Accepted manuscript of

Nury Elisa, Spadini Elena, 2020. “From giant despair to a new heaven: the early years of automatic collation”. *it - Information Technology*, 62 (2). https://dx.doi.org/10.1515/itit-2019-0047

The manuscript follows the template made available by the journal. Some parts of the template have been struck out to avoid confusion.

From Giant Despair to a New Heaven: the Early Years of Automatic Collation

Elisa Nury, Elena Spadini

Abstract: This article presents a commented history of automatic collation, from the 1940s until the end of the twentieth century. We look at how the collation was progressively mechanized and automatized with algorithms, and how the issues raised throughout this period carry on into today's scholarship. In particular, we examine the inner workings of early collation algorithms and their different steps in relation to the formalization of the Gothenburg Model. The scholars working with automatic collation also offer fascinating insights to study the collaborations between Humanists and Computer Scientists, and the reception of computers by philologists.

ACM CCS: Theory of computation → Design and analysis of algorithms; Social and professional topics → Professional topics → History of computing

Keywords: Collation, Textual Criticism, Philology, Digital Humanities, Agorithms.

1 Introduction

> Whereas in the past, then, textual editors, finding the way rough, got over the stile into By-Path Meadow and, benighted and storm-beaten, were captured by Giant Despair, now the genial pressure of automation, like the sun in the fable, will cause us to throw off what before we hugged about us. – Dearing 1962, p. 3.

> Today, however, the textual editor sees a new heaven opening. [...] I tell you with absolute humility that I hear the morning stars sing together and all the sons of God shout for joy. – Dearing 1962, p. 34.

In the early days of Humanities Computing, textual scholars saw the benefits of using computers to facilitate parts of their work when editing a text. Collation was a well-suited candidate for automation, as a repetitive and often tedious task, during which the risk of making errors is high. Collation is the practice of comparing multiple versions of the same work, called witnesses, in search of the specific differences – the variants – which may shine a light on the history of the text, its genesis or its transmission.

Collation can be considered a form of intertextuality in the very large sense of the term: both involve text comparison to find either differences, or similarities, which inform scholars on the meaning of the text and influence its interpretation. In this article we argue that a history of automatic collation is relevant to understanding the numerous new developments in this area of philology.[1]

Automatic collation makes use of computers or other devices to identify the variants in the texts, and display them on the screen. The workflow of automatic collation, as practiced today, includes first a transcription of the witnesses' texts into machine-readable format, whether by hand or by OCR. Afterwards, these transcriptions are aligned where their texts match with the help of algorithms.

Since the 1940s, mechanical devices and later computer programs have been developed to assist scholars during collation; as a result, the name and definition of automatic collation have evolved.[2] One of the earliest definitions is given by Gilbert [13, p. 139]: 'computer-aided critical editions, i.e., the use of the computer to compare texts or manuscripts and to indicate variants'. Interestingly, Gilbert refers to critical editions created with the help of a computer, but the actual definition describes very precisely the act of collating – comparing texts and identifying variant readings. In fact, several of the early tools

[1] See for instance Spadini [50, chapter 5] and Nury [33] for a critical panorama of automatic collation and lists of tools: at least 25 programs exist or have existed, eight of which were created in the past ten years. Before going any further in this article, we would like to thank the anonymous reviewers and the colleagues that contributed to our work.

[2] For brevity, we will use the terms 'automatic collation', whether or not collation is fully automatic: we consider synonyms the expressions 'semi-automatic collation', 'computer-supported collation', 'machine-assisted collation'.

were part of a larger infrastructure designed for the whole editing process, from gathering and collating variant forms of a work, to printing and publishing the edited text and the critical apparatus [47, p. 138].[3] One of the most successful of these early tools is TUSTEP, created by Ott at the University of Tübingen in Germany: it has been constantly developed until today and has served to prepare countless critical editions[36]. TUSTEP is highly modular and flexible, with numerous subroutines for comparing texts, editing and typesetting, preparing various indexes. Collation was complex and therefore such an important part of these editing tools that, in the case of Gilbert's tool COLLATE [13] and Robinson's Collate [40], it gave its name to the entire suite of tools.

In the 1990s, tools such as URICA! and Collate give full control of the collation to the users, letting them interrupt the program and interact with it at will. Hilton [18, p. 140] distinguishes therefore between fully automated and interactive collation: 'interactive collation [...] provid[es] the computer with human assistance whenever necessary', and not vice-versa. Collation, which had been considered a mechanical task well suited to automation [11, 20], had turned out to be more complex than anticipated [21, p. 125]. Hence the creation of interactive tools, where user input helps to correctly align the text. In the 2000s, new names appear, such as computer-supported collation or semi-automatic collation, which still hint at the necessity or possibility of integrating user input in the collation process.

The purpose of the tools has evolved as well: from creating a full critical edition, the main focus shifted to textual alignment or simply 'text comparison' [9, p. 2]. This evolution is reflected in tools names such as TRAViz (Text Re-use Alignment Visualization) or iAligner [24, 58]. While tools increasingly specialize in alignment, their purpose has become more open to include text reuse as well as plagiarism detection or intertextuality [24, 58, 42].

In this contribution, we wish to investigate in particular the forty years of early automatic collation starting in the 1960s and ending in the late 1990s. This period, as we shall see, laid the foundations of the approaches taken today. Scholars designed and implemented a variety of solutions, some of which still innovative.

The chronological boundaries of our study are imposed on one side: we consider everything related to computer-assisted collation from the very beginning, going back to the decades preceding the advent of computers. At the other end, we venture only very briefly into the new century, because there are substantial changes in collation algorithms, determined by two factors: the influence of Bioinformatics and sequence analysis algorithms, seeking to achieve multiple, instead of pairwise, alignment [52][4]; and the emergence of the graph data-structure, a fitting model for textual variants (implemented by Schmidt and Colomb in the Multi-Version Document [45, 44]). Furthermore, some of the tools developed in the 2000s are not yet well documented or still under development.

2 Prehistory

2.1 Formal notations

In the first half of the twentieth century, it is possible to distinguish a series of attempts, not necessarily directly related, to systematize the procedures used in scholarly editing. These attempts focus in particular on the second stage of the process, the *recensio*, when witnesses are collated and are organized in a genealogical tree of textual transmission, i.e. a *stemma codicum*. A considerable amount of effort is devoted towards the creation of a formal, sometimes mathematical, notation to represent the tasks carried out during the *recensio*. Examples of these attempts can be found in [38, 56, 28].[5]

In the same period, John Manly and Edith Rickert were working on Chaucer's manuscripts, collating all the known witnesses of the *Canterbury Tales* [29].[6] Manly, head of the Department of English at the University of Chicago, was enlisted during the First World War as a code breaker in the U.S. Army, together with his collaborator Rickert. After the war, they returned to philology, to produce what became the classic edition of Chaucer's *Canterbury Tales*; the endeavour lasted their entire lives. Their manual procedure for collation involved a card for each variant site: the lemma from the base text would be copied at the top of the card, followed by the variants in all witnesses and the indication of missing lines, sheets or sections.[7] As Dearing pointed out, Manly and Rickert contributed to the automation

[3] For example, Dearing presented a quite complex system, which consists of five programs for collating, proofreading, making word lists and create family trees. Gilbert's program comprised ten modules, from creating searchable files with the witnesses' text to printing the critical text and apparatus.

