COMMUNICATIVE GOAL-DRIVEN NL GENERATION AND DATA-DRIVEN GRAPHICS GENERATION: AN ARCHITECTURAL SYNTHESIS FOR MULTIMEDIA PAGE GENERATION

John Bateman
Centre for Communication and Language Research
School of English Studies
University of Stirling
SCOTLAND, U.K.
j.a.bateman@stir.ac.uk

Thomas Kamps
Industrial Process and System Communications
Dept. of Electrical Engineering
Darmstadt University of Technology

Jörg Kleinz **Klaus Reichenberger**
Integrated Publication and Information Systems Institute
German Center for Information Technology
DARMSTADT, GERMANY
{kamps,kleinz,reichen}@darmstadt.gmd.de

Abstract

In this paper we present a system for automatically producing multimedia pages of information that draws both from results in data-driven aggregation in information visualization and from results in communicative-goal oriented natural language generation. Our system constitutes an architectural synthesis of these two directions, allowing a beneficial cross-fertilization of research methods. We suggest that data-driven visualization provides a general approach to aggregation in NLG, and that text planning allows higher user-responsiveness in visualization via automatic diagram design.

1 Introduction

In this paper we present one of the most significant system-architectural results relevant for NLG achieved within the KOMET-PAVE multimedia page generation experiment (GMD-IPSI: 1994–1996).[1] Based on previous, separate experiences in natural language generation (see: Teich & Bateman 1994, Bateman & Teich 1995) and in automatic diagram design and visualization (see: Hüser, Reichenberger, Rostek & Streitz 1995), the KOMET-PAVE experiment sought to combine NLG and visualization into a single integrated information presentation system capable of producing effectively designed pages of information analogous to 'overviews' found in print-based publications such as encyclopediae or magazines. During this work, it became evident that there were significant overlaps both in the processes and organizations of data most supportive of information presentation. Moreover, the individual approaches offered complementary solutions for presentation subproblems that proved independent of the particular presentation modalities for which they were originally developed. A thorough architectural synthesis was therefore strongly indicated.

The particular complementarity that provides the focus of the present paper is the following. First, it is widely accepted in both NLG and graphic design that the design decisions adopted must be sensitive not only to communicative purposes and the 'user' but also to contingent and emergent organizational properties of the data. However, the effectiveness of the solutions proposed for these is in complementary distribution across the two modalities. Approaches to respecting communicative purpose are underdeveloped in graphic design, while NLG has powerful techniques for imposing adherence to communicative purpose (e.g., goal-driven text planning); and, similarly, approaches to data-driven organization (i.e., 'aggregation') are comparatively weak in NLG, while automatic visualization now has a range of powerful techniques for identifying emergent organizational properties of large datasets. The architecture constructed in KOMET-PAVE builds on a combination of these individually developed techniques, resulting in a significant 'cross-fertilization' of approaches.

[1] KOMET ('Knowledge-oriented production of multimodal documents') and PAVE ('Publication and advanced visualization environments') were two departments of the German National Research Center for Information Technology's (GMD) institute for Integrated Publication and Information Systems (IPSI) in Darmstadt that cooperated closely for the work described in this paper. The authors would therefore like to thank all the members of those departments who contributed, and particularly Lothar Rostek, Melina Alexa, Elke Teich, Wiebke Möhr and Klaas Jan Rondhuis.

We organize the discussion as follows. We first introduce the visualization and automatic diagram design methods developed within the PAVE component of our system, drawing explicit attention to the similarities between the decisions made during diagram generation and those necessary during NL generation (Section 2). This provides necessary background to our claim that the methods and algorithms developed for visualization can also serve as a general solution to the problem of aggregation in tactical generation (Section 3). We then briefly show the same algorithms at work at the level of text organization, helping to motivate informational structures necessary for constraining page layout and for allocating presentation modalities in the complete page generation scenario (Section 4). We conclude the paper by summarizing the main points of architectural synthesis that we have pursued and outlining some prominent lines of ongoing work and future development.

