

Information Visualization and Semiotic Morphisms

Joseph A. Goguen
Dept. Computer Science & Engineering
University of California at San Diego

Abstract: An approach to information visualization based on algebraic semiotics is introduced, and illustrated with examples. Semiotics is the general theory of signs, and algebraic semiotics is a new approach combining algebraic abstract data type theory with a grounding in social reality. The most important new ideas are to use semiotic spaces for systems of related signs, semiotic morphisms for representations of signs, and preservation orderings for the quality of representations.

1 Introduction and Motivation

Appropriate visualizations of large complex data sets can be an enormous aid to scientists in discovering, verifying, and predicting significant patterns. Unfortunately, it has proven difficult to find general principles for producing appropriate visualizations. Part of the reason is the lack of a precise definition for the word “appropriate” in the previous two sentences. The present state of HCI research is unsatisfactory, having a few precise laws of very limited scope (such as Fitt’s law), many case studies of unknown generality, and several methods of uncertain reliability (e.g., protocol analysis, usability studies, interviews – see [10] for a survey). Meanwhile, the user community and the technology base are both expanding rapidly, the commercial sector continues to produce exaggerated claims and mediocre products, and faith in experimental psychology and ergonomics as foundations is being eroded by developments in CSCW (Computer Supported Cooperative Work) and related areas which demonstrate that many difficulties arise from taking inadequate account of the meaning behind the interfaces, and of the social context in which interfaces are actually used. All this argues that we badly need to explore new directions for the construction of general theories.

Many fundamental issues in information visualization can be understood in terms of *representation*: a visualization *is* a representation of some aspects of the underlying information, and major questions are *what* to represent, and *how* to represent it. An adequate theory of information visualization must take account not just of current display technology capabilities, but also the structure of complex information such as scientific data, the capabilities and limitations of human perception and cognition, and the social context of work. For scientific visualization, the social context should include current scientific theories, conventional meanings of the signs and symbols used, the unequal importance of different patterns in the data, and the collaborative nature of scientific work. While it would be difficult to deny the importance of these factors for the design of visualizations and tools to support them, it would be foolish to believe that they are easy, and in particular, it would be foolish to believe that it is easy to get the designs of visualization or visualization tools right the first time,

or that design can be fully automated. For this reason, both theories and tools need to be broad and flexible, supporting relatively painless reconfiguration and evolution.

Although it seems natural to try to use semiotics as the basis for a theory of representation, classical semiotics has unfortunately not developed in a sufficiently rigorous way for our needs, nor has it explicitly addressed the representation of complex signs; also, its approach to meaning has been naive in some crucial respects, especially in neglecting (though not entirely ignoring) the social basis and contextualization of meaning. So it is not surprising that semiotics has mainly been used in the humanities, where scholars can compensate for these weaknesses, rather than in engineering, where descriptions need to be much more explicit. Another deficiency of classical semiotics is its inability to address dynamic signs and their representations, as is necessary for interfaces that involve change, instead of presenting a fixed static structure, e.g., for standard interactive features like buttons and fill-in forms, as well as for more complex situations like animations and virtual worlds. We will suggest approaches to overcoming these limitations.

Because we consider information visualization in particular, and user interface design in general, as problems in constructing appropriate representations, we need to know what representations are, and what makes them appropriate. For the first question, we consider a *representation* to be a mapping from one structured domain of signs, called a *semiotic space* or a *sign system*, to another such space. For the second question, we can measure the quality of a representation by how well it preserves what is most important to users, subject to any constraints imposed. These ideas seem simple, but it is not so obvious how to make them precise. Here we use some algebraic methods developed for the theory of abstract data types [12]. More specifically, the structure of a sign system is given by an *algebraic theory*, which consists of a syntax declaration (similar to a context free grammar) and a set of equations, plus some specifically semiotic features, including hierarchical levels for signs, and priorities on constructors; more details are given in the next section, and full details appear in [7]. Dynamic interfaces can be handled by generalizing from classical algebra to a new variant called hidden algebra [11].

The value of this approach can be judged by the analyses and suggestions for improvement it provides for examples, e.g., in Section 3 below. While sensitive designers might reach similar conclusions, algebraic semiotics does so in a systematic way, based on general principles (moreover, the original designers did not reach these conclusions). The mathematical formulation of the theory also raises hopes for partial automation of the design process. Finally, since all communication is mediated by signs, there is hope for applications well beyond information visualization.

