Bursty and Hierarchical Structure in Streams

Jon Kleinberg *

Abstract

A fundamental problem in text data mining is to extract meaningful structure from document streams that arrive continuously over time. E-mail and news articles are two natural examples of such streams, each characterized by topics that appear, grow in intensity for a period of time, and then fade away. The published literature in a particular research field can be seen to exhibit similar phenomena over a much longer time scale. Underlying much of the text mining work in this area is the following intuitive premise — that the appearance of a topic in a document stream is signaled by a "burst of activity," with certain features rising sharply in frequency as the topic emerges.

The goal of the present work is to develop a formal approach for modeling such "bursts," in such a way that they can be robustly and efficiently identified, and can provide an organizational framework for analyzing the underlying content. The approach is based on modeling the stream using an infinite-state automaton, in which bursts appear naturally as state transitions; in some ways, it can be viewed as drawing an analogy with models from queueing theory for bursty network traffic. The resulting algorithms are highly efficient, and yield a nested representation of the set of bursts that imposes a hierarchical structure on the overall stream. Experiments with e-mail and research paper archives suggest that the resulting structures have a natural meaning in terms of the content that gave rise to them.

*Department of Computer Science, Cornell University, Ithaca NY 14853. Email: kleinber@cs.cornell.edu.
Supported in part by a David and Lucile Packard Foundation Fellowship, an ONR Young Investigator Award, NSF ITR/IM Grant IIS-0081334, and NSF Faculty Early Career Development Award CCR-9701399.

1 Introduction

Documents can be naturally organized by topic, but in many settings we also experience their arrival over time. E-mail and news articles provide two clear examples of such *document streams*: in both cases, the strong temporal ordering of the content is necessary for making sense of it, as particular topics appear, grow in intensity, and then fade away again. Over a much longer time scale, the published literature in a particular research field can be meaningfully understood in this way as well, with particular research themes growing and diminishing in visibility across a period of years. Work in the areas of topic detection and tracking [2, 3, 5, 61, 62], text mining [36, 56, 57, 58], and visualization [26, 43, 60] has explored techniques for identifying topics in document streams comprised of news stories, using a combination of content analysis and time-series modeling.

Underlying a number of these techniques is the following intuitive premise — that the appearance of a topic in a document stream is signaled by a "burst of activity," with certain features rising sharply in frequency as the topic emerges. The goal of the present work is to develop a formal approach for modeling such "bursts," in such a way that they can be robustly and efficiently identified, and can provide an organizational framework for analyzing the underlying content. At one level, the approach presented here can be viewed as drawing an analogy with models from queueing theory for bursty network traffic (see e.g. [32]). In addition, however, the analysis of the underlying burst patterns reveals a latent hierarchical structure that often has a natural meaning in terms of the content of the stream.

My initial aim in studying this issue was a very concrete one: I wanted a better organizing principle for the enormous archives of personal e-mail that I was accumulating. Abundant anecdotal evidence, as well as academic research [6, 42, 59], suggested that my own experience with "e-mail overload" corresponded to a near-universal phenomenon — a consequence of both the rate at which e-mail arrives, and the demands of managing volumes of saved personal correspondence that can easily grow into tens and hundreds of megabytes of pure text content. And at a still larger scale, e-mail has become the raw material for legal proceedings [34] and historical investigation [8, 38, 44] — with the National Archives, for example, agreeing to accept tens of millions of e-mail messages from the Clinton White House [45]. In sum, there are several settings where it is a crucial problem to find structures that can help in making sense of large volumes of e-mail.

An active line of research has applied text indexing and classification to develop e-mail interfaces that organize incoming messages into *folders* on specific topics, sometimes recommending further actions on the part of a user [4, 9, 13, 29, 30, 39, 46, 47, 49, 50, 51, 53, 54] — in effect, this framework seeks to automate a kind of filing system that many users implement manually. There has also been work on developing query interfaces to fully-indexed collections of e-mail [7].

My interest here is in exploring organizing structures based more explicitly on the role of *time* in e-mail and other document streams. Indeed, even the flow of a single focused topic is modulated by the rate at which relevant messages or documents arrive, dividing naturally into more localized episodes that correspond to bursts of activity of the type suggested above. For example, my saved e-mail contains over a thousand messages relevant to the topic "grant proposals" — announcements of new funding programs, planning of proposals, and correspondence with co-authors. While one could divide this collection into sub-topics based on message content — certain people, programs, or funding agencies form the topics of some messages but not others — an equally natural and substantially orthogonal organization for this topic would take into account the sequence of episodes reflected in the set of messages — bursts that surround the planning and writing of certain proposals. Indeed, certain sub-topics (e.g. "the process of gathering people together for our large NSF ITR proposal") may be much more easily characterized by a sudden confluence of message-sending over a particular period of time than by textual features of the messages themselves. One can easily argue that many of the large topics represented in a document stream are naturally punctuated by bursts in this way, with the flow of relevant items intensifying in certain key periods. A general technique for highlighting these bursts thus has the potential to expose a great deal of fine-grained structure.

Before moving to a more technical overview of the methodology, let me suggest one further perspective on this issue, quite distant from computational concerns. If one were to view a particular folder of e-mail not simply as a document stream but also as something akin to a narrative that unfolds over time, then one immediately brings into play a body of work in narrative theory that deals explicitly with the bursty nature of time in stories, and the way in which particular events are signaled by a compression of the time-sense. In a concrete reference to this idea, E.M. Forster, lecturing on the structure of the novel in the 1920's, asserted that

> . . . there seems something else in life besides time, something which may conveniently be called "value," something which is measured not by minutes or hours but by intensity, so that when we look at our past it does not stretch back evenly but piles up into a few notable pinnacles, and when we look at the future it seems sometimes a wall, sometimes a cloud, sometimes a sun, but never a chronological chart [17].

This idea is developed more explicitly in the work of Genette [19, 20], Chatman [11], and others, which studies the non-uniform relation between the time span of a story's events and the amount of attention devoted to these events in the telling of the story. Such *anisochronies* [19] — points at which the perceived time-sense in a sequence of events abruptly speeds up or slows down — play a significant role in structuring an underlying narrative.

Modeling Bursty Streams. Suppose we were presented with a document stream — for concreteness, consider a large folder of e-mail on a single broad topic. How should we go about identifying the main bursts of activity, and how do they help impose additional structure on the stream? The basic point emerging from the discussion above is that such bursts correspond roughly to points at which the intensity of message arrivals increases sharply, perhaps from once every few weeks or days to once every few hours or minutes. But the rate of arrivals is in general very "rugged": it does not typically rise smoothly to a crescendo and then fall away, but rather exhibits frequent alternations of rapid flurries and longer pauses in close proximity. Thus, methods that analyze gaps between consecutive message arrivals in too simplistic a way can easily be pulled into identifying large numbers of short spurious bursts, as well as fragmenting long bursts into many smaller ones. Moreover, a simple enumeration of close-together sets of messages is only a first step toward more intricate structure. The broader goal is thus to extract global structure from a robust kind of data reduction — identifying bursts only when they have sufficient intensity, and in a way that allows a burst to persist smoothly across a fairly non-uniform pattern of message arrivals.

My approach here is to model the stream using an infinite-state automaton $\mathcal{A}$, which at any point in time can be in one of an underlying set of states, and emits messages at different rates depending on its state. Specifically, the automaton $\mathcal{A}$ has a set of states that correspond to increasingly rapid rates of emission, and the onset of a burst is signaled by a state transition — from a lower state to a higher state. By assigning costs to state transitions, one can control the frequency of such transitions, preventing very short bursts and making it easier to identify long bursts despite transient changes in the rate of the stream. The overall framework is developed in Section 2. It can be viewed as drawing an analogy to the use of on-off Markov sources in modeling bursty network traffic (see for example the overview article by Kelly [32]), as well as drawing on the formalism of hidden Markov models [48].

Using an automaton with states that correspond to higher and higher intensities provides an additional source of analytical leverage — the bursts associated with state transitions form a naturally nested structure, with a long burst of low intensity potentially containing several bursts of higher intensity inside it (and so on, recursively). For a folder of related e-mail messages, we will see in Sections 2 and 3 that this can provide a hierarchical decomposition of the temporal order, with long-running episodes intensifying into briefer ones according to a natural tree structure. This tree can thus be viewed as imposing a fine-grained organization on the sub-episodes within the message stream.

Following this development, Section 4 focuses on a case in which the document stream is comprised not of e-mail messages but of computer science conference paper titles over the past several decades; the set of bursts in this stream corresponds roughly to the appearance and disappearance of certain terms of interest in the papers. Section 5 discusses the connections to related work in a range of areas, particularly the striking recent work of Swan, Allan, and Jensen [56, 57, 58] on overview timelines, which forms the body of research closest to the approach here. Finally, Section 6 discusses some further applications of the methodology — how burstiness in arrivals can help to identify certain messages as "landmarks" in a large corpus of e-mail; and how the overall framework can be applied to logs of Web traffic.

