PLANNING COHERENT MULTISENTENTIAL TEXT

Eduard H. Hovy
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292-6695, U.S.A.
HOVY@VAXA.ISI.EDU

Abstract

Though most text generators are capable of simply stringing together more than one sentence, they cannot determine which order will ensure a coherent paragraph. A paragraph is coherent when the information in successive sentences follows some pattern of inference or of knowledge with which the hearer is familiar. To signal such inferences, speakers usually use relations that link successive sentences in fixed ways. A set of 20 relations that span most of what people usually say in English is proposed in the Rhetorical Structure Theory of Mann and Thompson. This paper describes the formalization of these relations and their use in a prototype text planner that structures input elements into coherent paragraphs.

1 The Problem of Coherence

The example texts in this paper are generated by Penman, a systemic grammar-based generator with larger coverage than probably any other existing text generator. Penman was developed at ISI (see [Mann & Matthiessen 83], [Mann 83], [Matthiessen 84]). The input to Penman is produced by PEA (Programming Enhancement Advisor; see [Moore 87]), a program that inspects a user's LISP program and suggests enhancements. PEA is being developed to interact with the user in order to answer his or her questions about the suggested enhancements. Its theoretical focus is the production of explanations over extended interactions in ways that are superior to the simple goal-tree traversal of systems such as TYRESIAS ([Davis 76]) and MYCIN ([Shortliffe 76]).

Supported by DARPA contract MDA903 81 C0335.

In answer to the question *how does the system enhance a program?*, the following text (not generated by Penman) is not satisfactory:

> (a). *The system performs the enhancement. Before that, the system resolves conflicts. First, the system asks the user to tell it the characteristic of the program to be enhanced. The system applies transformations to the program. It confirms the enhancement with the user. It scans the program in order to find opportunities to apply transformations to the program.*

...because you have to work too hard to make sense of it. In contrast, using the same propositions (now rearranged and linked with appropriate connectives), paragraph (b) (generated by Penman) is far easier to understand:

> (b). *The system asks the user to tell it the characteristic of the program to be enhanced. Then the system applies transformations to the program. In particular, the system scans the program in order to find opportunities to apply transformations to the program. Then the system resolves conflicts. It confirms the enhancement with the user. Finally, it performs the enhancement.*

Clearly, you do not get coherent text simply by stringing together sentences, even if they are related — note especially the underlined text in (b) and its corresponding three propositions in (a).

The goal of this paper is to describe a method of planning paragraphs to be coherent while avoiding unintended spurious effects that result from the juxtaposition of unrelated pieces of text.

2 Text Structuring

This planning work, which can be called *text structuring*, must obviously be done before the actual generating of language can begin. Text structuring is one of a number of pre-generation text planning tasks. For some of the other tasks Penman has special-purpose domain-specific solutions. They include:

- **aggregation:** determining, for input elements, the appropriate level of detail (see [Hovy 87]), the scoping of sentences, and the use of connectives
- **reference:** determining appropriate ways of referring to items (see [Appelt 87a, 87b])
- **hypotheticals:** determining the introduction, scope, and closing of hypothesis contexts (spans of text in which some values are assumed, as in "if you want to go to the game, then ...")

The problem of text coherence can be characterized in specific terms as follows. Assuming that input elements are sentence- or clause-sized chunks of representation, the permutation set of the input elements defines the space of possible paragraphs. A simplistic, brute-force way to achieve coherent text would be to search this space and pick out the coherent paragraphs. This search would be factorially expensive. For example, in paragraph (b) above, the 7 input clusters received from PEA provide 7! = 5,040 candidate paragraphs. However, by utilizing the constraints imposed by coherence, one can formulate operators that guide the search and significantly limit the search to a manageable size. In the example, the operators described below produced only 3 candidate paragraphs. Then, from this set of remaining candidates, the best paragraph can be found by applying a relatively simple evaluation metric.

The contention of this paper is that, exercising proper care, the coherence relations that hold between successive pieces of text can be formulated as the abovementioned search operators and used in a hierarchical-expansion planner to limit the search and to produce structures describing the coherent paragraphs.

