

# Assigning Trust to Wikipedia Content\*

B. Thomas Adler<sup>1</sup> Krishnendu Chatterjee<sup>2</sup> Luca de Alfaro<sup>2</sup>

Marco Faella<sup>3</sup> Ian Pye<sup>1</sup> Vishwanath Raman<sup>1</sup>

<sup>1</sup>Computer Science Dept.  
UC Santa Cruz, CA, USA  
{thumper, ipye,  
vraman}@ucsc.edu

<sup>2</sup>Computer Engineering Dept.  
UC Santa Cruz, CA, USA  
{c\_krish,luca}@soe.ucsc.edu

<sup>3</sup>Computer Science Division  
Dept. of Physics  
University of Naples, Italy  
mfaella@na.infn.it

## ABSTRACT

The Wikipedia is a collaborative encyclopedia: anyone can contribute to its articles simply by clicking on an “edit” button. The open nature of the Wikipedia has been key to its success, but has also created a challenge: how can readers develop an informed opinion on its reliability? We propose a system that computes quantitative values of trust for the text in Wikipedia articles; these trust values provide an indication of text reliability.

The system uses as input the revision history of each article, as well as information about the *reputation* of the contributing authors, as provided by a reputation system. The trust of a word in an article is computed on the basis of the reputation of the original author of the word, as well as the reputation of all authors who edited text near the word. The algorithm computes word trust values that vary smoothly across the text; the trust values can be visualized using varying text-background colors. The algorithm ensures that all changes to an article’s text are reflected in the trust values, preventing surreptitious content changes.

We have implemented the proposed system, and we have used it to compute and display the trust of the text of thousands of articles of the English Wikipedia. To validate our trust-computation algorithms, we show that text labeled as low-trust has a significantly higher probability of being edited in the future than text labeled as high-trust.

## Categories and Subject Descriptors

H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces—*Computer-supported cooperative work, Web-based interaction*; K.4.3 [Computers and Society]: Organizational Impacts—*Computer-supported collaborative work*

---

\*A preliminary version of this work is available as the Technical Report UCSC-CRL-07-09, School of Engineering, University of California, Santa Cruz, CA, USA, November 2007. This research has been partially supported by the CITRIS: Center for Information Technology Research in the Interest of Society.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WikiSym '08, September 8–10, Porto, Portugal.

Copyright 2008 ACM 978-1-60558-128-3/08/09 ...\$5.00.

## 1. INTRODUCTION

Wikipedia is an online encyclopedia who grew in the span of a few years to become one of the most widely used sources of information on the web. Wikipedia owes its growth and breadth of coverage to its ability to harness the contributions of millions of individuals, ranging from casual visitors, to domain experts, to dedicated editors. On the other hand, the open process that gives rise to Wikipedia content makes it difficult for visitors to form an idea of the reliability of the content. Wikipedia articles are constantly changing, and the contributors range from domain experts, to vandals, to dedicated editors, to superficial contributors not fully aware of the quality standards the Wikipedia aspires to attain. Wikipedia visitors are presented with the latest version of each article they visit: this latest version does not offer them any simple insight into how the article content has evolved into its most current form, nor does it offer a measure of how much the content can be relied upon. These considerations generated interest in algorithmic systems for estimating the trust of Wikipedia content [21, 34].

We introduce a *trust system* for Wikipedia that computes, and displays, a value of *trust* for each word of each article version of Wikipedia. The trust value of a word is computed according to the degree in which the word, and the immediately surrounding text, has been revised by previous authors and editors. The computation takes into account both the amount of revision, and the *reputation* of the people performing the revision, as computed by a separate reputation system [1]. The resulting trust values are displayed via different colors for the text background, providing an intuitive guide to the reliability of the content. The trust values allow visitors to easily spot new, unchecked content, as well as any content modification, including malicious attempts at corrupting information. An example of coloring produced by our system is given in Figure 1; the tampering with the prime minister’s last name is clearly indicated by the trust values.

Our emphasis on word-level trust reflects our goal of providing guidance to Wikipedia visitors over which portions of an article can be relied upon, and which others instead require closer scrutiny. This contrasts with approaches that assign a single, global value of trust or quality for an entire article [18, 7, 35, 23]. Such a global trust value is useful in many applications, including applications where *selecting* articles is important. We instead assume that a reader is interested in a given article, and we tackle the goal of providing an estimate of how much the different assertions in the article can be trusted.

A novel feature of our trust system is that it is resistant to tampering. Text that is deleted by vandalism, and then re-inserted, conserves its original trust, so that malicious users cannot lower the trust of text simply by deleting and re-inserting it. More importantly, users cannot tamper with the system and cause text of their

departments. Cabinet members are occasionally recruited from outside the Folketing.

Since 27 November 2001, the economist Anders Fogh Rasmussen has been Prime Minister to Denmark.

As known in other parliamentary systems of government, the executive,

(a) Immediately before the modification (revision id 77625823).

departments. Cabinet members are occasionally recruited from outside the Folketing.

Since 27 November 2001, the economist Anders Fjogh Rasmussen has been Prime Minister to Denmark.

As known in other parliamentary systems of government, the executive,

(b) Immediately after the modification (revision id 77692452).

**Figure 1: Trust coloring resulting from an attempt to modify the spelling of the Danish Prime Minister’s last name, from Fogh, to Fjogh (in Danish, a fjog is a fool). The text background is a shade of orange that is the darker, the lower the trust of the text. The sequence consists of two consecutive revisions. Notice how the trust coloring highlights the information that has not yet been sufficiently reviewed. Subtle changes such as the above can be hard for Wikipedia visitors to spot without the help of a trust coloring.**

choice to gain extra trust. Another novel feature of the proposed system is that it relies entirely on automatic content analysis. Both the reputation system [1], and the trust system, rely on an analysis of the evolution of the content of wiki articles, and require no user input. Consequently, our systems are applicable to any wiki. This is in contrast to previous approaches, which relied on a classification of users according to their Wikipedia role [21, 34]: such previous approaches were applicable only to wikis that developed a stratified classification of contributors.

## 1.1 The Trust Assignment Algorithm

The goal of our trust system is to convey information on the degree with which the text has been revised, and to flag any recent unchecked content modifications. We rely on a simple idea: the trust of text should depend on the reliability of the author, and on the reliability of the people who subsequently revised, checked, and edited the text [21, 34].

As a measure of author and revisor quality, we take the *author reputation* computed by the author reputation system of [1]. That reputation system, like the trust system described in this paper, is *content-driven*: it relies on content analysis, rather than user-to-user feedback. Users who contribute long-lived content gain reputation, while users who contribute content that is quickly removed lose reputation. The resulting author reputation was shown to correlate well with the quality of the author’s future contributions, justifying its use in the computation of text trust. Furthermore, gaining repu-

tion requires effort, and this will enable us to make the system resistant to tampering, as we will discuss in Section 2.4.

We compute the trust of the revisions  $v_1, v_2, v_3, \dots$  of a wiki article by analyzing how each revision is obtained from the previous one. When an author  $A$  edits revision  $v_k$ , obtaining revision  $v_{k+1}$ , we compare the text of  $v_k$  and  $v_{k+1}$ , tracking the blocks of text that have been inserted, deleted, and copied. Text that is new in  $v_{k+1}$  is given a trust proportional to  $A$ ’s reputation: the rationale is that authors of high reputation are likely to provide good quality, accurate contributions. We choose the proportionality constant so that even text by top-reputation authors does not initially have full trust: full trust requires the consensus of multiple authors. The text that used to be present in  $v_k$ , but has been deleted, is tracked as “dead text”, so that if it is reinserted in a later revision  $v_{k+m}$ , for  $m > 1$ , it can be assigned the trust it had in  $v_k$ . Tracking deleted text enables us to attribute text to its original author, and it ensures that vandalism has no lasting effect, and is not gratifying to the vandals.

If author  $A$  rearranged the order of blocks of text in producing revision  $v_{k+1}$  from  $v_k$ , we assign the end-points of the rearranged blocks the same trust value we assign to new text. Indeed, the meaning of the text may have changed due to the cut-and-paste, and it is no more reliable than other text inserted by author  $A$ . The trust of the block interior is instead inherited from the trust of the corresponding text in  $v_k$ . A consequence of this rule is that when text is deleted, the margins of the “wound” where the text has been cut are highlighted with low trust, as they correspond to end-points of rearranged text blocks. Thus, our system makes it hard to surreptitiously tamper with the content of Wikipedia articles: every change, including text rearrangements and deletions, leaves a low-trust mark that is prominently displayed via the trust coloring.