2 A Weighted Automaton Model

Perhaps the simplest randomized model for generating a sequence of message arrival times is based on an exponential distribution: messages are emitted in a probabilistic manner, so that the gap x in time between messages i and $i+1$ is distributed according to the "memoryless" exponential density function $f(x) = \alpha e^{-\alpha x}$, for a parameter $\alpha > 0$. (In other words, the probability that the gap exceeds x is equal to $e^{-\alpha x}$.) The expected value of the gap in this model is α^{-1}, and hence one can refer to α as the *rate* of message arrivals.

Intuitively, a "bursty" model should extend this simple formulation by exhibiting periods of lower rate interleaved with periods of higher rate. A natural way to do this is to construct a model with multiple *states*, where the rate depends on the current state. Let us start with a basic model that incorporates this idea, and then extend it to the models that will primarily be used in what follows.

A two-state model. Arguably the most basic bursty model of this type would be constructed from a probabilistic automaton $\mathcal{A}$ with two states q_0 and q_1, which we can think of as corresponding to "low" and "high." When $\mathcal{A}$ is in state q_0, messages are emitted at a slow rate, with gaps x between consecutive messages distributed independently according to a density function $f_0(x) = \alpha_0 e^{-\alpha_0 x}$ When $\mathcal{A}$ is in state q_1, messages are emitted at a faster rate, with gaps distributed independently according to $f_1(x) = \alpha_1 e^{-\alpha_1 x}$, where $\alpha_1 > \alpha_0$. Finally, between messages, $\mathcal{A}$ changes state with probability $p \in (0, 1)$, remaining in its current state with probability $1 - p$, independently of previous emissions and state changes.

Such a model could be used to generate a sequence of messages in the natural way. $\mathcal{A}$ begins in state q_0. Before each message (including the first) is emitted, $\mathcal{A}$ changes state with probability p. A message is then emitted, and the gap in time until the next message is determined by the distribution associated with $\mathcal{A}$'s current state.

One can apply this generative model to find a likely state sequence, given a set of messages. Suppose there is a given set of $n + 1$ messages, with specified arrival times; this determines a sequence of n *inter-arrival gaps* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. The development here will use the basic assumption that all gaps x_i are strictly positive. We can use the Bayes procedure (as in e.g. [14]) to determine the conditional probability of a state sequence $\mathbf{q} = (q_{i_1}, \ldots, q_{i_n})$; note that this must be done in terms of the underlying density functions, since the gaps are not drawn from discrete distributions. Each state sequence $\mathbf{q}$ induces a density function $f_{\mathbf{q}}$ over sequences of gaps, which has the form $f_{\mathbf{q}}(x_1, \ldots, x_n) = \prod_{t=1}^{n} f_{i_t}(x_t)$. If b denotes the number of state transitions in the sequence $\mathbf{q}$ — that is, the number of indices i_t so that $q_{i_t} \neq q_{i_{t+1}}$ — then the (prior) probability of $\mathbf{q}$ is equal to

$$\Big(\prod_{i_t \neq i_{t+1}} p\Big)\Big(\prod_{i_t = i_{t+1}} 1 - p\Big) = p^b (1-p)^{n-b} = \left(\frac{p}{1-p}\right)^b (1-p)^n.$$

(In this calculation, let $i_0 = 0$, since $\mathcal{A}$ starts in state q_0.) Now,

$$\begin{aligned}
\Pr[\mathbf{q} \mid \mathbf{x}] &= \frac{\Pr[\mathbf{q}]\, f_{\mathbf{q}}(\mathbf{x})}{\sum_{\mathbf{q}'} \Pr[\mathbf{q}']\, f_{\mathbf{q}'}(\mathbf{x})} \\
&= \frac{1}{Z}\left(\frac{p}{1-p}\right)^{b} (1-p)^{n} \prod_{t=1}^{n} f_{i_t}(x_t),
\end{aligned}$$

where Z is the normalizing constant $\sum_{\mathbf{q}'} \Pr[\mathbf{q}']\, f_{\mathbf{q}'}(\mathbf{x})$. Finding a state sequence $\mathbf{q}$ maximizing this probability is equivalent to finding one that minimizes

$$-\ln \Pr[\mathbf{q} \mid \mathbf{x}] = b \ln\left(\frac{1-p}{p}\right) + \left(\sum_{t=1}^{n} -\ln f_{i_t}(x_t)\right) - n\ln(1-p) + \ln Z.$$

Since the third and fourth terms are independent of the state sequence, this latter optimization problem is equivalent to finding a state sequence $\mathbf{q}$ that minimizes the following *cost function*:

$$c(\mathbf{q} \mid \mathbf{x}) = b \ln\left(\frac{1-p}{p}\right) + \left(\sum_{t=1}^{n} -\ln f_{i_t}(x_t)\right)$$

Finding a state sequence to minimize this cost function is a problem that can be motivated intuitively on its own terms, without recourse to the underlying probabilistic model. The first of the two terms in the expression for $c(\mathbf{q} \mid \mathbf{x})$ favors sequences with a small number of state transitions, while the second term favors state sequences that conform well to the sequence $\mathbf{x}$ of gap values. Thus, one expects the optimum to track the global structure of bursts in the gap sequence, while holding to a single state through local periods of non-uniformity. Varying the coefficient on b controls the amount of "inertia" fixing the automaton in its current state.

The next step is to extend this simple "high-low" model to one with a richer state set, using a cost model; this will lead to a method that also extracts hierarchical structure from the pattern of bursts.

An infinite-state model. Consider a sequence of $n+1$ messages that arrive over a period of time of length T. If the messages were spaced completely evenly over this time interval, then they would arrive with gaps of size $\overline{g} = T/n$. Bursts of greater and greater intensity would be associated with gaps smaller and smaller than $\overline{g}$. This suggests focusing on an infinite-state automaton whose states correspond to gap sizes that may be arbitrarily small, so as to capture the full range of possible bursts. The development here will use a cost model as in the two-state case, where the underlying goal is to find a state sequence of minimum cost.

Thus, consider an automaton with a "base state" q_0 that has an associated exponential density function f_0 with rate $\alpha_0 = \overline{g}^{-1} = n/T$ — consistent with completely uniform message arrivals. For each $i > 0$, there is a state q_i with associated exponential density f_i having

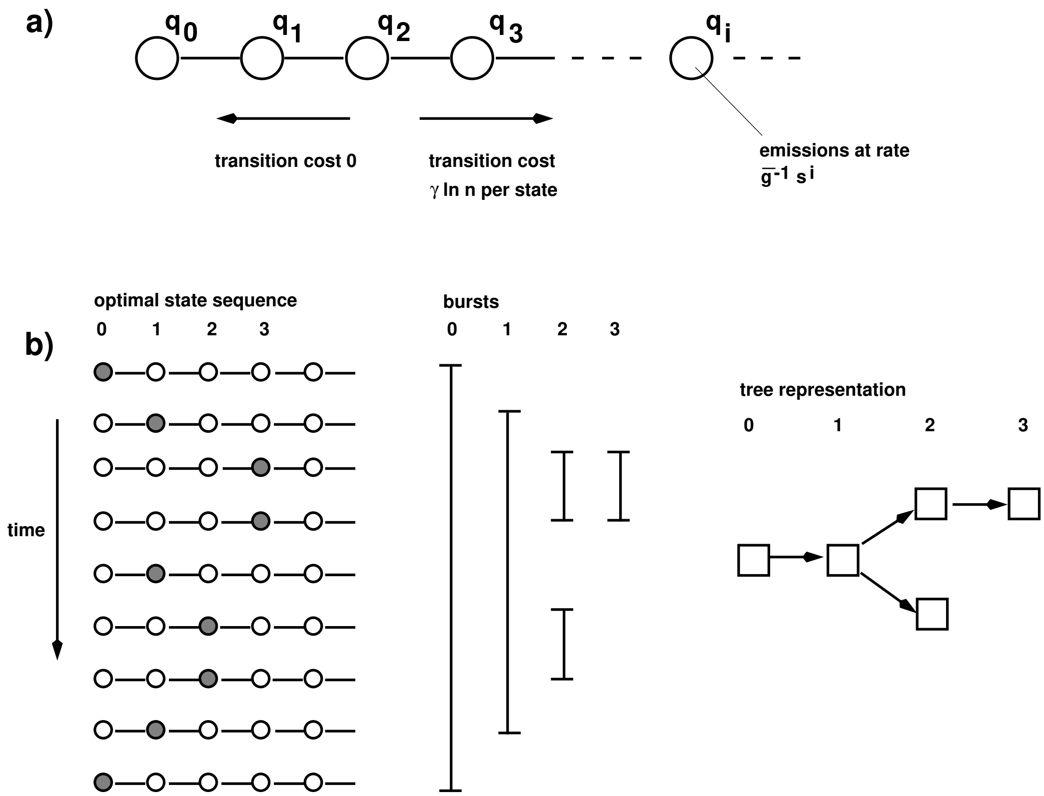

Figure 1: An infinite-state model for bursty sequences. (a) The infinite-state automaton $\mathcal{A}^*_{s,\gamma}$; in state q_i, messages are emitted at a spacing in time that is distributed according to $f(x) = \alpha_i e^{-\alpha_i x}$, where $\alpha_i = \overline{g}^{-1} s^i$. There is a cost to move to states of higher index, but not to states of lower index. (b) Given a sequence of gaps between message arrivals, an optimal state sequence in $\mathcal{A}^*_{s,\gamma}$ is computed. This gives rise to a set of nested *bursts*: intervals of time in which the optimal state has at least a certain index. The inclusions among the set of bursts can be naturally represented by a tree structure.