[4] The collaboration between biologists and philologists dates back to the early 90s [34, 35].

[5] See Greg [56, p. vi]: 'I wish at the outset to make it clear that there is nothing esoteric or mysterious about my so-called Calculus : it aims at nothing but defining and making precise for formal use the logical rules which textual critics have always applied. It is quite incapable of producing any results that could not have been attained by the traditional methods; only it aims at achieving them with less labour and greater certainty. Perhaps its chief merit – if it has any at all – will be found in the endeavour to give precision to terms and modes of inference which are frequently employed with quite astonishing looseness.'

[6] Vinton Dearing, i.e. 'the father of computer collation' [17, p. 46], in his overview of the automation of collation [7] mentions both Greg's *Calculus* and the Hinman Collator (which is described in the next section), and sets the roots in the work of Manly and Rickert.

[7] Manly and Rickert [29, vol. 2, pp. 3-12]. The procedure is recalled by Dearing [7, p. 16]; and Harris [17, p. 11]. For the presence of philologists among the code-breakers during WWI, see [25, pp. 351-354].

of collation, in the sense of 'systematizing and speeding routine procedures [...] with their printed and tabbed cards for recording variants' [7, p. 3].

2.2 Optical Collation

At the end of the 1940s Charlton Hinman, a Shakespearean scholar, invented a tool for mechanizing collation: he would apply it to compare different versions of the First Folio of Shakespeare's plays [19, 49, 26]. The Hinman Collator consisted of two projectors, presenting on the same screen the images of two witnesses. The superimposition of the two images produced oscillations and sparkles where the images diverge, and differences can be spotted easily. The technique was called 'mechanical collation' or 'optical collation'. The former, 'mechanical collation', was not exactly appropriate, since the collation performed with this machine was more optical than mechanical.

The Hinman Collator was designed to compare only printed books, and moreover copies printed from the same edition, with no variation in size, curvature distortion, nor typesetting [15]. Even if these conditions are satisfied, difficulties may arise: for instance, the collator would often block the light from the projectors while getting closer and trying to spot minute differences [49, p. 141].

In the next twenty years, other machines were created in addition to the Hinman Collator, such as the Levin Collator, the Dearing Mark IV, the Smith March VII, the Lindstrand Mark I Comparator [27], the Hailey's Comet and the McLeod's Portable Collator [49]. The optical collation method with analogue devices was popular until the end of the 1960s, at which point automatic collation with alignment algorithms started to grow in importance and became the preferred collation method. Nevertheless, optical collation could never be discarded completely. The preparation of accurate transcriptions is difficult enough that, for pages with the same typesetting, optical collation remains a valid option. To this aim, several tools for optical collation with a computer have been created in the 2010s, such as the Oxford Traherne Digital Collator, or Paragon at the University of South Carolina.[8]

Although in optical collation the process is made easier by the help of a device, it is still performed by a scholar who will manually align the images of the pages and locate variations in the text. For this reason, we do not consider optical collation as an example of automatic collation.[9]

[8] `https://web.archive.org/web/20191025101401/https://oxfordtraherne.org/traherne-digital-collator/` (Accessed Oct. 2019) and `https://web.archive.org/save/https://sc.edu/about/centers/digital_humanities/projects/paragon.php` (Accessed Oct. 2019).

[9] For a different opinion, see Dearing [7, p. 3], who considered optical collation as part of automated collation because it made use of a mechanical device to speed up the process.

3 Collation Algorithms

3.1 Basic Principles

Collation algorithms were first developed in the early 1960s, by Dearing [7] and Froger [10, 11]. While a scholar using mechanical collation would physically align two images of a page by superimposing them, the alignment is now achieved by an algorithm comparing strings of characters. The preliminary requirement is of course to have machine-readable transcriptions on punched cards, magnetic tapes, floppy or hard disks.

Early algorithms would imitate very closely what a human scholar would do: 'the machine is used to simulate a series of human operations' [20, p. 145]. One text is chosen as a base for comparison, the base text, and all other witnesses' texts are compared one by one to this base text. The texts are compared word by word until a difference is found, and the algorithm must find where the two texts correspond again. To do so, the algorithms would go through the possible categories of change until it discovered what makes the difference: an addition, omission, substitution or transposition of words.

Similar edit operations quantify the distance between two strings of characters, and are used by diff algorithms.[10] Although automatic collation can be likened to diff, as we examine collation algorithms more closely, we will highlight the most important differences between the two.

In the last decades, the main steps of a collation algorithm have been identified and separated in a series of easier sub-tasks. In 2009, a group of scholars specializing in automatic collation gathered in Gothenburg and discussed strategies to improve automatic collation. The resulting 'Gothenburg model' divides collation into three major stages: tokenization, alignment and visualization, with two optional steps for normalization and analysis or feedback [9].

3.2 Tokenization

In the Gothenburg model, the first task to accomplish is to split the text of each witness into smaller atomic units called tokens. In Froger's case, to take an early example, a token is a word or a white space. Other programs might consider the punctuation marks as tokens as well, while some attach punctuation marks to the previous token instead [4, p. 471].

The text can be divided at different levels, such as character, word, sentence, paragraph. In theory, there is no requirement to use a particular level of tokenization, but in practice, tokens at the word level have turned the

[10] Diff algorithms are used to compare two files – two electronic documents – line by line, and to signal the lines where the two files diverge.

most convenient for editors. Petty and Gibson attempted to tokenize at the sentence level, but with limited success. Punctuation marks can appear inside a sentence (e.g. 'Mrs.'), and there are many combinations of characters that can indicate the end of a sentence, which makes it complex to identify sentence boundaries.[11]

In case the tokens differ, it becomes a unit of variation.[12] Their size corresponds to the level at which the variation is spot: if the token is a word, the program will indicate the words that differ in the different witnesses, and not the character or the phrase, for example. In the tokenization phase, collation algorithms differ substantially from, for example, diff algorithms, which work at the character level, since they might adjust various length and complexity of tokens.