Once all the text of revision  $v_{k+1}$  is assigned a preliminary value of trust as described above, we perform one additional step, in which we may raise the trust of the text to account for the fact that it has been reviewed by author  $A$ . The idea is that author  $A$ , by leaving text unchanged from  $v_k$  to  $v_{k+1}$ , has given an implicit vote of confidence in the text. Thus, we raise the trust of the text in proportion to  $A$ ’s reputation, and in proportion to the attention that  $A$  is likely to have paid to each portion of text. To ensure that a single author cannot cause the trust of an article to raise more than due by performing repeated small edits, we keep track, for each word, of the list of the last  $n$  authors who raised the word’s trust. An author can cause a word to raise in trust only if she does not appear in this list. This ensures that text can only raise in trust if revised by multiple authors, preventing authors from single-handedly raising the trust of portions of text of their choice. We show (in Section 2.4) that this also ensures robustness against Sybil (or “sock-puppet”) attacks, in which attackers use multiple identities [6, 17, 3, 26].

We would like to point out some techniques and factors that we have chosen *not* to consider in computing text trust. We chose not to perform semantic analysis of the sentences affected by the edits. Undoubtedly, such an analysis would yield additional information. On the other hand, our methods have the advantage of simplicity, and they are suited to most languages with no adaptation required (as long as the text can be split into individual words); thus, we believe it is of interest to characterize how well trust can be associated with text without requiring semantic analysis. We also chose to consider all words equally, disregarding for instance the distinction between common words, and rarer ones. The meaning of a sentence can be drastically affected by changing common words, such as an “and” into a “not”, and we did not wish to build into the algorithm preconceived ideas of what changes were important.

## 1.2 Trust Quality Metrics

The trust values are computed from the past history of text, and reflect the degree with which text has been edited and revised. Ideally, we would like to show that high trust text conveys with high probability correct information. However, correctness is very difficult to define and measure. As a substitute, we study the correlation between trust, and future text stability, in the hypothesis that correct (or high-quality) content is less likely to be revised [34]. The quality metrics will also provide quantitative performance indices that will be useful in fine-tuning the behavior of the algorithms. We note that the quality metrics capture only in part the intent underlying our trust system: in particular, the goals of predicting future text stability, and warning readers about recent modifications, do not always coincide, as we will see in more detail later. Nevertheless, the metrics offer valuable insight in the performance of the system.

The first two metrics consider the precision and recall of low-trust with respect to immediate deletions. Let the possible range of trust values be the interval  $[0, T_{\max}]$ .

- *Recall of deletions.* For  $\rho \in [0, T_{\max}]$ , the  $\rho$ -recall of deletions is the percentage of deleted text that had trust lower than  $\rho$  in the revision preceding its deletion.
- *Precision of deletions.* For  $\rho \in [0, T_{\max}]$ , the  $\rho$ -precision of deletions is the percentage of text with trust lower than  $\rho$  that is deleted in the immediately subsequent revision.
- *Trust of average vs. deleted text.* We consider the average and median trust of all the text, compared with the average and median trust that deleted text possesses immediately prior to deletion.
- *Trust as a predictor of lifespan.* We select words uniformly at random, and we consider the statistical correlation between the trust of the word at the moment of sampling, and the future lifespan of the word. For  $\rho \in [0, T_{\max}]$ , the  $\rho$ -trust average lifespan of text is the average number of future revisions in which a word of trust  $\rho$  at sampling appears. We remark that this is a proper test, since the trust at the time of sampling depends only on the history of the word prior to sampling.

## 1.3 Implementation and Evaluation

We have implemented the trust system using, as a source of author reputation, the *content-driven* reputation system of [1]. The code of the reputation and trust systems has been made available in open-source format [29]; the code can be readily applied to wikis other than the Wikipedia.

The trust system has been used to process all the text of the English Wikipedia, as of February 2007. The resulting trust assignments can be viewed in a live demo, in which text is displayed with a background color that depends on its trust: white background for fully trusted text, and shades of orange that are the darker, the lower the text trust [29]. The demo provides information on both text trust and text provenance: when visitors click on a word in an article, they are redirected to the version of the article where the word was first introduced. The trust and provenance information complement each other: visitors are made aware of the less trusted portions of text by the coloring, and can then investigate the origin of such text via the text origin redirection.

The evaluation results can be summarized as follows (see Section 5 for the details):

- *Recall of deletions.* We show that text in the lowest 50% of trust values constitutes only 3.4% of the text of articles, yet corresponds to 66% of the text that is deleted from one version to the next.

- *Precision of deletions.* We show that text that is in the bottom half of trust values has a probability of 33% of being deleted in the very next version, in contrast with the 1.9% probability for general text. The deletion probability raises to 62% for text in the bottom 20% of trust values.
- *Trust of average vs. deleted text.* We show that 90% of the text overall had trust at least 76%, while the average trust for deleted text was 33%.
- *Trust as a predictor of lifespan.* We show that words with the highest trust have an expected future lifespan that is 4.5 times longer than words with no trust.

The above results were obtained by analyzing 1,000 articles selected randomly from the Wikipedia articles with at least 200 revisions.

The current implementation of the trust system relies on *batch* processing: the code examines all the content, and computes the trust value of each word of each article revision. We are currently working on an *on-line* implementation, in which new revisions of Wikipedia articles are colored according to trust in real-time, as they are created by users. No change in the basic trust (or reputation) algorithms is required for such an implementation; only the way the algorithms are applied to revisions changes.

## 1.4 Related Work

The problem of the reliability of Wikipedia content has often emerged both in the press (see, e.g., [27, 12]) and in scientific journals [8]. The idea of assigning trust to specific sections of text of Wikipedia articles as a guide to readers has been previously proposed in [21, 4, 34], as well as in white papers [14] and blogs [20]; these papers also contain the idea of using text background color to visualize trust values.

The work most closely related to ours is [34], where the trust of a piece of text is computed from the Wikipedia roles (anonymous, registered user, or editor) of the original author, and of the authors who subsequently revised the article. The Wikipedia roles of authors are thus used in lieu of author reputation; as a consequence, the algorithm can only be applied to wikis where authors are organized in a well-defined hierarchy. Text analysis is performed at the granularity level of sentences; all sentences introduced in the same revision form a *fragment*, and share the same trust. A change anywhere in a sentence causes the whole sentence to be considered new, and the position of the change in the sentence is not flagged via the trust labeling. The cut-and-paste edges of text deletions and reorderings are also not flagged via the trust labeling. Furthermore, deleted text is not tracked: when text is deleted, and then re-inserted, it is counted as new. Among other things, this creates an incentive to vandalism: blanking an article suffices to reset its entire trust assignment. To validate the trust assignment, [34] computes the correlation between the trust of a fragment, and the probability that the fragment appears in the most recent version of the article. We refine this criterion into one of our evaluation criteria, namely, the predictive power of trust with respect to word longevity.

In [21], the trust of authors and fragments is computed on the basis of the author-to-fragment and fragment-of-article graphs, together with the *link ratio* of article titles. The *link ratio* is the ratio of the number of times an article title appears as a link in other articles, and the number of times the title appears as normal text. The work provides trust values for some articles, but no comprehensive evaluation.

The white paper [14] focuses on the user interface aspects of displaying information related to trust and author contributions; we hope to include some of the suggestions in future versions of our

system. Related work that relies on an analysis of revision information to infer trust has been performed in the context of software, where logs are mined in order to find revision patterns that point to possible software defects and weak points (see, e.g., [19]).

Other studies have focused on trust as article-level, rather than word-level, information. These studies can be used to answer the question of whether an article is of good quality, or reliable overall, but cannot be used to locate in an article which portions of text deserve the most careful scrutiny, as our approach can. In [35], which inspired [34], the revision history of a Wikipedia article is used to compute a trust value for the entire article. In [7, 23], metrics derived via natural language processing are used to classify articles according to their quality. In [18], the number of edits and unique editors are used to estimate article quality. The use of revert times for quality estimation has been proposed in [30], where a visualization of the Wikipedia editing process is presented; an approach based on edit frequency and dynamics is discussed in [33]. There is a fast-growing body of literature reporting on statistical studies of the evolution of Wikipedia content, including [30, 31, 24]; we refer to [24] for an insightful overview of this line of work.

The notion of trust has been very widely studied in more general contexts (see, e.g., [2, 10]), as well as in e-commerce and social networks (see e.g. [15, 25, 5, 13, 11, 9]); these notions of trust however are generally based on user-to-user feedback, rather than on an algorithmic analysis of content evolution.

