Towards a qualitative analysis of diff algorithms

Gioele Barabucci, Paolo Ciancarini, Angelo Di Iorio, and Fabio Vitali

Department of Computer Science and Engineering, University of Bologna

Abstract. This paper presents an ongoing research on the qualitative evaluation of diff algorithms and the deltas they produce. Our analysis focuses on qualities that are seldom studied: instead of evaluating the speed or the memory requirements of an algorithm, we focus on how much *natural*, compact and fit for use in a certain context the produced deltas are. This analysis started as a way to measure the naturalness of the deltas produced by JNDiff, a diff algorithm for XML-based literary documents. The deltas were considered natural if they expressed the changes in a way similar to how a human expert would do, an analysis that could only be carried out manually. Our research efforts have expanded into the definition of a set of metrics that are, at the same time, more abstract (thus they capture a wider range of information about the delta) and completely objective (so they can be computed by automatic tools without human supervision).

1 Challenges in evaluating diff algorithms and deltas

The diff algorithms have been widely studied in literature and applied to very different domains (source code revision, software engineering, collaborative editing, law making, etc.) and data structures (plain text, trees, graphs, ontologies, etc.). Their output is usually expressed as *edit script*s, also called *deltas* or *patch*es. A delta is a set of operations that can be applied to the older document in order to obtain the newer one. Deltas are hardly ever unique, since multiple different sequences of operations can be devised, all capable of generating the newer document from the older one.

Each algorithm uses its own strategies and data-structures to calculate the "best" delta. Some of them are very fast, others use a limited amount of memory, others are specialized for use in a specific domain and data format. Surprisingly enough, the evaluation of the quality of the deltas has received little attention.

The historical reason is that most algorithms have been proposed by the database community focusing more on efficiency rather than quality. Another reason is that the produced deltas are not easily comparable: not only each algorithm choose different sequences of changes, but they even use their own internal model and recognize their own set of changes. For example, some algorithms detect moves while others do not, or the same name is used for different operations. Given this degree of heterogeneity, it is hard to evaluate the quality of these algorithms in an automatic and objective way.

Nonetheless, we believe that such an evaluation is essential for the final users and can effectively support them in selecting the best algorithm for their needs.

It is important, for instance, that the operations contained in a delta reflect meaningful changes to the documents, i.e., the detected changes are as close as possible to the editing operations that were actually performed by the author on the original document. Our research started by studying a way to make this quality more explicit.

2 A first manual approach: *naturalness* in JNDiff

In [2] we proposed an explicit metric useful in the evaluation, design and implementation of algorithms for diffing literary XML documents: the *naturalness*. Naturalness indicates how much an edit script resembles the changes effectively performed by the author on a document.

In relation to naturalness, in [2] we:

- discussed an extensible set of natural operations that diff algorithms should be able to detect, focusing on text-centric documents;
- presented an algorithm (NDiff) that detects many of these natural changes;
- described JNDiff, a Java implementation of the NDiff algorithm together with tools to apply the delta produced and highlight modifications;
- presented a case study in detecting changes in XML-encoded legislative bills, and described the benefits of natural deltas in improving the editing and publishing workflow of such documents.

In our view, naturalness is a property connected to the human application and interpretation of document editing, i.e., it can be fully validated only by people that know the editing process. That is why we started researching into other more objective ways to indirectly measure the naturalness.

The first approximation was to examine how close was the generated delta to the description of the changes given by an expert. The first step to calculate such approximation is to create a *gold standard* (an ideal edit script) by comparing pairs of document versions, both visually and structurally. The second step is to identify *clusters of changes* in the delta of each algorithm that correspond to the changes in the gold standard, and to assign a similarity value to each one, depending on a number of parameters. Most of these operations are manual: we manually generate the gold standard, then we manually link the changes in the delta to the clusters, dealing with different output formats.

The key part of the second step consists in assigning a score to each cluster of edit operations to assess its naturalness, taking into account these aspects:

1. *Minimality of the cluster*: we consider a cluster composed of a few sophisticated changes more natural than a cluster composed of many basic changes.
2. *Minimality of the number of nodes*: we rate as more natural clusters that affect fewer nodes, either elements or text characters: this penalizes imprecise edit scripts and rewards scripts in which only the needed nodes are modified.
3. *Minimality of the length of text nodes*: we regard as more natural the edit scripts in which the basic unit for text modifications are the single words, not whole paragraphs; the insertion/removal of big chunks of text where only few characters have been changed is considered verbose and not natural.

