Sublanguages in Text and Graphics

Ehud Reiter*
CoGenTex, Inc
840 Hanshaw Rd
Ithaca, NY 14850 USA

Abstract

We claim that *sublanguages*, a well known phenomena in natural-language, are also important in diagrams and related graphics. Automatic graph layout (AGL) systems need to conform to the rules and conventions of the specific graphical genre they are being used in, as well as, or even instead of, generic genre-independent rules. It is essential that genre-specific conventions be obeyed, even if this results in layouts that are 'in principle' suboptimal.

This is work in progress, and part of a broader effort to investigate if natural-language generation ideas and techniques can be applied to diagram-generation problems.

1 Introduction

Natural-language generation (NLG) is the study of how computers can automatically generate text; automatic graph layout (AGL) is the study of how computers can automatically create graphical diagrams. To date, there has been relatively little interaction between NLG and AGL researchers, which is a pity. Both NLG and AGL are, after all, just special cases of the more general task of generating effective presentations of information to human users, and no doubt each field could learn from techniques developed in the other.

CoGenTex is a small company which specializes in building applied NLG systems. We have recently been investigating whether ideas and techniques from NLG can be applied to AGL problems; our project is a very small one, and is aimed more at identifying promising areas for future work than at actually building AGL systems. In this paper, we discuss some preliminary findings about what we feel is one of the most promising such areas, sublanguages.

Sublanguages are an acknowledged and uncontroversial fact of life in the linguistic and NLP worlds; documents in different genres use different words, syntactic structures, pronominalization rules, etc. Almost everyone in the mid-1990s who is building an applied NLG system acknowledges this fact, and builds systems that are designed to produce texts in the target sublanguage, instead of 'general English.' But the concept of sublanguages seems to be less prominent in the graphics world; while AGL experts acknowledge that graphical sublanguages exist, most AGL work is still based on building generic heuristics and algorithms which are supposed to work 'for almost any diagram'. We believe that sublanguages are as important in graphics as in language, and therefore that AGL systems that use universal algorithms and layout rules, whether based on the author's intuition about what makes diagrams effective or on formal psychological knowledge of human visual perception, may not produce useful and effective diagrams in real-world applications.

We will summarize on our preliminary findings about graphical sublanguages in the rest of this paper. We have also in our project investigated the applicability to AGL tasks of the NLG ideas of document

*Currently at the Department of Computing Science, University of Aberdeen, King's College, Aberdeen AB9 2UE, BRITAIN. Email address is ereiter@csd.abdn.ac.uk

planning, pragmatic correctness, and user and task tailoring; unfortunately, space limitations prevent us from discussing these topics in this paper.

Our research, and this paper, focuses on automatic creation of *network diagrams*, such as flowcharts, computer network depictions, and class diagrams for object-oriented systems. We believe that sublanguage effects are also probably important in *data graphics* such as bar charts and scatter plots, but we have not (to date) investigated this.

2 Sublanguages in Natural Language Generation

The concept of 'sublanguages' [GK86] is central to CoGenTex's approach to natural-language generation [KGKP94]; its importance has also been acknowledged by other people building applied NLG systems (eg, [MKS94]). We strongly believe that practical language-generation systems need to produce output texts that obey numerous domain-specific constraints and conventions on syntax, word choice, aggregation, pronominalization, text structure, and probably every other aspect of language. In other words, it is not possible to build a 'general NLG' system that can be used in any domain without customization; at best one can build a 'general NLG shell' with appropriate hooks for domain-specific customization.

Sublanguages are not necessarily subsets of 'general English', as the name sublanguage unfortunately implies. For example, zero anaphora (missing pronouns) are not allowed in general English, but are allowed in recipes. So, *Bake for 20 minutes* is perfectly valid in the 'recipe' sublanguage, even though it is not a valid sentence in general English, which would instead require *Bake it for 20 minutes.*

There are many reasons for the evolution of different sublanguages in different genres. Much, perhaps most, of many sublanguages can be explained by historical accident and convention, but there are also more some more principled reasons, including:

- *Differences in readers:* For example, AECMA Simplified English [AEC86] is targeted towards texts that may be read by non-native English speakers, and hence prohibits some constructions (eg, gerunds) that pose no problems for native speakers, but are known to be difficult for some non-native speakers.

- *Differences in context:* For example, written technical instruction often avoids relative adjectives such as *big*, which are very common in spoken dialogs [RD92]. This is presumably because the writer cannot predict the context his text will be read in, which means he does not know how *big screw* (for example) will be interpreted.

