

## INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

**The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.**

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

# UMI

A Bell & Howell Information Company  
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA  
313/761-4700 800/521-0600



**CONTENT ANALYTIC CLASSIFICATION  
IN ELECTRONIC BRAINSTORMING:  
A STRUCTURAL MODEL AND EMPIRICAL ANALYSIS**

A Dissertation

by

DAVID ANDREW CHESLOW

Submitted to the Office of Graduate Studies of  
Texas A&M University  
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 1995

Major Subject: Business Analysis

**UMI Number: 9615788**

---

**UMI Microform 9615788**  
**Copyright 1996, by UMI Company. All rights reserved.**

**This microform edition is protected against unauthorized  
copying under Title 17, United States Code.**

---

**UMI**  
**300 North Zeeb Road**  
**Ann Arbor, MI 48103**

CONTENT ANALYTIC CLASSIFICATION  
IN ELECTRONIC BRAINSTORMING:  
A STRUCTURAL MODEL AND EMPIRICAL ANALYSIS

A Dissertation

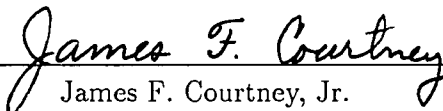
by


DAVID ANDREW CHESLOW


Submitted to Texas A&M University  
in partial fulfillment of the requirements  
for the degree of

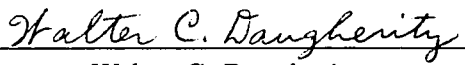
DOCTOR OF PHILOSOPHY

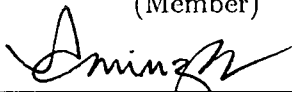
Approved as to style and content by:


  
James F. Courtney, Jr.  
(Co-Chair of Committee)

  
William L. Fuerst  
(Co-Chair of Committee)

  
Gerald R. Wagner  
(Member)

  
Walter C. Daugherty  
(Member)

  
Ajay S. Vinze  
(Member)

  
Frank P. Buffa  
(Head of Department)

December 1995

Major Subject: Business Analysis

## ABSTRACT

Content Analytic Classification in Electronic Brainstorming: A

Structural Model and Empirical Analysis. (December 1995)

David Andrew Cheslow, B.S., Radford University;

M.B.A., Virginia Polytechnic Institute & State University

Co-Chairs of Advisory Committee: Dr. James F. Courtney, Jr.  
Dr. William L. Fuerst

A model of electronic brainstorming communication structure is developed and tested. This model integrates and extends the Issue Based Information System (IBIS) and Interactive Process Analysis (IPA) theories. A computer system to store content analytic knowledge was developed. This computer system improves upon existing systems by storing the data and content analysis procedures in a relational database format which allows *ad hoc* access to relationships between elements in the knowledge base. A computer system was developed to make inferences about statements from electronic brainstorming transcripts (or other textual sources) using the aforementioned content analysis knowledge base. This inference engine outperforms an it predecessor by using a more conservative "forward tagging logic" procedure. The performance of an existing content analysis dictionary for categorizing electronic brainstorming transcript statements into the developed model was evaluated. Two studies were conducted to evaluate the reliability and validity of the results of content analysis on electronic brainstorming transcripts. Content analysis and correspondence analysis were used to determine whether or not statements in electronic brainstorming transcripts could be classified as issues, evaluative statements, restatements, or unrelated comments.

in memory of  
*Harold Taylor Pruitt*  
and for his namesake  
*Adam Taylor Cheslow*

## TABLE OF CONTENTS

	Page
INTRODUCTION . . . . .	1
BACKGROUND . . . . .	6
1 Electronic Brainstorming . . . . .	6
2 A Structural Model for EBS . . . . .	8
3 Content Analysis . . . . .	15
4 The Research Question . . . . .	24
METHODOLOGY AND RESULTS . . . . .	26
1 Pilot Studies . . . . .	26
2 Study #1 . . . . .	27
3 Study #2 . . . . .	45
SUMMARY AND CONCLUSION . . . . .	53
1 Contributions . . . . .	56
2 Limitations . . . . .	59
3 Future Research . . . . .	60
REFERENCES . . . . .	63
APPENDIX	
A THE DBMS(3) SYSTEM . . . . .	69
A.1 Introduction . . . . .	69
A.2 Schema Specification . . . . .	70
A.3 Application Interface . . . . .	73
A.4 Programmer Notes . . . . .	77
A.5 Future Development . . . . .	83
B LEXNET . . . . .	86
B.1 Function Overview . . . . .	86
B.2 Detailed Memory Representation of a Sentence . . . . .	87
C THE LEXNET KNOWLEDGE BASE SCHEMA . . . . .	90



APPENDIX	Page
D HARVARD IV'S NEW LEASE ON LIFE . . . . .	95
D.1 The LexNet Knowledge Base . . . . .	97
D.2 The Disambiguator . . . . .	106
D.3 Verification . . . . .	118
E ANALYSIS OF GI VERSUS LEXNET OUTPUT . . . . .	138
F QUESTIONNAIRE . . . . .	149
F.1 Instructions . . . . .	149
F.2 Demographic Questions . . . . .	150
F.3 MBA Success Factors . . . . .	151
F.4 Definition of Quality . . . . .	161
F.5 Customer Service . . . . .	167
F.6 Product Benefits . . . . .	173
F.7 Business Problem . . . . .	179
F.8 Staff Evaluation . . . . .	186
F.9 Business Prospect . . . . .	193
G SOURCE CODE LISTINGS (DIGITAL) . . . . .	200
G.1 DBMS . . . . .	201
G.2 DICTION . . . . .	202
G.3 CONVERT . . . . .	202
G.4 CONVERT . . . . .	202
G.5 SHOWWORD . . . . .	202
G.6 XREF . . . . .	202
VITA . . . . .	203

## LIST OF TABLES

Table	Page
1 The MMEBS node types. . . . .	14
2 Agreement frequency summary for study #1. . . . .	45
3 Cohen's kappa for each pair of experts. . . . .	50
4 Agreement frequency summary for study #2. . . . .	52

## LIST OF FIGURES

Figure		Page
1	The set of legal rhetorical moves in IBIS. . . . .	9
2	The categories of Bales' Interactive Process Analysis model. . . . .	11
3	The MMEBS structural model. . . . .	13
4	Generalized content analytic process. . . . .	16
5	The communications system model. . . . .	16
6	Frequency of ratings by expert. . . . .	37
8	Frequency of combined ratings by transcript. . . . .	38
9	Frequency of combined ratings by part. . . . .	39
10	Correspondence analysis (expert $\times$ transcript $\times$ part $\times$ rating). . . . .	41
11	Correspondence analysis (expert $\times$ transcript_part $\times$ combined rating). . . . .	43
12	Cumulative percent agreement on similarity by score. . . . .	44
13	Frequency of ratings by experience as a participant. . . . .	48
14	Frequency of ratings by expert. . . . .	49
15	Frequency of ratings by transcript. . . . .	50
16	Frequency of ratings by part. . . . .	51
17	A portion of a database schema. . . . .	74
18	Conversion between DBT data elements and 'C' data structures. . . . .	78
19	Primary versus other key DBTs. . . . .	81
20	The Harvard IV-4 CONTEXT program for the word <i>love</i> . . . . .	98
21	The LexNet CONTEXT program for the word <i>love</i> . . . . .	99