rate $\alpha_i = \overline{g}^{-1} s^i$, where $s > 1$ is a scaling parameter. (i will be referred to as the *index* of the state q_i.) In other words, the infinite sequence of states $q_0, q_1, \ldots$ models inter-arrival gaps that decrease geometrically from $\overline{g}$; there is an expected rate of message arrivals that intensifies for larger and larger values of i. Finally, for every i and j, there is a cost $\tau(i,j)$ associated with a state transition from q_i to q_j. The framework allows considerable flexibility in formulating the cost function; for the work described here, $\tau(\cdot,\cdot)$ is defined so that the cost of moving from a lower-intensity burst state to a higher-intensity one is proportional to the number of intervening states, but there is no cost for the automaton to end a higher-intensity burst and drop down to a lower-intensity one. Specifically, when $j > i$, moving from q_i to q_j incurs a cost of $(j-i)\gamma \ln n$, where $\gamma > 0$ is a parameter; and when $j < i$, the cost is 0. See Figure 1(a) for a schematic picture.

This automaton, with its associated parameters s and γ, will be denoted $\mathcal{A}^*_{s,\gamma}$. Given a sequence of positive gaps $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ between message arrivals, the goal — by analogy with the two-state model above — is to find a state sequence $\mathbf{q} = (q_{i_1}, \ldots, q_{i_n})$ that minimizes the cost function

$$c\,(\mathbf{q} \mid \mathbf{x}) \;\; = \;\; \left(\sum_{t=0}^{n-1} \tau(i_t, i_{t+1})\right) + \left(\sum_{t=1}^{n} -\ln f_{i_t}(x_t)\right).$$

(Let $i_0 = 0$ in this expression, so that $\mathcal{A}^*_{s,\gamma}$ starts in state q_0.) Since the set of possible $\mathbf{q}$ is infinite, one cannot automatically assert that the minimum is even well-defined; but this will be established in Theorem 2.1 below. As before, minimizing the first term is consistent with having few state transitions — and transitions that span only a few distinct states — while minimizing the second term is consistent with passing through states whose rates agree closely with the inter-arrival gaps. Thus, the combined goal is to track the sequence of gaps as well as possible without changing state too much.

Observe that the scaling parameter s controls the "resolution" with which the discrete rate values of the states are able to track the real-valued gaps; the parameter γ controls the ease with which the automaton can change states. In what follows, γ will often be set to a default value of 1; we can use $\mathcal{A}^*_s$ to denote $\mathcal{A}^*_{s,1}$.

Computing a minimum-cost state sequence. Given a a sequence of positive gaps $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ between message arrivals, consider the algorithmic problem of finding a state sequence $\mathbf{q} = (q_{i_1}, \ldots, q_{i_n})$ in $\mathcal{A}^*_{s,\gamma}$ that minimizes the cost $c\,(\mathbf{q} \mid \mathbf{x})$; such a sequence will be called *optimal*. To establish that the minimum is well-defined, and to provide a means of computing it, it is useful to first define a natural finite restriction of the automaton: for a natural number k, one simply deletes all states but $q_0, q_1, \ldots, q_{k-1}$ from $\mathcal{A}^*_{s,\gamma}$, and denotes the resulting k-state automaton by $\mathcal{A}^k_{s,\gamma}$. Note that the two-state automaton $\mathcal{A}^2_{s,\gamma}$ is essentially equivalent (by an amortization argument) to the probabilistic two-state model described earlier.

It is not hard to show that computing an optimal state sequence in $\mathcal{A}^*_{s,\gamma}$ is equivalent to doing so in one of its finite restrictions.

Theorem 2.1 *Let $\delta(\mathbf{x}) = \min_{i=1}^{n} x_i$ and $k = \lceil 1 + \log_s T + \log_s \delta(\mathbf{x})^{-1} \rceil$. (Note that $\delta(\mathbf{x}) > 0$, since all gaps are positive.) If $\mathbf{q}^*$ is an optimal state sequence in $\mathcal{A}^k_{s,\gamma}$, then it is also an optimal state sequence in $\mathcal{A}^*_{s,\gamma}$.*

Before proceeding to the proof, here are two key points to note. First, in all the experiments here, an optimal state sequence in $\mathcal{A}^*_{s,\gamma}$ can be found by restricting to a number of states k that is a very small constant, always at most 25.

Second, some condition requiring gaps to be positive is necessary in order for the theorem to hold, as the following example shows. Suppose that $\mathbf{x}$ were to consist of n gaps, each equal to 0, where n is large enough that $s^n > n^\gamma$. Then the state sequence $\mathbf{q}^{(\mathbf{j})}$ in which all n states are equal to q_j has cost

$$\begin{aligned} c\left(\mathbf{q}^{(\mathbf{j})} \mid \mathbf{x}\right) &= j(\gamma \ln n) - n \ln f_j(0) = j(\gamma \ln n) - n \ln \alpha_j \\ &= j(\gamma \ln n) - nj \ln s + n \ln \overline{g} = j(\gamma \ln n - n \ln s) + n \ln \overline{g}. \end{aligned}$$

For increasing values of j, these costs $c\left(\mathbf{q}^{(\mathbf{j})} \mid \mathbf{x}\right)$ form a sequence of negative numbers tending to $-\infty$, and hence there is no state sequence in $\mathcal{A}^*_{s,\gamma}$ that achieves a cost less than or equal to that of all others.

When all gaps are positive, however, no such example is possible, since Theorem 2.1 establishes that there is a state sequence in $\mathcal{A}^*_{s,\gamma}$ achieving the minimum cost.

Proof of Theorem 2.1. Let $\mathbf{q}^* = (q_{\ell_1}, \ldots, q_{\ell_n})$ be an optimal state sequence in $\mathcal{A}^k_{s,\gamma}$, and let $\mathbf{q} = (q_{i_1}, \ldots, q_{i_n})$ be an arbitrary state sequence in $\mathcal{A}^*_{s,\gamma}$. As always, set $\ell_0 = i_0 = 0$, since both sequences start in state q_0; for notational purposes, it is useful to define $\ell_{n+1} = i_{n+1} = 0$ as well. The goal is to show that $c\,(\mathbf{q}^* \mid \mathbf{x}) \leq c\,(\mathbf{q} \mid \mathbf{x})$.

If $\mathbf{q}$ does not contain any states of index greater than $k-1$, this inequality follows from the fact that $\mathbf{q}^*$ is an optimal state sequence in $\mathcal{A}^k_{s,\gamma}$. Otherwise, consider the state sequence $\mathbf{q}' = (q_{i'_1}, \ldots, q_{i'_n})$ where $i'_t = \min(i_t, k-1)$. It is straightforward to verify that

$$\sum_{t=0}^{n-1} \tau(i'_t, i'_{t+1}) \leq \sum_{t=0}^{n-1} \tau(i_t, i_{t+1}).$$

Now, for a particular choice of t between 1 and n, consider the expression $-\ln f_j(x_t) = \alpha_j x_t - \ln \alpha_j$; what is the value of j for which it is minimized? The function $h(\alpha) = \alpha x_t - \ln \alpha$ is concave upwards over the interval $(0, \infty)$, with a global minimum at $\alpha = x_t^{-1}$. Thus, if j^* is such that $\alpha_{j^*} \leq x_t^{-1} \leq \alpha_{j^*+1}$, then the minimum of $-\ln f_j(x_t)$ is achieved at one of j^* or j^*+1; moreover, if $j'' \geq j' \geq j^*+1$, then $-\ln f_{j''}(x_t) \geq -\ln f_{j'}(x)$.