3.3 Alignment

The alignment is the core of a collation program. All the programs under analysis here, and in general most collation programs perform pairwise alignments.[13] They compare two texts at a time and, in case more than two witnesses are to be aligned, merge the results of the pairwise comparisons to obtain the global result, that is the alignment between all the witnesses. This method has its limitations, first of which the dependency of the results on the order of the witnesses given to the machine for comparison; but it is also exponentially easier than multiple alignment [52, 51]. The ambition to collate multiple texts is another element for distinguishing diff algorithms, limited to pairwise comparison, to collation algorithm, that potentially aim at overcoming this situation. Nevertheless, since we only consider collation algorithms up to the 1990s, in what follows we can assume that witnesses are aligned in pairs.

During the alignment, the two texts (A and B) are compared token by token. A match happens when the token of text A (A1) is equal to the token of text B (B1). The texts can be compared starting from the beginning and moving forward, or from the end and moving backward, for instance starting at the end of a verse, to isolate the variants in the middle [8] or to identify corresponding portions of the texts [48].

When two tokens do not match, the program attempts to broaden the portion of text that the machine scans for finding matches, referred to as sliding window or context. Froger's program, for example, will first of all invert the order of two tokens, comparing A1 with B2 and B1 with A2. If no correspondence is found, five tokens are looked up in both texts, testing all possible matches, as in [59]; if this attempt also fails, twenty-five tokens are considered [11]. Gilbert's program, in the 1970s, would compare tokens first one by one, and then two to twenty, four to fifty and nine to one hundred [13, 14]. The length of the sliding window depends on the capacity of the computer memory: the maximum size of the window was only 5 words for Zarri [59], and could go up to 300 words for Petty and Gibson [12] or sixty lines of text for Silva and Love [48], who regretted the imposed limitation.

Several problems were identified with the sliding window technique: for instance, the algorithms would fail whenever a variant was longer than the maximum window size [45]. As soon as computing capacities increased, during the late 70's and 80's, algorithms became able to collate entire texts, without the need to use sliding windows.

3.4 Macro-alignment

The sliding window technique was used not only during the tokens alignment, but also in a preliminary phase that some algorithms included, and which we may call 'pre-alignment' or 'macro-alignment'. It consists in finding the corresponding portions of the texts, be they lines or paragraphs, before starting the actual comparison token by token. This is exactly what happens in the program to collate poetry devised by Silva and Love [48], already mentioned, which consists of two main routines, MATCH and EALV: the first finds corresponding verses and the second prints out the differences in each of them at character level.

In particular, the MATCH routine starts comparing the fifth lines of each text: if those match, the previous four lines are assumed to correspond; otherwise, a line by line collation is performed. When no match is found, the line is compared to forty lines of the second text, twenty before and twenty after the corresponding verse position. Two lines correspond if the first or last seven characters are equal. The program is also able to identify 'interchanged passages', that is lines which occur in different places in the texts.

A similar mechanism is the one used by Gibson and Petty [12], working on prose texts. As we have seen above, since they had difficulties with the sentence tokenization, they decided to align arbitrary portions of twelve words, before performing a more fine-grained alignment. In this first phase, twelve words of text A are compared to three hundred words in text B. Twenty years later, Stringer and Vilberg [53] used a mechanism analogous to those just described for collating poetry.

[11] Sentence segmentation tools are available nowadays for some languages. However, since most often collation is used to spot difference at the word or even character level, there are not many examples of sentence level tokenization in collation pipelines. See also 3.4 below.

[12] This is not the same as the philological 'lieu variant', which might be made of various units of variation, continuous or not, grouped together. See [10, p. 157]: 'La machine, qui procède autrement que le philologue pour découvrir les variantes, a aussi une façon différente de les présenter : ses ≪lieux variants ≫ne sont pas ceux du philologue'.

[13] With the exception of CollateX [9], for instance, or more recently LERA and LAKomp. See https://lera.uzi.uni-halle.de/ (Accessed Jan. 2020) and https://lakomp.uzi.uni-halle.de/ (Accessed Jan. 2020).

A different way to approach the macro-alignment is presented by Raben in his contribution to the 1979 edited volume *La pratique des ordinateurs dans la critique des textes.*[14] Raben aims at identifying corresponding chapters, before going into the details of the alignment. For doing so, the program compares the vocabulary of big portions of the texts, using a mechanism similar to those used at the time for authorial attribution [39, p. 260].

3.5 Normalization

Another difference between diff and collation algorithms, in addition to the multiple alignment and the flexible tokenization parameters in the second, is the need for handling text normalization. A form of normalization is needed when the editors are not interested in the results of an exact match, but wish to neutralize certain differences: orthographic variants are a widespread case of variants that are not deemed relevant by editors.

In order to obtain the expected results, multiple approaches are possible. Normalization can be performed directly on the transcriptions, possibly in the encoding, before the alignment. In Froger's transcriptions, for example, punctuation and accents are not registered [11], because of hardware constraints. However, normalizing during transcription causes a loss of information which may turn out to be significant and that cannot be retrieved later [59, 40]. In other cases, the algorithm is instructed to ignore a number of characters or symbols. This is what happens in the algorithm devised by Silva and Love [48, p. 93], which eliminates the signs leading to the most common types of variation, including 'letters e and s, the S-sign,[15] blanks, and all punctuation marks from both lines'.

In Gilbert [14], the STRIP parameter is designed for the editor to specify which punctuation marks and special symbols the program should ignore. A similar mechanism is used by Cabaniss with the exclusion list [2]. It is not indicated whether in these programs the signs discarded for the alignment are kept in memory and restored in the final output.

An innovative and versatile approach to normalization is introduced by Collate, a program constantly developed during twenty years by Robinson and numerous collaborators, in use in many projects until very recently. Collate was born to collate manuscripts of a medieval Icelandic text, whose orthography vary greatly. The problem of orthographic variation is therefore evident to its creator and it is tackled in two ways: a normalization phase, prior to collation, and a 'fuzzy match' during the alignment. In the normalization phase, an arbitrary normalized form is provided for some of the original tokens: for example, the normalized forms of both *color* and *colour* might be CLR. Furthermore, the application function defvars offers a way to indicate that two forms correspond: Robinson gives the example of *ok* and *en*, both meaning 'and' in Old Islandic [40, p. 103]. During the alignment, the fuzzy match finds matches for non identical, but graphically similar, tokens.

The similarity in the fuzzy match is based on a threshold. Some of the programs under analysis here allow the user to define values for parameters such as the fuzzy match threshold. An early example is the program devised by Cabaniss in 1970 [2], in which the users have an active role: they do not only prepare the text, but also configure a number of parameters which will be central in the process, among which the length of the match (MAX_MATCH), the length of the sliding window for slightly dissimilar texts (MAX_WORDS) and for highly dissimilar texts (MAX_LINES), and the 'degree of coarseness' for the latter (MAX_SCAN).[16] As shown in the article, changing these settings will result in considerably different outputs and suggestions on how to set them are offered across the article; the program also provides default values for them.