Figure	Page
22 The Harvard IV-4 CONTEXT program for the word <i>are</i> . . . . .	102
23 The Entity-Relationship Diagram for the LexNet knowledge base. . . . .	104
24 Sample output from the Inettrans program. . . . .	109
25 Memory representation of a sentence. . . . .	110
26 The LexNet disambiguation process. . . . .	116
27 Trace of a simple disambiguation . . . . .	120
28 Trace of a complicated disambiguation . . . . .	123

## INTRODUCTION

Brainstorming is a process in which the free flow of ideas from group participants is encouraged and evaluation of the thoughts expressed by the group members is delayed [45]. Typically a group coordinator, called a facilitator, has the responsibility for documenting and organizing the thoughts expressed by the members of the group as well as encouraging the group members to think freely [31]. The use of computers to support idea generation in group processes has received considerable attention in recent years. Referred to generically as electronic brainstorming, these tools have been shown to: allow group size to increase ([11], [40]); produce less inhibited expression of thoughts and increase participation ([29], [66], [10]); produce more ideas [29]; increase user satisfaction [40]; improve efficiency [41] and, in some cases, produce higher quality decisions ([29], [13], [50], [10]) over their non-computer-supported counterparts. To date, computer support for brainstorming has concentrated on the documentation aspect of the facilitator's role; little attention has been given to organizing the thoughts expressed by the group members. This dissertation investigates the application of content analysis to the organization of thoughts expressed during electronically-supported idea generation.

The theoretical foundation upon which brainstorming is based is simple: premature evaluation of ideas encourages group members to be inhibited about the expression of incomplete or unsupportable ideas. By delaying evaluation and encouraging the expression of every thought, however fantastic or nonsensical, a broader base of potential courses of action will be identified. Only after this broad base has been established should the ideas be evaluated for feasibility, practicality, or value [45]. Osborn [45] suggests that in order to achieve this freewheeling environment for idea generation, five general guidelines should be followed: all ideas should be anonymous; judgmental statements should be discouraged (particularly negative judgmental statements); quantity should be sought over quality; anything goes, ideas should not be

---

This dissertation follows the formatting conventions of Decision Support Systems.

constrained by what is possible, practical, or feasible; and, the group size should be kept relatively small.

Generally one individual, called the facilitator, is responsible for: recording and organizing the thoughts expressed by the group members, enforcing the guidelines listed above, and keeping the group moving. In manual brainstorming, the facilitator writes down (on large pieces of paper attached to the walls of the meeting room) all of the ideas expressed by the group, clustering together those ideas which seem related. The facilitator is typically not a member of the group itself and no specific knowledge of the subject matter is required by the facilitator (although an understanding of the terminology used by the group is helpful). The facilitator does not contribute content, but partially controls the process.

When computer communications technology is used to enhance the process of generating ideas in a group setting, the process is called electronic brainstorming (EBS). The key feature of EBS is that the members of a group can simultaneously generate ideas or thoughts without the information loss which would occur in a verbal exchange ([17], [40]). The computer network replaces the spoken word as the medium for communication, recording each message as it is transmitted. Group members can focus their attention on the formulation and expression of their idea without fear of missing what is being presented by another participant. Also, each member can respond to or reflect on the thoughts expressed by other group members at his or her own pace without slowing down the entire group.

Electronic brainstorming supports the generative portion of a group process. EBS is usually used in conjunction with other electronic meeting tools designed to support evaluative processes. A typical electronically-supported group process includes several generative phases interspersed with evaluative phases [17].

One of the side effects of simultaneous input is that it is possible (in fact, it frequently happens) that several participants will express very similar thoughts at the same time. Furthermore, participants may build upon the expressed thoughts of other group members, suggesting subtle variations on a single theme. While these incremental alterations are valuable and should be encouraged [45], they create a

problem during the evaluation phase. An example will help clarify. Assume that three participants have each suggested a similar solution to a problem. When the ideas are evaluated, the group may be evenly distributed as to which of the three ideas is best. The totals will suggest that there is a low level of interest in each of the ideas. But this interpretation is false because the three ideas represent concern with a single issue which, when aggregated, has a high level of interest. Similar problems occur with other evaluative techniques.

Despite the simultaneity of participant input, the EBS process is usually treated as a linear process in time. Two methodologies are common in current EBS systems: all thoughts are added to the group output in chronological order (e.g., SAMM<sup>®</sup>), or the author may insert a thought into the group output at a position of his/her choice (e.g., VisionQuest<sup>®</sup>). These schemes, and variants of them, are discussed by Gray and Olfman [17]. While the latter of these methods is clearly preferable for categorizing ideas, it does not allow group participants to focus their attention on a single issue. Even with the input of the participants grouped according to each author's intentions, the participants are faced with a conglomeration of all the issues being addressed by the group at all times. Plexsys<sup>®</sup> deals with this problem by building files, each pertaining to a single issue, which users can build upon. Unfortunately, the participants do not control the issue which they will address [41]. This method, while ostensibly called brainstorming, is conceptually much closer to a method called idea writing (see Moore [36] for a description of the idea writing methodology). In a complex EBS session the number of issues addressed may become large, causing participants to become overloaded with information [26].

EBS has typically been presented as a problem-solving or decision-making aid; the commonly used acronym GDSS (Group Decision Support System) reflects this perspective. This view, however, fails to consider the use of EBS as a communications

---

SAMM is a trademark of the University of Minnesota, Minneapolis, Minnesota.  
VisionQuest is a trademark of OmniQuest Inc., Dallas, Texas.  
Plexsys is a trademark of the University of Arizona, Tucson, Arizona (now under contract with IBM as GroupSystems<sup>®</sup>).

medium. Some research exists which suggests that only a relatively small proportion of the messages exchanged in an EBS session are directly issue related ([9], [67]). The remainder of the messages are a result of the EBS tool being used as a communication channel rather than as a structured problem solving tool. As a result, a significant number of the messages generated during an EBS session do not need to be included in the evaluation process. Examples of communicative messages might include: "I don't understand what you mean," "Let's take a break now," or "Can anyone think of a reason why this idea wouldn't work?"

Also due to the historical emphasis on EBS as a problem-solving tool (i.e., as a decision support system), EBS sessions have been considered to be isolated from one another. In reality, however, many group processes continue across time; issues resolved in one EBS session may spawn new issues which, in turn, spawn still more issues [41]. Semantic browsers have been suggested as one tool for accessing the organizational memory represented in EBS transcripts [59]. A semantic browser searches transcripts for a specified word or phrase and allows the user to view the context(s) in which that word or phrase was used. Unfortunately, such tools require significant *a priori* knowledge about the content of previous EBS sessions and the terminology which might have been used in those sessions. It is still difficult, using current EBS systems, to relate the resolution of a particular issue with the broader context in which that issue became important and to determine whether or not another group or session has addressed a similar issue in the past.