Since $k = \lceil 1 + \log_s T + \log_s \delta(\mathbf{x})^{-1} \rceil$, one has

$$\alpha_{k-1} = \overline{g}^{-1} s^{k-1} = \frac{n}{T} \cdot s^{k-1} \geq \frac{1}{T} \cdot s^{\log_s T + \log_s \delta(\mathbf{x})^{-1}} = \frac{1}{T} \frac{T}{\delta(\mathbf{x})} = \frac{1}{\delta(\mathbf{x})}.$$

Since $\delta(\mathbf{x})^{-1} \geq x_t^{-1}$ for any $t = 1, 2, \ldots, n$, the index $k-1$ is at least as large as the j for which $-\ln f_j(x_t)$ is minimized. It follows that for those t for which $i_t \neq i_{t'}$ one has $-\ln f_{i'_t}(x_t) \leq -\ln f_{i_t}(x_t)$, since $i_t > i_{t'} = k-1$.

Combining these inequalities for the state transition costs and the gap costs, one obtains

$$c\,(\mathbf{q}' \mid \mathbf{x}) = \left(\sum_{t=0}^{n-1} \tau(i'_t, i'_{t+1})\right) + \left(\sum_{t=1}^{n} -\ln f_{i'_t}(x_t)\right) \leq \left(\sum_{t=0}^{n-1} \tau(i_t, i_{t+1})\right) + \left(\sum_{t=1}^{n} -\ln f_{i_t}(x_t)\right) = c\,(\mathbf{q} \mid \mathbf{x}).$$

Since $\mathbf{q}'$ is a state sequence in $\mathcal{A}^k_{s,\gamma}$, and since $\mathbf{q}^*$ is an optimal state sequence for this automaton, it follows that $c\,(\mathbf{q}^* \mid \mathbf{x}) \leq c\,(\mathbf{q}' \mid \mathbf{x}) \leq c\,(\mathbf{q} \mid \mathbf{x})$. ■

In view of the theorem, it is enough to give an algorithm that computes an optimal state sequence in an automaton of the form $\mathcal{A}^k_{s,\gamma}$. This can be done by adapting the standard forward dynamic programming algorithm used for hidden Markov models [48] to the model and cost function defined here: One defines $C_j(t)$ to be the minimum cost of a state sequence for the input $x_1, x_2, \ldots, x_t$ that must end with state q_j, and then iteratively builds up the values of $C_j(t)$ in order of increasing t using the recurrence relation $C_j(t) = -\ln f_j(x_t) + \min_\ell(C_\ell(t-1) + \tau(\ell, j))$ with initial conditions $C_0(0) = 0$ and $C_j(0) = \infty$ for $j > 0$.

Note that although the final computation of an optimal state sequence is carried out by recourse to a finite-state model, working with the infinite model has the advantage that a number of states k is not fixed *a priori*; rather, it emerges in the course of the computation, and in this way the automaton $\mathcal{A}^*_{s,\gamma}$ essentially "conforms" to the particular input instance.

3 Hierarchical Structure and E-mail Streams

Extracting hierarchical structure. From an algorithm to compute an optimal state sequence, one can then define the basic representation of a set of bursts, according to a hierarchical structure.

For a set of messages generating a sequence of positive inter-arrival gaps $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, suppose that an optimal state sequence $\mathbf{q} = (q_{i_1}, q_{i_2}, \ldots, q_{i_n})$ in $\mathcal{A}^*_{s,\gamma}$ has been determined. Following the discussion of the previous section, we can formally define a *burst of intensity* j to be a maximal interval over which $\mathbf{q}$ is in a state of index j or higher. More precisely, it is an interval $[t, t']$ so that $i_t, \ldots, i_{t'} \geq j$ but i_{t-1} and $i_{t'+1}$ are less than j (or undefined if $t - 1 < 0$ or $t' + 1 > n$).

It follows that bursts exhibit a natural nested structure: a burst of intensity j may contain one or more sub-intervals that are bursts of intensity $j + 1$; these in turn may contain sub-intervals that are bursts of intensity $j+2$; and so forth. This relationship can be represented by a rooted tree Γ, as follows. There is a node corresponding to each burst; and node v is a child of node u if node u represents a burst B_u of intensity j (for some value of j), and node v represents a burst B_v of intensity $j + 1$ such that $B_v \subseteq B_u$. Note that the root of Γ corresponds to the single burst of intensity 0, which is equal to the whole interval $[0, n]$.

Thus, the tree Γ captures hierarchical structure that is implicit in the underlying stream. Figure 1(b) shows the transformation from an optimal state sequence, to a set of nested bursts, to a tree.

Hierarchy in an e-mail stream. Let us now return to one of the initial motivations for this model, and consider a stream of e-mail messages. What does the hierarchical structure of bursts look like in this setting?

I applied the algorithm to my own collection of saved e-mail, consisting of messages sent and received between June 9, 1997 and August 23, 2001. (The cut-off date is chosen here so as to roughly cover four academic years.) First, here is a brief summary of this collection. Every piece of mail I sent or received during this period of time, using my cs.cornell.edu e-mail address, can be viewed as belonging to one of two categories: first, messages consisting of one or more large files, such as drafts of papers mailed between co-authors (essentially, e-mail as file transfer); and second, all other messages. The collection I am considering here consists simply of all messages belonging to the second, much larger category; thus, to a rough approximation, it is all the mail I sent and received during this period, unfiltered by content but excluding long files. It contains 34344 messages in UNIX mailbox format, totaling 41.7 megabytes of ascii text, excluding message headers.[1]

Subsets of the collection can be chosen by selecting all messages that contain a particular string or set of strings; this can be viewed as an analogue of a "folder" of related messages, although messages in the present case are related not because they were manually filed together but because they are the response set to a particular query. Studying the stream induced by such a response set raises two distinct but related questions. First, is it in fact the case that the appearance of messages containing particular words exhibits a "spike," in some informal sense, in the (temporal) vicinity of significant times such as deadlines, scheduled events, or unexpected developments? And second, do the algorithms developed here provide a means for identifying this phenomenon?

In fact such spikes appear to be quite prevalent, and also rich enough that the algorithms of the previous section can extract hierarchical structure that in many cases is quite deep. Moreover, the algorithms are efficient enough that computing a representation for the bursts on a query to the full e-mail collection can be done in real-time, using a simple implementation on a standard PC.

To give a qualitative sense for the kind of structure one obtains, Figures 2 and 3 show the results of computing bursts for two different queries using the automaton $\mathcal{A}_2^*$. Figure 2 shows an analysis of the stream of all messages containing the word "ITR," which is prominent in my e-mail because it is the name of a large National Science Foundation program for which my colleagues and I wrote two proposals in 1999-2000. There are many possible ways to organize this stream of messages, but one general backdrop against which to view the stream is the set of deadlines imposed by the NSF for the first run of the program. Large proposals were submitted in a three-phase process, with deadlines of 11/15/99, 1/5/00, and 4/17/00 for letters of intent, pre-proposals, and full proposals respectively. Small proposals were submitted in a two-phase process, with deadlines of 1/5/00 and 2/14/00 for letters of intent and full proposals respectively. I participated in a group writing a proposal of each kind.

Turning to the figure, part (a) is a plot of the raw input to the automaton $\mathcal{A}_2^*$, showing the arrival time of each message in the response set. Part (b) shows a nested interval representation of the set of bursts for the optimal state sequence in $\mathcal{A}_2^*$; the intervals are annotated with the first and last dates of the messages they contain, and the dates of the NSF deadlines are lined up with the intervals that contain them. Note that this is a schematic representation, designed to show the inclusions that give rise to the tree Γ; the lengths and centering of the intervals in the drawing are not significant. Part (c) shows a drawing of the resulting tree Γ. The root corresponds to the single burst of intensity 0 that is present in any state sequence. One sees that the two children of the root span intervals surrounding the

[1] These figures reveal that I receive less e-mail per day than many of my colleagues; one contributing factor is that I do not subscribe to any high-volume mailing lists based outside Cornell.