The illustrate this contention, the Penman text structurer is a simplified top-down planner (as described first by [Sacerdoti 77]). It uses a formalized version of the relations of Rhetorical Structure Theory (see immediately below) as plans. Its output is one (or more) tree(s) that describe the structure(s) of coherent paragraphs built from the input elements. Input elements are the leaves of the tree(s); they are sent to the Penman generator to be transformed into sentences.

3 Previous Approaches

The heart of the problem is obviously *coherence*. Coherent text can be defined as text in which the hearer knows how each part of the text relates to the whole; i.e., (a) the hearer knows why it is said, and (b) the hearer can relate the semantics of each part to a single overarching framework.

In 1978, Hobbs ([Hobbs 78, 79, 82]) recognized that in coherent text successive pieces of text are related in a specified set of ways. He produced a set of relations organized into four categories, which he postulated as the four types of phenomena that occur during conversation. His argument, unfortunately, contains a number of shortcomings; not only is the categorization not well-motivated, but the list of relations is incomplete.

In her thesis work, McKeown took a different approach ([McKeown 82]). She defined a set of relatively static schemas that represent the structure of stereotypical paragraphs for describing objects. In essence, these schemas are paragraph templates; coherence is enforced by the correct nesting and filling-in of templates. No explicit theory of coherence was offered.

Mann and Thompson, after a wide-ranging study involving hundreds of paragraphs, proposed that a set of 20 relations suffice to represent the relations that hold within the texts that normally occur in English ([Mann & Thompson 87, 86, 83]). These relations, called RST (rhetorical structure theory), are used recursively; the assumption (never explicitly stated) is that a paragraph is only coherent if all its parts can eventually be made to fit into one overarching relation. The enterprise was completely descriptive; no formal definition of the relations or justification for their completeness were given. However, the relations do include most of Hobbs's relations and support McKeown's schemas.

A number of similar descriptions exist. The description of how parts of purposive text can relate goes back at least to Aristotle ([Aristotle 54]). Both Grimes and Shepherd categorize typical intersentential relations ([Grimes 75] and [Shepherd 26]). Hovy ([Hovy 86]) describes a program that uses some relations to slant text.

4 Formalizing RST Relations

As defined by Mann and Thompson, RST relations hold between two successive pieces of text (at the lowest level, between two clauses; at the highest level, between two parts that make up a paragraph)[1]. Therefore, each relation has two parts, a *nucleus* and a *satellite*. To determine the applicability of the relation, each part has a set of constraints on the entities that can be related. Relations may also have requirements on the combination of the two parts. In addition, each relation has an effect field, which is intended to denote the conditions which the speaker is attempting to achieve.

In formalizing these relations and using them generatively to plan paragraphs, rather than analytically to describe paragraph structure, a shift of focus is required. Relations must be seen as plans — the operators that guide the search through the permutation space. The nucleus and satellite constraints become requirements that must be met by any piece of text before it can be used in the relation (i.e., before it can be coherently juxtaposed with the preceding text). The effect field contains a description of the intended effect of the relation (i.e., the goal that the plan achieves, if properly executed). Since the goals in generation are communicative, the intended effect must be seen as the inferences that the speaker is licensed to make about the hearer's knowledge after the successful completion of the relation.

Since the relations are used as plans, and since their satellite and nucleus constraints must be reformulated as subgoals to the structurer, these constraints are best represented in terms of the communicative intent of the speaker. That is, they are best represented in terms of what the hearer will know — i.e., what inferences the hearer would run — upon being told the nucleus or satellite filler.

As it turns out, suitable terms for this purpose are provided by the formal theory of rational interaction currently being developed by, among others, Cohen, Levesque, and Perrault. For example, in [Cohen & Levesque 85], Cohen and Levesque present a proof that the indirect speech act of requesting can be derived from the following basic modal operators

- **(BEL x p)** — p follows from x's beliefs
- **(BMB x y p)** — p follows from x's beliefs about what x and y mutually believe
- **(GOAL x p)** — p follows from x's goals
- **(AFTER a p)** — p is true in all courses of events after action a

as well as from a few other operators such as **AND** and **OR**. They then define *summaries* as, essentially, speech act operators with activating conditions (*gates*) and *effects*. These summaries closely resemble, in structure, the RST plans described here, with gates corresponding to satellite and nucleus constraints and effects to intended effects.

