{Male, Female\}$. We split the domain of `Age` in a variable number of buckets k, so that for a fixed k a blog post can take $2 \times k$ different pairs of values for (`Age`, `Gender`).

In figure 13, we plot the running time (in μsec) required to solve problem 4.2 for both cases described above as a function of the number of buckets used to split the domain of attribute `Age`. We observe that these running times impose a very small delay in BlogScope.

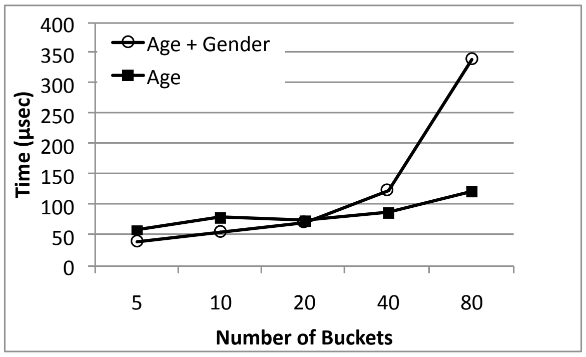


Figure 13: The running time needed to solve Problem 4.2 of section 4 for the case of a single attribute `Age` of blog posts and for a pair of attributes (`Age, Gender`), for a varying number of Age Buckets and for the regions of India and USA.

We now demonstrate the practical utility of the two problems defined in section 4 using real examples. We consider a single attribute `Age`, dividing its domain (0 - 80) into 8 buckets ([01-10], $[11-20]$, . . ., [70-80]).

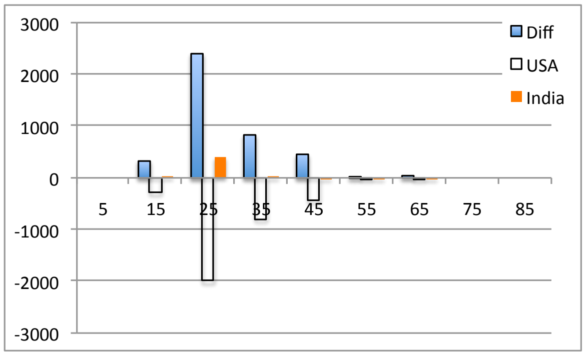
The first example is a query for keyword 'cricket' in the first 10 days of March 2008. For the same query (Figure 8), there are many bursts in India related to 'cricket' for that period, while there is none in USA. We performed burst detection (described in section 6.1.2) and then for every bucket c of attribute `Age`, we plotted the corresponding burst weight W_c for India (first bar) and USA (second bar) and their difference (third bar). All ages contribute non-negative burst weights in the spatial burst for India for that period of time. On the other hand, solving Problem 4.2 we obtain only the bucket $[21-30]$ as a solution. As a result, we observe that it is the bucket $[21-30]$ (corresponding to individuals in that age group) that contributes the big difference in the burstiness of the keyword 'cricket' between the two countries.

As another example, consider the keyword 'snowstorm' again for the first 10 days of March 2008. The keyword 'snowstorm' had large bursts in USA, while it had none in the UK. We plot the corresponding burst weights in figure 14(b). Following a similar reasoning, figure 14(b) indicates that individuals with ages between 21 and 60 years old contributed largely to the burst. Furthermore, *all* ages between 21 and 60 are returned as a solution to problem 4.2 for the two regions (the difference in the burstiness of the regions is attributed to all of them).