2 Algebraic Semiotics

We approach the representation of signs and the quality of representation through precise notions of “semiotic system” and “semiotic morphism,” the latter being a systematic translation between semiotic systems. Though transformations are fundamental in many areas of mathematics and its applications (e.g., linear transformations, i.e., matrices), they have not been considered in classical semiotics. This section gives an intuitive introduction to some basic concepts. The main reference for algebraic semiotics is [7]; an informal exposition of some main ideas and their motivation is given in the webnote [4], and an (intendedly) amusing introduction is given in the UC San Diego Semiotic Zoo [5]. Further applications are developed in the author’s course on user interface design, some of which can be browsed on the class website [9].

2.1 Semiotic Spaces

Signs need not be the simple little things that we usually call “signs,” like the letters of an alphabet, or traffic signs. In written natural language, sentences are composed from words, and words are composed from letters; also, user interfaces are often very complex systems that are usefully considered single complex signs. Semiotic systems¹ capture the systematic structure of signs. This subsection introduces some elements of this notion informally; see [7] for details.

An important insight due to Ferdinand de Saussure [20] is that signs always come in systems. A typical example considered by Saussure is the tense system for the verbs of a language. For example, in English, adding “ed” to the end of a present tense (regular) verb makes it past tense, and adding “will” in front makes it future tense, as in “walk”, “walked”, and “will walk”. Saussure’s emphasis on the structure of systems of signs rather than isolated signs has been very influential, for example, in French structuralism and post-structuralism.

A basic strategy for making complex combinations of signs easier to understand is to divide their potential parts into **sorts**, and then discover rules for the ways that each sort of part can be used. For example, newspapers are composed from articles, ads, cartoons, etc., while articles are composed from headlines, paragraphs, photos, diagrams, etc., and paragraphs are composed from sentences. The so-called parts of speech in traditional grammars are also sorts in this sense. Sorts may have a hierarchical structure under a **subsort** partial ordering. For example, the sort NOUN is a subsort of the sort NOUN-PHRASE.

The rules for composing signs into more complex signs are of two kinds, called constructors and axioms. **Constructors** are functions that build new signs from others signs of given sorts, plus perhaps additional parameters. For example, a computer graphics image of a cat may be given as a constructor, with parameters that determine its size, color, and location on the screen. There may also be functions and predicates defined on signs; for example, a LOCATION function for graphical objects, and a HIGHLIGHTED predicate for text. **Axioms** are logical formulae built from constructors, functions and predicates; they constrain the set of possible signs.

In many examples, some constructors for signs of a given sort are more important than others. For example, a warning popup window is more important than a virtual pet cat. This gives rise to a **priority** partial ordering on the constructors for each sort. For another example, the pollutants in a lake may be prioritized by their toxicity.

Another fundamental strategy for managing complexity is to have a hierarchy of **levels**, with signs that are not atomic being constructed from other signs that are at lower (or possibly the same) levels. Thus linguistics has levels for phonology, morphology, lexicography, syntax, and discourse (multisentential units). Similarly, standard GUI displays have windows, which may contain other windows.

It is clear that context, including the physical setting of a given sign, can be at least as important for meaning as the sign itself. In an extreme example, the sentence “Yes” can mean almost anything, given the right context. This corresponds to an important insight of Peirce [17], that meaning is *relational*, not just denotational (i.e., functional); this is part of the point of his famous semiotic triangle. In the language of this paper, there may be constructors that place signs in some larger context, and that thus provide new parts for larger, contextualized signs. For example, the familiar 12 hour clock tells the correct 24 hour time in the context of external illumination, which can be considered an argument of a higher level constructor for clocks-in-context.

¹This paper uses the terms “sign system,” “semiotic system,” and “semiotic space” interchangeably.

2.2 Semiotic Morphisms and Design

Crafting a helpful explanation or a good “icon” (in the informal sense of computer graphics rather than Peirce’s technical sense), choosing a good file name, or using a mixture of media to present a given content in a satisfactory way, are all problems of translating signs in one system to signs in another system. In such cases, we know the source system, and we seek a suitable target system and an appropriate transformation that presents the information of interest in a suitable way; often we even know the target system. This is the problem of *design*. Conversely, we may know the target sign system, and seek to infer properties of signs in the source system from their images in the target system, for example, when we try to understand a poem, an equation, a sentence, or indeed, anything at all. Call this the *inverse problem*, as opposed to the *direct problem* of design.