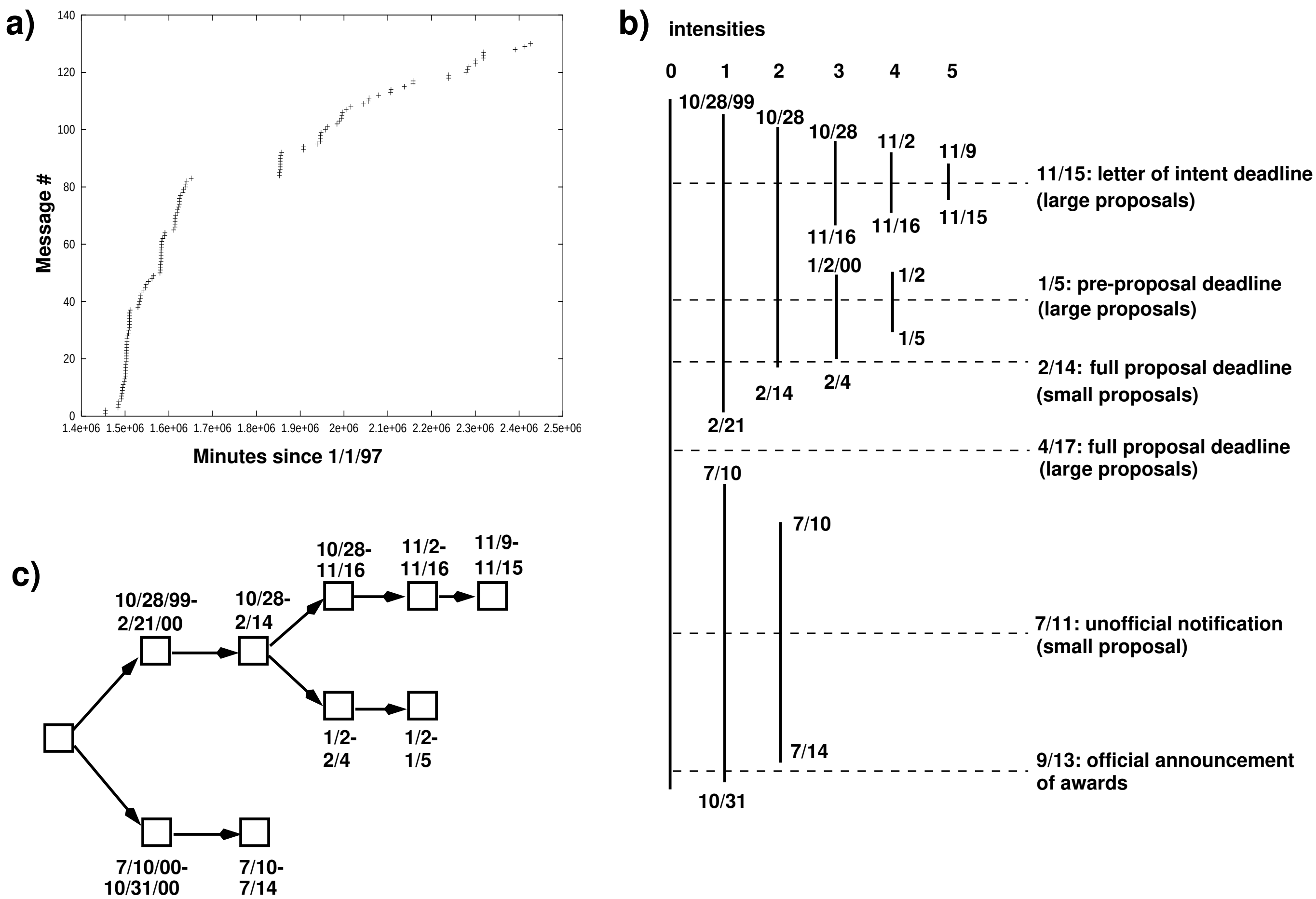

Figure 2: The stream of all messages containing the word "ITR," analyzed using the automaton $\mathcal{A}_2^*$. (a) The raw input data: the x-axis shows message arrival time; the y-axis shows message sequence number. (b) The set of bursts in the optimal state sequence for $\mathcal{A}_2^*$, drawn schematically to show the inclusions that form the tree Γ. (Lengths of intervals are standardized and hence not to scale.) Intervals are annotated with starting and ending dates, and the dates of the NSF ITR program deadlines are lined up with the intervals that contain them. (c) A representation of the tree Γ, showing inclusions among the bursts.

submission deadlines and notification dates, respectively. Moreover, the sub-tree rooted at the first of these children splits further into two sub-trees that are concentrated over a week leading up to the deadline for letters of intent (11/15/99), and four days leading up to the pre-proposal deadline (1/5/00). Finally, note that there is no burst of positive intensity over the final deadline for large proposal, since we did not continue our large submission past the pre-proposal stage.

Figure 3 shows an analysis of the stream of all messages containing the word "prelim," which is the term used at Cornell for (non-final) exams in undergraduate courses. One sees that the raw data in this example (part (a) of the figure) exhibits an arguably more regular structure than in the previous example. I taught undergraduate courses in four of the eight semesters covered by the collection of e-mail, and each of these courses had two prelims.

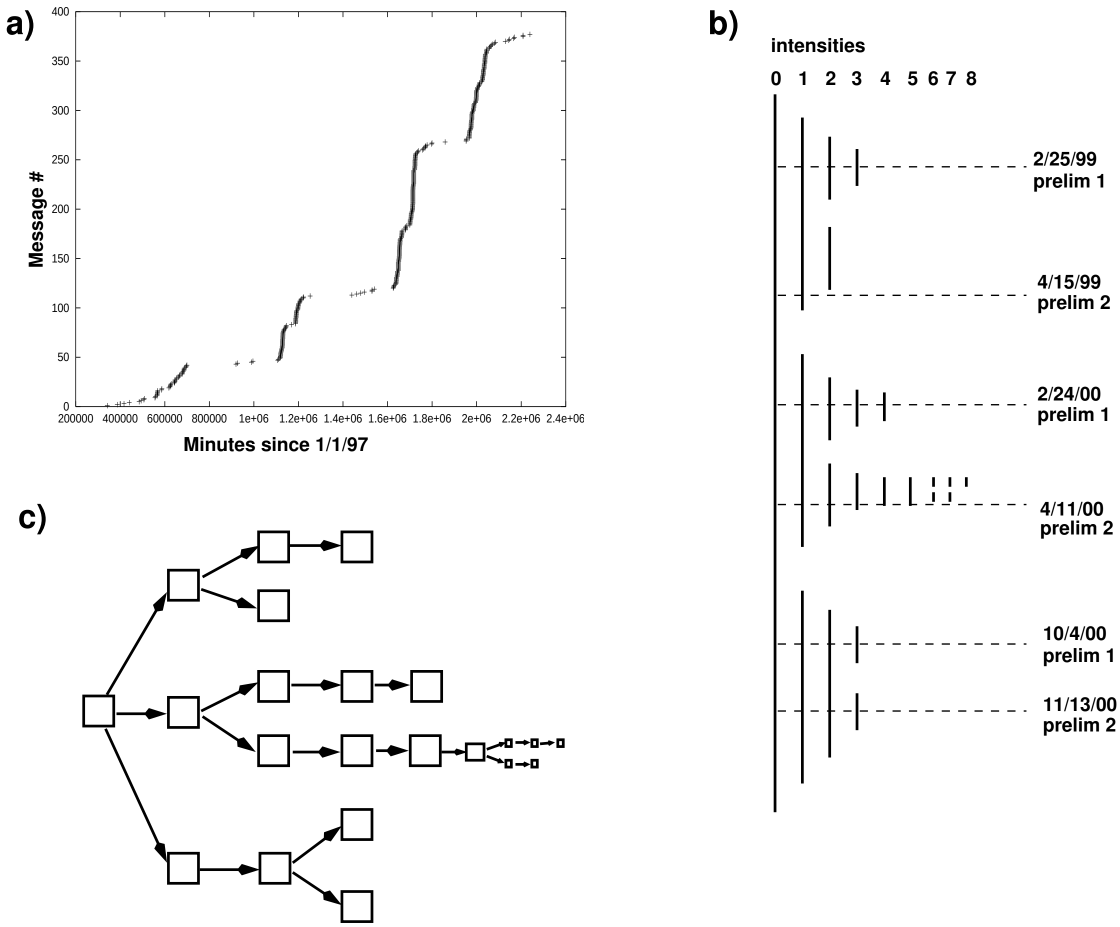

Figure 3: The stream of all messages containing the word "prelim," analyzed using the automaton $\mathcal{A}_2^*$. Parts (a), (b), and (c) are analogous to Figure 2, but date annotations are omitted. In part (b), the dates of prelims (exams) are lined up with the intervals that contain them.

For the first of these courses, correspondence with students was restricted almost exclusively to a special course e-mail account, and hence very little appears in my own saved e-mail. The remaining three courses are captured very cleanly by the tree Γ computed from the optimal state sequence of $\mathcal{A}_2^*$ (parts (b) and (c) of the figure) — each course corresponds to a long burst, and each contains two shorter, more intense bursts for the particular prelims. Specifically, the three children of the root are centered over the semesters in which the three undergraduate courses were taught (Spring 1999, Spring 2000, and Fall 2000); and the sub-trees below these children split further into two sub-trees each, concentrated either directly over or slightly preceding the two prelims given that semester.