Four major problems with current EBS systems have been identified thus far: redundant messages may result in misleading evaluative results; an inability to filter, condense and group messages may result in information overload; identification of messages suitable for evaluation is tedious; and linkages between EBS sessions are difficult to maintain and search. These problems are a reflection of the lack of structure in EBS. Because EBS transcripts are typically stored as simple text, nearly all of the semantic processing and understanding must be performed by the user. Without an underlying structural model and a consistent methodology for classifying thoughts



within that model, EBS becomes little more than a real-time electronic mail facility. This dissertation addresses these issues by presenting a comprehensive structural model for EBS communications, empirically testing the applicability of that model to actual EBS transcripts, and testing the feasibility of using content analysis as a methodology for automating the process of categorizing expressed thoughts within that structural model. In so doing, potentially valuable tools for solving the four problems mentioned above may be discovered.

The structural model developed in this dissertation symbolically represents the relationships between the component statement types which make up a typical brainstorming transcript<sup>†</sup>. Content Analytic tools are used to qualitatively support or refute the existence of each component statement type.

---

<sup>†</sup> The definition of structural model adopted here should not be confused with Structural Equation Models which use mathematical notation and are quantitative in nature.

## BACKGROUND

### 1 Electronic Brainstorming

EBS works in a manner similar to conventional brainstorming. Group members type their ideas at a terminal or personal computer. The ideas are transmitted to a network fileserver which distributes the ideas entered to the other group participants. Thus, every group member can see the ideas expressed by the other group members. The precise mechanism by which this distribution takes place varies from system to system; in SAMM<sup>®</sup>, all ideas are displayed on an overhead projection monitor, while VisionQuest<sup>®</sup> provides group members with their own copy of the entire list of ideas which may be independently scrolled. The latter arrangement has two advantages over the former: group members can be spatially dispersed, and group members can review preceding group ideas at will.

In keeping with the guidelines of brainstorming, as set forth by Osborn, many EBS tools allow group members to contribute anonymously, either totally anonymous (e.g., VisionQuest<sup>®</sup>) or anonymous from other participants, but not from the facilitator (e.g., Plexsys<sup>®</sup>, SAMM<sup>®</sup>). As previously stated, anonymity was identified by Osborn as a key to reducing the inhibitions of group process participants. Significant research effort has been expended to validate this claim in an EBS setting; see Hiltz [21] and Connolly, Jessup and Valacich [9]. The conclusions of these studies have lent credence to Osborn's hypothesis that anonymous participants are more likely to express their true feelings than are identifiable participants. To date, no EBS systems have attempted to enforce the guideline that evaluative comments should be restricted. Plexsys can be further differentiated from the systems examined in this dissertation in that the idea generation process in that system differs from brainstorming as originally developed by Osborn. The Plexsys system forces a structure on the idea generation process in which participants are restricted to a single subtopic of the discussion at any given time.

Rawlinson [46] found that a group size of twelve was optimal for conventional brainstorming and that groups larger than twenty result in decreased per person productivity due to competition for “speaking time.” Since all participants may input ideas simultaneously under EBS, there is no competition for speaking time and, therefore, group size could presumably increase. Fellers [11] supports this belief, finding that the presence of an EBS tool allows group size to increase without a loss of productivity per person. Nunamaker, Vogel and Konsynski [40] also found evidence to suggest that larger groups can be supported in the presence of computer support but suggest that there exists a threshold (higher than in manual brainstorming) beyond which the amount of information being transmitted exceeds the participant’s processing ability. Kerr and Hiltz [26] suggest that information overload is likely to occur when communications are nonsequential and when multiple topic threads (issues) are being addressed. By allowing simultaneous input from multiple participants, EBS has eliminated one barrier to increasing group size. The next barrier seems to be the participant’s ability to deal with the unstructured form of EBS messages.

EBS systems, as they currently exist, distribute the documentation process to the individual members of the group; group members are responsible for providing their ideas in a form which can easily be distributed to the other group members. Thus, one of the major responsibilities of the facilitator has been eliminated. To date, no system provides computer support for the other activities performed by the facilitator, namely, organization of ideas, enforcement of the guidelines of brainstorming, and keeping the group focused and productive. This dissertation directly investigates the potential of content analysis as a mechanism for automating the first of these facilitator roles and, to a limited degree, the second.

To the list of four problems presented in the previous section can be added two opportunities: structuring of EBS messages could reduce the cognitive load on the participants, thereby allowing more information to be processed and, presumably, allowing group size to increase; and automated or semiautomated identification of specific types of messages, notably evaluative comments, could be useful for enforcing

the brainstorming guidelines. It might even be possible to identify conditions under which the group is no longer making significant progress toward its goals.

The following section describes a structural model which has been successfully used in certain types of group generative processes. The model is then expanded to accommodate a broader range of interpersonal messages, such as might be found in unstructured EBS sessions.

## 2 A Structural Model for EBS

The developed structural model for EBS is an extension of the Issue Based Information System (IBIS) approach first proposed by Kunz and Rittle [28] for documenting software design activities. The modifications and extensions made to the IBIS model reflect the unique aspects of brainstorming in general and electronic brainstorming specifically. The extensions reflect a combination of experiences of the author facilitating electronic brainstorming sessions and parallel the theoretical foundations of Bale's Interactive Process Analysis in several ways.

### 2.1 The IBIS Model

The IBIS model, as developed, consists of three types of nodes connected by as many as six relationships. *Issues* are problems, concerns or questions which must be resolved in order for the design process to continue, *positions* are alternative resolutions of the issues, and *arguments* are supporting or refuting statements which suggest that one position should be preferred over another. Arguments either support or refute positions; positions respond to issues; issues may question or be suggested by issues, positions or arguments; and issues may replace other issues.

Conklin and other researchers [8] at the Microelectronic Computer Corporation (MCC) in Austin, Texas, have worked extensively with the IBIS model and have suggested several additions to the model. Specifically, the MCC extended model allows issues to specialize or generalize other issues, and adds two new node types (*external* and *other*) to the original IBIS model. An *external* node contains documents from outside the design process which may be used as corroborating evidence and, finally,

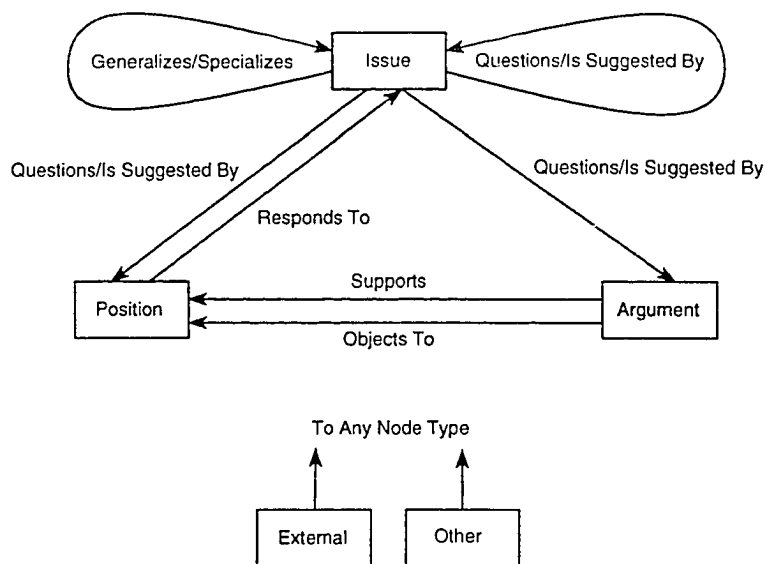


Fig. 1. The set of legal rhetorical moves in IBIS.

an *other* node may contain any messages which do not fit into the IBIS framework. Figure 1 shows the set of legal rhetorical moves in the IBIS model [47].