Because semiotic systems are theories rather than models, their morphisms translate from the theory of one semiotic system into the theory of another, instead of translating concrete signs for one model to concrete signs for another model; that is, they translate descriptions to descriptions. This may seem indirect, but it has important advantages; in particular, it allows multiple models, and it remains open to additional structure at a later time.

A semiotic morphism maps structure in its source space to structure in its target space, taking sorts to sorts, subsorts to subsorts, constructors to constructors, etc. But in many real world applications, not everything *can* be preserved, so these maps must be *partial*. Axioms should also be preserved – but again in practice, not all axioms are preserved. Design is the problem of massaging a source space, a target space, and a morphism, to achieve suitable quality, subject to constraints. The extent to which structure is actually preserved gives a way to compare the quality of semiotic morphisms, as discussed further in the next subsection.

2.3 Quality of Semiotic Morphisms

Each feature of semiotic spaces that might be preserved gives rise of a measure of quality, defined by the degree to which this feature is preserved. For example, given semiotic morphisms M_1 and M_2 from one semiotic space S_1 to another S_2 , we may define $M_1 \sqsubseteq_C M_2$ if M_2 preserves every constructor that M_1 preserves, and $M_1 \sqsubseteq_A M_2$ if M_2 preserves every axiom that M_1 preserves. Other preservation relations are defined similarly [7]. There are also more refined orderings, e.g., $M_1 \sqsubseteq_{C,s} M_2$ if M_2 preserves every constructor of sort s that M_1 preserves; and we can define Boolean combinations of all these orderings, to get something appropriate for a particular application. For example, [8] applies these ideas in justifying design decisions for the user interface to a theorem proving system. Note that these quality measures are partial orderings, rather than linear numerical scales; this is appropriate because semiotic spaces are qualitative, in that they are concerned with structure.

2.4 Further Topics

Harvey Sacks’ notion of “category system” [18] from the branch of ethnomethodology [2] called conversation analysis [19] is related to semiotic systems, but is less formal. Our previous work on the nature of information [6] also uses ideas from ethnomethodology, and can be seen as providing a philosophical and methodological foundation for algebraic semiotics, that takes account of the social nature of signs. Lakoff, Johnson and others have developed the flourishing new field of cognitive linguistics, building on their careful studies of metaphor [15, 14, 16]. The cognitive linguists Fauconnier

and Turner have introduced the notion of “blending” conceptual metaphors [1], and demonstrated its importance for many aspects of cognition. A blend is built from two (or more) semiotic morphisms having a common source, called the **generic space**, with targets called the **input spaces**, by providing two (or more) semiotic morphisms from the input spaces to a **blend space**. See the blending website for much more information [23]. Simple examples from natural language include “house boat”, “road kill”, “artificial life”, and “computer virus”, each of which is a blend of its two component words. It happens that “boat house” has a different meaning from “house boat” because a different blend is computed. This is not because the order of the words is different, but because the same two spaces can have many different blends [7]. Semiotic spaces significantly generalizes the conceptual spaces used in cognitive linguistics, and [7] gives an appropriate generalization of blending, which allows many interesting examples in user interface design and information visualization.

Hidden algebra extends the algebraic theory of abstract data types to handle states and dynamics, as well as concurrency and nondeterminism [11]. This is the same extension needed to move algebraic semiotics from static signs to dynamic signs, including interactive interfaces, animated visualizations, virtual environments, etc. Our approach requires that the cognitive and social dimensions of this extension should also be addressed. These issues can be explore using Gibson’s notion of affordance, which he defined as “a capability for a specific kind of action, involving an animal and a part of its environment” [3]. For example, a BACK button on a browser provides an affordance for returning to the previously viewed page. Werner Kuhn has used semiotic morphisms, Gibsonian affordances, and blending to develop semantics for geographic information system interfaces [13].

3 Some Examples

Because a major intuition of semiotic morphisms is that they should preserve what is most important, it may be surprising that, if there is a conflict between structure and content (e.g., because not all the data can be displayed at once), it is more important to preserve structure than content. This is nicely illustrated by Figure 1, a code browser built at Bell Labs, because its content – the code of some program – has been sacrificed in favor of its structure, which is its division into files and procedures; there is also one key attribute, which is the age of the code. Two spatial dimensions are used to represent structure, and color is very effectively used to represent age. (The superimposed window on the bottom gives an overview of the whole program, plus a closeup showing some actual text. This illustrates the overview and zoom features of the system.)