Overall, these structures suggest how a large folder of e-mail might naturally be divided into a hierarchical set of sub-folders around certain key events, based only on the rate of message arrivals. The appropriateness of Forster's comments on the time-sense in narratives is also fairly striking here: when organized by burst intensities, the period of time covered in the e-mail collection very clearly "piles up into a few notable pinnacles" [17], rather than proceeding uniformly.

4 Enumerating Bursts

Given a framework for identifying bursts, it becomes possible to perform a type of enumeration: for every word w that appears in the collection, one computes all the bursts in the stream of messages containing w. Combined with a method for computing a *weight* associated with each burst, and for then ranking by weight, this essentially provides a way to find the terms that exhibit the most prominent rising and falling pattern over a limited period of time. This can be applied to e-mail, and it can be done very efficiently even on the scale of the e-mail corpus from the previous section; roughly speaking, it can be performed in a single pass over an inverted index for the collection. Here, however, I consider a different application of this technique: extracting bursts in term usage from the titles of conference papers. Two distinct sources of data will be used here: the titles of all papers from the database conferences SIGMOD and VLDB for the years 1975-2001; and the titles of all papers from the theory conferences STOC and FOCS for the years 1969-2001.

The first issue that must be addressed concerns the underlying model: unlike e-mail messages, which arrive continuously over time, conference papers appear in large batches — essentially, twenty to sixty new papers appear together every half year. As a result, the automaton $\mathcal{A}^*_{s,\gamma}$ is not appropriate, since it is fundamentally based on analyzing the distribution of inter-arrival gaps. Instead, one needs to model a related kind of phenomenon: documents arrive in discrete *batches*; in each new batch of documents, some are *relevant* (in the present case, their titles contain a particular word w) and some are *irrelevant*. The idea is thus to find an automaton model that generates batched arrivals, with particular fractions of relevant documents. A sequence of batched arrivals could be considered bursty if the fraction of relevant documents alternates between reasonably long periods in which the fraction is small and other periods in which it is large.

Suppose there are n batches of documents; the t^{th} batch contains r_t relevant documents out of a total of d_t. Let $R = \sum_{t=1}^{n} r_t$ and $D = \sum_{t=1}^{n} d_t$. Now, define an automaton $\mathcal{B}^*_{s,\gamma}$ as follows, by close analogy with the construction of $\mathcal{A}^*_{s,\gamma}$. For each state q_i of $\mathcal{B}^*_{s,\gamma}$, for $i \geq 0$, there is an expected fraction of relevant documents p_i. Set $p_0 = \overline{p} = R/D$, and $p_i = p_0 s^i$. Since it does not make sense for p_i to exceed 1, the state q_i will only be defined for i such that $p_i \leq 1$; thus, $\mathcal{B}^*_{s,\gamma}$ will be a finite-state automaton. One can further restrict $\mathcal{B}^*_{s,\gamma}$ to k states, resulting in the automaton $\mathcal{B}^k_{s,\gamma}$. Viewed in a generative fashion, one can imagine state q_i in these models as producing a mixture of relevant and irrelevant documents according to a binomial distribution with probability p_i.

The cost of a state sequence $\mathbf{q} = (q_{i_1}, \ldots, q_{i_n})$ in $\mathcal{B}^*_{s,\gamma}$ is defined as follows. If the automa-

Word	Interval of burst
data	1975 SIGMOD — 1979 SIGMOD
base	1975 SIGMOD — 1981 VLDB
application	1975 SIGMOD — 1982 SIGMOD
bases	1975 SIGMOD — 1982 VLDB
design	1975 SIGMOD — 1985 VLDB
relational	1975 SIGMOD — 1989 VLDB
model	1975 SIGMOD — 1992 VLDB
large	1975 VLDB — 1977 VLDB
schema	1975 VLDB — 1980 VLDB
theory	1977 VLDB — 1984 SIGMOD
distributed	1977 VLDB — 1985 SIGMOD
data	1980 VLDB — 1981 VLDB
statistical	1981 VLDB — 1984 VLDB
database	1982 SIGMOD — 1987 VLDB
nested	1984 VLDB — 1991 VLDB
deductive	1985 VLDB — 1994 VLDB
transaction	1987 SIGMOD — 1992 SIGMOD
objects	1987 VLDB — 1992 SIGMOD
object-oriented	1987 SIGMOD — 1994 VLDB
parallel	1989 VLDB — 1996 VLDB
object	1990 SIGMOD — 1996 VLDB
mining	1995 VLDB —
server	1996 SIGMOD — 2000 VLDB
sql	1996 VLDB — 2000 VLDB
warehouse	1996 VLDB —
similarity	1997 SIGMOD —
approximate	1997 VLDB —
web	1998 SIGMOD —
indexing	1999 SIGMOD —
xml	1999 VLDB —

Figure 4: The 30 bursts of highest weight in $\mathcal{B}_2^2$, using titles of all papers from the database conferences SIGMOD and VLDB, 1975-2001.

ton is in state i when the t^{th} batch arrives, a cost of

$$\sigma(i,t) = -\ln\left[\binom{d_t}{r_t} p_i^{r_t}(1-p_i)^{d_t-r_t}\right]$$

is incurred, since this is the negative logarithm of the probability that r_t relevant documents would be generated using a binomial distribution with probability p_i. There is also a cost of $\tau(i_t, i_{t+1})$ associated with the state transition from q_{i_t} to $q_{i_{t+1}}$, where this cost is defined precisely as for $\mathcal{A}^*_{s,\gamma}$. A state sequence of minimum total cost can then be computed as in Section 2.

In the analysis of conference paper titles here, the main goal is to enumerate bursts of

Word	Interval of burst
grammars	1969 STOC — 1973 FOCS
automata	1969 STOC — 1974 STOC
languages	1969 STOC — 1977 STOC
machines	1969 STOC — 1978 STOC
recursive	1969 STOC — 1979 FOCS
classes	1969 STOC — 1981 FOCS
some	1969 STOC — 1980 FOCS
sequential	1969 FOCS — 1972 FOCS
equivalence	1969 FOCS — 1981 FOCS
programs	1969 FOCS — 1986 FOCS
program	1970 FOCS — 1978 STOC
on	1973 FOCS — 1976 STOC
complexity	1974 STOC — 1975 FOCS
problems	1975 FOCS — 1976 FOCS
relational	1975 FOCS — 1982 FOCS
logic	1976 FOCS — 1984 STOC
vlsi	1980 FOCS — 1986 STOC
probabilistic	1981 FOCS — 1986 FOCS
how	1982 STOC — 1988 STOC
parallel	1984 STOC — 1987 FOCS
algorithm	1984 FOCS — 1987 FOCS
graphs	1987 STOC — 1989 STOC
learning	1987 FOCS — 1997 FOCS
competitive	1990 FOCS — 1994 FOCS
randomized	1992 STOC — 1995 STOC
approximation	1993 STOC —
improved	1994 STOC — 2000 STOC
codes	1994 FOCS —
approximating	1995 FOCS —
quantum	1996 FOCS —

Figure 5: The 30 bursts of highest weight in $\mathcal{B}_2^2$, using titles of all papers from the theory conferences STOC and FOCS, 1969-2001.

positive intensity, but not to emphasize hierarchical structure. Thus, the *two-state* automaton $\mathcal{B}_2^2$ is used; given an optimal state sequence, bursts of positive intensity correspond to intervals in which the state is q_1 rather than q_0. For such a burst $[t_1, t_2]$, we can define the *weight* of the burst to be

$$\sum_{t=t_1}^{t_2} (\sigma(1,t) - \sigma(0,t)).$$

In other words, the weight is equal to the improvement in cost incurred by using state 1 over the interval rather than state 0. Observe that in an optimal sequence, the weight of every burst is non-negative. Intuitively, then, bursts of larger weight correspond to more prominent periods of elevated activity. (This notion of weight can be naturally extended to larger numbers of states, as well as to the automaton model from Section 2.)

In Figure 4, this framework is applied to the titles of SIGMOD and VLDB papers for the years 1975-2001. For each word w (including stop-words), an input to $\mathcal{B}_2^2$ is constructed in which r_t is the number of titles at the t^{th} conference (chronologically) that contain the word w, and d_t is the total number of titles at the t^{th} conference. The 30 bursts with the highest weight, over all possible words w, are then depicted in the figure, sorted by year of appearance. The bursts with no given ending date ('mining', 'warehouse', 'similarity', 'approximate', 'web', 'indexing', and 'xml') are those for which the interval extends to the most recent conference, suggesting terms that are in the middle of a large-weight burst at present. Note that no pre-processing is done on the titles, other than to convert each word to lower-case. One observes that the words in Figure 4 are almost all quickly recognizable as carrying technical content, even though they are the top results in an enumeration where bursts were computed and ranked for *all* words, including stop-words.[2]

Figure 5 shows the results of the same computation on the titles of STOC and FOCS papers for the years 1969-2001. For both these collections, it is important to note that the number of occurrences of a word w is in general a quantity that, at a local scale, changes very rapidly from one conference to the next; thus, many of the intervals depicted in the figures span conferences in which the indicated word did not appear at all, and omit ones with large numbers of occurrences. The non-trivial cost of state transitions in $\mathcal{B}_2^2$ is crucial in making it possible for intervals of any reasonable length to form in the presence of this data.

