Version Control in Collaborative Writing

B R Gaines
Tel: 403-220-5901
Fax: 403-284-4707
gaines@cpsc.ucalgary.ca

Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4

An architecture for automatic version control in collaborative writing is described which enables authors to operate independently using normal word processing facilities to edit the 'same' document, and supports them in understanding and managing what has happened when the document files are brought together. The paragraph and document level versioning mechanisms are described and illustrated through screen dumps and an application scenario.

Introduction

The increasing use of word processors directly by authors, together with the ease of interchange of documents electronically on disks or through networks, is supporting and promoting collaborative writing in which joint authors work simultaneously on an article. Conventional software version control techniques which rely on network control of access are inappropriate to a situation in which authors wish to work on personal computers not connected to the same network. Even the assumption that versions are checked in and out of central archives would be a major restriction. What is required is a version management system that is effective in an anarchic environment in which different authors edit different versions of the same document at the same time, and are then supported in understanding and managing what has happened when the document files are brought together on their word processors. Additionally, the interface to versioning features of the document production environment should be simple and minimal so that authors are not distracted from their primary task of writing.

This paper described the architecture of *GroupWrite*, a document processor with simple and effective version control at the document and paragraph level (Gaines and Malcolm 1993; Malcolm and Gaines 1991) .

Paragraph-level versioning

GroupWrite supports the tracking of two distinct levels of versioning: document versioning and paragraph versioning. Document versions are based on document files and a new version is created when a user opens an old version, edits it, and re-commits it to the filing system as a new version; paragraph changes are automatically tracked during editing, and the new version status of a changed paragraph is recorded when the document is re-committed.

The paragraph was chosen as the unit of granularity because it is both a natural conceptual unit for document producers (Halliday and Hasan 1976) and a basic data structure in word processors. Each paragraph in the archive file has a unique identification (ID) number, and a link to the paragraph from which it derived. This results in a paragraph derivation tree as shown in Figure 1.

0-7803-1936-2/94/$4.00

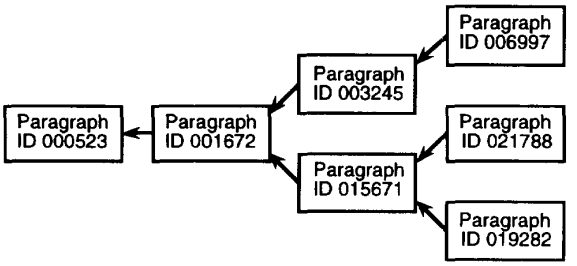

Figure 1 Paragraph derivation tree

A document version is identified by a user given name, and its structure is stored as a linked list of paragraphs together with a link to the document version from which it derived as shown in Figure 2. At the beginning of an editing or merging session a user selects a document version, designated the primary version. Additionally, other versions of a document may be designated secondary versions. Secondary version paragraphs that are alternatives of paragraphs in the primary version are available during editing as shown in Figure 3. They may be modified and incorporated into the primary document version. During editing each paragraph has an associated flag to indicate whether or not it has been modified.