The score of the i-th cluster of the delta is calculated as the inverse of a weighted sum of the above mentioned parameters. Further coefficients m_e, m_n and m_c are needed to balance the weights (because, for instance, the number of characters is on average much higher than the number of edit actions or of affected nodes).

$$nat\,(\Delta, i) = (w_e \times m_e \times EDITS_i + w_n \times m_n \times NODES_i + w_c \times m_c \times CHARS_i)^{-1}$$

Eventually, after various experiments on real-world documents, we instantiated the general formula as:

$$nat(\Delta, i) = (0.2 \times EDITS_i + 0.1 \times NODES_i + 0.0077 \times CHARS_i)^{-1}$$

3 Automated analysis

The JNDiff naturalness formula as presented has two main issues. First, its evaluation is impossible to automate as it requires the identification of a gold standard for each cluster of each delta and the ability to match the generated changes to the corresponding changes in the gold standard. Second, different users have different requests and expectations for a diff system and a single metric is not enough to show how well these requests are matched. Not only these requests are different, often they also conflict with each other: for example in certain cases a cursory summary of what has changed is enough, while in others an extreme level of detail is needed.

The idea of measuring objective indicators on the delta, on the other hand, is promising. The same idea of naturalness could be generalized and seen as one of the many qualities of a delta.

In order to define metrics for analyzing deltas under multiple aspects and in an objective way, one has to rely on properties that can be extracted and elaborated by automatic tools without resorting to human evaluations. To reach this goal we elaborated a universal delta model that works on linear texts, trees and graphs. This universal delta model is based on the concept of iterative recognition of more meaningful changes starting from simple changes. Using this model we extracted the properties shown in table 1.

By themselves, the values of these properties say little about the various qualities of the delta. However, once these properties have been extracted they form the base upon which we build several metrics, each of which focused on a single aspect of the analyzed delta. For instance the measure of how much redundant information has been included in a delta (the so called *conciseness* metric) uses two properties of deltas: the number of modified elements and the number of referenced-yet-not-modified elements. We derived four key metrics, described in table 2.

Preliminary experiments showed that these metrics are useful to characterize diff algorithms. In particular, we compared the deltas produced by three well-known XML diff tools (JNDiff [2], XyDiff [1] and Faxma [3]) on a small dataset of real documents. The metrics highlighted, for example, the tendency of some

Property	Definition
population	The total number of changes of which a change is composed of, including itself.
depth	The maximum number of encapsulation layers that must be crossed to reach an atomic change.
width	The number of distinct changes encapsulated inside the change.
touched elements	The number of distinct pieces of information that are included as part of the change or of the encapsulated changes.
modified elements	The minimum number of pieces of information that must be modified by the change to fulfill its purpose.
number of top-level	The number of changes that are not encapsulated in any other change.

Table 1. Properties of changes and deltas

Metric	Definition	Formula
Precision	How many non modified elements have been included in the delta.	$\frac{modified-elements(\delta)}{touched-elements(\delta)}$
Conciseness	How much the changes found in the delta have been grouped into bigger changes.	$1 - \frac{\#top-level(\delta)}{population(\delta)}$
Meaningfulness	How much of the delta conciseness is due to the use of complex changes.	$\frac{\#top-level_{complex}(\delta)}{\#top-level(\delta)}$
Aggregation	How much of the inner parts of the delta is expressed using complex changes instead of atomic changes.	$1 - \frac{\#top-level_{atomic}(\delta)}{population_{atomic}(\delta)}$

Table 2. Metrics

algorithms to detect many localized small changes instead of fewer big changes, or to aggregate changes, or to produce verbose output. In the future we plan to further investigate new metrics (together with their related qualities) and new applications to discover information about the editing process of documents.

References

1. Cóbena, G., Abiteboul, S., Marian, A.: Detecting changes in XML documents. In: Agrawal, R., Dittrich, K.R. (eds.) Proceedings of the 18th International Conference on Data Engineering, San Jose, CA, USA, February 26 - March 1, 2002. pp. 41–52. IEEE Computer Society (2002)
2. Di Iorio, A., Schirinzi, M., Vitali, F., Marchetti, C.: A natural and multi-layered approach to detect changes in tree-based textual documents. In: Filipe, J., Cordeiro, J. (eds.) Enterprise Information Systems, 11th International Conference, ICEIS 2009, Milan, Italy, May 6-10, 2009. Proceedings. Lecture Notes in Business Information Processing, vol. 24, pp. 90–101. Springer (2009)
3. Lindholm, T., Kangasharju, J., Tarkoma, S.: Fast and simple XML tree differencing by sequence alignment. In: Bulterman, D.C.A., Brailsford, D.F. (eds.) Proceedings of the 2006 ACM Symposium on Document Engineering, Amsterdam, The Netherlands, October 10-13, 2006. pp. 75–84. ACM (2006)