2 Automatic Diagram Generation using Dependency Lattices

The approach to diagram generation adopted within the KOMET-PAVE experiment has been developed both theoretically and practically. The practical side was originally built as part of an 'Editor's Workbench' aimed at facilitating the work of an editor preparing large-scale publications such as encyclopediae (Rostek, Möhr & Fischer 1994). A range of flexible automatic visualisation tools (cf. Reichenberger, Kamps & Golovchinsky 1995, Hüser et al. 1995) were developed in this context. To illustrate our discussions below, we will adopt one trial application domain in which the Editor's Workbench has been used and for which a significant knowledge base has been constructed—that is, the art and art history domain already used as a basis for NLG in Teich & Bateman (1994) and Bateman & Teich (1995). Typical information maintained by this knowledge base involves information about artists (particularly biographical information such as birthdates, dates of working in particular institutions, dates of movements, works of art created, etc.), details of works of art and art movements, as well as pictures and full text representations of several thousand biographies.

Visualization in the context of the Editor's Workbench focused on providing a high degree of control over all the visual aspects of its presentations: including layout of information and diagram design. The particular aim of visualization was to be able to present *overviews* of datasets rather than elaborating on specifics, and this required methods for discovering regularities in the data that could then be used to motivate particular presentation strategies. The theoretical basis for the methods developed is given in detail in Kamps (1997) and rests on a new application of Formal Concept Analysis (FCA: Wille 1982). We now show briefly how FCA allows the construction of *dependency lattices* that support flexible diagram design. We adopt as a simple example the set of 'facts' displayed in the following table. These facts together show the subject areas, institutions, and time periods in which the shown artists were active.[2]

	Person	Profession	School	Workperiod
g1	Gropius	Architect	Harvard	1937–1951
g2	Breuer	Architect	Harvard	1937–1946
g3	A. Albers	Designer	Black Mountain College	1933–1949
g4	J. Albers	Urban Planner	Black Mountain College	1933–1949
g5	Moholy-Nagy	Urban Planner	New Bauhaus	1937–1938
g6	Hilberseimer	Architect	Illinois Institute of Technology	1938–1967

2.1 Algorithm for the construction of the concept lattice

Dependency lattices represent effectively the functional and set-valued functional dependencies that are established among the domains of a data relation. They can be computed from plain relation tables such as

[2]The names, institutions, periods, etc. used in this paper are selected primarily for illustrative purposes and should not be taken as reliable statements of art history!

	Architect	Designer	Urban Planner
Gropius	X		X
Breuer	X		X
A. Albers		X	
J. Albers			X
Moholy-Nagy			X
Hilberseimer	X		

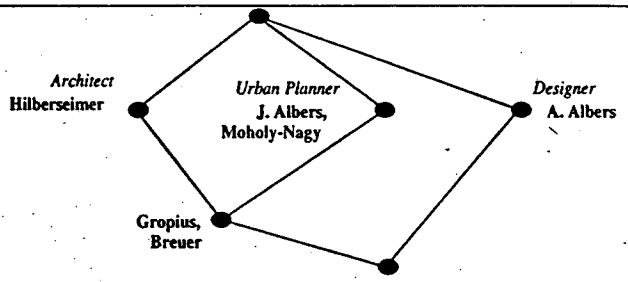

Figure 1: Example for a one-valued context and corresponding lattice

the one shown above, where the columns represent the domain sets on which the relation is defined and the rows represent the relation tuples. Dependency lattices are a particular kind of concept lattice as defined in Formal Concept Analysis. FCA starts from the notion of a formal context (G, M, I) representing the data in which G is a set of objects, M is a set of attributes and I establishes a binary relation between the two sets. $I(g, m)$ is read "object g has property m" if $g \in G$ and $m \in M$. Such a context is called a *one-valued context*. The one-valued context corresponding to the Profession-attribute of our example dataset is shown in the table to the left of Figure 1.

The formal concepts of concern in FCA are defined as consisting of an *extension* and an *intension*, where the extension is a subset A of the set of objects G and the intension is a subset B of the set of attributes M. We call the pair (A, B) a *formal concept* if each element of the extension may be assigned each attribute of the intension. Thus, the pairs ({Gropius, Breuer}, {Urban Planner, Architect}) and ({A.Albers}, {Designer}) represent concepts with respect to the example one-valued context of Figure 1. More intuitively, in a formal context concepts represent rectangles of maximum size, completely filled with x's after permutation of rows and columns. The set of all concepts may be computed effectively using the algorithm "Next Closure" developed by Ganter & Wille (1996). The hierarchy relation "subconcept", established between the set of concepts, is based on inclusions of the respective extensions and intensions of the concepts. Concretely, a concept (A, B) is a subconcept of (A^*, B^*) if and only if $A \subseteq A^* \Leftrightarrow B^* \subseteq B$. The main theorem of concept analysis shows that this "subconcept" relationship represents a complete lattice structure (see Wille 1982).