- *Differences in risk and costs:* For example, technical documents in general usually contain many fewer pronouns than novels. This may partially be explained by the fact that the cost of inappropriate pronominalization can be very high in a technical document (eg, the reader may misuse the device and seriously injure herself), while the cost of inappropriate pronominalization in a novel is relatively low.

3 Sublanguages in Automatic Graph Layout

Most AGL research (ie, most papers mentioned in the bibliographies [ET89] and [FMM93]) has focused on generic (genre-independent) algorithms and layout rules, usually based on some combination of the author's intuition, psychological knowledge of human visual perception, and computational complexity considerations. When genre-specific algorithms or rules have been proposed, they typically have been relatively small variants of a pre-existing set of 'universal' rules.

For example, [TBT83] give a set of criteria for drawing Entity-Relationship (E-R) diagrams which consist of three generic rules (minimize edge crossings, bends, and line lengths) which the authors have

used in other genres, plus a single E-R specific rule, namely "Use equal-length connections between relationship boxes and related entity boxes." In our study, however, we observed many additional E-R specific conventions. For example (the below list is largely based on the layout rules given on pages 3-16 and 3-17 of Barker's textbook [Bar90]):

- E-R diagrams use mostly horizontal and vertical edges, but often have a few diagonal edges as well. Flowcharts, in contrast, often have exclusively horizontal and vertical edges, while 'circular' computer network diagrams (where the server is placed in the middle of a circle of clients) have mostly diagonal (ie, neither vertical nor horizontal) edges.

- In E-R diagrams, related entities (boxes) are typically grouped together within a close-to-square 'bounding box' (ie, the box's width is similar to its height). In flowcharts, in contrast, related boxes are often grouped in high-aspect ratio bounding boxes (eg, in a vertical or horizontal row).

- E-R diagrams should be laid out with the 'many' end of a relation at the left or on top. This means that the bottom right-hand corner of the diagram will contain "highly significant entities that are used to define other [entities]" [Bar90, page 3-17]. This goes against the more usual layout principle, which is to put the most important objects in the middle or on top of the diagram.

- According to Barker, relationship names should be written close to the entity boxes they modify; this contradicts the "equal-length connection" rule used by [TBT83], which usually causes relationship names to be centered between entity boxes (Martin's textbook [Mar87] also contains many diagrams with centered relationship names). This is an example of a variation between different E-R 'dialects'.

These are just a few examples of sublanguage-specific layout rules. We have seen many others, and indeed would not be at all surprised if it turned out that there were in fact no universal layout rules, but rather simply rules that are used in a large number of graphical sublanguages and rules that are used in a small number of graphical sublanguages.

Commercial AGL systems have in many ways been more sensitive than research systems to the need to conform to genre-specific conventions. During the course of this investigation, we experimented with several such packages (including Tom Sayer's Graph Layout Toolkit, allClear, and the ATT *dotty* system), and observed that all of these systems were *de facto* tuned to particular kinds of diagrams (although this fact was not always emphasized in the marketing literature). Brendan Madden, President of Tom Sawyer Software, agreed (in a personal communication) that a substantial amount of "specialization" was needed to get his general system to produce acceptable layouts in particular domains and genres; and that this specialization included adjusting the system to conform to pre-existing conventions in the genre.

3.1 The Motivation for Graphical Sublanguages

From a theoretical perspective, graphical sublanguages can be justified as a way of making it easier for experienced users to comprehend diagrams. As the psychologist Steven Kosslyn points out [Kos94], diagrams will be comprehended more quickly if the user can use her visual pattern-recognition skills to identify patterns that are similar to patterns she has seen in the past, instead of having to analyze the diagram in terms of individual nodes and edges. And pattern-recognition will only work if diagrams with similar semantics are given a similar visual appearance; ie, if a sublanguage is conformed to.

Kosslyn's observation leads to an interesting hypothesis about a *difference* between graphics and text generation. Namely, since the human visual system probably has much better pattern-recognition abilities than the human language system, perhaps conformance to a strong sublanguage is more universally important in graphics than in text. We certainly cannot prove this conjecture, but one piece of circumstantial evidence that may support it is the differing attitude towards variation in textual and graphical genres. In many text genres, including literary work, newspaper articles, and even (to a limited degree)

AECMA Simplified English, authors are supposed to deliberately vary their texts, eg, use different synonyms and syntactic structures in order to 'keep the text interesting.' In contrast, no graphical genre that we are aware of has a 'vary when possible' rule; the emphasis is instead on presenting similar information in a visually similar way.