3.6 Base Text

Until the 1990s, all algorithms operate with a base text. It may be either the text of a previous edition, or the text of one witness, chosen because it is complete, or without major lacunae and minimal corrections by later scribes [10, p. 139], or finally it may be an artificial creation: an existing text modified to optimize the collation output and make the results easier to interpret for the editor. As Robinson puts it: 'the final [base text] was worthless as a text – but it provided a splendid series of pegs on which the variants might hang' [40, p. 102]. However, this base text was not without problems.

First of all, if a a part of the text is absent from the base text, this means that similarities between other witnesses will not be noted at this point in the text. For Robinson, another issue is that the base text was the main point of access to the text and to the variant readings for readers, who could become biased in favour of an artificially constructed text [41]. Moreover, the base text was seriously limiting the display possibilities: the collation can only be displayed in relation to the base text, and not in relation to a witness chosen by the user. The last problem is that the collation output could not

[14] This seminal volume, edited by Irigoin and Zarri, contains papers about automatic collation by Gilbert, Raben and Waite, in addition to many others on related Digital Philology topics.

[15] For neutralizing differences such as lived versus liv'd and capitalization, which is indicated by the S-sign.

[16] 'Before this program is run, certain definitions have to be made as to the nature of the processing. The following questions must be answered in quantitative terms: 1. What is a variant? 2. Once a variant is found, what constitutes a match? 3. How long should a word by word search be made in looking for a difference of 'medium size'? 4. What is the maximum amount of text that should be searched in a 'large difference'? 5. What degree of coarseness should be used in scanning the texts for a match within a 'large difference' [2, p. 7].

be reused for phylogenetic purposes, i.e. the creation of the *stemma*: the variants were grouped according to the tokens of the base text, which resulted in overlapping variation (see Figure 1); while for phylogenetic analysis it is fundamental that the collation results do not overlap.

Towards the end of the 1990s, a major change was brought by a collaboration between Robinson, the Institute for New Testament Research in Münster (Institut für neutestamentliche Textforschung), and the Cambridge University Department of Molecular Biology. They introduced the parallel segmentation method, which groups variants according to the longest variant present at any point in the witnesses, avoiding overlapping.[17] Figures 1 and 2 illustrate the difference between the former apparatus and the new parallel segmentation. Although this is out of scope of the present paper, later on Robinson identifies limitations of the parallel segmentation method and seeks 'a collation system which does not stop at the point where it has identified the parallel segments, but actually carries on within the segments, seeking to link them at a finer level of detail' [41, par. 4].[18]

hors] *37 witnesses*; horsbak *1 witness*
hors and faire] a goode hors and *1 witness*
and] *33 witnesses*; and therto *5 witnesses*
faire] *36 witnesses*; wel koude he *1 witness*; faire therto *1 witness*

Figure 1: Collate's critical apparatus [41].

hors and faire] hors and faire *29 witnesses*
hors and therto faire *5 witnesses*
a goode hors and *1 witness*
hors and wel koude he *1 witness*
hors and faire therto *1 witness*
horsbak and faire *1 witness*

Figure 2: The parallel segmentation [41].

3.7 Evaluation

Before moving away from collation algorithms, some words are in order about their evaluation. In 1976 an article by Robert L. Cannon [3], a computer scientist, evaluates the efficiency of the early available algorithms (Petty and Gibson, Cabaniss, Gilbert), calculating the amount of comparisons between the words of text A and of text B needed by the program. The algorithms Cannon takes into account have been created for processing prose texts, where, as mentioned, there is a difficulty due to the absence of reference points in the match query: when text A and B differ, a variant begins; but where does it end? In lyric texts, the beginning of the new line offers a possible reference for looking for a match, while in prose texts different strategies are needed. The algorithms analysed by Cannon, as said, had more than one step, in which various sliding windows are used: overall, for aligning a text A of 100 words with a text B with 100 words, the number of comparisons effectuated by the algorithms are:

- Gilbert: 83527
- Cabaniss: 1699
- Petty and Gibson: 17424

The comparisons needed for finding the alignment are equal to the length of text A multiplied by the length of text B: for a text of 100 words (text A) compared with another one of 100 words (text B), the number of the comparisons would amount to 10000 (100 x 100). On this basis, Cannon concluded that Cabaniss' algorithm uses information that is not sufficient for aligning the texts, while the one by Gilbert is redundant and makes several times the same comparisons.

Cannon proposed a new algorithm called Optimal Collation (OPCOL), which stores tokens in tables, in order to avoid needless comparisons. OPCOL is 'optimal' in the sense that it is not possible to collate two texts with a lower number of comparisons, and still produce a valid output in any situation. In OPCOL, when a match fails and a variant begins, words are progressively added and the edit distance calculated; when the distance decreases, instead of increasing, the variant ends. The program, in short, builds a table populated with the values of the edit distance for each comparison and infers the alignment from it, using numbers, which can be more quickly processed than strings of characters.

The approach of Cannon is focusing on the number of operations performed by the algorithms, but Humanists tended to evaluate the output in terms of quality of the results and the accuracy of the alignment, while the speed and overall efficiency of the algorithm is deemed less important [12, 13, 20]. Evaluating the quality of a collation algorithm output is still a challenge, since we need already aligned texts in machine-readable format to serve as benchmark and there are not yet many available [9].

3.8 Visualization

The tools under consideration in this article were almost all created or in use before the spread of the internet

[17] The same principle was used by Manly and Rickert to record variants on cards, one lemma per card: 'The lemma is by preference a single word or a simple phrase, but it is sometimes necessarily longer [...]. It is chiefly wild variation in word order [...] that necessitate these long lemmata' [29, vol. 2, p. 7].

[18] Robinson's wish here is to identify the 'lieu variant' in the common philological sense, see footnote 12 above.

and of digital editions: the prevalent result of the editorial work was on paper and included a critical text and a critical apparatus. This is why most of the software analysed have an output close to a printed apparatus, which can be reused in the publication.

The visualization that is probably most popular today, the variant table, inherited from the matrix in use in bioinformatics for sequence alignment, would only appear in the twenty-first century. The variant graph is another visualization, which is particularly useful to display transpositions. As said in the introduction, the emergence of the graph data-structure is one of the factors causing a substantial change in automatic collation in the new century. While its theorization is due to Schmidt and Colomb [44], and is later used in CollateX, the graph has been used as a form of visualization long before and then forgotten. Its origin can be traced back to 1964 in an article of biblical textual criticism by Colwell and Tune [6]. They have suggested that variants should be grouped into variation-units, and expected that their model could easily be adapted for electronic manipulations.[19] Figures 3 and 4 show the variant graph, from its first schematization to the more recent CollateX visualization.

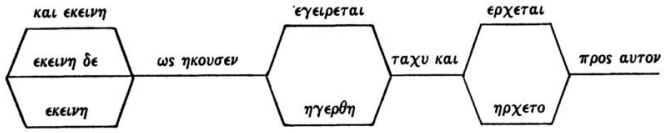

Figure 3: Colwell and Tune's early variant graph [6].