## 2. TEXT TRUST ALGORITHMS

We compute the trust of Wikipedia text on the basis of an algorithmic analysis of how the content of Wikipedia articles evolve across revisions. We assume that, in addition to the text of all article revisions, we have access to a reputation system that, at every point in time, can give us a value of reputation for each author; we assume that reputations take values in a fixed interval  $[0, T_{\max}]$ , for some  $T_{\max} > 0$ . Our goal consists in associating a value of trust in the interval  $[0, T_{\max}]$  to every word of every article revision. We rely on the reputation system of [1], which also computes reputation values based on an analysis of content evolution: thus, the whole system is content-driven.

We present our algorithm for trust assignment in three steps. First, we will illustrate the basic idea via a simplified algorithm that does not cope with reversion, nor in general, with the situation when text is deleted, and later re-inserted. Next, we present an improved algorithm for assigning trust to Wikipedia content that deals with removed-and-reinserted text, and that also contains a tuned model of user attention during the process of article revision. Finally, we discuss the modifications to the algorithm that we introduced to make the trust system robust to tampering.

### 2.1 Notation

We denote the sequence of revisions of a Wikipedia article by  $v_0, v_1, v_2, \dots$ . Version  $v_0$  is empty, and version  $v_i$ , for  $i > 0$ , is obtained by author  $a_i$  performing an edit  $e_i = v_{i-1} \rightsquigarrow v_i$ . When editing a versioned document, authors often save intermediate results, thus performing multiple consecutive edits. Before processing the versions, we filter them, keeping only the last of consecutive versions by the same author; we assume thus that for  $1 \leq i < n$  we have  $a_i \neq a_{i+1}$ . Every version  $v_i$ , for  $0 \leq i \leq n$ , consists of a sequence  $[w_1^i, \dots, w_{m_i}^i]$  of words, where  $m_i$  is the number of words of  $v_i$ ; we have  $m_0 = 0$ . Our system works at the level of the Mediawiki markup language in which authors write article content, rather than at the level of the HTML produced by the wiki engine; a word is a whitespace-delimited alphanumerical string in the Mediawiki markup language.

### 2.2 A simplified text-trust algorithm

Our trust algorithms will assign a trust value in the interval  $[0, T_{\max}]$  for each word of each article revision. Given an edit  $e_i = v_{i-1} \rightsquigarrow v_i$ , a trust value  $t_1, t_2, \dots, t_{m_{i-1}}$  for each word of  $v_{i-1}$ , and a value  $r \in [0, T_{\max}]$  for the reputation of the author  $a_i$  of the revision, the algorithm computes trust values  $\hat{t}_1, \hat{t}_2, \dots, \hat{t}_{m_i}$  for all words of  $v_i$ . The algorithm first computes an *edit list*  $L_i$  detailing how  $v_i$  is obtained from  $v_{i-1}$  [28]. The edit list  $L_i$  consists of one or more of the following elements:

- $I(j, n)$ :  $n$  words are inserted at position  $j$  of  $v_i$  (i.e., words of indices from  $j$  to  $j + n - 1$  are new in  $v_i$ );
- $R(j, n)$ :  $n$  words are deleted at position  $j$  of  $v_{i-1}$ ;
- $M(j, j', n)$ :  $n$  words are moved from position  $j$  in  $v_{i-1}$  to position  $j'$  in  $v_i$  (it may be  $j = j'$ ).

Each word in  $v_i$  is part of exactly one of the above  $I(\cdot)$  or  $M(\cdot)$  elements, and the algorithm to generate edit lists tries to maximize text block matches [1]. The trust computation algorithm uses the following constants:

- $0 \leq c_l < 1$  is the *trust inheritance constant*: it specifies how much trust should a word inherit from the reputation of its author.
- $0 \leq c_r < 1$  is the *revision constant*: it specifies how much trust does the author reputation confer to the text of the article.
- $c_e > 0$  is the *edge effect constant*: when blocks of text are displaced, this constant specifies how far into the blocks is the trust of the text affected by the move.

The values of these constants are obtained via optimization techniques that will be described later. We first compute preliminary trust values  $t'_0, t'_1, \dots, t'_{m_i}$  by considering all elements in the edit list  $L_i$ :

1. *Insertions*. If  $I(j, n) \in L_i$ , then  $t'_k := c_l \cdot r$  for all  $j \leq k < j + n$ : thus, inserted text is assigned a trust value equal to the reputation of the author inserting it, multiplied by the trust inheritance constant.
2. *Block moves*. If  $M(j, j', n) \in L_i$ , then for all  $0 \leq k < n$ ,  $k$  is the distance of the  $k$ -th word in the block from the beginning of the block, and  $\bar{k} = n - 1 - k$  is the distance from the end of the block. We apply an *edge effect*, whereby the text at the block boundary acquires the same trust as new text; this edge effect weakens exponentially towards the interior of the block. The edge effect is not applied to block move boundaries that remain at the beginning or end of the article. Precisely:

- (a) If  $j \neq 0$  or  $j' \neq 0$  then the left endpoint of the block has changed context, and we let:

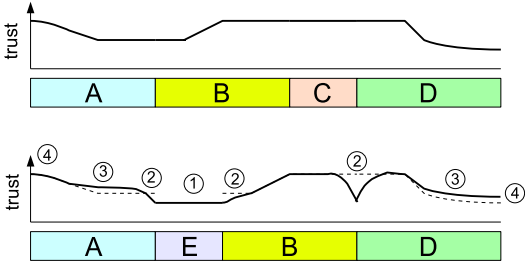
$$t''_{j'+k} = t_{j+k} + (c_l \cdot r - t_{j+k}) \cdot e^{-c_e k}$$

Otherwise, if  $j = 0$  and  $j' = 0$ , we let  $t''_{j'+k} = t_{j+k}$ .

- (b) If  $j+n \neq m_{i-1}$  or  $j'+n \neq m_i$ , then the right endpoint has changed context, and we let:

$$t'_{j'+k} = t''_{j'+k} + (c_l \cdot r - t''_{j'+k}) \cdot e^{-c_e \bar{k}}$$

Otherwise, if  $j+n = m_{i-1}$  and  $j'+n = m_i$ , we let  $t'_{j'+k} = t''_{j'+k}$ .



**Figure 2: Update process for text trust.** The text is shown before (top) and after (bottom) an edit, together with its trust. In the bottom figure, the new values of trust (continuous line) are obtained from the inherited values of trust (dashed line) as follows: **1: Trust value for newly inserted text (E).** **2: Edge effect: the text at the edge of blocks has the same trust as newly inserted text.** **3: Revision effect: old text may increase in trust, if the author reputation is higher than the old text trust.** **4: The edge effect is applied at the beginning and end of the article only if text changes there (which is not the case here).**

If  $R(j, n) \in L_i$ , then the text is deleted, and there is no trust assignment to be made (the edge effect of adjacent blocks to  $R(j, n)$  will take care of flagging the deletion in the new version). Once all elements of the edit list  $L_i$  have been processed, we have preliminary trust values  $t'_1, t'_2, \dots, t'_{m_i}$  which take into account of insertions, block moves, and edge effects. The final trust values  $\hat{t}_0, \hat{t}_1, \dots, \hat{t}_{m_i}$  of the words of  $v_i$  are then computed by accounting for the fact that the author  $a_i$  lends some of her reputation  $r$  to the revision  $v_i$  she just performed. For  $0 \leq k < m_i$ , we let:

$$\hat{t}_k = \begin{cases} t'_k & \text{if } t'_k \geq r \\ t'_k + (r - t'_k) \cdot c_r & \text{if } t'_k < r \end{cases} \quad (1)$$

The trust update process is illustrated in Figure 2. The trust labeling computed by the algorithm is such that high trust requires consensus: only text that survives scrutiny by multiple authors can gain high trust. The trust labeling also provides a warning when text is deleted or reordered. However, this simplified algorithm has a fatal flaw: it does not cope with text that is deleted in a revision, only to be reinserted in a later one. Deletion and reinsertion is a common phenomenon in the evolution of Wikipedia articles: it occurs in many disputes about article content, and even more devastatingly, it occurs when visitors deface articles by removing part or all of their text. If this algorithm were applied to the Wikipedia, a vandal would simply need to delete, and then re-insert, existing text in order to reset its trust to zero. Thus, it would be extremely easy for vandals to destroy trust information and deface the coloring provided by the trust system.

## 2.3 An improved text-trust algorithm

We describe now an improved text-trust algorithm, which keeps track not only of the trust of the text present in an article, but also of the trust of the text that used to be present, but that has subsequently been deleted. The algorithm also models the attention focus of the author performing an edit, raising by a larger amount the trust of the text that is most likely to have been read by the author in the course of the edit.