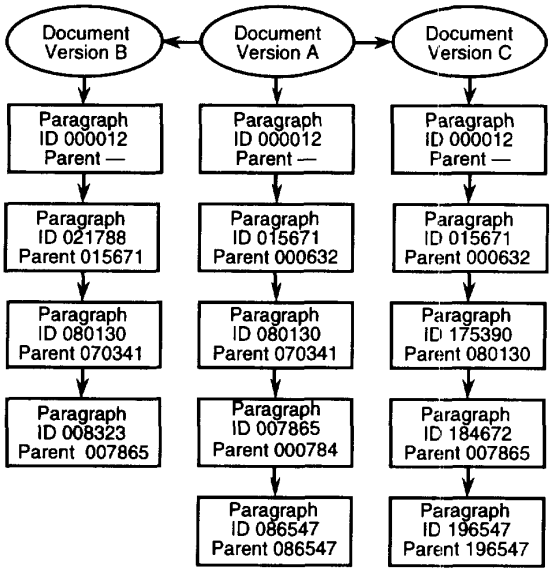

Figure 2 Document structures and derivation tree

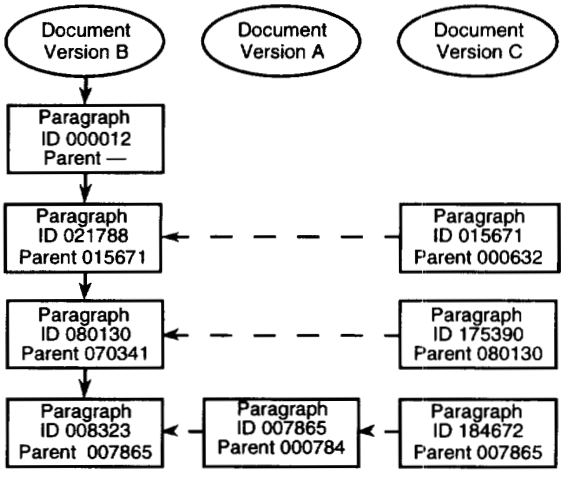

Figure 3 Alternative paragraphs

User interface design

A graphic browser showing relationships between document versions is provided as shown in Figure 4. This display is created dynamically by examination of the document version information in the files within the working directory when the browser is selected. Double-clicking on a document version opens it for editing as the primary document version, with any other selected versions being opened as secondary versions.

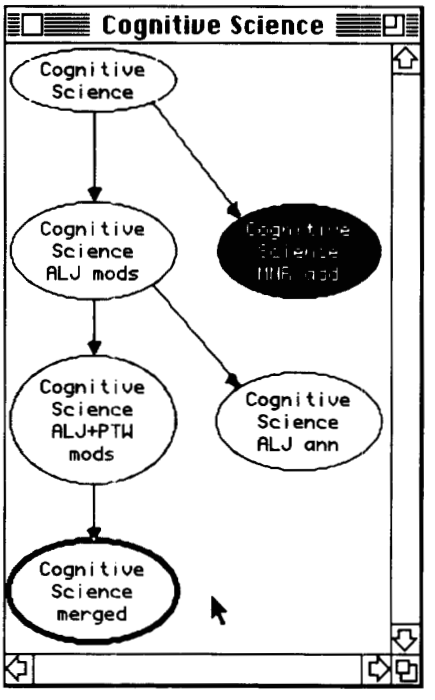

Figure 4 Document version browser

The primary document version appears in a normal edit window as shown in Figure 5 with lines in the left margin indicating the existence of alternative paragraphs or of annotations. The appearance of the line varies depending on what types of annotation are present, and whether alternative paragraphs are present. The mouse pointer changes shape when moved over the line, to indicate that a popup menu is available. This menu may also be accessed by holding down the option key while clicking the mouse in the marked paragraph. The popup menu provides a number of operations that may be performed on annotations and alternative paragraphs. They may be displayed in a separate window by the *Show* operation; inserted into the document text by the *Place Before*, *Place After*, or the *Replace* operations; removed by the *Remove* operation. Sound annotations may be heard by selecting the *Play* operation. Annotations and alternative paragraphs are selected from the sub menu, with annotations appearing below the dotted line.