Given all concepts, we may construct the concept lattice starting from the top concept (the one that has no superconcepts) and proceed top-down recursively. In each step we must compute the set of direct subconcepts and link them to the respective superconcept until we reach the greatest lower bound of the lattice itself (the existence of the bounds is always guaranteed if we consider finite input data structures). One efficient implementation of this algorithm is explained in greater detail in Kamps (1997). The corresponding lattice for the one-valued context shown in Figure 1 is shown to the right of the figure. The labelling of the nodes of the lattice makes full use of the dependencies and redundancies that the lattice captures. Elements of the extensions are shown moving up the lattice, the extension label for each node consists of just those elements which are *added* at each node, while the members of the intensions are shown moving down the lattice, again adding just those elements that are new for each node. Thus, for example, the node simply labelled **Gropius, Breuer** corresponds to the full formal concept ({Gropius, Breuer}, {Architect, Urban Planner}) since both Gropius and Breuer are added new to the extension at that node, while no new elements are added to the intension ('Architect' and 'Urban Planner' are both inherited from the two nodes above in the lattice, where they are already present).

	Person	Profession	School	Workperiod
g1g2		X	X	
g1g6		X		
g2g6		X		
g3g4			X	X
g4g5		X		

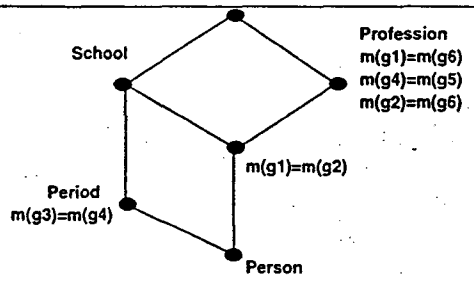

Figure 2: Example dependency context and corresponding lattice

2.2 How to find functional dependencies in the data

The original table of facts with which we started above is not a one-valued context: it is a *multivalued context*. A multivalued context is a generalisation of a one-valued context that may formally be represented as a quadruple (G, M, W, I) where G, M and I are as before. Here, however, the set of values W of the attributes is not trivial: to identify the value $w \in W$ of attribute $m \in M$ for an object $g \in G$ we adopt the notation m(g) = w and read this as "attribute m of object g has value w". Thus relation tables in general, such as the original table above, may all be considered as multivalued contexts.

Given an n-ary relation, functional relationships may generally be established between subsets of the n domains. However, we adopt the following particular construction of the dependency context: for the set of objects choose the set of subsets of two elements of the given multi-valued context $P_2(G)$, for the set of attributes choose the set of domains M, and for the connecting incidence relation choose $I_N(\{g,h\},m) :\Leftrightarrow m(g) = m(h)$, so that the resulting dependency context is represented by the triple $(P_2(G), M, I_N)$. Although this only considers pairwise mappings—that is such functional relationships that hold between two single domains—it simplifies the problem drastically and is a sensible approach for two reasons: first, the isolated functional relationships may, as we will see, be arranged in the form of a dependency lattice that allows a wholistic view on the dependency structure, and second, it is computationally simple to achieve.

The underlying principle is then straightforward: compute a (one-valued) dependency context from the given n-ary relation table and apply the techniques described above for the construction of the corresponding dependency lattice. This is illustrated in the table to the left of Figure 2, which shows the dependency context corresponding to our original full table of facts above. An entry in this table indicates that the identified attribute has the same value for both the facts identified in the object labels of the leftmost column: for example, 'g1' and 'g2' share the values of their Professions and Schools attributes. The corresponding dependency lattice, built in the same manner as shown for one-valued contexts, is shown in the lattice on the right of the figure.

The arcs in this lattice represent the functional dependencies between the involved domains whereas the equalities (e.g., m(g1)=m(g2)) represent the redundancies that may be observed in the table: for example, the lower left node labelled Period indicates not only that the third and fourth row entries under Period (g3 and g4) are identical but also, following the upward arc that these entries are equal with respect to School; similarly, following the upward arcs (which is possible because functional dependencies are transitive), the middle node (m(g1)=m(g2)) indicates that the first and second row table entries are shared with respect to both School and Profession. The lattice as a whole indicates that there are functional relationships from the set of persons into the set of professions, the set of periods, and the set of schools. A further functional relationship exists from the set of periods into the set of schools.