### 2.3.1 Tracking deleted text.

We track deleted text by representing each article version  $v_i$ , for

$1 \leq i \leq n$ , as a non-empty list  $C_i = [c_0^i, c_1^i, \dots, c_{h_i}^i]$  of *chunks*, where each chunk  $c_k^i$ , for  $0 \leq k \leq h_i$ , is a sequence of words. The *live* chunk  $c_0^i$  corresponds to the words that are present in  $v_i$ ; the *dead* chunks  $c_1^i, \dots, c_{h_i}^i$ , if present, correspond to contiguous portions of text that used to be present in some prior version  $v_0, \dots, v_{i-1}$  of the article, but have been deleted. The chunks  $C_i$  are computed from the chunks  $C_{i-1} = [c_0^{i-1}, c_1^{i-1}, \dots, c_{h_{i-1}}^{i-1}]$  for  $v_{i-1}$  as described in [1]. Specifically, we match the text of  $v_i$  with the text of all the chunks in  $C_{i-1}$ , looking for the longest possible matches of contiguous sequences of words. We break ties in favor of matches between  $v_i$  and the text  $c_0^{i-1}$  that was present in  $v_{i-1}$ , thus preferring matches between  $v_i$  and the live text in  $v_{i-1}$ , to matches between  $v_i$  and the text  $c_1^{i-1}, \dots, c_{h_{i-1}}^{i-1}$  that was present before  $v_{i-1}$  but is “dead” in  $v_{i-1}$ . Furthermore, we allow the text in  $C_{i-1}$  to be matched multiple times, modeling the fact that an author can replicate existing text; the text in  $v_i$  can be matched at most once. The portions of unmatched text in  $C_{i-1}$  go on to form the new dead chunks  $[c_1^{i-1}, \dots, c_{h_{i-1}}^{i-1}]$  for  $v_i$ . In this matching process, lower bounds on the length of acceptable matches ensure that common sequence of words (such as “the” or “in fact”) appearing in new contexts are not considered as copied or re-introduced text.

We update the trust of deleted and reinserted text as follows.

- For text that is moved from the live chunk  $c_0^{i-1}$  to some dead chunk  $c_{h'}^i$ ,  $h' > 0$ , we multiply the trust of the text by  $e^{-rc_k}$ . The idea is that when text is deleted, its trust is decreased in proportion to the reputation  $r$  of the author deleting the text. In particular, text does not lose trust when deleted by anonymous users or novices ( $r = 0$ ). This ensures that when vandals remove all text of an article, once the text is re-inserted it has the same trust as before the vandalism occurred. In our implementation, we have taken  $c_k = (\log 2)/T_{\max}$ , so that the trust of a word is halved when deleted by an author of maximum reputation.
- For text that is moved from a dead chunk  $c_h^{i-1}$ ,  $h > 0$ , to another dead chunk  $c_{h'}^i$ ,  $h' > 0$ , we simply copy the trust.
- For text that is moved from a dead chunk  $c_h^{i-1}$ ,  $h > 0$ , to the live chunk  $c_0^i$ , we update the trust in a manner completely equivalent to the one used for block moves  $M(j, j', n)$  in the previous section, applying the edge effect to both text endpoints.

### 2.3.2 Modeling author attention.

In equation (1) of the previous algorithm, we increase the trust of the text uniformly — this assumes that the author of the revision pays equal attention to the entire text being revised. This assumption is unlikely to be correct, as authors are more likely to pay greater attention to text that is closer to their edits; raising the trust of all the text in the article may impart too much trust to text that has not been the focus of author attention. We decided therefore to experiment with a variation of the algorithm that models author attention in a rudimentary fashion.

When parsing the text of the revision  $v_i$ , we split it into paragraphs, where section titles, items in a bulleted or numbered list, image captions, and table cell entries also count as “paragraphs”. Our algorithm then follows the simple idea that authors are likely to pay more attention to the text in the same paragraph as the edits they are performing. To this end, we mark as *modified* all paragraphs where (a) either new text has been inserted (corresponding to an  $I$  element in the edit list), or (b) the paragraph contains the endpoint of a block move (elements  $M$  in the edit list) to which the edge effect applies. For modified paragraphs we apply, after (1), the

following update:

$$\hat{t}_k := \begin{cases} \hat{t}_k & \text{if } \hat{t}_k \geq r \\ \hat{t}_k + (r - \hat{t}_k) \cdot c_p & \text{otherwise,} \end{cases} \quad (2)$$

where  $0 \leq c_p < 1$  is the *paragraph constant*: it specifies how much additional trust the author reputation confers to the paragraph of the article she modified. Thus, text in modified paragraphs receives an additional trust increment.

## 2.4 Robust trust

The current implementation of the trust system is a batch one, in which the wiki revision history is analyzed off-line. Our goal, however, is to develop algorithms that are suited for on-line implementation and deployment on live wikis. If the trust system is deployed on a high-traffic, and high-visibility wiki, it most likely will come under attack. We consider two types of attacks: *vandalism* attacks, aimed at destroying the trust information, and *tampering* attacks, aimed at causing the text to increase unduly in trust, perhaps to mask malicious changes. We present here methods that make the trust system robust to such attacks. In making the trust system robust to attacks, we assume that the reputation system itself is reliable, in the sense that it is hard for authors to gain reputation in a short time, without strong justification. Thus, we deal with the robustness problem in modular fashion: this paper concerns itself with a robust trust system, while the problem of implementing a robust reputation system will be dealt with elsewhere.

### 2.4.1 Vandalism

The algorithms for text trust that we have presented so far are already robust with respect to vandalism attacks in which portions of text are deleted. Deleted text is tracked by the system, as described in Section 2.3.1. Since vandals typically have a reputation close to 0, the trust of the text is lowered by a small amount when the deletion occurs, as the multiplicative factor  $e^{-rc_k}$  is very close to 1. When the deleted text is re-inserted, its trust value will be essentially unchanged. In a more malicious version of this attack, vandals can perform extensive text re-arrangements, causing much text to be assigned the low trust value used for block-move endpoints (see the edge effect in Figure 2). To defend against this attack, the on-line system we are developing compares the text of revision  $v_k$  with the text of revisions  $v_{k-m}, v_{k-m+1}, \dots, v_{k-1}$ ; special data structures make this comparison efficient even for values of  $m$  that range up to 50 or more. We then identify the past revision  $v_j$  that is closest, in edit distance, to  $v_k$ . The trust assigned to each word of  $v_k$  is then equal to the *largest* of these two trust values:

- the trust value resulting from the edit  $v_{k-1} \rightsquigarrow v_k$ ;
- the trust value computed as if the edit  $v_j \rightsquigarrow v_k$  occurred (thus short-circuiting revisions  $v_{j+1}, \dots, v_{k-1}$ ).

In this fashion, as long as vandalism is reverted within a small number of revisions (no larger than  $m$ ), the original trust of the text is restored.

### 2.4.2 Tampering

The above vandalism attacks have the aim of lowering the trust value of text in an article. The attacks can cause visual distractions for the readers of the article, as much text is labeled and colored as low trust, until the vandalism is corrected. Nevertheless, these attacks never cause text to be labeled with too high a trust value. A more malicious type of attack, which we call *tampering attack*, aims instead at raising the trust value of the text of an article, in

spite of the fact that the text has not been properly revised by the wiki community of authors.

The algorithms described in Sections 2.2–2.3 are not robust with respect to tampering attacks by high-reputation users. According to the algorithms presented so far, new text inserted by an author  $A$  of reputation  $r \in [0, T_{\max}]$  initially has the value of trust

$$c_l r + (r - c_l r) c_r + \left( r - (c_l r + (r - c_l r) c_r) c_p \right) < r .$$

However, if the author  $A$  performs multiple small edits on an unrelated portion of the same article, the trust of this text grows, until it approaches  $r$ . Thus, author  $A$  could first add arbitrary text to one portion of the article, and then perform multiple small edits to another portion of the article. After such sequence of edits, the arbitrary text would have trust very close to  $r$ .

To defend against this type of attack, we allow authors to increase the trust of a word only if they have not already done so recently. Precisely, for each word, we keep track of the list of the last  $m$  authors who have increased the trust of the word. When an author  $A$  performs a revision, for each word  $w$  of the new revision, we first check whether steps (1) (of Section 2.2) and (2) (of Section 2.3.2) would lead to a trust increase for  $w$ . If so, we proceed as follows:

- If  $A$  appears in the list  $l$  associated with  $w$ , we leave the trust of the word  $w$  unchanged.
- If  $A$  does not appear the list  $l$  associated with  $w$ , we insert  $A$  at the beginning of  $l$  and, if the resulting list is longer than  $m$ , we truncate the list to the first  $m$  elements.

This scheme ensures that, after an author raises the trust of a word, at least  $m$  different authors need to raise the trust of the word before  $A$  can do so again.

