Sirix - Beyond Versioning of Persistent Trees

Storing, Querying and Analysing Differences

Johannes Lichtenberger *

October 10, 2013

Abstract

Nowadays, disk space is cheap and versioning becomes increasingly interesting. Besides, the days of mechanical disks are numbered and flash drives will certainly predominate in the future. We propose a system which takes full advantage of fast random reads and sequential log-structured writes of flash drives to store fine grained temporal trees. Retaining not only the most recent version but also snapshots of the past facilitates the analysis of temporal patterns.

Keywords: Versioning, Git, SVN, Mercurial, Revision, Diffing, XML, JSON, XQuery, XPath, Storage, Visual Analytics

1 Introduction

Storage space follows Moore's Law and doubles approximately every two years. It becomes increasingly tempting to store not only the current version, but all past snapshots of data sets. Hence, it is possible to go back in time in the advent of any kind of failures or unintentional modifications. Furthermore analytical tasks to detect patterns in temporal data are supported, due to the availability of each stored snapshot. Moreover, semi-structured data in the form of tree-structures is very common either in the serialized form as XML-dialects or in the JSON format.

1.1 Problem Statement

By retaining old versions of fine grained trees we face the problem of having to cope with increasingly large data sets. Storing new versions of collections of whole trees is very expensive and inefficient. The storage layer thus has to focus on an effective and efficient page-to-block management for the on-disk storage. In order to support fast queries while not restricting the write-performance suitable versioning strategies have to be found. Besides, query languages as for instance *XPath* and its big brother *XQuery* have no notion of temporal aspects and thus have to be extended. In order to navigate not only in space but also in time, XPath has to be extended. XQuery on the other hand supports user-defined functions. It must be possible to open and serialize specific revisions, that is snapshots.

1.2 Contribution

2 Conclusion

3 Acknowledgements

*email: lichtenberger.johannes@gmail.com