The need for an *other* node type supports the contention that the IBIS model is incomplete, even for the specialized group design process. In fact, the researchers at MCC point to a number of shortcomings of the IBIS model which were identified through extensive use of the system. Primary among these inadequacies is the inability to represent issue resolution within the IBIS framework. Another concern which is central to the modeling of EBS is the absence of tools for documenting messages which do not relate directly to an issue but which are substantive, annotative or procedural in nature (called metadiscussions). A message from a participant that a break from the group session is in order is a valid message but does not relate in any way to the primary purpose of the EBS session.

In a software design process, there is no question as to the topic of discussion — how best to design the product. EBS, however, has a much broader audience and can be used for a wide variety of purposes. In order to reflect this in the model

developed herein, an additional node type, *topic*, has been added. The distinguishing characteristic between issues and topics is that of premeditation. A topic is the stated goal or purpose of the EBS session. In a sense it is the primary issue, although this is not quite accurate. A topic might not be an issue at all (although all issues can be promoted to topic status). A topic might be “to discuss the results of last month’s sales efforts.” While it would be possible to restate this goal in the form of an issue (e.g., “what were the important results of last month’s sales efforts”), this seems artificial and does not in any way reflect the fact that this is the stated goal of the EBS session. When the stated goal has been reached, the EBS session is completed, regardless of the elapsed time. In a design process every issue must be resolved; otherwise the design cannot be completed. This restriction is inappropriate for generalized EBS; in fact, some issues may have no possible resolution. Such issues would remain unresolved (perhaps indefinitely); they should not prevent the group from moving forward on other issues. The introduction of a topic node into the model allows for an EBS session to be completed without resolution of all of the issues (although a distinction must still be maintained between resolved and unresolved issues). This added distinction of topics permits another degree of flexibility which is needed in EBS, but not a design process.

## 2.2 Bales’s IPA model

Another model in which EBS messages might be expressed is Bales’ Interaction Process Analysis (IPA) [2]. Bales theorized that group members have two broad types of concerns: *instrumental*, or task oriented; and *expressive*, or socio-emotional concerns. Instrumental concerns are further subdivided into active and passive, and expressive concerns are either positive or negative. These categories and their component

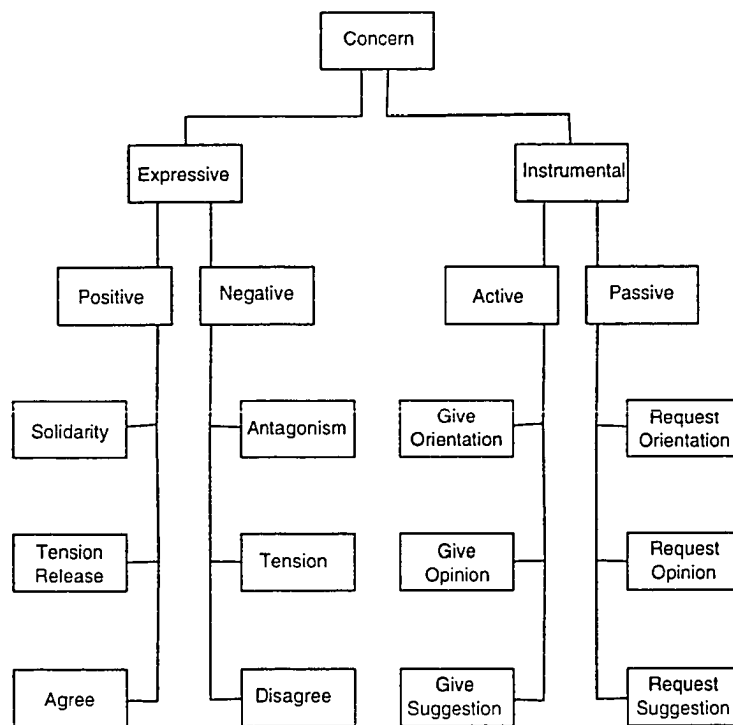


Fig. 2. The categories of Bales' Interactive Process Analysis model.

messages are shown in Figure 2.

Bale's model [1], as its name implies, is concerned primarily with the group process, while the IBIS model attempts to identify the artifacts of the group process. The two models are not in conflict, however. To *give orientation* is to raise an issue, and to *give an opinion* or *suggestion* is to propose a position or an argument. Expressive concerns represent evaluative thoughts and positions. Unlike the IBIS model, the IPA model recognizes (and in fact emphasizes) the communications aspect of a group process. The ability to express a request (other than by raising an issue) is absent in the IBIS framework. While it is possible to express positive and negative messages in an IBIS network, these expressions are limited to the single node type of argument. The IPA model provides a more flexible categorization scheme for expressive messages. Further, the semantics used in the IPA model do not suggest,

as does the term argument, that these expressive messages must be supportable by evidence, facts, or logic. Finally, the IPA model explicitly recognizes the existence of non-issue related communications.

Lest it appear that the IPA model is preferable to the IBIS model, several significant shortcomings should be noted. Because the model was designed for direct observation of group activity and incorporates a number of messages which may not be verbal in nature, it is difficult to use for some *post hoc* analysis (e.g., from a written transcript). The unit of measurement is not at all clear in the IPA model; is fidgeting with a lighter to light a cigarette a single expression of tension or an expression of tension followed by a tension release? Finally, the use of abstract (contentless) categories makes it difficult to view a message from the context of the group perspective [33].

### 2.3 The MMEBS model

The model developed for this dissertation's research, called the Message Model for Electronic Brainstorming or MMEBS (see Figure 3), is based primarily on the IBIS model but incorporates communicative components from the IPA model. In addition to the topic node described previously, the following node types have been added: *query* and *restatement* nodes to allow for communication directly with the author of another node; a *remark* node to accommodate non-issue related messages and evaluative comments; an *evidence* node to allow attachment of external documents supporting an argument; and a *resolution* node to explicate the culmination of the process, if one exists. The nodes of the MMEBS and their descriptions are given in table 1.

Every issue is subordinate to some topic, although an issue might be promoted to the status of topic at some future EBS session. In this way, the temporal links between EBS sessions may be maintained. Queries and clarifications may be attached to any node, as may remarks. Evidence nodes may only be attached to the argument they support.