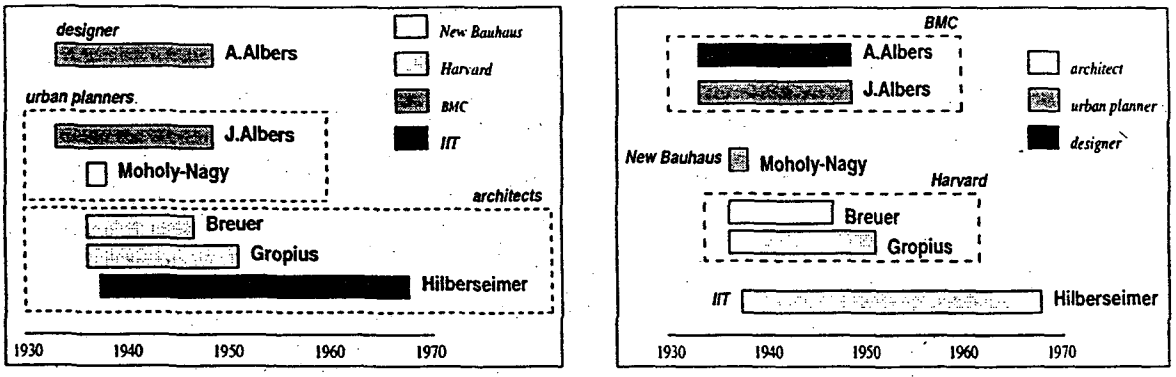

Figure 3: Example generated diagrams for the example data

2.3 How dependency lattices are used for visualisation

A dependency lattice, in which the edges represent functions between the domains and the non-existing edges represent set-valued mappings, may be interpreted as a set of classifications of the relational input. For graphics generation it is important that all domains of the relation become graphically encoded. This means the encoding is complete. To this purpose, Kamps (1997) proposes a graphical encoding algorithm that starts encoding the bottom domain and walks up the lattice in a bottom-up / left-to-right approach encoding the upper domains. The idea of this model, much abbreviated, is that the cardinality of the bottom domain is the largest, whereas the domains further up in the lattice contain less and less elements. Thus, the bottom domain is graphically encoded using so-called graphical elements (rectangle, circle, line, etc.), whereas the upper domains are encoded using their graphical attributes (colour, width, radius) as well as set-valued attributes (attachments of graphical elements) to keep graphical complexity moderate. Since functions and set-valued functions are binary relations, the encoding of a structured n-tuple is composed of a set of binary encodings. In the algorithm proposed by Kamps (1997), each domain is visited and encoded once which implies one walk through the lattice representing exactly one classification and one visualisation of the data. Many alternative diagrams may thus be generated for such a data set and the visualization algorithm contains extensive perceptual heuristics for evaluating among these.

Figure 3 shows two example diagrams that are produced from the dataset of our example table via the dependency lattice shown to the right of Figure 2. Informally, from the lattice we can see directly that artists ('Person') can be classified on one hand according to work period and, on the other hand, *jointly* according to school and profession. The 'attribute' person, indicated in the lowest node of the lattice, is first allocated to the basic graphical element 'rectangle'; the individual identities of the set members are given by a graphical attachment: a string giving the artist's name. The functional relationship between the set of persons and the set of time periods is then represented by the further graphical attribute of the *length* of the rectangle. This is motivated by the equivalence of the properties of temporal intervals in the data and the properties of the graphical relationship of spatial 'intervals' on the page. Two paths are then open: first following the functional relationship to a set of schools or to a set of professions. Diagram (a) in Figure 3 adopts the first path and encodes the school relationship by means of the further graphical attribute of the *color* of the rectangle, followed by a nesting rectangle for the relationship to professions; diagram (b) illustrates the second path, in which the selection of graphical encodings is reversed. Both the selection of color and of nesting rectangles are again motivated by the correspondence between the formal properties of the graphical relations and those of the dependencies observed in the data.

2.4 The partial equivalence of diagram design and text design

Our brief description of the process of producing alternative diagrams can now be considered from the perspective of producing alternative *texts*. The selection of particular graphical elements, and the commitments that follow for expressing particular functional dependencies, are closely analogous to decisions that need to be made when generating a text from the given dataset. Indeed, textual representations of the example diagrams may be motivated from the dependency lattice structure by proceeding over all functional groupings and taking into account the position of the equalities in the lattice just as in the diagram generation.