5 Related Work

The Topic Detection and Tracking (TDT) study [2, 3, 61, 62] articulated the problem of extracting significant topics and events from a stream of news articles, thereby framing the type of document stream analysis questions considered here. Much of the emphasis in the TDT study was on techniques for the *on-line* version of the problem, in which events must be detected in real-time; but there was also a *retrospective* version in which the whole stream could be analyzed. Similar issues have recently been addressed in the visualization community [26, 43, 60], where the problem of visualizing the appearance and disappearance of themes in a sequence of news stories has been explored.

Following on the TDT work, Swan, Allan, and Jensen [56, 57, 58] developed a method for constructing *overview timelines* of a set of news stories. For each named entity and noun phrase in the corpus, they perform a χ^2 test to identify days on which the number of occurrences yields a value above a certain threshold; contiguous sets of days meeting this condition are then grouped into an interval that is added to the timeline. Thus, the high-level structure of their approach is parallel to the enumerative method in Section 4. However, the

[2]The bursts for 'data,' 'base,' and 'bases' in the years 1975-1981 arise in large part from the fact that the term "database" was written as two words in a significant number of the paper titles during this period.

underlying methodology is quite different from the present work in two key respects. First, Swan et al. note that the use of thresholds makes it difficult to construct long intervals of activity for a single feature — such intervals are often broken apart by brief gaps in which the feature does not occur frequently enough, and subsequent heuristics are needed to piece them together. The present work, by modeling a burst as a state transition with costs, allows for a long interval to naturally persist across such gaps; essentially, in place of thresholds, the optimization problem inherent in finding a minimum-cost state sequence adaptively groups nearby high-intensity intervals together when it is advantageous to do so. Second, the work of Swan et al. does not attempt to infer any type of hierarchical structure in the appearance of a feature.

Lewis and Knowles analyze the dynamics of message-sending over a very short time scale, searching for features that can determine whether one message is a response to another [37]. This is applied to develop robust techniques for identifying *threads*, a popular metaphor for organizing e-mail and newsgroup postings [15, 22]. In a very different context, Grosz and Sidner develop structural models for discourse as a means of analyzing communication [21]; their use of stack models in particular results in a nested organization that bears an intriguing, though distant, relationship to the nested structure of bursts studied here.

The present work clearly overlaps with the large areas of time series analysis and sequence mining [10, 25]; connections to related probabilistic frameworks such as bursty on-off sources [32] and hidden Markov models [48] have already been discussed above. Ehrich and Foith [16] proposed a method for constructing a tree from a one-dimensional time series, essentially by introducing a branch-point at each local minimum and a leaf at each local maximum (see also [55]). In the context of the applications here, such an approach would yield trees of enormous complexity, due to the ruggedness of the underlying temporal data, with many local minima and maxima.

The search for a minimum-cost state sequence in the automata of Section 2 and 4 can also be viewed as a search for approximate level sets in a time series, and hence related to the large body of work on piece-wise function approximation in both statistics and data mining (see e.g. [23, 24, 27, 31, 33, 35, 41]). In a discrete framework, work on mining episodes and sequential patterns (e.g. [1, 12, 25, 40]) has developed algorithms to identify particular configurations of discrete events clustered in time, in some cases obeying partial precedence constraints on their order. Finally, there is an interesting general relationship to work on *traffic analysis* in the areas of cryptography and security [52]; in that context, temporal analysis of a message stream is crucial because the content of the messages has been explicitly obscured.

6 Extensions and Conclusions

In the settings discussed above, the analysis has made use of both the temporal information and the underlying content. The role of temporal data is clear; but the content of course plays an integral role as well: Section 3 deals with streams consisting of the response set for a particular query to a larger stream; and Section 4 considers streams with batched arrivals, in which a particular subset of each batch is designated as relevant. And in fact, there is strong evidence that the interplay between content and time is crucial here — that an arbitrary set of messages with same sequence of arrival times would not exhibit an equally strong set of bursts. Adapting a *permutation test* from Swan and Jensen [58], one can start with a complete e-mail corpus having arrival times $t_1, t_2, \ldots, t_N$, choose a random permutation π, and shuffle the corpus so that message $\pi(i)$ arrives at time t_i (instead of message i), for $i = 1, 2, \ldots, N$. The resulting shuffled corpus has the same set of arrival times and the same messages, but the original correspondence between the two is broken; do equivalently strong "spurious" bursts appear in this new sequence? In fact, they clearly do not: when the weight of bursts for all words (with respect to $\mathcal{A}_2^*$) is computed using the e-mail corpus in Section 3, the total weight associated with the true corpus is more than an order of magnitude greater than the average total weight over 100 randomly shuffled versions (369,980 versus 25,141). Moreover, the shuffled versions exhibit almost no non-trivial hierarchical structure; the average total number of words generating bursts of intensity at least 2 (i.e. inducing trees Γ with two or more levels below the root) is 16.7 over the randomly shuffled versions, compared with 3865 in the true corpus.

I have also applied the overall framework developed here to Web clickstream data collected by Gay et al. [18]. The dataset in [18] was compiled as part of a study of student usage of wireless laptops: The browser clicks of roughly 80 undergraduate students in two particular classes at Cornell were collected (with consent) from wireless laptops over a period of two and a half months in Spring 2000. Bursts with respect to $\mathcal{A}_{s,\gamma}^*$ can be computed by an enumerative method, as in Section 4: for every URL w, all bursts in the stream of visits to w are determined; the full set of bursts is then ordered by weight. Each burst, associated with a URL w, now has an additional quantity associated with it: the number of distinct users who visited w during the interval of the burst. This allows one to distinguish between collective activity involving much of the class and that of just a single user. As it turns out, if one focuses on bursts that involve at least 10 distinct users, then many of those with the highest weight involve the URLs of the on-line class reading assignments, centered on intervals shortly before and during the weekly sessions at which they were discussed.

A final observation is that the use of a model based on state transitions leads to bursts with sharp boundaries; they have clear beginnings and ends. In particular, this means that for every burst, one can identify a single message on which the associated state transition occurred. This is akin to the TDT study's notion of (retrospective) *first story detection* [2], although in the automaton model of the present work, identifying initial messages does not constitute a separate problem since it follows directly from the definition of the state transitions. In the context of e-mail, the contents of such an initial message can often serve as a concentrated summary of the circumstances precipitating the burst — in other words, there is frequently something in the message itself to frame the flurry of message-sending that is about to occur. And for messages on which bursts for several different terms are initiated simultaneously, this phenomenon is even more apparent; such messages often represent natural "landmarks" at the beginning of a long-running episode.

In many domains, we accumulate extensive and detailed records of our own behavior — in the e-mail we send and receive, the Web pages we visit, the queries we issue to search engines. An underlying theme, of which several aspects have been developed here, is that a great number of these settings have a fundamental temporal aspect; they are punctuated by the sharp and sudden onset of particular episodes, and can be organized around rising and falling patterns of activity. There is a great amount of complexity underpinning such a picture. But by developing a better understanding of it, one can hope ultimately to find structure in the raw data that we generate through the basic process of interacting and communicating.

Acknowledgements. I thank Lillian Lee for valuable discussions and suggestions throughout the course of this work.

References

[1] R. Agrawal, R. Srikant, "Mining sequential patterns," *Proc. Intl. Conf. on Data Engineering*, 1995.

[2] J. Allan, J.G. Carbonell, G. Doddington, J. Yamron, Y. Yang, "Topic Detection and Tracking Pilot Study: Final Report," *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, Feb. 1998.

[3] J. Allan, R. Papka, V. Lavrenko, "On-line new event detection and tracking," *Proc. SIGIR Intl. Conf. Information Retrieval*, 1998.

[4] K. Becker, M. Cardoso, "Mail-by-Example: A visual query interface for managing large volumes of electronic messages,' *Proc. 15th Brazilian Symposium on Databases*, 2000.

[5] D. Beeferman, A. Berger, J. Lafferty, "Statistical Models for Text Segmentation," *Machine Learning* 34(1999), pp. 177-210.