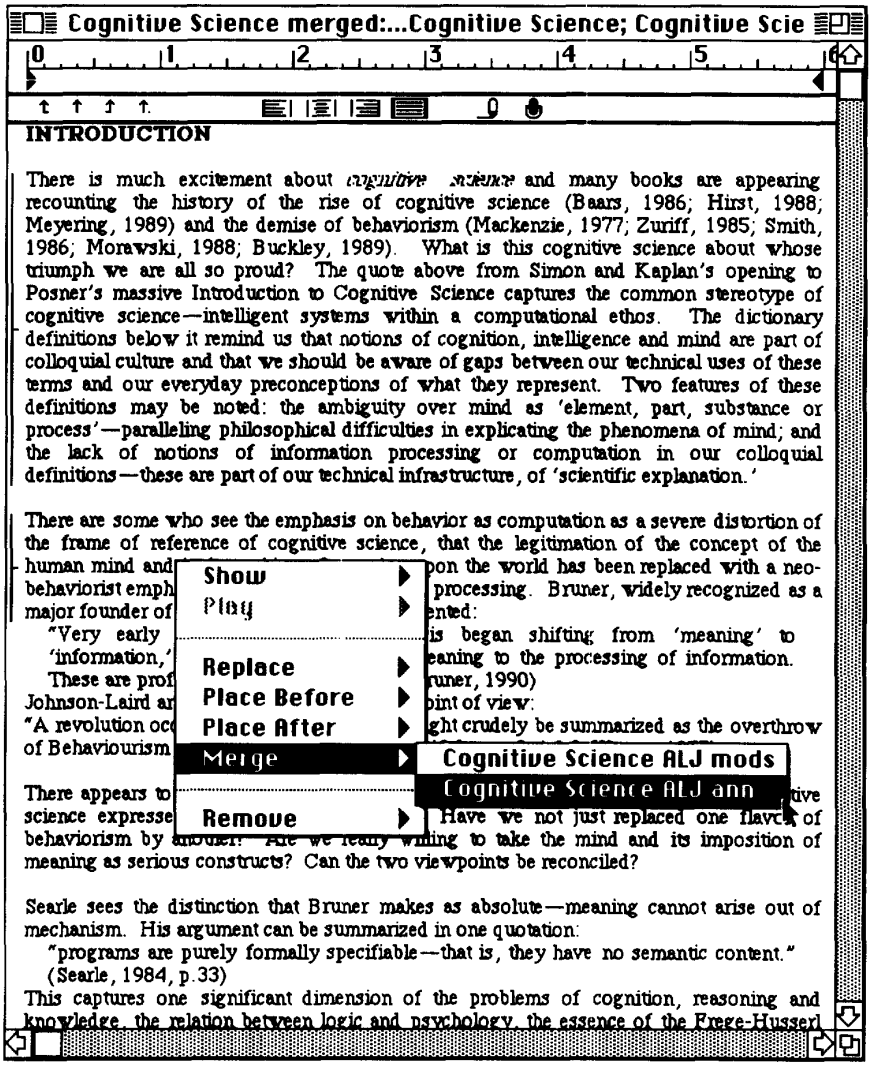

Figure 5 Alternative paragraph markers and the popup menu

Conclusions—a scenario of use

The versioning architecture described above provides a simple and natural environment for collaborative writing based on existing common practice in the use of personal computers. A scenario of use is:

1 Co-author A outlines a paper in draft form (V1) and sends a copy on disk through the mail to co-author B.

2 Co-author B edits and extends the paper (V1.1) and sends a copy through email to co-author C.

3 Co-author C edits and extends the paper (V1.1.1) and sends a copy back through email to co-author B.

4 Co-author A continues to work on the original paper (V1.2) and sends a copy on disk through the mail to co-author B.

5 Co-author B opens C's version (V1.1.1) in the context of his original (V1.1) and A's amended version (V1.2), and creates a merged version (V1.1.1.1) which he sends to both A and C with annotation asking them to make final changes.

6 Co-author B receives back A's final version (V1.1.1.1.1) and B's final version (V1.1.1.1.2), both annotated with requests for final checks and amendments, opens one in the context of the other, creates a merged version, checks all the annotations, opens the merged version in the context of all the previous versions and checks all variants for possible improvements, and then prints out the paper to be sent to a journal.

In this scenario, co-author B has acted as overall coordinator using the versioning and merging facilities, and both A and C have used GroupWrite as an ordinary word processor with annotation capabilities. Compared with current practice using ordinary word processors, B's task has been made very much easier because the changes being made by all three co-authors in parallel are being tracked automatically and made visible through *GroupWrite*.

References

Gaines, B.R. and N. Malcolm. "Supporting collaboration in digital journal production." *Journal of Organizational Computing* 3 (2 1993): 195-213.

Halliday, M.A.K. and R. Hasan. *Cohesion in English.* London, UK: Longmans, 1976.

Malcolm, N. and B.R. Gaines. "A minimalist approach to the development of a word processor supporting group writing activities." In *COCS'91: Proceedings of Conference on Organizational Computing Systems,* 147-152. ACM Press, 1991.