4 Future Work

We obviously have only taken the first steps in the investigation of graphical sublanguages and their impact on automatic graph layout systems. Much work remains to be done, and many questions need to be answered, before sublanguage-sensitive AGL systems can be built. Two particularly important issues that we hope to work on are:

- Where do graphical sublanguages typically vary? Are there 'families' of related and similar graphical sublanguages? Are there any layout rules that are in fact (almost) universal? We hope that the corpus-based sublanguage analysis techniques that have been developed in the NL community can be used to answer these questions.
- What is the best way to make an AGL system adjustable for different sublanguages? Is it better to build completely separate systems for each sublanguage; to add numerous parameters to 'conventional' AGL systems; or to switch to the constraint-satisfaction approach suggested by [KMS94, GN94]? Answering these questions will probably require empirical system-building experimentation.

5 Conclusion

We think it is undeniable that the sublanguage phenomena affects graphics as well as text. We also believe that the graphics-generation community can profit from one of the hard-earned lessons of the language-generation community, namely that useful applied systems usually need to be targeted towards specific sublanguages instead of general English.

We also are excited by the fact that this phenomena does seem to be important in both media. There have been other findings about similar underlying principles between text generation and graphics generation (eg, [MR90]), and this suggests that that perhaps it may be possible to develop a media-independent theory of computer-to-human communication, of which text generation and diagram generation would be special cases. Such a theory would be of great practical benefit in building multimedia systems, and also of great theoretical interest, and we can only hope that our efforts here provide a small stepping stone towards the construction of a general theory of multimedia communication.

6 Acknowledgements

This research has been funded by Rome Laboratory (US Air Force) under contract F30602-94-C-0281.

References

[AEC86] AECMA. A guide for the preparation of aircraft maintenance documentation in the international aerospace maintenance language, 1986. Available from BDC Publishing Services, Slack Lane, Derby, UK.

[Bar90] Richard Barker. *CASE* Method: Entity Relationship Modelling.* Addison-Wesley, 1990.

[ET89] P. Eades and R. Tamassia. Algorithms for drawing graphs: An annotated bibliography. Technical Report CS-89-09, Dept. of Computer Science, Brown University, 1989.

[FMM93] Steve Feiner, Jock Mackinlay, and Joe Marks. Automating the design of effective graphics, 1993. Notes from tutorial presented at AAAI-1993.

[GK86] Ralph Grishman and Richard Kittredge, editors. *Analyzing Language in Restricted Domains: Sublanguage Description and Processing.* Lawrence Erlbaum, 1986.

[GN94] Winfried Graf and Stefan Neurohr. Using graphical style and visibility constraints for a meaningful layout in visual programming interfaces. Research Report RR-94-15, DFKI, Saarbruecken, Germany, 1994.

[KGKP94] Richard Kittredge, Eli Goldberg, Myunghee Kim, and Alain Polguère. Sublanguage engineering in the FOG system. In *Proceedings of the Fourth Conference on Applied Natural Language Processing (ANLP-1994)*, pages 215–216, 1994.

[KMS94] Corey Kosak, Joseph Marks, and Stuart Shieber. Automating the layout of network diagrams with specified visual organization. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(3), 1994.

[Kos94] Stephen Kosslyn. *Elements of Graphic Design.* W.H. Freeman, New York, 1994.

[Mar87] James Martin. *Recommended Diagramming Standards for Analysts and Programmers.* Prentice-Hall, 1987.

[MKS94] Kathleen McKeown, Karen Kukich, and James Shaw. Practical issues in automatic document generation. In *Proceedings of the Fourth Conference on Applied Natural-Language Processing (ANLP-1994)*, pages 7–14, 1994.

[MR90] Joseph Marks and Ehud Reiter. Avoiding unwanted conversational implicatures in text and graphics. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-1990)*, pages 450–456, 1990.

[RD92] Ehud Reiter and Robert Dale. A fast algorithm for the generation of referring expressions. In *Proceedings of the Fourteenth International Conference on Computational Linguistics (COLING-1992)*, volume 1, pages 232–238, 1992.

[TBT83] R. Tamassia, C. Batini, and M. Talamo. An algorithm for automatic layout of entity relationship diagrams. In C. Davis, S. Jajodia, , P. Ng, and R. Yeh, editors, *Entity-Relationship Approach to Software Engineering (Proceedings of the Third International Conference on the Entity-Relationship Approach)*, pages 421–439. Elsevier, 1983.