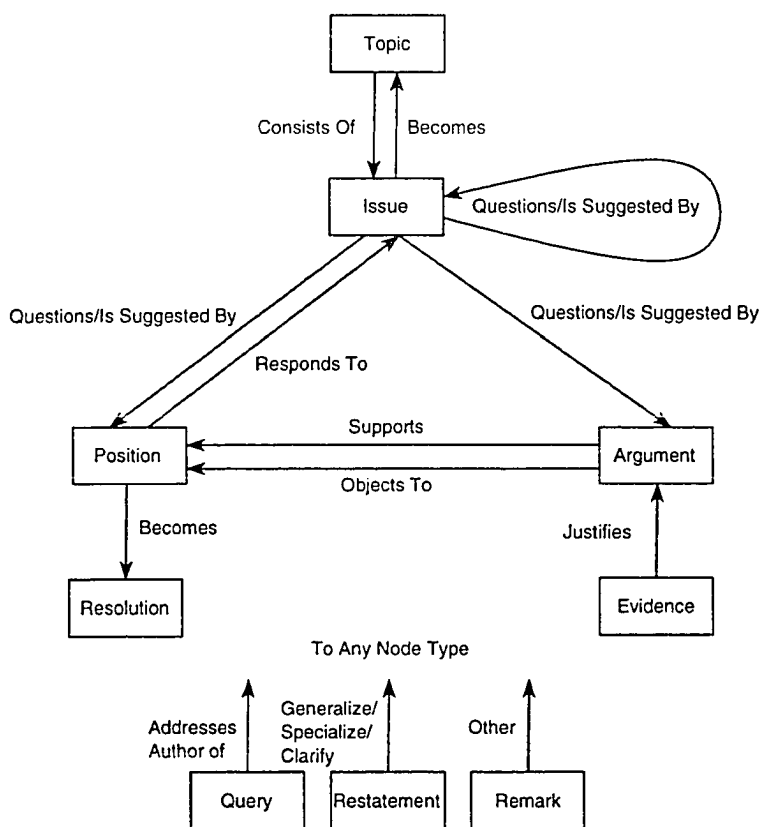


Fig. 3. The MMEBS structural model.

VisionQuest<sup>®</sup> uses an hierarchical arrangement similar to the one developed here; however, there is a clear distinction. In VisionQuest<sup>®</sup>, a *dialogue* subordinates one or more *topics*, just as a topic subordinates one or more issues in the developed model. The dialogue/topic hierarchy is static; only the dialogue initiator(s) may establish new topics. Issues, on the other hand, are dynamically established by the participants in the group process; only the topic need be established by the session initiator. This is consistent with the IBIS model.

Given the MMEBS structural model for EBS communications, it should be possible to categorize the messages transmitted during an EBS session into one or

Table 1. The MMEBS node types.

MMEBS Node	Definition
Topic	The primary Issue. The issue that needs to be resolved. Originally part of the IBIS model, later removed by IBIS researchers, later reintroduced. Most closely related to Request Suggestion or Request Opinion in the IPA model.
Issue	A point of discussion. A potential source of disagreement. Part of the IBIS model. Most closely related to Request Suggestion or Request Opinion in the IPA model.
Position	A potential resolution to an Issue. Part of the IBIS model. Corresponds with Give Orientation in the IPA model.
Argument	A reason for preferring one position over another. Part of the IBIS model. Corresponds with either Solidarity or Antagonism in the IPA model. Also incorporates elements of the Agree and Disagree nodes of the IPA model.
Resolution	The Position selected by the group. Present in some later versions of the IBIS model. No corresponding IPA node.
Evidence	Documentation which substantiates an Argument. Most similar to the External node in IBIS. May (or may not) correspond with Give Orientation in the IPA model.
Query	A request for Evidence or Restatement. Corresponds with Request Orientation in the IPA model.
Restatement	A clarification or adjustment to an Issue, Position, or Argument. May specialize or generalize the target statement. Corresponds with Give Orientation and/or Give Suggestion in the IPA model.
Remark	Any statement that is unrelated to the resolution of the Topic or one of its Issues. May correspond to an Other node in IBIS, but usually not part of an IBIS model. May correspond with any IPA model node type.

more of the aforementioned node types. While this has been demonstrated by researchers at MCC [8], their implementation required the participants to perform the categorization explicitly prior to transmission of the message. This methodology is contraindicated by the theory behind brainstorming that group members should freely express whatever thoughts they have without concern for form, completeness or feasibility. The problem is how to categorize free form thoughts, as expressed during a true brainstorming session, into the structural model so that they may be stored and manipulated efficiently.

### 3 Content Analysis

Krippendorff provides a broad definition of content analysis as “the use of replicable and valid methods for making specific inferences from text to other states or properties of its source” (Gerbner, Holsti, Krippendorff, Paisley and Stone [14], p. 11). The ultimate goal of content analysis is to condense a symbolic message (usually in the form of text) into a standard set of categories so that inferences can be made about the issues of concern to the message sender (see Figure 4). As such, content analysis fits cleanly into the often cited communications system model [30] shown in Figure 5.

In this model, the message sender maintains a world view (cognitive map, if you will) and selects some portion of that map to communicate to a receiver; the sender intends to produce an affect in the receiver. The message is encoded (into words, gestures, images, etc.) and transmitted, via some communications channel, to the receiver who then decodes the message and experiences some affect, which may or may not be the affect intended. The communication channel is responsible not only for physically transmitting the message but for partially encoding and decoding the message. The choice of communication channel can restrict or enhance the ability of the sender to produce the desired affect in the receiver. This research is directed toward increasing the amount of decoding which can be done by the communication channel (the computer network).

Historically, content analysis has been used after the fact of the communication to make inferences about the motivation, disposition or intent of the sender. That is,

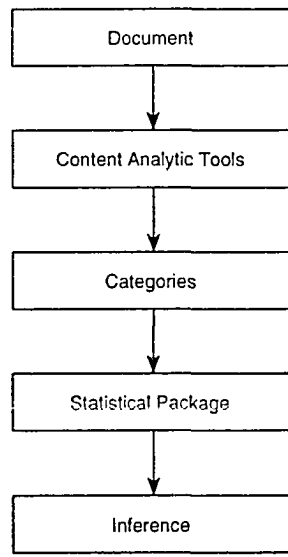


Fig. 4. Generalized content analytic process.

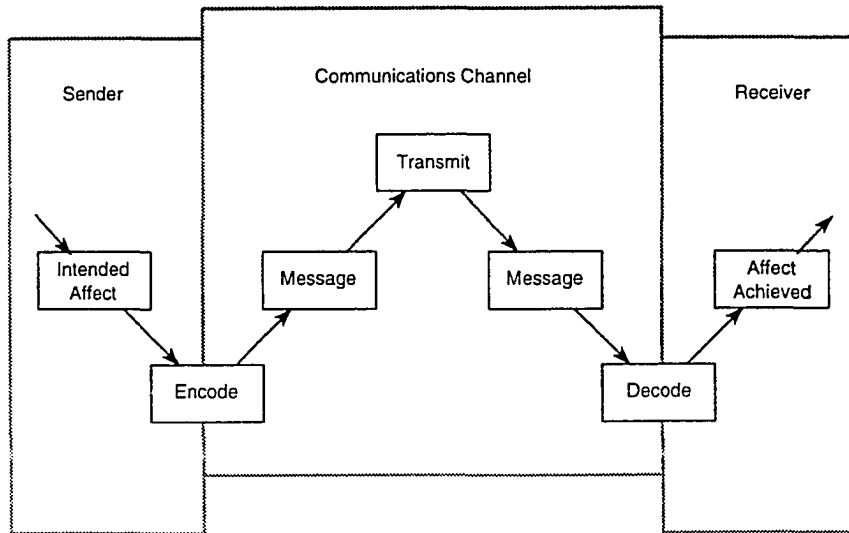


Fig. 5. The communications system model.

to make inferences about the sender's cognitive map. In this case, however, content analysis will be evaluated as a tool for performing a partial decoding of the message in order to assist the receiver. This subtle distinction will prove quite useful as it ameliorates some of the concerns which researchers have expressed with respect to content analysis in recent years.