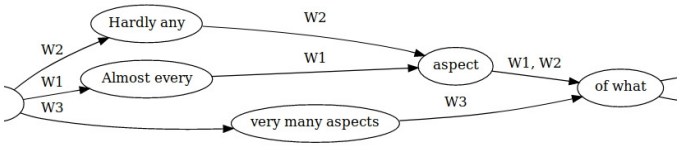

Figure 4: CollateX variant graph.

4 Humans and Machines

4.1 Human Interactions

Scholars who worked on automatic collation did not only describe alignment algorithms; they also commented on more social and economic dimensions of their research. Therefore, studying the history of automated collation offers insights into a Digital Humanities community, and the relationships between the various groups of people involved: Humanities researchers, computer scientists, and students. Their interactions highlight several friction points still relevant today. The relationship between human and machine plays a role as well, and scholars did not always react positively to the introduction of computers into their workflow. Dearing [7, p. 28] states that scholars have three options to acquire a collation program: obtain it from others, get an expert's help, or learn to code. In the last two cases, this means active collaboration with a computer scientist.

The collaboration may occasionally lead to communication problems, 'one of the most trying difficulties' for Gibson and Petty, whose scholarly decisions were 'circumvented when they were translated to machine code' [12, pp. 280-281]. As a result, they decided to code themselves the next program, but coding for humanists is fraught with challenges and frustrations: 'The actual coding of the program took about three months of trial and error, in which the machine repeatedly tried to explain how silly the instructions it had been given were' [12, p. 288]. Dearing also recounts that he 'wiped out' parts of a program which he wanted to modify, but did not fully understand [8, p. 260].

Most often, the programmers were mentioned by name in publications, and sometimes thanked [7, 10, 12, 57, 13, 53]. However, their remuneration and the recognition of their work as a part of digital textual scholarship was not assured.[20] Dearing admits that he 'paid [Mr. Bandat] a lump sum on delivery, but it did not fully compensate him for his time' and explains that the costs of a programmer's salary may be 'offset by their tendency to do more than what they are paid for' [8, pp. 260-261].[21]

While computer scientists were sometimes viewed as a hindrance to scholarship, for instance by Petty and Gibson, some of them, like Vilberg, participated in writing both the program and the scholarly article [53]. In between those positions, it is difficult to judge how much programmers contributed to the scholarly output. Students did also take part into the collation, usually doing the tedious work under a scholar's supervision [46, p.

[19] Another attempt at a graph was proposed by Sperberg-McQueen (1989), using the metaphor of a river's delta to illustrate how texts can separate at a variant location and then merge back together like a stream.

[20] The recognition of code as scholarship is nevertheless gaining attention: criteria have been proposed to evaluate the scholarly quality of a piece of code [54]; standards are developed in order to quote software in bibliographies, Such as Codemeta (`https://web.archive.org/web/20191115075258/https://codemeta.github.io/`, Accessed Nov. 2019) or the Citation File Format (`https://web.archive.org/web/20190924225247/https://citation-file-format.github.io/`, Accessed Sept. 2019); finally specialized journals such as RIDE will publish this year its first issue on tools for textual scholarship (`https://web.archive.org/web/20191015161704/https://ride.i-d-e.de`, Accessed Oct. 2019).

[21] Mr. Bandat did, as Dearing recounts, even work in his spare time on the project [8, p. 261].

448-449]. They work part-time and are known to make mistakes [12, p. 280], but it is worth noting that they may be paid. Widmann [57], at the Department of English in the University of Pennsylvania, paid student assistants 2$ an hour, which was more than the minimum wage of 1.60$ at the time.[22]

4.2 Humans and Machines Interactions

New attention to the man-machine relation, and in this case, the editor-computer one, characterized the development of URICA! by Cannon and Oakmann [4]. URICA! is the acronym of User Response Interactive Collation Assistant, a program in which the user has control over the entire procedure. The program is expressly designed for microcomputers, more accessible for single researchers and in Humanities departments. The two texts, the base one and that to be compared, are both visualized in a window on the screen, which is an innovative and successful solution. When the machine finds a difference, the process stops and asks to the user to which category the variant belongs (addition, deletion, substitution) and if the alignment should re-start; variants are then recorded in a specific file.

The quote at the beginning of Cannon and Oakman's article is emblematic: 'T. H. Howard-Hill had envisioned such a collation procedure of man-machine interaction in 1973 before the era of widespread personal computers. He foresaw 'a close flexible relationship between the editor and the computer where the first does what he is good at (perceiving and evaluating the significance of complex differences and making judgments), the second does what it was designed for (speedy manipulation of large quantifies of data, retention and accurate copying of data sets, creation and updating of extensive records) and the functions of each complement the other" [4, p. 469].

The introduction of computers into textual scholarship was viewed enthusiastically by the creators of collation tools. The computer works 'swiftly, efficiently, and tirelessly' [57, p. 63], it is alternatively a 'high grade clerical assistant' (or 'heedful slave') [57, p. 63] and 'a more industrious and attentive committee of editors' [39]; furthermore, 'in performing these mechanical tasks, the computer never slips up nor grows weary' [53, p. 85]. Robinson's opinion is well known and often quoted: 'The collation of manuscripts requires the infuriating accuracy of a pedant and the obsessive stamina of an idiot. It is therefore an ideal task for a computer' [40, p. 99].

The accuracy of the machine, when correctly instructed, is also appreciated: 'A computer operates at the speed of light and never makes a mistake' [8, p. 255]; methodologically, the absence of interpretation – the neutrality – during the collation that only a machine can ensure, is valued by Zarri, who recalls the desiderata of Maas [59]. In fact, as we hope to have shown in this article, a degree of interpretation is always present in the choices operated while collating, not to mention those made during preliminary transcription and encoding of the texts; but the machine makes sure that the same criteria are applied consistently during the entire process and that the judgement of the editors does not intervene randomly.

Despite these praises, in the same period, and still today, computers might also be met with 'indifference, fear, and even antagonism' [39, p. 258]. We can recognize several reasons for this 'mechanophobia'. Among them, there is 'the unconscious fear of the supposed ties of formalism' [30, p. 608], that is the concern of viewing a complex argument or object of study such as a textual criticism reduced to information in cells, to 'counting and alphabetizing' [39, p. 258]. Another aspect of the resistance to computers is the fear of the scholars to be replaced by the machine, a sentiment not only limited to editors of critical editions. Finally, the refusal might be directed to the amount of new skills to acquire in order to manage the computational aspects of the research, which comes in addition to the considerable knowledge required in philology.