Without knowing the use of this system, it is impossible to know how appropriate its representation really is. Still, we can infer from the display that the designer thought that the age of code was the most important attribute, presumably because of its value in debugging. However, such a tool would be even more useful if it could be configured to highlight with colors a variety of features of interest for a variety of problems; such features might include references to certain variables, certain uses of pointers, some kinds of recursion, etc. (e.g., consider what might be needed to work on the Y2K problem).

Figure 2, FilmFinder, from Ben Shneiderman’s group at the University of Maryland (see [21]) is a display for films, with the vertical axis indicating popularity, the horizontal axis indicating date, and color indicating genre; the area on the right side is for controlling the system. We can see this

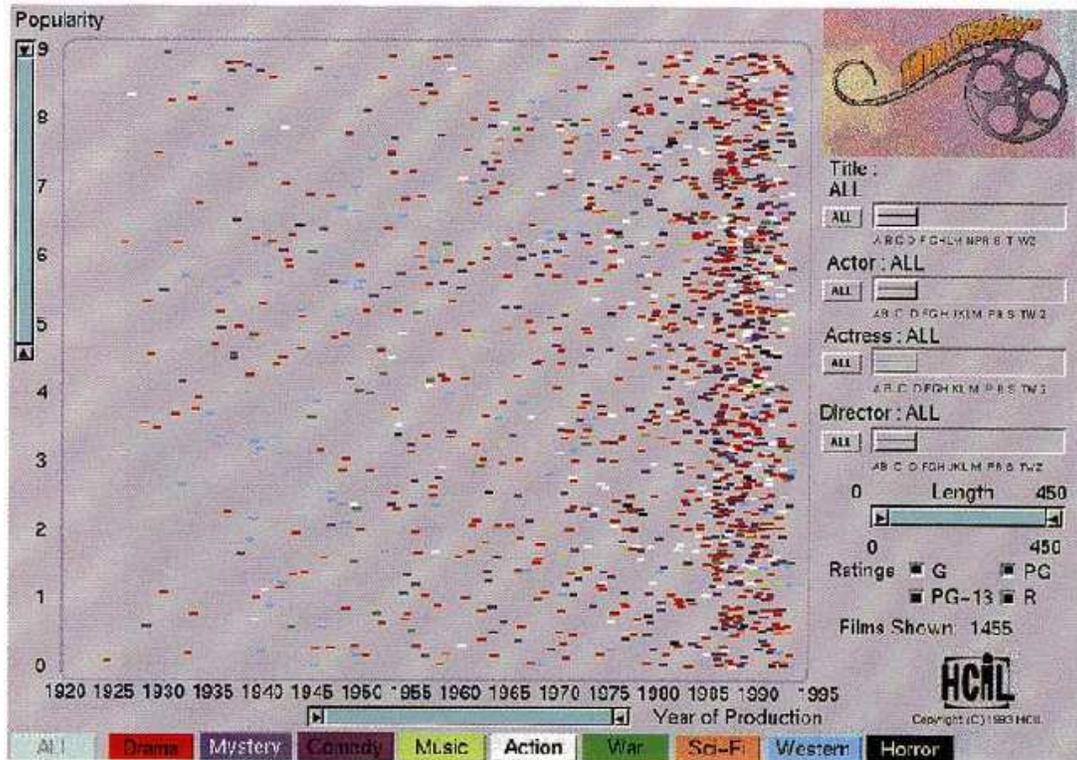


Figure 2: FilmFinder

can observe a clustering at around 90 minutes length, and we can again observe that there are too many dots to be useful, even though the display cuts off at 1990! If the user is looking for a particular film or class of films, she will have to narrow the focus by imposing additional constraints, and this single display does not give us enough information to know how effectively that can be done. We may presume that the (possibly imaginary) user who created this display thought that these particular attributes were the most interesting at a certain point during a sequence of displays constituting a search; but in fact, they do not seem particularly useful.