The scheme obviously prevents the simple attack in which  $A$  tries to raise the trust of the word by editing the article frequently. More subtly, the scheme also prevents  $A$  from raising the trust of a word via Sybil (or sock-puppet) attacks [6, 17, 3, 26]. In these attacks,  $A$  uses multiple identities (all under her control) to try to raise the trust of the word  $w$ . To see this, consider the situation after  $A$  raises a first time the trust of  $w$  to the value  $t$ . After this happens, authors (or sock-puppets) can raise the trust of  $w$  further only if their reputation is above  $t$ . Since we assume that it is difficult for an author or sock-puppet to acquire reputation, it will be difficult for  $A$  to have a sufficient number of high-reputation sock-puppets to cause the trust of  $A$  to raise.

We prefer to associate the list of past revisors with each word, rather than with an entire page. All our algorithms are word-based, so this choice leads to a more uniform setting. Moreover, we believe that the word-level accounting we use leads to a more natural, and fairer, setting. For instance, consider the case where  $A$  raises the trust of a version  $v$  of an article, and shortly afterwards, an author  $B$  of lower reputation inserts some text in the article. If  $A$  edits the page immediately after  $B$ , our word-level accounting enables  $A$  to raise the trust of the text inserted by  $B$ , while preventing  $A$  from raising twice the trust of the text that was already present in  $v$ . Indeed, there would be no reason to disallow  $A$  from raising the trust of the text inserted by  $B$ . The revisor lists can be stored in more compact form via hashing.

We call the trust computed with the help of the anti-tampering algorithm above *tamper-resistant* trust, to contrast it with the *non tamper-resistant* trust described in Sections 2.2–2.3.

### 3. TRUST QUALITY METRICS

We present trust quality metrics that quantify the ability of trust values to predict the future stability of text. The idea is that higher-trust text should be less likely to be revised in the future [34].

We remark that the ability of trust to predict future text stability provides only a partial assessment of the effectiveness of the trust labeling. In fact, the trust labeling is meant both to assess how well text has been revised, and to highlight recent content modifications. These two goals sometimes cannot be reconciled. As an example, consider the case of an author removing a sentence from a paragraph. Our trust labeling will label low-trust both the end of the sentence preceding the removal, and the beginning of the sentence immediately following the removal. This low-trust labeling, and the resulting orange coloring, is used to make readers aware that some edit has occurred — that text was removed. However, the sentences that precede and follow the removal are unlikely to be themselves deleted, so that labeling them low-trust lowers our measured quality of the trust labeling.

Even with these limitations, the quality metrics will be useful to us, providing an estimate of the predictive value of the trust we compute, and offering quantitative data for the optimization of the algorithms.

#### 3.1 Low trust as a predictor of deletions

Two of our quality metrics measure the precision recall of low-trust with respect to text deletions. For each trust value  $t \in [0, T_{\max}]$ , we consider the fact of a word  $w$  having trust  $t_w \leq t$  as a “warning bell”, and we ask what is the recall, and the precision, of this warning bell with respect to the event of the word being deleted in the next revision. The recall  $recl(t)$  measures the fraction of deleted text that had trust smaller than or equal to  $t$  immediately prior to deletion; the precision  $prec(t)$  measures the fraction of text with trust smaller than or equal to  $t$  which is deleted in the next revision. More formally, let:

- $W_{i,p}^{\leq}(t)$  be the number of words in version  $i$  of article  $p$  that have trust no larger than  $t$ ;
- $D_{i,p}^{\leq}(t)$  be the number of words in version  $i$  of article  $p$  that have trust no larger than  $t$  and which are deleted in the revision from version  $i$  to  $i + 1$ ;
- $D_{i,p} = D_{i,p}^{\leq}(T_{\max})$  be the number of words in version  $i$  of article  $p$  which are deleted in the revision from version  $i$  to  $i + 1$ .

Then, we have:

$$recl(t) = \sum_{i,p} D_{i,p}^{\leq}(t) / \sum_{i,p} D_{i,p} \quad (3)$$

$$prec(t) = \sum_{i,p} D_{i,p}^{\leq}(t) / \sum_{i,p} W_{i,p}^{\leq}(t), \quad (4)$$

where the summation is taken for all versions of all articles that are used to evaluate the quality of the trust labeling.

While recall and precision of low-trust are good indicators, they suffer from the fact that text can be deleted by vandals, only to be re-added in the next revision. This source of error can be significant: while people intent on improving an article often delete small amounts of text at a time, vandals often delete the entire text of an article. To obtain better metrics, we would like to give more weight to deletions that happen due to well-thought-out editorial concerns, rather than vandalism. To this end, we employ the notion of *edit longevity* developed in [1]. The edit longevity  $\alpha_{i,p} \in [-1, 1]$  is a measure of how long-lived is the change  $e_i = v_{i-1} \rightsquigarrow v_i$  for article  $p$ . In particular, if  $\alpha_{i,p}$  is  $-1$ , then the change  $e_i$  is reverted

immediately, and if  $e_i$  is a deletion, then practically this should not be considered as a valid deletion. On the other hand, if  $\alpha_{i,p}$  is close to 1, the change will live through many subsequent revisions, and if  $e_i$  is a deletion, then it should be considered as a valid deletion [1]. We use the *edit quality*  $q_{i,p} = (\alpha_{i,p} + 1)/2$  to weigh the data points in (5)–(6), thus giving weight close to 1 to deletions that happen due to authoritative revisions, and no weight to deletions performed by vandals (which have longevity  $-1$ ). We thus define the *quality-weighted* recall and precision of low-trust with respect to deletions as follows:

$$w\_recl(t) = \frac{\sum_{i,p} q_{i,p} D_{i,p}^{\leq}(t)}{\sum_{i,p} q_{i,p} D_{i,p}} \quad (5)$$

$$w\_prec(t) = \frac{\sum_{i,p} q_{i,p} D_{i,p}^{\leq}(t)}{\sum_{i,p} q_{i,p} W_{i,p}^{\leq}(t)}. \quad (6)$$

#### 3.2 Trust distribution of general vs. deleted text

Another quality metric for trust labelings is obtained by comparing the trust value distribution of all text, and of deleted text. Recall that, in our system, we display the text of revisions with a background color that reflects text trust, and which ranges from white for fully trusted text, to orange for text with trust 0. Site visitors are going to use the orange background as an indication that the information may be unreliable. If too much text on an article has orange background, the alert loses effectiveness, as visitors habituate to the constant flagging of text. Thus, we prefer trust labeling in which text, on average, is as trusted as possible. On the other hand, we clearly want text to be flagged as low-trust when it is about to be deleted.

To make these notions precise, we define the following quantities. Given a function  $f : [0, T_{\max}] \mapsto \mathbb{R}$  with  $\int_0^{T_{\max}} f(t) dt < \infty$ , and  $\rho \in [0, 1]$ , we define the  $\rho$ -median of  $f$  the quantity  $a$  satisfying

$$\int_0^a f(t) dt = \rho \int_0^{T_{\max}} f(t) dt.$$

We also denote with  $W_{i,p}^{\bar{}}(t)$  the amount of text having trust  $t$  in version  $i$  of article  $p$ , and we denote with  $D_{i,p}^{\bar{}}(t)$  the amount of text in version  $i$  of article  $p$  having trust  $t$  which will be deleted in version  $i + 1$ . We define the following notations:

$$tot\_txt(t) = \sum_{i,p} W_{i,p}^{\bar{}}(t)$$

$$del\_txt(t) = \sum_{i,p} D_{i,p}^{\bar{}}(t)$$

$$w\_del\_txt(t) = \sum_{i,p} q_{i,p} D_{i,p}^{\bar{}}(t).$$

We assess the quality of a trust labeling via the following quantities, for  $\rho \in [0, 1]$ :

- The  $\rho$ -white point is the  $\rho$ -median of  $tot\_txt(t)$ .
- The *weighed orange average* is the average value of  $t$  for  $w\_del\_txt(t)$ .

We will use  $W_{0.9}$  and  $Org_{avg}$  to denote the 0.9-white point and weighed orange average, respectively. Again, the weighing used in the definition of orange average is used to give more weight to deletions that occur in the course of higher-quality revisions.