For instance, starting from equality $m(g1) = m(g2)$ in the lattice, it is sensible to relate the fact that this dependency holds both for the schools and for the professions so that we may connect them in a single sentence: i.e., 'g1' (concerning Gropius) and 'g2' (concerning Breuer) can be compactly expressed by collapsing their (identical) school and profession attributes. A similar phenomenon holds for grouping $m(g3) = m(g4)$, which is shared by the periods and the schools: here, 'g3' (concerning A. Albers) and 'g4' (concerning J. Albers) may be succinctly expressed by collapsing their identical period and school attributes. This would motivate the following approximate textual re-rendering of diagram (b):

> Anni Albers (who was a designer) and J. Albers (who was an urban planner) both taught at the BMC from 1933 until 1949. Moholy-Nagy (who was also an urban planner) taught from 1937 until 1938 at the New Bauhaus. Gropius and Breuer (both architects) were, at partially overlapping times (1937–1951 and 1937–1946 respectively), at Harvard. Hilberseimer (who was an architect too) taught at the IIT from 1938 until 1967.

In contrast, the other three groupings (indicated by the equalities on the profession node in the lattice) are "simple"—i.e., not shared by more than one domain—so that selecting these does not result in a further compaction of a text being possible.

3 Towards a general treatment of aggregation for NLG

The extraction of partial commonalities held constant over subsets of the data to be presented—be they expressed via an allocation of common graphical elements or by textual groupings—is naturally similar to one aspect of the problem of *aggregation* in NLG. In fact, the functional redundancies that are captured by the lattice construction technique are also precisely those redundances that indicate opportunities for structurally-induced aggregation. Selecting a particular graphical element or attribute to realize some aspect of the data *is* an aggregation step. In this section, we show this in terms more familiar to NLG by briefly sketching how the approach handles one example of aggregation discussed in the literature: the production of concise telephone network planning reports illustrated by McKeown, Robin & Kukich (1995).

One example from McKeown et al. (1995) concerns the data shown in Figure 4, again re-represented in tabular form. The attributes taken here are the *semantic roles* that might be used to provide input concerning 3 individual 'facts' (g1, g2, g3) to a tactical generation component. We consider the problem of providing possible 'aggregations' of these facts in order to improve the resulting sentences that would be generated. This is managed by means of the corresponding dependency lattice, which we also show in Figure 4, abbreviated and annotated somewhat here for ease of discussion. Analogously to the case for diagram generation, where several diagrams may be generated from a single lattice, a dependency lattice represents not a particular aggregation, but rather *all possible aggregations* in a single compact form. Input expressions for tactical generation can be constructed by working upwards from the bottom of the lattice. Each node with associated functional dependencies represents a point of possible aggregation.

In the diagram, therefore, the lowest nodes in the lattice represent three starting points; from left to right: (i) aggregations of type, source and destination with respect to the major dimensions of actor, process, etc., and (ii) and (iii) source and destination with respect to a type. The righthand Type node then represents

	Actor	Process	Actee	Type	Object	Source	Destination	Period	Year
g1	it	requested	placement	48 fiber	cable	CO	1103	Q2	1995
g2	it	requested	placement	24 fiber	cable	1201	1301	Q2	1995
g3	it	requested	placement	24 fiber	cable	1401	1501	Q2	1995

McKeown et al. (1995, p718; ex.4)

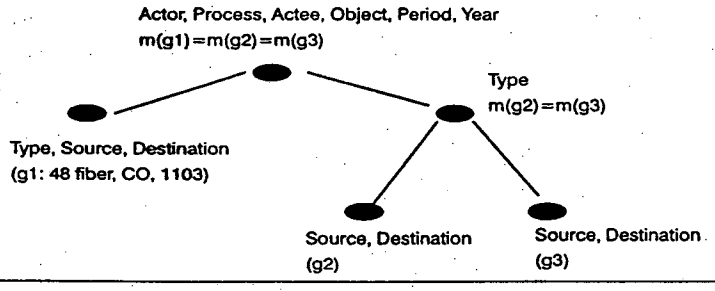

Figure 4: Example data and corresponding (annotated) dependency lattice

aggregation with respect to the major dimensions analogously to the left hand node. Respecting these dependencies results in the following maximally compact rendering of this information:

> It requested placement in the second quarter of 1995 of a 48-fiber cable from CO to 1103 and 24-fiber from 1201 to 1301 and from 1401 to 1501.

Thus, the dependency lattice directly determines the *logical dependency structure* of the clause (cf. Halliday 1994).