We can also infer what the designer of this version thought would be most important, by examining the controls on the right of the display; we may hope that these were determined by polling an adequate pool of typical users, but the key issue should be how easy it is to use these controls in scenarios that have been found to be of particular importance. Presumably typical users are more likely to be looking for a good video to rent, than they are to be analyzing trends in the movie industry. So once again, the controls should reflect the key features involved in typical searches, rather than just the most important attributes of films in general. It would take some experimental work to determine what these key search relevant attributes might be. But we can still criticize the design of the control console, because of its exclusive focus on simple attributes instead of structure. And we can criticize the fine grain control given to users over length and year, suggesting instead that soft constraints would be more appropriate; it also seems doubtful that length is a highly significant attribute for search. We can also criticize its design philosophy, advocating instead a more socially oriented approach that relates the profile of one user to the profiles of other users to select films that similar users have found interesting (there are numerous variations on this theme, such as listing films that a user's friends have liked). Finally, we can note that the design ideas proposed to improve

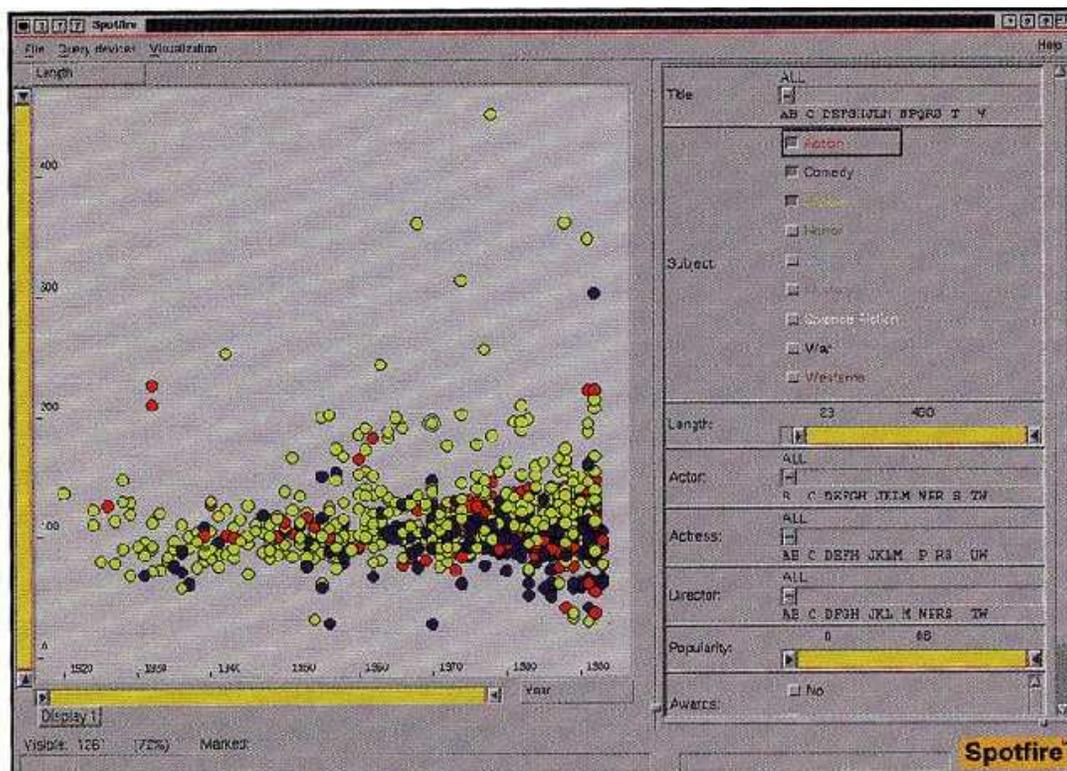


Figure 3: The SpotFire version of FilmFinder

the previous version of this system still apply to this version.

4 Discussion

As the examples illustrate, it is often best to apply algebraic semiotics in an informal way, calling on precise definitions only when needed for difficult design decisions, and otherwise using the formal framework mainly as a way to guide the analysis. The examples also illustrate that by drawing on even a little relevant theory, we can pinpoint significant deficiencies and suggest improvements. The “world famous” UCSD Semiotic Zoo [5] displays a number of other graphical designs, and uses algebraic semiotics to analyze their deficiencies.

Measuring quality by what is preserved and how it is preserved seems a novel idea, at least when formulated with the precision suggested here. The claim that it is more important to preserve structure than content when a trade-off is forced, has surprised even some user interface design professionals, although it is in the literature for many special cases. The need to take account of social issues in user interface design, e.g., in the discussion of Figure 3, is also surprising to many people; this need is the reason that our version of semiotics is not just algebraic but also social. Another non-obvious result is that preserving high level sorts is more important than preserving priorities. Of course, not all these results are unique to algebraic semiotics. The importance of social factors in HCI is the focus of the entire subfield of CSCW, and many good papers exploring various aspects of this issue have been produced. Also, many examples where content has been sacrificed in favor of form can be found in the books of Edward Tufte, e.g., [22].