### 3.3 Trust as predictor of text life-span

Our final quality metric for the trust labeling consists in quantifying the predictive value of word trust with respect to the subsequent life-span of the word. To measure this predictive value, we sample word occurrences from all versions uniformly at random (applying the algorithm to all words would be computationally very expensive), and we observe for how many consecutive article versions the words are present after their sampled occurrence.<sup>1</sup>

The simplest approach consists in studying the correlation between the trust  $t$  of the word at the moment it is sampled, with the life-span  $l$  of the word, measured as the number of consecutive subsequent versions in which the word is present. However, such a measurement would be biased by the *horizon effect* induced by the fact that we have only a finite sequence  $v_0, v_1, \dots, v_n$  of versions to analyze. Words sampled in a version  $v_i$ , and that are still present in the last version  $v_n$ , have a life-span of  $n - i + 1$ , even though they may live much longer once the wiki evolves and versions beyond  $v_n$  are introduced. This horizon effect causes us to under-estimate the true life-span of high-longevity words sampled close to the last existing version of an article.

To obtain a measurement that is unaffected by this horizon effect, we model the life-span of a word as a memoryless decay process, in which the word has a constant probability (dependent on the word, but not on its past life-span) of being deleted at every revision. Thus, we assume that the probability that a word that is alive at  $v_i$  is still alive at  $v_k$ , for  $k \geq i$ , is  $e^{-(k-i)/\lambda}$ , where  $\lambda$  is the half-life of the word under infinite-horizon. Our task is then to estimate the correlation between the trust  $t$  that the word has in  $v_i$  and the half-life  $\lambda$  of the word. Note that this definition of half-life eliminates the horizon effect due to the finite number of versions.

For every word sampled at  $v_i$ , and last present in  $v_k$ , with  $i \leq k \leq n$ , we output a triple  $(t, l, h)$  consisting of the trust  $t$  of the word in  $v_i$ , the life-span  $l = k - i + 1$ , and the observation horizon  $h = n - i + 1$ . To estimate  $\lambda$ , we use the following observation: if  $l < h$ , then the word would have lived for  $l$  even under infinite horizon; if  $l = h$ , then the word has an average life-span of  $l + \lambda$  under infinite horizon, since the distribution is memoryless. Let  $A$  be the set of triples sampled for a trust level  $t$ . Let:

- $m$  be the number of samples in  $A$  with  $l < h$ ;
- $M = \sum \{l \mid l < h \wedge (t, l, h) \in A\}$ ;
- $k$  be the number of samples in  $A$  with  $l = h$ ;
- $K = \sum \{l \mid l = h \wedge (t, l, h) \in A\}$ .

We can estimate  $\lambda$  via

$$\lambda = \frac{M + K + k \cdot \lambda}{m + k}$$

which yields

$$\lambda = \frac{M + K}{m}.$$

A trust labeling will have high predictive value for life-span if larger values for the trust of the word in  $v_i$  correspond to larger values of  $\lambda$ .

<sup>1</sup>As we have seen in Section 2.3.1, a word in a version can correspond to multiple occurrences in the next version, when text is duplicated. When tracking a word to measure its life-span, whenever the word is duplicated, we track all occurrences separately.

## 4. IMPLEMENTATION

We have implemented a trust tool that computes text trust and provenance for the Wikipedia. The trust tool takes as input an XML dump containing all the text of all the revisions of the Wikipedia; such dumps are periodically made available from the Wikimedia Foundation. The trust tool is written in Ocaml [16]; we chose this language for its combination of speed and excellent memory management. On an Intel Core 2 Duo 2 GHz CPU, our tool is capable of assigning trust to versions of Wikipedia articles<sup>2</sup> at over 15 versions/second, or roughly 1.5 millions versions per day, an edit rate much higher than the one of the on-line Wikipedia [32]. We have run the trust tool over the entire English Wikipedia, as of its February 6, 2007 dump; the results can be viewed on a live demo [29]. To save disk space on the server, the demo contains only the last 100 versions of each article, but all versions were considered in trust computation.

The current implementation of the tool is a batch one. The first step consists in computing the *reputation history* of all Wikipedia authors. When the trust system examines a revision  $v_k \rightsquigarrow v_{k+1}$  performed by an author  $A$ , it looks up the value of trust of author  $A$  in the reputation history of  $A$ , corresponding to the time  $t_{k+1}$  when  $v_{k+1}$  was created. The trust system uses the reputation of  $A$  at time  $t_{k+1}$ , rather than the “final” or “average” reputation of  $A$ , in order to mimick faithfully the trust computation that is used in the on-line system we are developing.

The reputation histories are computed using the content-driven reputation system for Wikipedia authors proposed in [1]. In this system, authors of contributions which prove long-lasting gain in reputation, while authors whose contributions are reverted lose reputation. Specifically, whenever an author  $A$  edits an article that had been previously edited by another author  $B$ , a change in reputation is generated for  $B$ : the reputation of  $B$  increases if  $A$  preserves  $B$ 's contribution, and decreases if  $A$  undoes  $B$ 's contribution. The reputation system is thus *chronological*: the reputation is computed from the chronological sequence of increments received by authors. The reputation system is such that  $T_{\max} = 9$ .

Once the reputation histories of all users have been computed, we feed the reputation histories, and the Wikipedia XML dump, to the trust tool. The tool produces as output a *colorized XML dump*, containing the original text annotated with the computed trust and provenance information. The colorized dump is in the same format as the input XML dump, except that two additional markup tags are interposed in the text:

- the tag  $\{\{\#t:x\}\}$  indicates that the subsequent text has trust  $x \in \{0, 1, \dots, 9\}$  (trust is rounded to the nearest integer for display purposes);
- the tag  $\{\{\tau o:i\}\}$  indicates that the subsequent text was first inserted in version  $i$  (Mediawiki assigns to each version a global integer identifier).

To save storage, these tags are not added for each word, but only when the information changes from one word to the next.

The colorized XML dump can be loaded in a Mediawiki installation using the standard tools made available as part of Mediawiki (Mediawiki [22] is the software package responsible for implementing the wiki behind Wikipedia). The additional tags are then interpreted by a Mediawiki extension we developed, following the Mediawiki extension framework. We display the trust of each word by coloring the background of the word: white for fully trusted words, and increasingly intense gradations of orange for progressively less

<sup>2</sup>Measured on a randomly-selected subset of articles with at least 200 versions each.

trusted text. For text origin, our extension defines a on-click action in JavaScript. When a user clicks on a word, the user is sent to the article version where the word was first inserted. The two types of information, trust and origin, augment each other, and together provide Wikipedia visitors with effective tools to judge the accuracy of article content.

## Towards an on-line implementation

We are currently working on an *on-line* implementation of the trust system, capable of coloring the revisions of Wikipedia articles as they are created. The on-line system will be suited to any Mediawiki-based wiki, and indeed to any wiki, via minor adaptations.

While in the batch system the computation of author reputation histories, and the computation of text trust, happen in two separate passes, in the on-line system author reputations and trust are updated in real-time, every time a new article revision is created. When a revision is created, the on-line system first analyzes the text difference between the revision and the previous article revisions. This information is passed to the reputation system, which updates the reputation of the authors of previous article revisions: past authors whose contributions are preserved in the latest revision gain reputation, while authors whose contributions have been undone lose reputation [1]. The information on text tracking is then passed to the trust system, which updates word trust according to the algorithms of Section 2. Consequently, in the on-line implementation, every new revision causes both reputation and trust updates.

We note that this on-line system, and the batch system we have used for evaluation purposes, compute the same values, due to our use of author reputation histories to compute trust values. Thus, the performance figures that we report for the batch system are representative of the on-line system.

## 5. EVALUATION

Our first step in the performance evaluation of the trust labeling consisted in choosing values for the constants appearing in the trust labeling algorithm.

Choosing values for the constants involves seeking a balance between the opposite goals of alerting visitors to as much unreliable content as possible, and avoiding visual clutter and information overload. We found it helpful to reason about how “white” a mature article should be on average, and about how “orange” the deleted text should be: thus, we performed the optimization using the white point and orange average, as defined in Section 3. We let  $W'_{0.9} = W_{0.9}/T_{\max} \in [0, 1]$  be the normalized 90%-white-point, and we let  $Org'_{avg} = (T_{\max} - Org_{avg})/T_{\max} \in [0, 1]$  be the normalized weighed orange average, where  $T_{\max} = 9$  for our system. We wanted to find parameter values that would make the article, overall, as white as possible (maximize  $W'_{0.9}$ ), while ensuring the deleted text was as orange as possible (maximize  $Org'_{avg}$ ). To this end, we used linear search on the space of the parameters to optimize the value of the *weighed harmonic mean* of  $W'_{0.9}$  and  $Org'_{avg}$ , i.e., we optimize

$$F(W'_{0.9}, Org'_{avg}) = \frac{2 \cdot W'_{0.9} \cdot Org'_{avg}}{W'_{0.9} + Org'_{avg}},$$

for a set of 100 articles used for training. We use the weighed harmonic function since it weighs both of its arguments evenly. This led to the following values for the parameters, for non tamper-resistant trust:

$$c_r = 0.2 \quad c_l = 0.4 \quad c_e = 2 \quad c_p = 0.2 \quad c_k = (\log 2)/T_{\max}. \quad (7)$$

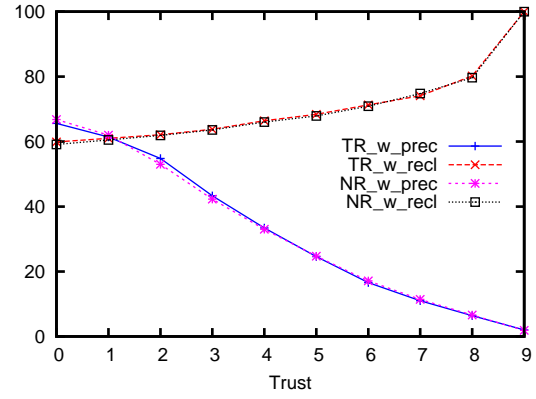


Figure 3: Low-trust as a predictor of deletions: quality weighted precision and recall. Lines prefaced TR are produced by tamper resistant algorithms. Those prefaced by NR are not tamper resistant. w\_prec and w\_recl are weighted precision and recall, respectively.