As McKeown et al. (1995) note, however, it is sometimes ill-advised to carry out a maximal aggregation. We can also model this restraint using the dependency lattice by bringing more generic (higher) nodes down and 'distributing' them over lower lattice nodes. The motivation for such lowering is typically to be found in registerial constraints and the method of textual development being used in the text at hand. If the 'objects' of the domain (e.g., in this case, the *cable*) are to remain salient, then these can be re-distributed from the uppermost node to enforce redundant expression; for example:

> It requested placement ...of a 48-fiber cable from CO to 1103 and 24-fiber cables from 1201 to 1301 and from 1401 to 1501

The other examples presented by McKeown et al. (1995), as well as other examples of similar phenomena presented in the literature (e.g., Dalianis & Hovy 1996) are handled similarly.

Since the dependency lattice does not itself determine which of the possible aggregations is taken up, but simply represents what is possible, this approach turns aggregation into a *process of communicative choice* along exactly the same lines as all other choices in the grammar, semantics, text organization, etc. One of the major benefits of the dependency lattice is then to represent this space of possibilities compactly, allowing a more systematic weighing of alternatives. The possibilities for aggregation captured by a dependency lattice then largely remove the need for *ad hoc* specific rules of grouping. Nevertheless, the extraction of those dimensions of organization or aggregation that are particularly *relevant* for a specific text or diagram can only be determined from the *communicative purpose* of the text or diagram that is being constructed: i.e., which 'question' is the text/diagram answering. Therefore, the kinds of grouping and organization that we have illustrated in the paper so far cannot replace communicative-goal driven NLG; they need rather to be properly integrated in a goal-driven architecture. This we illustrate in the section following.

4 Page generation

Within the KOMET-PAVE page generation experiment, we attempted to make use of the close analogies we have illustrated above between data-driven aggregation for diagram design and for text production. Moreover, the existence of a general aggregation tool allows us to consider aggregation as a general property of all levels of linguistic representation constructed during the generation process. The lattice construction algorithm is robust and fast and we are now aiming to construct a dependency lattice after the production of each level of structure during generation. This should apply to grammar and rhetorical structure as well as to the more semantic or domain oriented aggregations discussed above. In our final example in this paper, therefore, we briefly sketch the utility of performing data-driven aggregation on the results of a text planning process aimed at producing rhetorically motivated page specifications.

The purpose of the KOMET-PAVE experiment was to provide a system where the response of the system to a user's request for information is a single 'page' of information combining generated text, generated graphics, and retrieved visual information (pictures, etc.) within a communicative-functionally motivated layout. The *multimedia page* is therefore seen as the basic unit of information presentation, while these units are themselves seen as moves in a multimodal dialogue (cf. Stein & Thiel 1993); the analogy to (and extension of) web-based information services should be obvious. Given our use of the art and art history domain, the particular goal of the pages generated by the system was to present useful 'starting-off points', or overviews, of the information maintained in the knowledge base. Our example in this section concerns possible answers of the system to a question concerning the spread of the Bauhaus movement. The input to the page synthesis process was taken as a set of artists selected during the previous 'conversational move' and some generic features determined for such pages.[3]

When planning the information to be expressed by a page as a whole, it is possible to construct an RST-like structure as is familiar from NLG for individual texts (e.g., Hovy, Lavid, Maier, Mittal & Paris 1992, Moore & Paris 1993)—indeed, prior to further information chunking, the structure could well *be* a single text. An example of such a structure is shown on the left of Figure 5[4]. We assume that generic constraints on this type of text predispose the planning system to pursue presentations of evidence for assertions made and, at almost any excuse, short biographies of any artists mentioned as additional background.

The information present in this RST-structure can be made amenable to formal concept analysis in a number of ways; it is simply necessary to make available the relations and their arguments so that the data is structured as in our examples above. Then, constructing a dependency lattice on the basis of this information yields a number of possible aggregations: most useful here are two sets of functional dependencies, one grouping the acts of teaching around the predicate of teaching and one grouping the biographies. These points of aggregation in effect 're-structure' the corresponding RST, as shown to the right of Figure 5. This restructuring factors out commonalities so that information from lower leaves of the tree has been placed at higher branches. This results in an alternative, more richly structured presentation plan, the leaves of which are then analyzed in order to estimate how appropriate particular realizations and media-allocations would be.