[1]This is not strictly true; a small number of relations, such as **Sequence**, relate more than two pieces of text. However, for ease of use, they have been implemented as binary relations in the structurer.

5 An Example

The RST relation **Purpose** expresses the relation between an action and its intended result:

Purpose
Nucleus Constraints:
1. (BMB S H (ACTION ?act-1))
2. (BMB S H (ACTOR ?act-1 ?agt-1))

Satellite Constraints:
1. (BMB S H (STATE ?state-1))
2. (BMB S H (GOAL ?agt-1 ?state-1))
3. (BMB S H (RESULT ?act-1 ?act-2))
4. (BMB S H (OBJ ?act-2 ?state-1))

Intended Effects:
1. (BMB S H (BEL ?agt-1 (RESULT ?act-1 ?state-1)))
2. (BMB S H (PURPOSE ?act-1 ?state-1))

For example, when used to produce the sentence *The system scans the program in order to find opportunities to apply transformations to the program*, this relation is instantiated as

Purpose
Nucleus Constraints:
1. (BMB S H (ACTION SCAN-1))
   The program is scanned
2. (BMB S H (ACTOR SCAN-1 SYS-1))
   The system scans it

Satellite Constraints:
1. (BMB S H (STATE OPP-1))
   Opportunities to apply transformations exist
2. (BMB S H (GOAL SYS-1 OPP-1))
   The system "wants" to find them
3. (BMB S H (RESULT SCAN-1 FIND-1))
   Scanning will result in finding
4. (BMB S H (OBJ FIND-1 OPP-1))
   the opportunities

Intended Effects:
1. (BMB S H (BEL SYS-1 (RESULT SCAN-1 OPP-1)))
   The system "believes" that scanning will disclose the opportunities
2. (BMB S H (PURPOSE SCAN-1 OPP-1))
   This is the purpose of the scanning

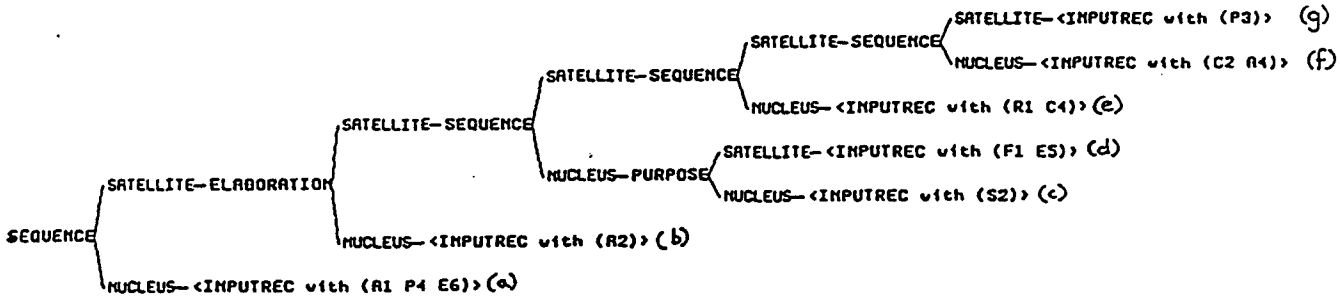

Figure 1: Paragraph Structure Tree

The elements SCAN-1, OPP-1, etc., are part of a network provided to the Penman structurer by PEA. These elements are defined as propositions in a property-inheritance network of the usual kind written in NIKL ([Schmolze & Lipkis 83], [Kaczmarek et al. 86]), a descendant of KL-ONE ([Brachman 78]). Some input for this example sentence is:

```
(PEA-SYSTEM SYS-1)      (OPPORTUNITY OPP-1)
(PROGRAM PROG-1)        (ENABLEMENT ENAB-5)
(SCAN SCAN-1)             (DOMAIN ENAB-5 OPP-1)
  (ACTOR SCAN-1 SYS-1)    (RANGE ENAB-5 APPLY-3)
  (OBJ SCAN-1 PROG-1)   (APPLY APPLY-3)
  (RESULT SCAN-1 FIND-1) (ACTOR APPLY-3 SYS-1)
(FIND FIND-1)             (OBJ APPLY-3 TRANS-2)
  (ACTOR FIND-1 SYS-1)    (RECIP APPLY-3 PROG-1)
  (OBJ FIND-1 OPP-1)    (TRANSFORMATION TRANS-2)
```

