Computer Standards & Interfaces 46 (2016) 52–65

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

# Measuring the quality of diff algorithms: a formalization

Gioele Barabucci [b], Paolo Ciancarini [a], Angelo Di Iorio [a,*], Fabio Vitali [a]

[a] *University of Bologna, Italy*
[b] *Universität zu Köln, Germany*

ARTICLE INFO

*Article history:*
Received 22 December 2014
Received in revised form 15 November 2015
Accepted 28 December 2015
Available online 22 January 2016

*Keywords:*
Diff algorithms
Output quality
Metrics
Delta model
XML diff

ABSTRACT

The automatic detection of differences between documents is a very common task in several domains. This paper introduces a formal way to compare diff algorithms and to analyze the deltas they produce. There is no one-fits-all definition for the quality of a delta, because it is strongly related to the application domain and the final use of the detected changes. Researchers have historically focused on minimality: reducing the size of the produced edit scripts and/or taming the computational complexity of the algorithms. Recently they started giving more relevance to the human interpretability of the deltas, designing tools that produce more readable, usable and domain-oriented results. We propose a universal delta model and a set of metrics to characterize and compare effectively deltas produced by different algorithms, in order to highlight what are the most suitable ones for use in a given task and domain.

## 1. Introduction

The automatic comparison of two different versions of a document and the compilation of a list of changes between them is a common task. A *diff* algorithm is used for this purpose: it takes two files as input and computes their difference, according to a given set of change operations. The outputs of diff algorithms, usually called *deltas*, *diffs* or *patches*, are used for many purposes: programmers review source code diffs to avoid adding bugs and to understand which parts of the code has changed; editors highlight the changes made on drafts and pre-prints; law makers compare proposals during the discussion and approval of a bill; philologists use the differences between documents to recreate the *stemma codicum* of a text, the history of its development. An exhaustive survey on change detection and versioning tools can be found in [1].

Historically the research on diff algorithms has been carried out by the database community, that has to deal with huge quantities of data and seeks to reduce space and time consumption. In fact, these algorithms have been evaluated mainly by comparing their time and space performance. Almost all the experiments in the literature follow the same pattern: the authors first compare the computational complexity and the execution time of the algorithms, then evaluate the quality of the results, see for instance [2–5].

The quality is often expressed in terms of the ability to reduce the size of the produced delta. As summarized in [6]: "quality is described by some minimality criteria [...] Minimality is important because it captures to some extent the semantics that a human would give when presented with the two versions".

Surprisingly only a few other quality measures have been defined and applied. There is now a growing interest in characterizing more precisely the quality of deltas, in order to design algorithms that produce an output that is *easier to interpret* and more adequate for human readers, for instance specialized for literary documents [7] or ontological data [8].

The focus of this paper is on comparing the quality of the deltas produced by diff algorithms.

We introduce a framework for measuring the quality of diffs through an objective evaluation process. The basic idea consists of extracting numerical indicators from deltas (such as the number of detected changes, the number of high-level changes, the number of elements listed in the description of each change) and aggregating them into more complex quantitative metrics. These indicators can be associated to quality requirements and evaluated to decide whether or not the algorithm that produced that delta is 'better' than others in a given context.

This work is built on top of UniDM [9,10], a unified conceptual model able to abstract the characteristics of deltas. Each diff algorithm uses its own strategy either in computing deltas or in serializing. Design choices are strictly dependent on the application domain and, very often, prescribed by the tools that are meant to apply deltas. Such a unified model, along with the evaluation metrics on top of it, gives users a powerful tool to analyze in a more precise way the behavior of the algorithms. It is also worth remarking that this model is general enough to deal with streams of text, lists, trees or graphs, so that the same evaluation process can be applied to heterogeneous algorithms and domains.

* Corresponding author at: Department of Computer Science and Engineering, University of Bologna, Mura Anteo Zamboni 7, 40126, Bologna, Italy. Tel.: +390 512 094 898; fax: +390 512 094 510.
*E-mail address:* angelo.diiorio@unibo.it (A. Di Iorio).

http://dx.doi.org/10.1016/j.csi.2015.12.005
0920-5489/© 2016 Elsevier B.V. All rights reserved.

The paper is structured as follows. Section 2 describes in more depth the solutions adopted to evaluate the quality of diff algorithms. Section 3 discusses how the same concept of 'quality' can have different meanings according to users' needs and preferences. Section 4 introduces our solution, that is detailed in each part in the following sections: Section 5 introduces UniDM (the model used to analyze the deltas) and some quantitative indicators, while Sections 6 describes the metrics in a formal way. The application of the metrics is presented in Section 7 where we introduce a two-phase method to evaluate the existing algorithms and we present experimental results on some well-known XML diff tools, before concluding in Section 8.

## 2. Related work

It is hard to compare the quality of the output of diff algorithms. First of all, because different algorithms might produce different deltas that are all correct. There is a further tricky issue. As highlighted by [3] "*all approaches make use of different delta models, which makes it difficult to measure the quality of the resulting deltas*". In fact, each algorithm uses its own internal model and recognizes its own set of changes.