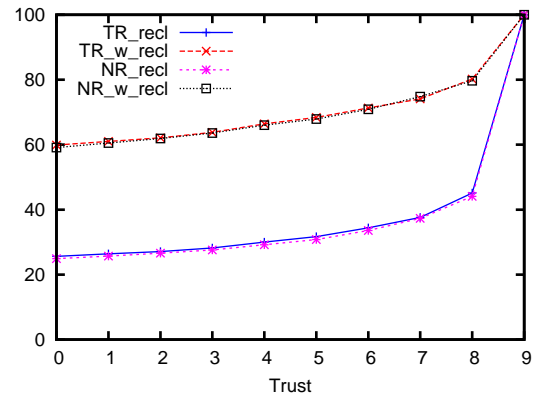


Figure 4: Comparison of recall and weighed recall. w\_recl, recl is the weighed, un-weighted recall.

For the tamper-resistant trust, we choose  $m = 3$ , so that an author needs to wait until three other authors of similar reputation raise the trust of a word, before being able to raise it herself again. As tamper-resistant trust yields slightly lower trust (as authors are occasionally prevented from raising the trust of words), we compensate by taking  $c_r = 0.3$ , which yields essentially the same values for the white point and orange average; the other coefficients are as in (7). With this choice, the results we obtained for normal trust, and for tamper-resistant trust, are quite similar. In the figures, we indicate with *NR* the non tamper-resistant trust, and with *TR* the tamper-resistant trust.

We proceeded to evaluate the performance of the trust coloring on a set of 1,000 articles selected uniformly at random among the articles with at least 200 revisions; the articles comprised 544,250 versions all together, for a total of 13.7 GB of text. We focused on articles with long revision histories for two reasons. From a technical point of view, the long revision history enables us to better estimate the predictive power of trust with respect to text stability. From a user-interface point of view, our trust is especially useful for mature articles: it is relatively easy for visitors to conclude that incomplete articles, with short revision history, cannot (yet) be trusted.

Figure 3 gives the quality-weighted precision and recall of low

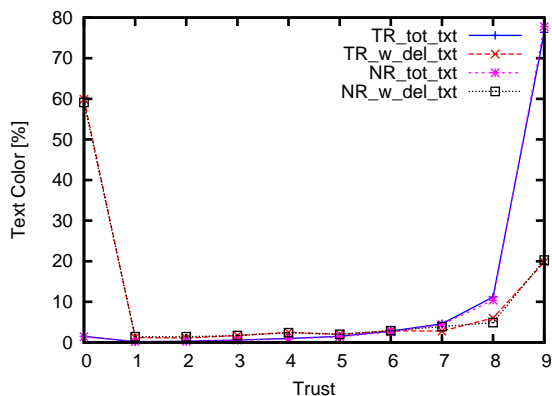


Figure 5: Color of general and deleted text. *tot\_text* shows the percent of all text in each trust bin. *w\_del\_text* shows the weighed trust of deleted text only.

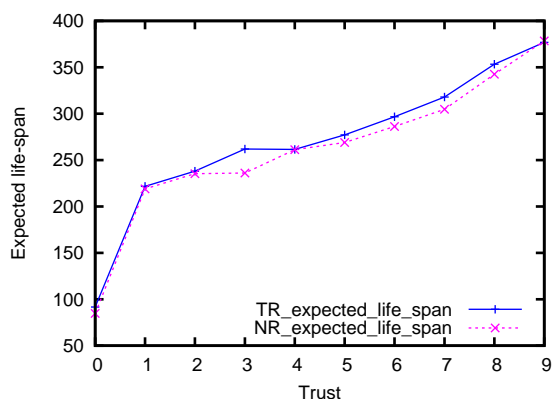


Figure 6: Expected future life-span  $\lambda$  of words.

trust with respect to text deletions. We see that the recall is always at 60% or above; in practice, a mid-range orange background, which is sure to attract a reader’s attention, is able to warn the reader to 2/3 of the text that will be deleted in the next revision. We believe that this is a good performance figure, given that text can be deleted for many reasons other than poor quality, such as rewording: thus, some deletions are never likely to be anticipated by low trust. The precision figures give the probability that text marked as low-trust will be deleted in the very next revision; low precision figures would be a sign of excessive warnings to visitors. We see that text with trust 0 has a 2/3 probability of being deleted in the next revision, and text with mid-level trust has a 1/3 probability of deletion; we consider this to be an acceptable level, especially since not all text that will be deleted is going to be deleted in the very next revision. In Figure 4 we compare weighed and unweighed recalls: as we see, if we also include deletions due to vandalism (*recl*), our recall drops, reflecting the fact that such vandalistic deletions are hard to predict.

The color profiles of general and deleted text are compared in Figure 5. We can see that deleted text, on average, is much lower in trust: indeed, the average trust of deleted text was 2.96, while 90% of text had a trust above 7.60 (out of a maximum of  $T_{max} = 9$ ).

Figure 6 depicts the correlation between the trust of a word occurrence, and the subsequent life-span of the word. The data is obtained by random sampling of word occurrences, and tracing the future of the word from the sampling point. We note that the trust is the trust of the *word occurrence*: over the subsequent life-span,

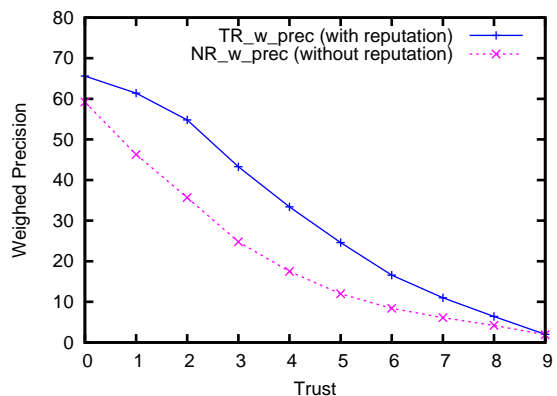


Figure 7: The weighed precision with and without reputation systems.

the word trust may well vary (and typically increases, as the article is revised). We see that there is a clear correlation: higher trust corresponds to a longer expected life-span. We also see that there is a sharp increase in expected life-span as we go from words labeled with trust 0 to words labeled with trust 1. This can be explained by the high “early mortality” of words with trust 0: over 60% of them, as indicated by the recall graph in Figure 3, do not make it to the next version.

We also evaluated the magnitude performance improvement due to the use of the author attention modeling presented in Section 2.3.2. To our surprise, we discovered that the author attention modeling does not appreciably improve the results, in spite of introducing additional degrees of freedom in the trust algorithms. We believe this is due to the fact that authors usually edit the sections of an article that have received the most recent edits. Thus, outside of the paragraph being edited, there is not much text which can benefit from a trust increase, and distinguishing between edited and non-edited paragraphs has little effect.

## 5.1 Trust quality in absence of a reputation system

The reputation system provides two key benefits to our trust system: it provides information on the quality of the authors, and (most importantly) it enables us to obtain a system that is resistant to tampering. The present evaluation, however, is performed on past data, where tampering cannot have occurred, as authors were unaware even of the proposal for such a system. This provides us with the opportunity to evaluate the quality improvement of the trust system that can be ascribed to the use of a reputation system.

To this end, we compared the performance to the regular trust system, with the performance of a modified trust system that does not rely on a reputation system, and instead assigns everybody, from anonymous visitors to well-established editors, the maximum value of trust. Fresh text, as well as block-move edges, received initially trust 0,<sup>3</sup> and the trust of text would then increase according to the algorithms of Section 2 (no change was made to the trust algorithms). We note that this is in fact equivalent to using the *age* of text, measured in number of revisions, to compute the trust. We chose coefficients for the trust computation that would yield a weighed orange average similar to the one obtained using a reputation system.