We have already seen some results of attempting further realization of the set of teaching facts since our original starting table in Section 2 was just such a set. Diagrams such as those in Figure 3 can readily be produced, whereas the corresponding texts (see above) are not particularly smooth. We account for this by considering many co-varying dimensions of functional dependencies, as in the combined nucleus of

[3]The Bauhaus example is taken from Kamps, Hüser, Möhr & Schmidt's (1996) discussion of interface design and the kinds of interaction that a multimodal information system should support. Several examples of pages actually generated by the system are available on the web at URL: `'http://www.darmstadt.gmd.de/publish/komet/kometpave-pics-96.html'`. The presentation environment is implemented in Smalltalk, the visualization and layout engines in C, the text generation component in Common Lisp; page generation is in real-time.

[4]Note that currently we do not generate the initial nucleus, the overview paragraph.

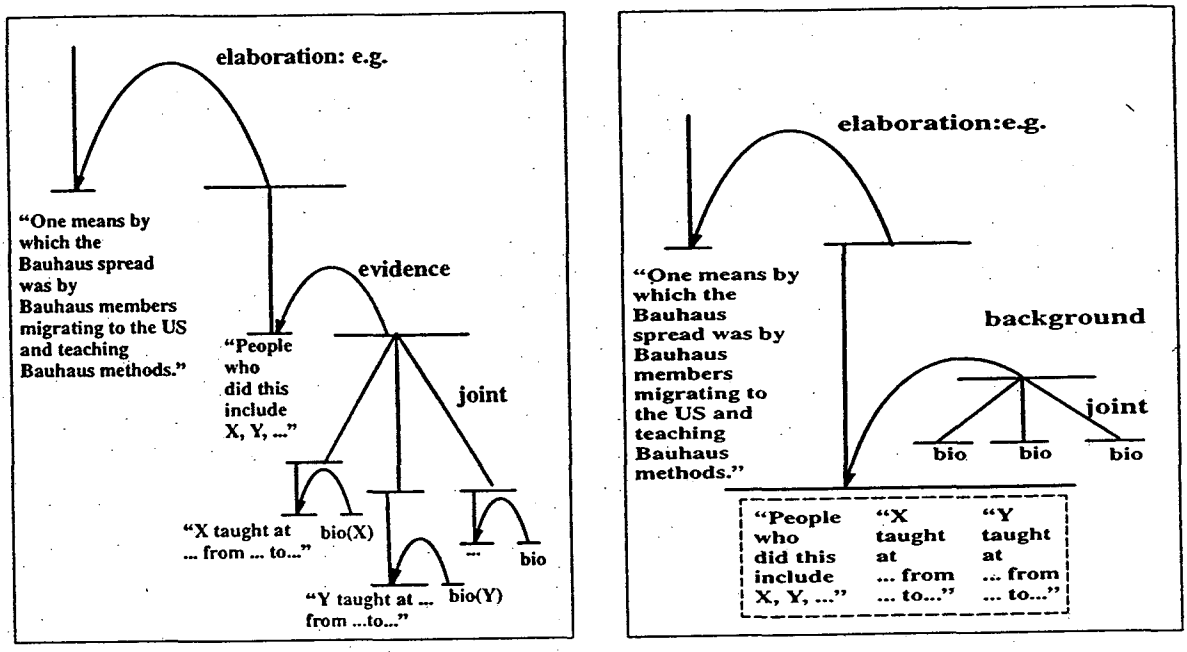

Figure 5: RST-like structuring of the contents of a potential page: before and after aggregation

the first embedded elaboration, to more strongly motivate a diagram.[5] This then serves as the input for the visualization process described above resulting in, for example, a timeline diagram. In contrast, the dependency lattices constructed for the individual biographies exhibit far fewer dimensions of reoccuring commonalities (e.g., simple progression in time with accompanying changes in location or state revolving around a single individual) and so are considered good candidates for textual expression. And, indeed, texts appropriate for these chunks of information are in fact precisely the simple biographies produced by the genre-driven text generation component described previously in Bateman & Teich (1995).

Finally, passing the revised RST-structure on to layout planning (cf. Reichenberger, Rondhuis, Kleinz & Bateman 1996), complete with its leaves filled in with text and diagrams as motivated here, results in a synthesized multimedia page with communicatively appropriate layout as required.

5 Conclusion: directions and future work

In this paper, we have very briefly presented an extended architecture for generation that attempts to combine generic methods for data-driven organization with top-down organizing principles. There are several further lines of development that are now required to establish the full utility of the architecture. At present, we have not evaluated the kinds of variation that occur when aggregation is sought at all levels of representation as we propose: in particular, generic text stages and grammatical structures have not been included. In addition, the relationship between the top-down communicative goals and the particular selections of organizing dimensions to be exploited during aggregation needs further work. Nevertheless, it seems clear that, in its combination of modes and techniques of processing from the NL-generation and visualisation traditions, an improved level of overall functionality has been achieved.