[6] H. Berghel, "E-mail: The good, the bad, and the ugly," *Communications of the ACM*, 40:4(April 1997), pp. 11-15.

[7] A. Birrell, S. Perl, M. Schroeder, T. Wobber, *The Pachyderm E-mail System*, 1997, at http://www.research.compaq.com/SRC/pachyderm/.

[8] T. Blanton, Ed., *White House E-mail*, New Press, 1995.

[9] G. Boone, "Concept features in Re:Agent, an intelligent e-mail agent," *Proc. 2nd Intl. Conf. Autonomous Agents*, 1998.

[10] C. Chatfield, *The Analysis of Time Series: An Introduction*, Chapman and Hall, 1996.

[11] S. Chatman, *Story and Discourse: Narrative Structure in Fiction and Film*, Cornell Univ. Press, 1978.

[12] D. Chudova, P. Smyth, "Unsupervised identification of sequential patterns under a Markov assumption," *KDD Workshop on Temporal Data Mining*, 2001.

[13] W. Cohen. "Learning rules that classify e-mail." *Proc. AAAI Spring Symp. Machine Learning and Information Access*, 1996.

[14] T. Cover, P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Information Theory* IT-13(1967), pp. 21-27.

[15] W. Davison, L. Wall, S. Barber, *trn*, 1993
http://web.mit.edu/afs/sipb/project/trn/src/trn-3.6/.

[16] R. Ehrich, J. Foith, "Representation of Random Waveforms by Relational Trees," *IEEE Trans. Computers*, C25:7(1976).

[17] E.M. Forster, *Aspects of the Novel*, Harcourt, Brace, and World, Inc. 1927.

[18] G. Gay, M. Stefanone, M. Grace-Martin, H. Hembrooke, "The effect of wireless computing in collaborative learning environments," *Intl. J. Human-Computer Interaction*, to appear.

[19] G. Genette, *Narrative Discourse: An Essay in Method*, English translation (J.E. Lewin), Cornell Univ. Press, 1980.

[20] G. Genette, *Narrative Discourse Revisited*, English translation (J.E. Lewin), Cornell Univ. Press, 1988.

[21] B. Grosz, C. Sidner, "Attention, intentions, and the structure of discourse," *Computational Linguistics* 12(1986).

[22] T. Gruber, *Hypermail*, Enterprise Integration Technologies.

[23] V. Guralnik, J. Srivastava, "Event detection from time series data," *Intl. Conf. Knowledge Discovery and Data Mining*, 1999.

[24] J. Han, W. Gong, Y. Yin, "Mining Segment-Wise Periodic Patterns in Time-Related Databases", *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, 1998.

[25] D. Hand, H. Mannila, P. Smyth, *Principles of Data Mining*, MIT Press, 2001.

[26] S. Havre, B. Hetzler, L. Nowell, "ThemeRiver: Visualizing Theme Changes over Time," *Proc. IEEE Symposium on Information Visualization*, 2000.

[27] D. Hawkins, "Point estimation of the parameters of piecewise regression models," *Applied Statistics* 25(1976)

[28] B. Heckel, B. Hamann, "EmVis – A Visual e-Mail Analysis Tool," *Proc. Workshop on New Paradigms in Information Visualization and Manipulation*, in conjunction with Conf. on Information and Knowledge Management, 1997.

[29] J. Helfman, C. Isbell, "Ishmail: Immediate identification of important information," AT&T Labs Technical Report, 1995.

[30] E. Horvitz, "Principles of Mixed-Initiative User Interfaces," *Proc. ACM Conf. Human Factors in Computing Systems*, 1999.

[31] D. Hudson, "Fitting segmented curves whose join points have to be estimated," *Journal of the American Statistical Association* 61(1966) pp. 1097–1129.

[32] F.P. Kelly, "Notes on effective bandwidths," in *Stochastic Networks: Theory and Applications*, (F.P. Kelly, S. Zachary, I. Ziedins, eds.) Oxford Univ. Press, 1996.

[33] E. Keogh, P. Smyth, "A probabilistic approach to fast pattern matching in time series databases," *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, 1997.

[34] J.I. Klein et al., Plaintiffs' Memorandum in Support of Proposed Final Judgment, United States of America v. Microsoft Corporation and State of New York, ex rel. Attorney General Eliot Spitzer, et al., v. Microsoft Corporation, Civil Actions No. 98-1232 (TPJ) and 98-1233 (TPJ), April 2000.

[35] M. Last, Y. Klein, A. Kandel, "Knowledge Discovery in Time Series Databases," *IEEE Transactions on Systems, Man, and Cybernetics* 31B(2001).

[36] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, J. Allan, "Mining of Concurrent Text and Time-Series," *KDD-2000 Workshop on Text Mining*, 2000.

[37] D.D. Lewis, K.A. Knowles, "Threading electronic mail: A preliminary study," *Inf. Proc. Management* 33(1997).

[38] S.S. Lukesh, "E-mail and potential loss to future archives and scholarship, or, The dog that didn't bark," *First Monday* 4(9) (September 1999), at http://firstmonday.org

[39] P. Maes, "Agents that reduce work and information overload," *Communications of the ACM* 37:7(1994), pp. 30-40.

[40] H. Mannila, H. Toivonen, A.I. Verkamo, "Discovering frequent episodes in sequences," *Proc. Intl. Conf. on Knowledge Discovery and Data Mining*, 1995.

[41] R. Martin, V. Yohai, "Data mining for unusual movements in temporal data," *KDD Workshop on Temporal Data Mining*, 2001.

[42] M.L. Markus, "Finding a Happy Medium: Explaining the Negative Effects of Electronic Communication on Social Life at Work," *ACM Trans. Info. Sys.* 12(1994), pp. 119-149.

[43] N. Miller, P. Wong, M. Brewster, H. Foote, "Topic Islands: A Wavelet-Based Text Visualization System," *Proc. IEEE Visualization*, 1998.

[44] R. Moore, C. Baru, A. Rajasekar, B. Ludaescher, R. Marciano, M. Wan, W. Schroeder, A. Gupta, "Collection-Based Persistent Digital Archives – Part 2," *D-Lib Magazine*, 6(2000).

[45] F. Olsen, "Facing Flood of E-Mail, Archives Seeks Help From Supercomputer Researchers," *Chronicle of Higher Education*, August 24, 1999.

[46] T. Payne, P. Edwards, "Interface agents that learn: An investigation of learning issues in a mail agent interface," *Applied Artificial Intelligence* 11(1997), pp. 1–32.

[47] S. Pollock, "A rule-based message filtering system," *ACM Trans. Office Automation Systems* 6(3):232–254, 1988.

[48] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE* 77(1989).

[49] M. Redmond, B. Adelson, "AlterEgo e-mail filtering agent," *Proc. AAAI Workshop on Case-Based Reasoning*, 1998.

[50] J. Rennie, "ifile: An application of machine learning to e-mail filtering," *Proc. KDD Workshop on Text Mining*, 2000.

[51] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz. "A Bayesian approach to filtering junk email," *Proc. AAAI Workshop on Learning for Text Categorization*, 1998.

[52] B. Schneier, *Applied Cryptography* Wiley, 1996.

[53] R. Segal, J. Kephart. "MailCat: An intelligent assistant for organizing e-mail," *Proc. Intl. Conf. Autonomous Agents*, 1999.

[54] R. Segal, J. Kephart. "Incremental Learning in SwiftFile," *Proc. Intl. Conf. on Machine Learning*, 2000.

[55] S. Shaw, R. DeFigueiredo, "Structural Processing of Waveforms as Trees," *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38:2(1990)

[56] R. Swan, J. Allan, "Extracting significant time-varying features from text," *Proc. 8th Intl. Conf. on Information Knowledge Management*, 1999.

[57] R. Swan, J. Allan, "Automatic generation of overview timelines," *Proc. SIGIR Intl. Conf. Information Retrieval*, 2000.

[58] R. Swan, D. Jensen, "TimeMines: Constructing Timelines with Statistical Models of Word Usage," *KDD-2000 Workshop on Text Mining*, 2000.

[59] S. Whittaker, C. Sidner, "E-mail overload: Exploring personal information management of e-mail," *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems*, 1996.

[60] P. Wong, W. Cowley, H. Foote, E. Jurrus, J. Thomas, "Visualizing sequential patterns for text mining," *Proc. IEEE Information Visualization*, 2000

[61] Y. Yang, T. Ault, T. Pierce, C.W. Lattimer, "Improving text categorization methods for event tracking," *Proc. SIGIR Intl. Conf. Information Retrieval*, 2000.

[62] Y. Yang, T. Pierce, J.G. Carbonell, "A Study on Retrospective and On-line Event Detection," *Proc. SIGIR Intl. Conf. Information Retrieval*, 1998.