The trust labeling computed without the aid of a reputation system performed worse than the one that made use of the reputation

<sup>3</sup>Had we used a trust value greater than 0 as initial value, no text would ever get trust 0.

system of [1]. The performance gap was most noticeable with respect to the precision, as illustrated in Figure 7: for trust 4, for instance, the precision was nearly double (33%) with the reputation system than without (17.5%). The gap for recall was narrower: for trust 4, the quality-weighted recall was 66% using a reputation system, and 72.5% without. Furthermore, while deleted text had similar colors, the average text was noticeably more orange in the tests not using the reputation system: the 90% white point went from 7.6 using reputation, to 5.43 when reputation was not used.

This performance difference can be explained as follows. One of the benefits of using a reputation system is that text which is inserted or moved by high-reputation authors receives a non-zero initial value of trust (in our system,  $0.616 \cdot 9 \approx 5.5$ ). This reflects the fact that high-reputation authors are statistically more likely to perform good contributions [1]. If we do without a reputation system, all newly inserted or rearranged text instead has trust 0 initially. This makes the text lower-trust overall (thus the lower 90% white point), and this decreases precision, since among the low-trust text is plenty of text that is due to authors who are statistically likely to perform good contributions.

## 5.2 Discussion

The results on precision and recall, word longevity prediction, and trust distribution overall indicate that the trust we compute has indeed a predictive value with respect to future text stability. As mentioned in the introduction, this is an indication that the trust system provides valuable information; the visitors to our on-line demo seemed, in anedoctical fashion, to corroborate this finding.

A natural question is whether a similar performance could be obtained more simply by considering the “age” of text in articles. To answer this question, first consider how “age” can be measured. There are two natural choices: to measure text age via the number of revisions, or via the amount of time, for which the text survived.

Consider first the case of age measured via the number of revisions for which text survived. This is the scenario described in Section 5.1 above, and as indicated there, it leads to somewhat inferior performance. The biggest drawback of this approach, however, is that it would lead to a trust system that is extremely susceptible to tampering: to raise the trust of a portion of text, all an authour would need to do is to edit the article multiple times, perhaps with the help of sock puppets.

Measuring age as the amount of time for which text survived, on the other hand, would lead to problems due to the varying rate at which Wikipedia articles are edited. Choosing a fast time-constant for trust increase would most likely work well for popular articles, but would enable text in seldom-visited, seldom-edited pages to quickly gain trust in thee near absence of actual revision. Choosing a slow time-constant, on the other hand, would prevent text on topical articles, subject to frequent edits, from gaining much trust.

Furthermore, we note that a trust system based on text age would not be markedly more computationally efficient than the present one. In terms of efficiency, the main challenge in the trust system, as in the reputation system of [1], consists in parsing and tracking the text across revisions. This parsing and tracking would be required even if some notion of age was adopted as a trust metric.

In conclusion, we believe that the trust system we proposed provides a good balance between implementation complexity, performance, and resistance to tampering.

## Acknowledgements.

We would like to thank Jason Benterou for implementing the Javascript that made the display of provenance information possible.

## 6. REFERENCES

- [1] B.T. Adler and L. de Alfaro. A content-driven reputation system for the Wikipedia. In *Proc. of the 16th Intl. World Wide Web Conf. (WWW 2007)*. ACM Press, 2007.
- [2] C. Castelfranchi and eds. Y. Tan. *Trust and Deception in Virtual Societies*. Kluwer Academic Publishers, 2001.
- [3] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *Proc. of the ACM SIGCOMM workshop on Economics of peer-to-peer systems*. ACM Press, 2005.
- [4] T. Cross. Puppy smoothies: Improving the reliability of open, collaborative wikis. *First Monday*, 11(9), September 2006.
- [5] C. Dellarocas. The digitization of word-of-mouth: Promises and challenges of online reputation systems. *Management Science*, October 2003.
- [6] J.R. Douceur. The sybil attack. In *Peer-to-Peer Systems: First Intl. Workshop*, volume 2429 of *Lect. Notes in Comp. Sci.*, pages 251–260, 2002.
- [7] W. Emigh and S. Herring. Collaborative authoring on the Web. In *Proc. of HSCC*, 2005.
- [8] J. Giles. Internet encyclopaedias go head to head. *Nature*, pages 900–901, December 2005.
- [9] J.A. Golbeck. *Computing and Applying Trust in Web-Based Social Networks*. PhD thesis, University of Maryland, 2005.
- [10] T. Grandison and M. Sloman. A survey of trust in internet application. *IEEE Comm. Surveys Tutorials*, 3(4), 2000.
- [11] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proc. of the 13th Intl. Conf. on World Wide Web*, pages 403–412. ACM Press, 2004.
- [12] M. Hickman and G. Roberts. Wikipedia — separating fact from fiction. *The New Zealand Herald*, Feb. 13 2006.
- [13] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proc. of the 12th Intl. Conf. on World Wide Web*, pages 640–651. ACM Press, 2003.
- [14] R. King. Contributor ranking system, 2007. White paper available from [http://trust.cse.ucsc.edu/Related\\_Work](http://trust.cse.ucsc.edu/Related_Work).
- [15] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.
- [16] Xavier Leroy. Objective caml. <http://caml.inria.fr/ocaml/index.en.html>.
- [17] B.N. Levine, C. Shields, and N.B. Margolin. A survey of solutions to the sybil attack. Technical Report Technical Report 2006-052, Univ. of Massachussets Amherst, 2006.
- [18] A. Lih. Wikipedia as participatory journalism. In *Proc. 5th International Symposium on Online Journalism*, 2004.
- [19] V.B. Livshits and T. Zimmerman. Dynamine: Finding common error patterns by mining software revision histories. In *ESEC/FSE*, pages 296–305, 2005.
- [20] P. Massa. Wikipedia trust network, 2007. [http://www.gnuband.org/2007/06/26/wikipedia\\_trust\\_network/](http://www.gnuband.org/2007/06/26/wikipedia_trust_network/).
- [21] D.L. McGuinness, H. Zeng, P.P. da Silva, L. Ding, D. Narayanan, and M. Bhaowal. Investigation into trust for collaborative information repositories: A Wikipedia case study. In *Proceedings of the Workshop on Models of Trust for the Web*, 2006.
- [22] <http://www.mediawiki.org/>.
- [23] B. Mingus, T. Pincock, and L. Rassbach. Using natural language processing to determine the quality of Wikipedia articles. In *Wikimania, Taipei, Taiwan*, 2007.

<http://wikimania2007.wikimedia.org/wiki/Proceedings:BM1>.

- [24] F. Ortega and J.M. Gonzales-Barahona. Quantitative analysis of the Wikipedia community of users. In *Proc. of Wikisym*. ACM Press, 2007.
- [25] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kiwabara. Reputation systems. *Comm. ACM*, 43(12):45–48, 2000.
- [26] J.-M. Seigneur, A. Gray, and C.D. Jensen. Trust transfer: Encouraging self-recommendations without sybil attack. In *Trust Management*, volume 3477 of *Lect. Notes in Comp. Sci.* Springer-Verlag, 2005.
- [27] R. Stross. Anonymous source is not the same as open source. *The New York Times*, Mar. 12, 2006.
- [28] W.F. Tichy. The string-to-string correction problem with block move. *ACM Trans. on Computer Systems*, 2(4), 1984.
- [29] The ucsc wikipedia trust project, 2007. <http://trust.cse.ucsc.edu>.
- [30] F. Viégas, M. Wattenberg, and K. Dave. Studying cooperation and conflict between authors with history flow visualizations. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 575–582, 2004.
- [31] J. Voss. Measuring Wikipedia. In *Proc. of ISSI*, 2005.
- [32] <http://stats.wikimedia.org/EN/TablesDatabaseEdits.htm>.
- [33] D. Wilkinson and B. Huberman. Cooperation and quality in Wikipedia. In *Proc. of WikiSym*. ACM Press, 2007.
- [34] H. Zeng, M. Alhossaini, R. Fikes, and D.L. McGuinness. Mining revision history to assess trustworthiness of article fragments. In *Proc. of the 2nd Intl. Conf. on Collaborative Computing: Networking, Applications, and Worksharing (COLLABORATECOM)*, 2006.
- [35] H. Zeng, M.A. Alhossaini, L. Ding, R. Fikes, and D.L. McGuinness. Computing trust from revision history. In *Intl. Conf. on Privacy, Security and Trust*, 2006.