The relations are used as plans; their intended effects are interpreted as the goals they achieve. In other words, in order to bring about the state in which both speaker and hearer know that OPP-1 is the purpose of SCAN-1 (and know that they both know it, etc.), the structurer uses **Purpose** as a plan and tries to satisfy its constraints.

In this system, constraints and goals are interchangable; for example, in the event that (RESULT SCAN-1 FIND-1) is believed not known by the hearer, satellite constraint 3 of the **Purpose** relation simply becomes the goal to achieve (BMB S H (RESULT SCAN-1 FIND-1)). Similarly, the propositions (BMB S H (RESULT SCAN-1 ?ACT-2)) (BMB S H (OBJ ?ACT-2 OPP-1)) are interpreted as the goal to find some element that could legitimately take the place of ?ACT-2.

In order to enable the relations to nest recursively, some relations' nucleuses and satellites contain requirements that specify additional relations, such as examples, contrasts, etc. Of course, these additional requirements may only be included if such material can coherently follow the content of the nucleus or satellite. The question of ordering such additional constituents is still under investigation. The question of whether such additional material should be included at all is not addressed; the structurer tries to say everything it is given.

The structurer produces all coherent paragraphs (that is, coherent as defined by the relations) that satisfy the given goal(s) for any set of input elements. For example, paragraph (b) is produced to satisfy the initial goal (BMB S H (SEQUENCE ASK-1 ?NEXT)). This goal is produced by PEA, together with the appropriate representation elements (ASK-1, SCAN-1, etc.) in response to the question *how does the system enhance a program?*. Different initial goals will result in different paragraphs.

Each paragraph is represented as a tree in which branch points are RST relations and leaves are input elements. Figure 1 is the tree for paragraph (b). It contains the relations **Sequence** (signalled by "then" and "finally"), **Elaboration** ("in particular"), and **Purpose** ("in order to"). In the corresponding paragraph produced by Penman, the relations' characteristic words or phrases (boldfaced below) appear between the blocks of text they relate:

> [*The system asks the user to tell it the characteristic of the program to be enhanced.*]$_{(a)}$ **Then** [*the system applies transformations to the program.*]$_{(b)}$ **In particular,** [*the system scans the program*]$_{(c)}$ **in order to** [*find opportunities to apply transformations to the program.*]$_{(d)}$ **Then** [*the system resolves conflicts.*]$_{(e)}$ [*It confirms the enhancement with the user.*]$_{(f)}$ **Finally,** [*it performs the enhancement.*]$_{(g)}$