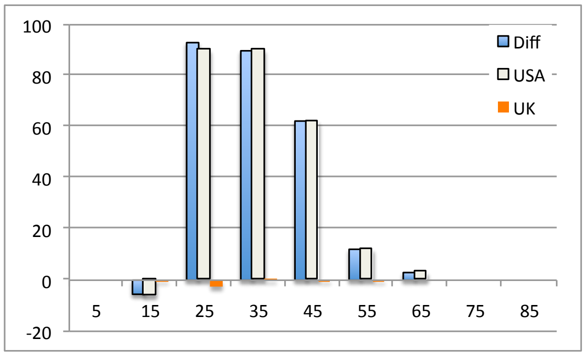
E. EXPERIMENTS: BURST DESCRIPTION

We now proceed to test the scalability and the quality of results for the three algorithms, `ContentBased`, `CurveCorr` and `CurveEstim`. We test the three algorithms for a set of keywords that had a burst at the beginning of year 2008 (from Jan. 4th to Jan. 8th). This set of keywords is SQ = {'giuliani', 'clinton', 'obama', 'mccain', 'romney'}; it consists of the names of well known politicians that participated in the presidential US primaries. The burst is related to the Iowa caucus on Jan. 4th. For each algorithm we request top-3 results for each query expansion and vary the depth of the expansion from 2 to 5. We compute the average running time of the three algorithms as a performance indicator.

In the graph of figure 15, we report the running time of the al-



(a) 'cricket', India and USA



(b) 'snowstorm', USA and UK

Figure 14: Burst weights (and their differences) for different age buckets produced for queries 'cricket' and 'snowstorm' for the 10 first days of March 2008.

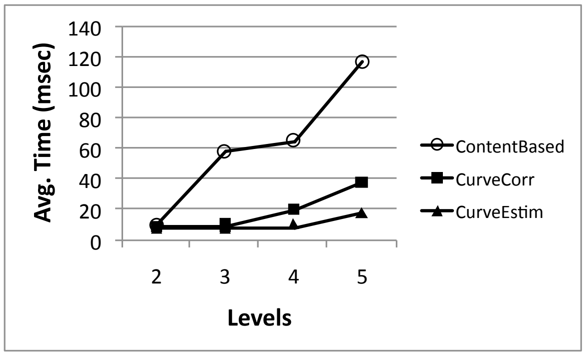


Figure 15: The average running time of the three keyword hierarchy construction algorithms for different number of hierarchy levels.

gorithms (in *seconds*) as a function of the number of levels in the expanded hierarchy. Both `CurveCorr` and `CurveEstim` scale much better than `ContentBased`, since their overheads are much lower (with $CurveEstim$ being faster than $CurveCorr$ since it has lower overheads).

Finally, in figures 16(a) and 16(b) we provide examples of hierarchies returned by the three algorithms for the keywords 'giuliani' and 'romney'. In figure 16(a), we show the expansion of the keyword 'giuliani' by 3 keywords in one level. Algorithms `CurveCorr` and `CurveEstim` produce keywords that are highly related to the particular presidential candidate (Hillary Clinton and

Mike Huckabee were candidates as well and the keywords 'mike', 'hillary' and 'candidate' are also in the top results returned by ContentBased). The situation is similar when we expand the keyword 'romney' with one keyword at the first level and two keywords at the second level. The keywords are closely related to this presidential candidate; the results of CurveCorr and CurveEstim overlap with the results of ContentBased (for keywords 'obama' and 'mccain') and the keywords 'clinton', 'ron' and 'candidate' are again among the top keywords returned by ContentBased.

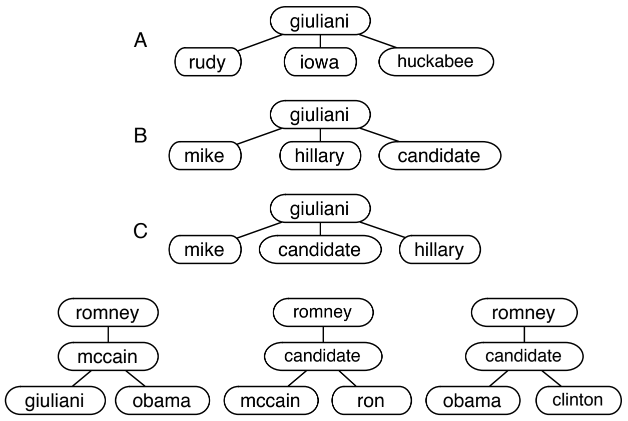


Figure 16: Examples of expansion for the keywords 'giuliani' and 'romney', produced by the three algorithms described in section 5 – (A) ContentBased (B) CurveCorr (C) CurveEstim