Content analytic techniques range in complexity from simple word counts to artificial intelligence algorithms which evaluate grammatical structure (see Hick, Rush, and Strong [20] for an overview of content analytic procedures). The most commonly cited (and used) technique is to assign units of text, called signs, to categories called *tags*. Some signs are simply words like *empire* or *building*, while others are phrases, clauses or proper nouns like *the sands of time* or *Empire State Building*. Categories are usually rather broad subject headings like *wealth* or *affection*. This method is variously referred to as a thesaurus or dictionary technique; it will be referred to as the dictionary technique here.

The dictionary technique has been used successfully in a number of predominantly psychological and political studies (see Stone, Dunphy, Smith and Ogilvie [53] for an overview of several key studies). Basically, the dictionary techniques work as follows: the dictionary creator establishes a set of categories; these are considered to be general areas with which the authors of the text to be examined might concern themselves. The dictionary creator then "tags" each sign with one or more of these categories; in some cases many signs are categorized while in others only a few highly relevant signs are considered. Signs sharing a common tag infer concern by the author(s) with similar issues. The frequency of these categories may then be analyzed in a variety of ways to make inferences about the author's mental model.

The dictionary technique became so popular in the 1960's that a number of researchers [53] developed a general purpose computerized sign categorizer called the General Inquirer (GI). The GI is independent of the dictionary used in the analysis and of the language used in the text(s) to be analyzed. The researcher provides a dictionary, in a standard format, and the text to be analyzed; from these inputs the GI produces lists of the tags referenced in the text. This output may be used as input

to a variety of statistical summary programs to assist researchers in their analyses. Recent versions of the GI include a rule base for disambiguating homographs and proper pronouns ([63], [64]). Two types of rules are used to accomplish this: sign proximity rules, and tag reference rules. The former of these is used to identify sign combinations such as *Empire State Building*. In the case that the sign *Building* is preceded by the signs *Empire* and *State*, the tags associated with *Empire* and *State* are replaced in the tag list with the appropriate tag(s) for *Empire State Building*. The latter rule type is used to distinguish between the various senses of signs like *Mercury*. During primary processing the sign is given its most likely tag (e.g., *chemical*); after primary processing is completed, the sign is reevaluated. If the processed text is identified as containing tags for *astronomical* or *theological* categories, then the tag for *Mercury* is appropriately altered. While these rule types can be used to disambiguate a wide variety of homographs and nearly all proper pronouns, a number of references require a larger context in order to be disambiguated. For example, the sentence “The astronomer saw his favorite star tonight” cannot be disambiguated (either by the GI or by a human) without further information. The sign *star* might not refer to a *celestial body*, but rather to a *popular performing artist*. This example highlights one of the major controversies surrounding content analytic techniques: what should the unit of analysis be?

Weber [64] and Holsti [22] discuss the issue of the unit of analysis and report that not only is the reliability of classification variables dependent upon the unit of analysis but, more surprisingly, this relationship is non-linear. Using either sentences or entire documents as the unit of analysis results in more reliable category assignments than does the use of paragraphs [18]. For efficiency reasons the GI uses sentences as the unit of analysis. The issue of unit size is somewhat moot for the analysis of EBS messages as such messages are typically only a sentence or two in length.

Much controversy surrounds the appropriateness of a given dictionary to specific context domains. This is perhaps best stated by Berelson [5], p. 147:

Content analysis stands or falls by its categories. Particular studies have been productive to the extent that the categories

were clearly formulated and well adapted to the problem and to the content. Content analysis studies done on a hit or miss basis, without clearly formulated problems for investigation and with vaguely drawn or poorly articulated categories are almost certain to be of indifferent or low quality as research productions. Although competent performance in other parts of the analytic process is also necessary, the formulation and the definition of appropriate categories takes a central importance. Since the categories contain the substance of the investigation, a content analysis can be no better than its system of categories.

Unfortunately, there is no good theory of how individual cognitive maps will be expressed symbolically ([18], [53]). This suggests that there may be as many correct content category systems as there are message senders. Not surprisingly, a wide variety of dictionaries has emerged for use in an equally wide variety of textual contexts. Several researchers, however, have directed their efforts toward development of dictionaries which have broad applicability. One of these dictionaries, the Harvard Psychosociological Dictionary [55], has been used successfully in a variety of contexts. Now in its third revision, the dictionary classifies over 3,500 English language signs into 83 tag categories. Although the number of signs categorized appears small, it was sufficient for categorizing over 92% of the words in a random 500,000 word sample of texts [53]. Once proper names, noun plurals and regular verb paradigms were controlled for (noun plurals and regular verb paradigms are controlled by the GI), the percentage increases to more than 95%.

Content analytic tools, like all measurement instruments, are subject to criticism if they are not reliable. In order for an observation to be considered factual in a Lockean sense, there must be agreement between individuals and across time concerning what was observed. Similarly, a research measurement instrument must be valid in order to be useful. In a more Leibnitzian approach to determining the factuality of an observation, validity requires that an observation and the inferences made from that observation be consistent with what is already known and accepted as fact. The following two subsections will address the reliability and validity of content analysis respectively.

### 3.1 Validity

#### 3.1.1 Content Analytic Categories

Weber [62] identifies three pertinent measures of validity with respect to content analysis: criterion, content, and construct validity. To this list Holsti [22] adds concurrent validity.

Criterion validity, also called predictive validity, is established when a measure can be used to predict, with some accuracy, future events or facts unknown at the time of the measurement. Criterion validity has been demonstrated in a number of content analytic studies. Content analysis has been used successfully on Nazi World War II documents to predict certain aspects of Axis behavior [4] and to predict the authenticity of suicide notes [42].

According to Weber, content validity is the weakest and least measurable form of validity. Simply stated, an instrument has content validity (also called face validity by Weber) if it appears to measure what it is intended to measure. The construction of dictionaries has relied heavily on content validity. While great efforts have been taken to ensure the content validity of the Harvard IV-4 General Inquirer Dictionary, there are a number of areas of contention. For example, should the sign *love* be categorized similarly to the sign *like* (taken in the sense of affection as opposed to the sense of similarity)? It is unlikely that these issues will be resolved in the foreseeable future. Arguably, some categories have more content validity than do others.

Construct validity, and particularly external construct validity, is perhaps the strongest form of validity; it measures the degree to which the measure behaves as it is theoretically postulated to behave. A number of studies ([37], [61], [48]) has demonstrated the external construct validity of content analysis when used to measure the degree of concern over social, political, and economic issues in documents produced over extended periods of time. As would be expected, the levels of concern for certain issues as measured by content analytic tools has been shown to be highly consistent with pertinent interpretations of societal concerns at the time.



Concurrent validity refers to the degree to which a measure agrees with other measures believed to have validity in their own right. One study [37] analyzed newspaper editorials using two different dictionaries. While the categorizations were quite different using the two dictionaries, the important factors identified and the substantive conclusions were quite similar.

While the preceding discussion demonstrates that content analysis can produce valid results, it has not been established that the findings are generalizable to other domains or studies. It is, therefore, imperative that researchers evaluate the validity of their results. This research effort relies at first on content validity for the development of categorization heuristics. Once developed, the performance of the heuristics will be evaluated by individuals knowledgeable about the techniques of facilitation. This evaluation will serve as a measure of construct and concurrent validity.