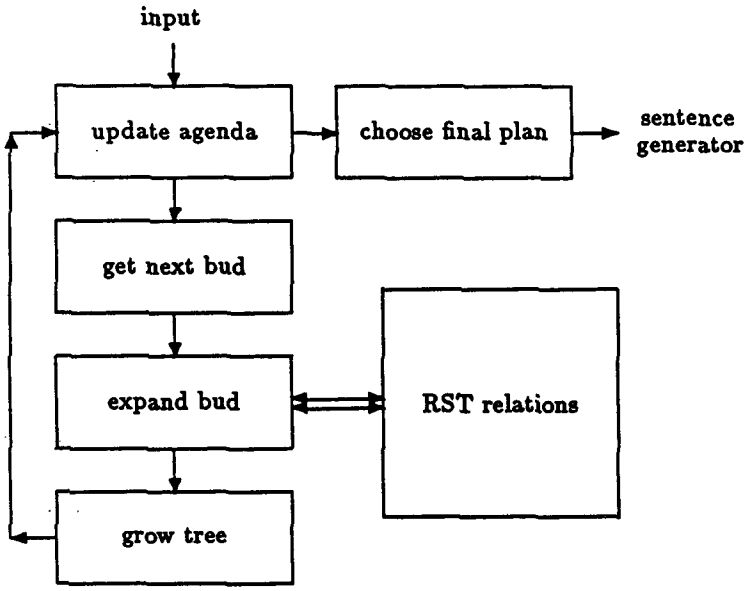

Figure 2: Hierarchical Planning Structurer

6 The Structurer

As stated above, the structurer is a simplified top-down hierarchical expansion planner (see Figure 2). It operates as follows: given one or more communicative goals, it finds RST relations whose intended effects match (some of) these goals; it then inspects which of the input elements match the nucleus and subgoal constraints for each relation. Unmatched constraints become subgoals which are posted on an agenda for the next level of planning. The tree can be expanded in either depth-first or breadth-first fashion. Eventually, the structuring process bottoms out when either: (a) all input elements have been used and unsatisfied subgoals remain (in which case the structurer could request more input with desired properties from the encapsulating system); or (b) all goals are satisfied. If more than one plan (i.e., paragraph tree structure) is produced, the results are ordered by preferring trees with the minimum unused number of input elements and the minimum number of remaining unsatisfied subgoals. The best tree is then traversed in left-to-right order; leaves provide input to Penman to be generated in English and relations at branch points provide typical interclausal relation words or phrases. In this way the structurer performs top-down goal refinement down to the level of the input elements.

7 Shortcomings and Further Work

This work is also being tested in a completely separate domain: the generation of text in a multimedia system that answers database queries. Penman produces the following description of the ship Knox (where CTG 070.10 designates a group of ships):

> (c). *Knox is en route in order to rendezvous with CTG 070.10, arriving in Pearl Harbor on 4/24, for port visit until 4/30.*

In this text, each clause (en route, rendezvous, arrive, visit) is a separate input element; the structurer linked them using the relations **Sequence** and **Purpose** (the same **Purpose** as shown above; it is signalled by "in order to"). However, Penman can also be made to produce

> (d). *Knox is en route in order to rendezvous with CTG 070.10. It will arrive in Pearl Harbor on 4/24. It will be on port visit until 4/30.*

The problem is clear: how should sentences in the paragraph be scoped? At present, avoiding any claims about a theory, the structurer can feed

Penman either extreme: make everything one sentence, or make each input element a separate sentence. However, neither extreme is satisfactory; as is clear from paragraph (b), "short" spans of text can be linked and "long" ones left separate. A simple way to implement this is to count the number of leaves under each branch (nucleus or satellite) in the paragraph structure tree.

Another shortcoming is the treatment of input elements as indivisible entities. This shortcoming is a result of factoring out the problem of aggregation as a separate text planning task. Chunking together input elements (to eliminate detail) or taking them apart (to be more detailed) has received scant mention — see [Hovy 87], and for the related problem of paraphrase see [Schank 75] — but this task should interact with text structuring in order to provide text that is both optimally detailed and coherent.

At the present time, only about 20% of the RST relations have been formalized to the extent that they can be used by the structurer. This formalization process is difficult, because it goes hand-in-hand with the development of terms with which to characterize the relations' goals/constraints. Though the formalization can never be completely finalized — who can hope to represent something like motivation or justification complete with all ramifications? — the hope is that, by having the requirements stated in rather basic terms, the relations will be easily adaptable to any new representation scheme and domain. (It should be noted, of course, that, to be useful, these formalizations need only be as specific and as detailed as the domain model and representation requires.) In addition, the availability of a set of communicative goals more detailed than just say or ask (for example), should make it easier for programs that require output text to interface with the generator. This is one focus of current text planning work at ISI.

8 Acknowledgments

For help with Penman, Robert Albano, John Bateman, Bob Kasper, Christian Matthiessen, Lynn Poulton, and Richard Whitney. For help with the input, Bill Mann and Johanna Moore. For general comments, all the above, and Cecile Paris, Stuart Shapiro, and Norm Sondheimer.