These negative judgements on machine-assisted workflows are important to understand the role often assigned to computers, and to automatic collation in particular, in scholarly editing. Collation might be seen as a tedious preliminary activity to textual criticism, which 'truly begins only once all variants have been gathered' [47, p. 140]. As shown here, on the contrary, we consider collation a complex task, in which a good amount of interpretation and scientific choices are at stake: deciding which manuscripts to compare and how, or whether or not to record a difference, will affect the establishment of the text and are an integral part of textual criticism.

5 Latest developments

The developments of the last two decades will be very briefly addresses here, in relation to those of the last century. Between 2000 and the time of this article, major innovations were introduced in the field, such as the adoption of the variant graph or the formalization of automatic collation in the Gothenburg model. Two of the most well known collation tools, Juxta and CollateX were created, as well as other promising ones. Notable changes that happened during this period concern in particular algorithms, visualizations and workflows.

For what regards the algorithms, document-to-document matrices have been applied to sequence alignment, for instance in CollateX. As said in the beginning of this article, bioinformatics algorithms for mul-

[22] Detailed information about the cost of computer time (renting machines), student assistants, professional keypunchers can be found in the papers mentioned; see, in particular, Widmann [57, p. 59, note 1] and Gibson and Petty [12, p. 291]. Dearing explains as well that assistants' wages increase with experience [8, p. 260].

tiple alignment, mostly progressive, are now in use in order to compare more than two witnesses at a time: this is the case for CollateX, LERA and LAKomp.

As we have seen above, the purpose of the tools moved away from printing a critical text and apparatus, which encouraged the development of new visualizations. Such visualizations include for instance the histogram that shows a distribution of the variation in the entire text. The histogram was implemented in the Cervantes Project [32] and Juxta.[23] The variant graph may as well be displayed by some tools such as TRAViz and the CollateX online demo.[24] The heatmap is another visualization in Juxta that displays a single base text, and highlights variant locations in varying shades of blue: the darker the shade, the more witnesses disagree with the base text.

Despite the emergence of innovative visualizations, the need for a print output of collation is not outmoded, on the contrary print and digital have to be integrated. The question of the support, paper or electronic, of the output is exemplary of the variety of workflows that can be adopted, today as in the past. A tension is still present between the all-in-one solution of environments such as Tustep and the pipeline approach consisting in concatenating independent pieces of software.[25] The importance of the modularity of the software architecture, equivalent to the principle of the separation of concerns in nowadays computing, had been highlighted already in the past [13, p. 144], [55, p. 244]. In the choice of the workflow, the data model is also to be considered: the widespread availability of texts encoded following the Guidelines of the Text Encoding Initiative (TEI) calls for XML collation [1, 43] and inspires new textual models [16].

6 Conclusion

The early days of automatic collation were certainly a moment of great hope and creativity. Some of the scholars who contributed to the history of the field, outlined here, insisted on the innovative character of their enterprises. Gibson and Petty, for example, believe 'OCCULT is revolutionary' [12, p. 279]. Dearing offers celestial metaphors and Widmann considers that 'the uniting of human efforts to machine capabilities [...] makes us see that there is indeed something remarkable left beneath the visiting moon' [57, p. 63]. But it is probably Raben who reflects the most on the methodological implications of the use of computers for collation.

Raben's methodological considerations directly shaped his conception of automatic collation and how it should be performed. This approach is still in vogue today and contributes to innovations in the field of Digital Philology: the primary purpose of computers is not to be a simple secretary, automatizing and expediting the operations carried out by the editor, but to stimulate a critical re-thinking of the methodology.[26] Raben uses the metaphor of looking for a needle in a haystack: for isolating the needles, one can 'examine each long thin bit in turn, establish that it is either hay or steel' (p. 256), or using a magnet; the most revolutionary change is not inventing a machine that check each bit of hay to make sure it is not a needle, but using the magnet.

On the same line, we find Robinson, author of one of the most influential programs for automatic collation: 'along the way, I learnt several computer languages and found myself re-thinking some of the fundamental notions of textual criticism' [40, p. 99]; and Ott, the creator of TUSTEP: 'To sum up: by means of this new tool, which we have in electronic data processing, new and higher standards are imposed not only on the results of others sciences, but also on critical editions [...]. The question whether it is possible or not to save time and / or money by these methods is only of secondary importance. The expenses necessary for future critical editions may possibly be even higher than they have been in the past when these tools were not yet available' [36, p. 222]. In the field of New Testament studies, for example, automatic collation is now the preferred method of edition, and the computer is used to achieve a new understanding of the New Testament textual tradition which would not have been possible otherwise [37, 23].

We hope to have shown in this paper that it is worth going back in time and thoroughly examine what has been achieved in terms of theoretical reflections and code. Some of the problems, and even some of the solutions, addressed in the scholarship that made the history of automatic collation is still relevant today. The importance to look at 'words' beyond the level of the graphic string, for example, and to deal effectively with variant spellings, multiple manuscripts, and lemmatization –mentioned in Hockey's analysis of Digital Humanities early days [22]– is still valid. The same applies to the necessity of having texts in digital formats, first requi-

[23] See `http://juxtacommons.org/` (Accessed Jan. 2020). CATview can also be mentioned, although it is not a collation tool but a visualization widget for synoptic text views: `https://catview.uzi.uni-halle.de/` (Accessed Jan. 2020).

[24] `https://collatex.net/demo/` (Accessed Jan. 2020). See also Stemmaweb for a variant graph visualisation: `https://stemmaweb.net/` (Accessed Jan. 2020).

[25] For the second approach, see [9] and [43, par. 45].

[26] 'We are in many ways in the situation of all generations caught in a cataclysmic change: we find difficulty in making a totally new orientation toward our intellectual environment. In many ways our approach to the computer is controlled by attitudes more appropriate to other, older environments. [...] We still have not asked ourselves 'What is the full range of functions that the computer can perform?' Instead we ask 'Are there functions we are now performing that the computer can take over?' The limit we have placed on ourselves by this narrowed line of approach has cost us loss of time, loss of effort, and loss of opportunity' [39, p. 256].

rement for their computational processing;[27] as well as to the need of aligning sections of the texts before going into the details of collation (see section 3.4 above).[28]

This history also tells us something about where and when collation started to be automatized: if we look at the late 60's, it is mostly in the USA (California), Australia, France and Italy that the developments took place, almost independently; already in the 70's, automatic collation became much more widespread. Eventually, through the analysis of early research on automatic collation, we are also witnessing some of the first interactions between humans and computers, the struggles and the hopes of this two-faced relationship. Something remarkable about those early days is the willing of some Humanities scholars to penetrate the mysteries of machines: in their papers, they illustrate the computational process step-by-step, often accompanying explanations with flowcharts and complete program listings. It is clear that nowadays software is much more complicated; but another lesson that this history might teach us is the fact that the interactions of humans and machines work well when the first understand what the second do: translating algorithms into words and flowcharts is something those scholars seemed well-equipped for and a duty that we might re-discover looking at these old papers.