### 3.1.2 System Findings

Researchers have identified a plethora of definitions for validity. Beard [3] lists ten “kinds” of validity defined with respect to expert systems: completeness, content, construct, criterion, face, inference engine, input/output, predictive and subsystem. Researchers disagree as to the best way to validate software systems; “Many people are working on the validation problem, but no one has yet advanced a good proposal for solving it (any more than anyone knows how to test human professionals to assure that they won’t make mistakes)” ([19] p. 281). As a practical matter, the question of validity can be simply stated as ‘Is the system doing what we believe it should be doing?’ There are a number of broad classes of tests to help determine the validity of system output: if known correct test cases are available, then predictive validation is possible; if the system is decomposable, then subsystem validation (using one of the other techniques) can be used; if the system is one that can be modelled graphically, then visual interaction with a recognized expert may be possible; once a system is “good enough” to produce useful results it can be field validated — used in actual field decision making situations; if the system responds to continuous variable inputs, then sensitivity analysis can be used to identify the range of valid responses; finally,

variations on the famous Turing test can be used to implicitly support claims of validity [43]. This dissertation uses a variation on the Turing test.

The basic premise of a Turing test is that if the output of a system is indistinguishable from output generated by a human, then the system is exhibiting intelligence. The original Turing test involved two humans and a computer. One human was designated as the “interrogator”. The interrogator was not able to see either the computer or the other human. The interrogator’s goal was to distinguish between the two solely on the basis of their responses to textual question that the interrogator produced [23]. If the interrogator could not distinguish between the human and the computer, then the computer must have some intelligence.

A variety of variations on the original Turing test have been used to validate expert systems. Most often this involves a set of experts examining the output of the system and judging whether it is correct or incorrect. For example, R1/Xcon, a system that designs computer configurations, was validated by six experts who examined fifty suggested configurations for correctness [32]. MYCIN, a system that diagnosis blood disorders, was originally validated by a panel of five experts who approved or disapproved of the system’s recommendations [23].

Ideally, a system can be validated by using some sort of double blind methodology. A double blind methodology allows direct comparison between conclusions drawn by human experts with those arrived at by the computer system. The original Turing test was a double blind test and subsequent validations of MYCIN used double blind methodologies. Unfortunately, it is sometimes impossible or impractical to use human subjects to perform the task that the computer is performing. Such is the case for this system. Pilot studies (described in the Methodologies section) showed that the time required for a human expert to perform categorization of brainstorming transcript statements is much too long to conduct double blind tests.

### 3.2 Reliability

Krippendorff [27] identifies three kinds of reliability with respect to content analytic categorization: stability, reproducibility, and accuracy. Stability refers to intra-rater reliability, reproducibility to inter-rater reliability, and accuracy to performance

against a known standard. Stability can be measured by having the same rater categorize the same set of signs more than once, and reproducibility can be measured by using multiple raters and comparing the results. In this research a computer will be used to perform the needed categorizations. This eliminates concerns about stability; given the same input and algorithms, a computer will always produce the same output. Similarly, because the algorithms implemented will represent heuristics developed by a single “expert,” there is no measure for reproducibility. Known standards very seldom exist for any text and may never exist for EBS sessions.

In a double blind experimental design it would be possible and desirable to examine the reliability of the system findings compared with those of human experts. For reasons explained above (and in more detail below), a double blind design was not feasible for this dissertation. It was possible to examine the reliability of the human experts when compared with each other. Although this does not give a measure of the system’s reliability, it does provide a sense for the performance criteria. That is, if the human experts can agree that the system has (or has not) made correct category assignments, then we can conclude that the system has (or has not) performed the task it was designed to do. If, on the other hand, the experts do not agree as to whether the system produced acceptable category assignments, then it is impossible to make conclusions about the reliability of the system, but it is possible to conclude that the system need not produce highly reliable results to perform as well as human experts. That is, lack of agreement between the experts may show that the problem is simply a “wicked”, or unreproducible, one that defies reliable interpretation. If unreproducible human categorizations are found to be valuable, then unreproducible computer generated categorizations might also prove to be valuable. Cohen’s kappa [7] was calculated in study #2, described below, to measure the level of agreement between human experts, controlled for agreement that occurs by chance.

While electronic brainstorming offers a number of benefits over its manual counterpart, several obstacles stand in the way of organizations being able to use electronic brainstorming effectively. These obstacles center around two characteristics of brainstorming in general and electronic brainstorming in particular: a lack of structure

in brainstorming sessions, and a high cognitive load for imposing structure after the fact.

A necessary first step toward discovery of the structure of brainstorming sessions is the development of models that can be subjected to empirical scrutiny. One such model has been developed and has been presented here.

Content analysis may be a useful tool for applying structure to brainstorming sessions after the fact. It can be implemented in the form of an expert system, as it is here, to reduce the cognitive load on human facilitators and other users of brainstorming records.

#### 4 The Research Question

The primary research question to be addressed in this dissertation is:

Can the MMEBS model be implemented via content analytic tools to produce a plausible and useful categorization of electronic brainstorming thoughts?

Implicit in this research question is the notion that there exists some “true,” or shared, interpretation of the statements in an EBS transcript — that humans will agree to some extent that certain issues, positions, etc. are indeed present. By doing this, the author of the dissertation is “taking sides” in a long standing debate among content analysis experts. One side of this debate holds that there exists some “true” meaning in statements that can be identified. The opposing view is that “truth” depends entirely upon the reader’s cognitive models and that the best that a tool such as content analysis can hope for is to be useful to some of the readers some of the time.

Without digressing too much into the philosophical dimensions of communications, the author of this dissertation believes that each of these positions has merit. In many cases, groups are formed of people with a shared cognitive model and so, for those people, there will be a “true” interpretation. As the differences in the group participants cognitive maps increase, so does the likelihood that they will interpret

a given statement differently. It is precisely these differences that brainstorming as a technique is intended to uncover. A dilemma results from this condition — the “best” brainstorming groups may be the most likely to have conflicting cognitive models and therefore the least likely to agree on what issues, positions, etc. were raised by the other group members.

This research takes the assumption of a “true” model one step further, however. Because the actual participants of the EBS sessions were not available to serve as experts in the study, persons experienced with electronic brainstorming and facilitation were used (see the section on Experts for a more detailed description). The assumption, then, is that the Experts who analyzed the EBS transcripts for this study share the same cognitive models as the participants in the EBS session. So, to the (fairly practical) research question above can be added:

To what extent does this research support or refute the contention that there exists some “true” interpretation of EBS statements.

Implicit in the answering of the primary research question are three additional research questions:

1. How appropriate is the proposed MMEBS model for the classification of EBS thoughts and what additions or changes can be made to the model to improve its applicability to business decision making?
2. What kinds of heuristics and content analytic techniques show the greatest promise for categorizing EBS transcripts?
3. Are the developed heuristics general enough to be applied to other sets of EBS transcripts?

## METHODOLOGY AND RESULTS

This dissertation is primarily directed toward validation of the components (issues, positions, etc.) of the MMEBS model. Towards that end a series of experiments was devised to find out if content analytic classification — the assignment of category tags and the statistical analysis of those tags — can be used to identify the elements of the MMEBS model within electronic brainstorming transcripts. This section describes the design and implementation of those experiments.