9 References

1. Appelt, D.E., 1987a.
A Computational Model of Referring, SRI Technical Note 409.

2. Appelt, D.E., 1987b.
Towards a Plan-Based Theory of Referring Actions, in *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, Kempen, G. (ed), (Kluwer Academic Publishers, Boston) 63-70.

3. Aristotle, 1954.
The Rhetoric, in *The Rhetoric and the Poetics of Aristotle*, W. Rhys Roberts (trans), (Random House, New York).

4. Brachman, R.J., 1987.
A Structural Paradigm for Representing Knowledge, Ph.D. dissertation, Harvard University; also BBN Research Report 3605.

5. Cohen, P.R. & Levesque, H.J., 1985.
Speech Acts and Rationality, *Proceedings of the ACL Conference*, Chicago (49-59).

6. Davis, R., 1976.
Applications of Meta-Level Knowledge to the Constructions, Maintenance, and Use of Large Knowledge Bases, Ph.D. dissertation, Stanford University.

7. Grimes, J.E., 1975.
The Thread of Discourse (Mouton, The Hague).

8. Hobbs, J.R., 1978.
Why is Discourse Coherent?, SRI Technical Note 176.

9. Hobbs, J.R., 1979.
Coherence and Coreference, in *Cognitive Science* 3(1), 67-90.

10. Hobbs, J.R., 1982.
Coherence in Discourse, in *Strategies for Natural Language Processing*, Lehnert, W.G. & Ringle, M.H. (eds), (Lawrence Erlbaum Associates, Hillsdale NJ) 223-243.

11. Hovy, E.H., 1986.
Putting Affect into Text, *Proceedings of the Cognitive Science Society Conference*, Amherst (669-671).

12. Hovy, E.H., 1987.
Interpretation in Generation, *Proceedings of the AAAI Conference*, Seattle (545-549).

13. Kaczmarek, T.S., Bates, R. & Robins, G., 1986.
Recent Developments in NIKL, *Proceedings of the AAAI Conference*, Philadelphia (978-985).

14. Mann, W.C., 1983.
An Overview of the Nigel Text Generation Grammar, USC/Information Sciences Institute Research Report RR-83-113.

15. Mann, W.C. & Matthiessen, C.M.I.M., 1983.
Nigel: A Systemic Grammar for Text Generation, USC/Information Sciences Institute Research Report RR-83-105.

16. Mann, W.C. & Thompson, S.A., 1983.
Relational Propositions in Discourse, USC/-Information Sciences Institute Research Report RR-83-115.

17. Mann, W.C. & Thompson, S.A., 1986.
Rhetorical Structure Theory: Description and Construction of Text Structures, in *Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics*, Kempen, G. (ed), (Kluwer Academic Publishers, Dordrecht, Boston MA) 279-300.

18. Mann, W.C. & Thompson, S.A., 1987.
Rhetorical Structure Theory: A Theory of Text Organization, USC/Information Sciences Institute Research Report RR-87-190.

19. Matthiessen, C.M.I.M., 1984.
Systemic Grammar in Computation: the Nigel Case, USC/Information Sciences Institute Research Report RR-84-121.

20. McKeown, K.R., 1982.
Generating Natural Language Text in Response to Questions about Database Queries, Ph.D. dissertation, University Of Pennsylvania.

21. Moore, J.D., 1988.
Enhanced Explanations in Expert and Advice-Giving Systems, USC/Information Sciences Institute Research Report (forthcoming).

22. Sacerdoti, E., 1977.
A Structure for Plans and Behavior (North-Holland, Amsterdam).

23. Schank, R.C., 1975.
Conceptual Information Processing, (North-Holland, Amsterdam).

24. Schmolze, J.G. & Lipkis, T.A., 1983.
Classification in the KL-ONE Knowledge Representation System, *Proceedings of the IJCAI Conference*, Karlsruhe (330-332).

25. Shepherd, H.R., 1926.
The Fine Art of Writing, (The Macmillan Co, New York).

26. Shortliffe, E.H., 1976.
Computer-Based Medical Consultations: MYCIN.