Literature

[1] Elli Bleeker, Bram Buitendijk, and Ronald Haentjens Dekker. Including XML markup in the automated collation of literary text. In *XML Prague 2018 Conference Proceedings*, pages 77–95.

[2] Margaret Scanlon Cabaniss. Using the Computer for Text Collation. *Computer Studies in the Humanities and Verbal Behaviour*, 3:1–33, 1970.

[3] Robert L. Cannon. OPCOL: An Optimal Text Collation Algorithm. *Computers and the Humanities*, 10(1):33–40, 1976.

[4] Robert L. Cannon and Robert Oakman. Interactive Collation on a Micro-Computer. The URICA! Approach. *Computers and the Humanities*, 23:469–472, 1989.

[27] Already in Dearing [7, pp. 19-20] and Froger [10, p. 136] we can find mentions of OCR techniques and confidence that they will soon be available also for handwritten documents. Cf. [12, p. 300, note 13]: 'The preparation of natural language machine-readable text is a large and knotty problem which needs to be studied thoroughly very soon. Optical scanners which would read the text and translate it directly into machine-readable form have been constructed, but have not reached the necessary accuracy. The difficulty of preparing accurate text is now the major obstacle to the use of computer technology in the humanities'.

[28] The macro-alignment, followed by a micro-alignment, essentially means that the alignment (and sometimes the tokenization) step of the Gothenburg model is repeated twice; but this strategy of multiple alignments is not generally addressed by current collation programs and falls back on the editors today, who do it manually. Among the recent exceptions, there are the tools LERA and LAKomp, which operate on two stages of tokenization and alignment, first for larger segments, and then at the word level [31]; the same concept is applied in the tool 'Prahbed' [5].

[5] Sukanta Chaudhuri, editor. *Bichitra: The Making of an Online Tagore Variorum*. Springer, 2015.

[6] Ernest Cadman Colwell and Ernest W. Tune. Variant readings : classification and use. *Journal of Biblical Literature*, 83(3):253–261, 1964.

[7] Vinton A. Dearing. *Methods of Textual Editing. A Paper Delivered by Vinton A. Dearing at a Seminar on Bibliography Held at the Clark LIbrary, 12 May 1962*. William Andrews Clark Memorial Library, Los Angeles, 1962.

[8] Vinton A. Dearing. Computer Aids to Editing the Text of Dryden. In *Art and Error: Modern Textual Editing. Essays Compiled and Edited by Ronald Gottesman and Scott Bennett.*, pages 254–278. Methuen and Co. Ldt., London, 1970.

[9] Ronald Haentjens Dekker, Dirk van Hulle, Gregor Middell, Vincent Neyt, and Joris van Zundert. Computer-supported collation of modern manuscripts: CollateX and the Beckett Digital Manuscript Project. *Literary and Linguistic Computing*, 30(3):452–470, 2015.

[10] Jacques Froger. La collation des manuscrits à la machine électronique. *Bulletin d'information de l'Institut de Recherche et d'Histoire des Textes*, 13:135–171, 1966.

[11] Jacques Froger. *La Critique Des Textes et Son Automatisation*. Dunod, Paris, 1968.

[12] William M. Gibson and George R. Petty. Project OCCULT: The Ordered Computer Collation of Unprepared Literary Text. In *Art and Error: Modern Textual Editing. Essays Compiled and Edited by Ronald Gottesman and Scott Bennett.*, pages 279–300. Methuen and Co. Ldt., London, 1970.

[13] Penny Gilbert. Automatic Collation: A Technique for Medieval Texts. *Computers and the Humanities*, 7(3):139–147, 1973.

[14] Penny Gilbert. The Preparation of Prose-Text Editions with the COLLATE System. In Jean Irigoin and G. P. Zarri, editors, *La Pratique Des Ordinateurs Dans La Critique Des Textes. Paris, 29-31 Mars 1978*, pages 245–254, Paris, 1979. Éditions du Centre National de la Recherche Scientifique.

[15] George Robert Guffey. Standardization of Photographic Reproductions for Mechanical Collation. *The Papers of the Bibliographical Society of America*, 62(2):237–240, 1968.

[16] Ronald Haentjens Dekker and David J. Birnbaum. It's more than just overlap: Text as graph. In *Proceedings of Balisage: The Markup Conference 2017*, Balisage Series on Markup Technologies, vol. 19.

[17] Mary Harris. *Computer collation of manuscript poetry: Dylan Thomas' "Poem on his birthday"*. PhD thesis, The University of Texas at Austin, 1975.

[18] Michael L. Hilton. The URICA! II Interactive Collation System. *Computers and the Humanities*, 26:139–144, 1992.

[19] Charlton Hinman. Mechanized collation; a preliminary report. *Papers of the Bibliographical Society of America*, 41:99–106, 1947.

[20] Susan Hockey. *A Guide to Computer Applications in the Humanities*. Duckworth, London, 1980.

[21] Susan Hockey. *Electronic Texts in the Humanities : Principles and Practice*. Oxford University Press, Oxford, 2000.

[22] Susan Hockey. The History of Humanities Computing. In Susan Schreibman, Raymond Siemens, and John Unsworth, editors, *A Companion to Digital Humanities*. Blackwell Publishing, Oxford, 2004.

[23] Hugh Houghton and Catherine Smith. Digital Editing and the Greek New Testament. In Claire Clivaz, Paul Dilley, and David Hamidovic, editors, *Ancient Worlds in Digital Culture*, pages 110–127. Brill, 2016.

[24] Stefan Jänicke, Annette Geßner, Greta Franzini, Melissa

Terras, Simon Mahony, and Gerik Scheuermann. TRAViz: A Visualization for Variant Graphs. *Digital Scholarship in the Humanities*, 30(Issue suppl_1):i83–i99, 2015.

[25] David Kahn. *The codebreakers: the story of secret writing*. Macmillan, 1967.

[26] Folger Shakespeare Library. Welcome to the collation.

[27] Gordon Lindstrand. Mechanized Textual Collation and Recent Designs. *Studies in Bibliography*, 24:204–214, 1971.

[28] Paul Maas. *Textkritik*. Teubner, 1927.

[29] John M. Manly and Edith Rickert, editors. *The Text of the Canterbury tales, studied on the basis of all known manuscripts by John M. Manly and Edith Rickert, with the aid of Mabel Dean, Helen McIntosh and others. With a chapter on illuminations by Margaret Rickert.* the University of Chicago press, 1940.

[30] E. Maretti and G. P. Zarri. Collatio Codicum: An Exercise in COMIT Programming. *La ricerca scientifica*, 37:608–611, 1967.

[31] A. Medek (*Gießler), Jörg Ritter, Paul Molitor, and S. Koesser. Interactive Similarity Analysis of Early New High German Text Variants, 2015.