References

- [1] Gilles Fauconnier and Mark Turner. Conceptual integration networks. *Cognitive Science*, 22(2):133–187, 1998.
- [2] Harold Garfinkel. *Studies in Ethnomethodology*. Prentice-Hall, 1967.
- [3] James Gibson. The theory of affordances. In Robert Shaw and John Bransford, editors, *Perceiving, Acting and Knowing: Toward an Ecological Psychology*. Erlbaum, 1977.
- [4] Joseph Goguen. Semiotic morphisms, 1996. www.cs.ucsd.edu/users/goguen/papers/smm.html. Early version in *Proc., Conf. Intelligent Systems: A Semiotic Perspective, Vol. II*, ed. J. Albus, A. Meystel and R. Quintero, Nat. Inst. Science & Technology, pages 26–31.
- [5] Joseph Goguen. UCSD Semiotic Zoo, 1996–2001. www.cs.ucsd.edu/users/goguen/zoo/.
- [6] Joseph Goguen. Towards a social, ethical theory of information. In Geoffrey Bowker, Leigh Star, William Turner, and Les Gasser, editors, *Social Science, Technical Systems and Cooperative Work: Beyond the Great Divide*, pages 27–56. Erlbaum, 1997.
- [7] Joseph Goguen. An introduction to algebraic semiotics, with applications to user interface design. In Chrystopher Nehaniv, editor, *Computation for Metaphors, Analogy and Agents*, pages 242–291. Springer, 1999. Lecture Notes in Artificial Intelligence, Volume 1562.
- [8] Joseph Goguen. Social and semiotic analyses for theorem prover user interface design. *Formal Aspects of Computing*, 11:272–301, 1999. Special issue on user interfaces for theorem provers.
- [9] Joseph Goguen. CSE 271 website, Winter 2002. www.cs.ucsd.edu/users/goguen/courses/271.
- [10] Joseph Goguen and Charlotte Linde. Techniques for requirements elicitation. In Stephen Fickas and Anthony Finkelstein, editors, *Requirements Engineering '93*, pages 152–164. IEEE, 1993. Reprinted in *Software Requirements Engineering (Second Edition)*, ed. Richard Thayer and Merlin Dorfman, IEEE Computer Society, 1996.
- [11] Joseph Goguen and Grant Malcolm. A hidden agenda. *Theoretical Computer Science*, 245(1):55–101, August 2000. Also UCSD Dept. Computer Science & Eng. Technical Report CS97–538, May 1997.
- [12] Joseph Goguen, James Thatcher, and Eric Wagner. An initial algebra approach to the specification, correctness and implementation of abstract data types. In Raymond Yeh, editor, *Current Trends in Programming Methodology, IV*, pages 80–149. Prentice-Hall, 1978.
- [13] Werner Kuhn. Modeling semantics through blendings, 2002. Technical report, Inst. for Geoinformatics, Univ. Muenster.
- [14] George Lakoff. *Women, Fire and Other Dangerous Things: What categories reveal about the mind*. Chicago, 1987.
- [15] George Lakoff and Mark Johnson. *Metaphors We Live By*. Chicago, 1980.

- [16] George Lakoff and Rafael Núñez. *Where Mathematics Comes from: How the Embodied Mind Brings Mathematics into Being*. Basic Books, 2000.
- [17] Charles Saunders Peirce. *Collected Papers*. Harvard, 1965. In 6 volumes; see especially Volume 2: Elements of Logic.
- [18] Harvey Sacks. On the analyzability of stories by children. In John Gumpertz and Del Hymes, editors, *Directions in Sociolinguistics*, pages 325–345. Holt, Rinehart and Winston, 1972.
- [19] Harvey Sacks. *Lectures on Conversation*. Blackwell, 1992. Edited by Gail Jefferson.
- [20] Ferdinand de Saussure. *Course in General Linguistics*. Duckworth, 1976. Translated by Roy Harris.
- [21] Ben Shneiderman. *Designing the User Interface*. Addison Wesley, 1998. Third edition.
- [22] Edward Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 1983.
- [23] Mark Turner. The blending website. www.wam.umd.edu/mturn/WWW/blending.html.