[5]This is, of course, only a heuristic at this time and could easily require alteration—for example, with different communicative purposes or different output modalities (e.g., spoken language).

Work in progress or preparation is now providing more efficient and robust implementations of the general dependency analyses and their encoding in graphical form, furthering the relationship between rhetorical structure and motivated layout, and seeking more empirically based statements of generic document layout, visualization and text type constraints that can provide more detailed constraints for the page generation process.

References

Bateman, J. A. & Teich, E. (1995), 'Selective information presentation in an integrated publication system: an application of genre-driven text generation', *Information Processing and Management: an international journal* **31**(5), 753–768.

Dalianis, H. & Hovy, E. (1996), Aggregation in natural language generation, *in* G. Adorni & M. Zock, eds, 'Trends in natural language generation: an artificial intelligence perspective', Springer-Verlag, pp. 88–105.

Ganter, B. & Wille, R. (1996), *Formale Begriffsanalyse—Mathematische Grundlagen*, Springer-Verlag.

Halliday, M. A. K. (1994), *An Introduction to Functional Grammar*, Edward Arnold, London. 2nd. edition.

Hovy, E. H., Lavid, J., Maier, E., Mittal, V. & Paris, C. (1992), Employing knowledge resources in a new text planner architecture, *in* R. Dale, E. Hovy, D. Rösner & O. Stock, eds, 'Aspects of automated natural language generation', Springer-Verlag, pp. 57 – 72.

Hüser, C., Reichenberger, K., Rostek, L. & Streitz, N. (1995), 'Knowledge-based editing and visualization for hypermedia encyclopedias', *Communications of the ACM* **38**(4), 49–51.

Kamps, T. (1997), A constructive theory for diagram design and its algorithmic implementation, PhD thesis, Darmstadt University of Technology, Darmstadt, Germany.

Kamps, T., Hüser, C., Möhr, W. & Schmidt, I. (1996), Knowledge-based information access for hypermedia reference works: exploring the spread of the Bauhaus movement, *in* M. Agosti & A. F. Smeaton, eds, 'Information retrieval and hypertext', Kluwer Academic Publishers, Boston/London/Dordrecht, pp. 225–255.

McKeown, K., Robin, J. & Kukich, K. (1995), 'Generating concise natural language summaries', *Information Processing and Management* **31**(5), 703–733.

Moore, J. D. & Paris, C. L. (1993), 'Planning texts for advisory dialogs: capturing intentional and rhetorical information', *Computational Linguistics* **19**(4), 651 – 694.

Reichenberger, K., Kamps, T. & Golovchinsky, G. (1995), Towards a generative theory of diagram design, *in* 'Proceedings of 1995 IEEE Symposium on Information Visualization', IEEE Computer Society Press, Los Alamitos, USA, pp. 217–223.

Reichenberger, K., Rondhuis, K., Kleinz, J. & Bateman, J. A. (1996), 'Effective presentation of information through page layout: a linguistically-based approach'. In: 'Effective Abstractions in Multimedia, Layout and Interaction', workshop held in conjunction with ACM Multimedia '95, November 1995, San Francisco, California.

Rostek, L., Möhr, W. & Fischer, D. H. (1994), Weaving a web: The structure and creation of an object network representing an electronic reference network, *in* C. Hüser and W. Möhr and V. Quint, ed., 'Proceedings of Electronic Publishing (EP) '94', Wiley, Chichester, pp. 495 – 506.

Stein, A. & Thiel, U. (1993), A conversational model of multimodal interaction in information systems, *in* 'Proceedings of the 11th National Conference on Artificial Intelligence (AAAI '93), Washington DC, USA', AAAI Press/ MIT Press, pp. 283–288.

Teich, E. & Bateman, J. A. (1994), Towards an application of text generation in an integrated publication system, *in* 'Proceedings of the Seventh International Workshop on Natural Language Generation, Kennebunkport, Maine, USA, June 21-24, 1994', pp. 153–162.

Wille, R. (1982), Restructuring lattice theory: an approach based on hierarchies of concept, *in* I. Rival, ed., 'Ordered Sets', Reidel, Dordecht/Boston, pp. 445–470.