[32] Carlos Monroy, Rajiv Kochumman, Richard Furuta, Eduardo Urbina, Eréndira Melgoza, and Arpita Goenka. Visualization of Variants in Textual Collations to Analyze the Evolution of Literary Works in the Cervantes Project. In M. Agosti and C. Thanos, editors, *Research and Advanced Technology for Digital Libraries. ECDL 2002. Lecture Notes in Computer Science*, volume 2458, pages 638–653, Berlin, Heidelberg, 2002. Springer.

[33] Elisa Nury. *Automated Collation and Digital Editions: From Theory to Practice*. PhD thesis, King's College London, 2018.

[34] Robert O'Hara and Peter Robinson. Report on the Textual Criticism Challenge 1991. *Bryn Mawr Classical Review*, 3(4):331–337, 1992.

[35] Robert O'Hara and Peter Robinson. Cladistic Analysis of an Old Norse Manuscript Tradition. In Susan Hockey and Nancy Ide, editors, *Research in Humanities Computing*, pages 115–137. Oxford University Press, Oxford, 1996.

[36] Wilhelm Ott. Computer Applications in Textual Criticism. In A.J. Aitken, R.W. Bailey, and N. Hamilton-Smith, editors, *The Computer and Literary Studies*, pages 199–223. Edinburgh University Press, Edinburgh, 1973.

[37] David C. Parker. *An Introduction to the New Testament Manuscripts and their Texts*. Cambridge University Press, Cambridge, 2008.

[38] Henri Quentin. *Essais de critique textuelle*. A. Picard, 1926.

[39] Joseph Raben. De Acibus et Faeni Acervis: Text Comparison as a Means of Collation. In Jean Irigoin and G. P. Zarri, editors, *La Pratique Des Ordinateurs Dans La Critique Des Textes. Paris, 29-31 Mars 1978*, pages 256–261, Paris, 1979. Éditions du Centre National de la Recherche Scientifique.

[40] Peter Robinson. The Collation and Textual Criticism of Icelandic Manuscripts (1): Collation. *Literary and Linguistic Computing*, 4(2):99–105, 1989.

[41] Peter Robinson. Rationale and Implementation of the Collation System Used on this CD-ROM. In *The Miller's Tale on CD-ROM*. Scholarly Digital Editions, Leicester, UK, 2004.

[42] Peter Robinson. Scholarly Digital Editions: Collate 2, and the design for its successor: CollateXML (now, CollateX), 2014.

[43] Torsten Roeder. Juxta web service, LERA, and variance viewer. web based collation tools for TEI | RIDE. 11.

[44] Desmond Schmidt and Robert Colomb. A Data Structure for Representing Multi-version Texts Online. *Int. J. Hum.-Comput. Stud.*, 67(6):497–514, 2009.

[45] Desmond Allan Schmidt. Merging Multi-Version Texts: A General Solution to the Overlap Problem. In *Proceedings of Balisage: The Markup Conference 2009*, volume vol. 3 of *Balisage Series on Markup Technologies*, Montréal, Canada, August 11 - 14, 2009, 2009.

[46] Miriam J. Shillingsburg. Computer Assistance to Scholarly Editing. *Bulletin of Research in the Humanities*, 81:448–473, 1978.

[47] Peter Shillingsburg. *Scholarly Editing in the Computer Age: Theory and Practice*. University of Michigan Press, Ann Arbor, 3rd ed. edition, 1996.

[48] Georgette Silva and Harold Love. The identification of text variants by computer. *Information Storage and Retrieval*, 5(3):89–108, October 1969.

[49] Steven Escar Smith. "The Eternal Verities Verified": Charlton Hinman and the Roots of Mechanical Collation. *Bibliographical Society of the University of Virginia*, 53:129–161, 2000.

[50] Elena Spadini. *Studi sul Lancelot en prose*. PhD thesis, Sapienza Università di Roma, 2016.

[51] Elena Spadini. The role of the base manuscript in the collation of medieval texts. In Peter Boot and alii, editors, *Advances in Digital Scholarly Editing. Papers presented at the DiXiT Conferences in The Hague, Cologne, and Antwerp*, pages 345–350, Leiden, 2017. Sidestone Press.

[52] Matthew Spencer and Christopher J. Howe. Collating Texts Using Progressive Multiple Alignment. *Computers and the Humanities*, 38(3):253–270, 2004.

[53] Gary A. Stringer and William R. Vilberg. The Donne Variorum Textual Collation Program. *Computers and the Humanities*, 21(2):83–89, 1987.

[54] Joris van Zundert and Ronald Haentjens Dekker. Code, scholarship, and criticism: When is code scholarship and when is it not? *Digital Scholarship in the Humanities*, 32(suppl_1):i121–i133, 2017.

[55] Stephen Waite. Two programs for Comparing Texts. In Jean Irigoin and G. P. Zarri, editors, *La Pratique Des Ordinateurs Dans La Critique Des Textes. Paris, 29-31 Mars 1978*, pages 241–244, Paris, 1979. Éditions du Centre National de la Recherche Scientifique.

[56] Walter W. Greg. *The Calculus of Variants: An Essay on Textual Criticism*. Clarendon Press, Oxford, 1927.

[57] R. L. Widmann. The computer in historical collation: Use of the IBM 360/75 in collating multiple editions of A Midsummer Night's Dream. In R. A. Wisbey, editor, *The Computer in Literary and Linguistic Research*, pages 57–63, Cambridge, 1971. Cambridge University Press.

[58] Tariq Yousef and Chiara Palladino. iAligner : A tool for syntax-based intra-language text alignment, 2016.

[59] Gian Piero Zarri. Linguistica algoritmica e meccanizzazione della 'collation codicum'. *Lingua e stile*, III:21–40, 1968.

Dr. Elisa Nury Elisa Nury is a post-doctoral researcher at the University of Geneva for the Grammateus project on Greek documentary papyri. In 2018, she completed a Ph.D. in Digital Humanities at the University of King's College London, UK, on the topic of automated collation tools and digital critical editions. She graduated from the University of Lausanne, Switzerland, with a specialisation in History of the Book and Critical Edition. Her research interests include Latin literature, digital humanities and digital scholarly editing.

Address: Université de Genève, Département des sciences de l'antiquité, 1211 Genève 4, E-Mail: elisa.nury@unige.ch

Dr. Elena Spadini Elena Spadini is a post-doctoral researcher at the University of Lausanne, where she is developing a digital edition for the project 'Gustave Roud's. Œuvres complétes'. Elena holds a PhD in Romance Philology from Sapienza Università di Roma and specialized in Digital Humanities at the École nationale des chartes. From 2014 to 2017 she was a Marie Curie fellow in the ITN DiXiT (Digital Scholarly Editions Initial Training) program. Her research focuses on Digital Philology.

Address: Université de Lausanne, Centre des littérature en Suisse romande, 1015 Lausanne, E-Mail: elena.spadini@unil.ch