The quality of a delta has often been associated to some kind of minimality; we show here a few alternative characterizations have been proposed and that we classified in two groups.

### 2.1. Quality as minimality

The most used parameter to measure the quality of diff algorithms is its ability to reduce the dimension of the delta. There are two main approaches to calculate such a dimension: either measuring the size of the file or counting the number of edits listed in the delta. The first approach has been used, for instance, in [11] and [6]. This evaluation can be fully automated and it makes it possible to compare directly heterogeneous deltas. It is, however, not precise, as it does not investigate the content of the file. The measurement of the number of edits, known as *edit distance*, was experimented in many other works, for instance in the evaluation of DocTreeDiff [3], Xandy [12], X-Diff [4] and XRel_Change_SQL [13].

A refinement of the first approach has been proposed for measuring the quality of Faxma [2]. The authors compared some algorithms by comparing the size of compressed deltas. The motivation is that: "*[the authors] expect to get results that are less dependent on the encoding and more closely related to the amount of actual information. The difference in output size due to some tools generating XML and others binary diffs should be mitigated by compression*". This approach reduces the noise generated by implementation choices of each algorithm.

The second approach, the edit distance approach, is more precise, though it is heavily influenced by the set of available operations. In fact, this model initially considered insertion and deletion operations only. Later, other edits have also been considered such as the substitution of elements' labels and names, for intermediate nodes (including their attributes) and for entire subtrees. The introduction of these complex operations required more flexible models for calculating the weights of each operation. The minimization of the edit distance was further refined using *edit cost models*. This is the solution proposed in the early days of tree-based diff algorithms by [14]. The idea is to define a cost for each type of change and to measure the overall cost of the delta as the sum of the costs of each detected change. In the same paper the authors introduce an algorithm that minimizes this cost.

### 2.2. Quality as interpretability

In other cases, the quality of a delta has been associated to the capability of humans to interpret and exploit the changes it contains. This quality is much more difficult to define, as it involves the nature of changes and the human analysis of the output.

For instance, in [7] the authors introduced the notion of naturalness of a diff algorithm. The naturalness indicates the "*capability of producing an edit script that an author would recognize as containing the changes she/he effectively performed when editing a document*". The authors presented a taxonomy of natural operations on literary documents and an algorithm, called JNDiff, able to capture (most of) those operations. The focus is on the quality of deltas in terms of readability and accuracy for human users, so that JNDiff is slower than other algorithms, but still acceptable.

The idea of looking for deltas that better describe operations on literary documents has also been investigated by DocTreeDiff [3]. In that paper, the authors sketched out an original approach to measure quality. They suggested to compare the mixture of changes listed in the deltas. The analysis is quite preliminary but shows a great variety of the types of changes detected by the algorithms. The ability of an algorithm to detect a larger and more precise set of changes is considered a good indicator of quality.

The identification of higher level changes that capture appropriately the editing process has also been studied for XML database schema evolution. In [15] the authors discuss a taxonomy of high-level changes and how each change can be expressed as combination of smaller units. As stated in the paper, however, their model is incomplete and does not cover all possible XML DTDs and schemas.

Other interesting ways of assessing the quality of deltas have been proposed in the area of ontology diffing. In [16] the authors discussed the need of a high-level set of changes that should be detected in order to produce deltas that are "*more intuitive, concise, closer to the intentions of the ontology editors*" and that "*capture more accurately the semantics of changes*". They proposed both a set of high-level changes, described in a formal way, and an algorithm to detect them. From authors' perspective, in fact, the presence of high-level changes in the delta increases its quality and effectiveness. In [8] the authors measured the quality of the deltas between ontologies as a combination of heterogeneous properties such as reversibility, size minimality and redundancy elimination. The authors introduced multiple differential functions to compute deltas and argued that the quality depends on types of information extraction and reasoning that are expected on changes.

The importance of letting users to tune the quality of a diff algorithm in relation to the set of detectable high-level changes was also stressed by [17]. The authors, in fact, introduced the idea of *viewpoints* (i.e., each ontology designer has her own needs and should be able to define the set of complex changes she is interested in) and proposed a language to describe complex changes, called CDL (Change Definition Language).

We agree that measuring the quality of a delta as a single fixed value is not enough. A better approach is to think of deltas as objects that have multiple measurable dimensions, each able to capture one facet of quality. There are in fact contrasting needs and expectations in characterizing such dimension.

## 3. Quality of deltas in different scenarios

Users in different domains have different requests and expectations on deltas and their quality. In this section we present some application scenarios and we explain how a more precise characterization of the quality of (the output of) diff algorithms would help users in selecting the most suitable solution for their purposes. The scenarios are identified with the labels S1 to S8 that will be used throughout the paper.

The diff algorithms are widely used by programmers. Different programmers, or even the same programmer in different moments, might have different needs. Consider, for instance, the following two scenarios:

S1: **Programmer developing code:** some verbosity is appreciated by developers who review source code during the development. It is often required to have contextual information that wraps the actual changes detected by a diff algorithm. Notice that this context is not strictly necessary but helps developers to understand what