### 1 Pilot Studies

In two separate pilot studies, subjects were asked to identify the issues, positions and arguments present in an electronic brainstorming transcript. The subjects of the first pilot study were asked to identify the exact statement which signalled the introduction of a new MMEBS node. The results were analyzed to determine the level of agreement as to the “correct” classification of each brainstorming thought. The results indicated a very low (24%) level of agreement among the subjects.

The second pilot study used a more subjective categorization process. Each subject was asked to read the transcripts and then list, in their own words, the MMEBS components. Having completed this task, the subjects were asked to return to the transcript and identify the thought or thoughts which prompted them to create each node. In some cases, there was significant agreement as to the issues represented in the transcript (as high as 81%); other issues were identified by only one of the sixteen subjects (0.0625%). A closer examination of the identified issues revealed that many of the conflicting issue identifications were plausible, given the transcript content. This suggested that there may, in fact, be no single “correct” classification for EBS thoughts but that several, equally valid, interpretations are possible.

Based on the results of these pilot studies, an expert support system approach was adopted over an expert system approach. Rather than attempt to identify “the correct” list of MMEBS components represented in the transcript, the system was

designed to identify a reasonable and supportable interpretation of the transcript content. This interpretation might then be altered by the group process participants as they see fit. Implicit in this approach is a two-part idea generation process consisting of a freestyle generation process followed by a separate structuring process. Plexsys<sup>®</sup> already supports this two-phase generation process through their group process tool called the Issue Analyzer. The Issue Analyzer does not provide suggested categorizations, however [59].

The content analysis research community is, more or less, divided into two camps (along the philosophical lines drawn in the previous section) concerning the usefulness of researcher-developed categories [25]. One side of this argument contends that the researcher is performing an additional, and perhaps unnecessary, translation of the evidence. This translation will often introduce bias that will affect the outcome of the research. The alternate point of view is that the researchers always decide what phenomena are important and necessarily has expectations of what the outcome will be — otherwise they would not engage in the research in the first place. The evidence produced in the pilot studies for this experiment supports the former position but, for purely practical reasons, the author has chosen the latter approach; there are simply too many possible interpretations of any given brainstorming transcript to permit categorization based on the participants' internal mental model(s).

Another observation made during the pilot studies was that a very significant amount of time was needed to produce a reasonable categorization. Subjects self-reported between two and six hours for categorizing an EBS transcript containing fifty thoughts (a relatively short transcript). An approach requiring *a priori* classification by research volunteers was ruled out because of the exorbitant amount of time they would have to contribute in order to produce a sample of sufficient size to allow generalization.

## 2 Study #1

### 2.1 Procedure

In order to answer the research questions, five separate tests were performed on the EBS transcripts.

1. The general purpose of content analysis, as stated earlier, is to identify a set of categories which the EBS author(s) are concerned about (issues). With this in mind, a simple summation of the number of times that each category tag appears in an EBS transcript should be useful in identifying major issues. Marker and non-substantive categories (like STRONG, which serves as a modifying category) are necessarily ignored. The ability of the system to identify meaningful categories of concern depends entirely upon the dictionary being employed. In all likelihood the Harvard IV-4 dictionary will be better suited to some transcripts than to others.
2. The Harvard IV-4 dictionary contains categories specifically intended to identify evaluative statements. These categories are: AFFIL (supportiveness and affiliation), COMP (comparative), EMOT (emotional responses), EVAL (evaluation), FAIL (indications of failure), HOSTILE (aggression), NGTV (negative attitudes) and PSTV (positive attitudes). Statements assigned these tags have a higher likelihood of being evaluative statements than other statements.
3. The similarity of transcript entries is a somewhat more complicated matter to address because the number of words in a statement affects the number of categories assigned. Consider a pairwise comparison between two hypothetical statements which have been content analyzed. Our hypothetical category scheme has only five categories. The first statement has been assigned the first, third and fifth category and can be represented using the binary digits 10101. Clearly, the best possible match with this statement would be another having the representation 10101 — anything else should be considered less similar.

It is possible that the second statement will have tags which the first does not, for example 10111. Such a situation does not indicate that the author of the first is not concerned with category 4, only that he or she simply did not express that concern in this statement. But we are interested in the similarity



between the statements and not necessarily the concerns of the author at this point. The pattern 10111 is not as similar to 10101 as the perfect match 10101 is. An identical argument can be made in the situation where the second statement lacks a concept expressed in the first. This observation leads us to conclude that an *exclusive or* (XOR) of the two binary patterns, summed, will provide a useful measure of similarity. An XOR of two binary digits yields 1 if one or the other of the digits is non-zero but yields 0 if both digits are zero or if both digits are non-zero. A logical XOR will yield identical result regardless of the order of the operands, so there is no need to perform a pairwise comparison more than once. For the three examples above (comparison of 10101 with 10101, 10111 and 10001), the results are:

10101 XOR 10101 = 00000

10101 XOR 10111 = 00010

and

10101 XOR 10001 = 00100

Summing the digits yields 0, 1 and 1, respectively, meaning that statements 3 and 4 are equally dissimilar to statement 1 and that statement 2 is not at all dissimilar to statement 1.

Obviously, all marker tags must be ignored. Fortunately markers are identified clearly in the content analysis dictionaries. If markers were not eliminated, then they would bias the similarity measure between statements. For example, two statements containing only “leftover” words might be identified as similar because they both have the tags B (beginning of sentence) and E (end of sentence) which hardly constitutes similarity.

The differences in length of the statements (as reflected in the differences in the number of categories identified with each statement) have still not been controlled for; although the problem has been successfully delayed to this point. At this point we have a triangular matrix representing the number of differences between each pair of statements. This matrix is easily normalized

by dividing by the maximum number of categories represented in the two statements. In our identical statement example above, each statement has 3 associated categories there were at most 3 possible mismatches, none of which occurred, yielding a normalized measure of  $\frac{0}{3}$ . The other two comparisons yield  $\frac{1}{4}$  and  $\frac{1}{3}$  respectively. This makes intuitive sense because the comparison of statement 3 with statement 1 shows that one author expressed 3 concerns, the other expressed all of those concerns and one more. In the comparison between statements 4 and 1, there are only two mutual concerns.

Using this normalized measure of statement similarity, the user can select an arbitrary cut-off point at which statements are no longer considered “similar”. In all likelihood, this threshold will vary from one EBS session to another.

The remaining two tests rely on an exploratory statistical technique called *correspondence analysis*. Correspondence analysis is a tabular and graphical technique for identifying patterns in multidimensional data. The mathematical foundation of correspondence analysis is shared by techniques variously called “reciprocal averaging,” “principal components analysis,” “dual scaling” and “optimal scaling.” Each of these techniques seeks to identify an  $N - 1$  dimensional plane through an  $N$  dimensional cloud of points so as to minimize some function of the errors between the points and the plane. Unlike regression analysis which seeks to minimize the errors on a single (dependent) axis, these techniques calculate error terms based on the orthogonal distance from the point to the plane. The graphical technique now called correspondence analysis is attributed to Jean-Paul Benzécri, who formalized the mathematics of the technique in the 1960s and 1970s [15].

Simply put, correspondence analysis takes all the variables (dimensions) provided and finds an  $N - 1$  dimensional plane through the cloud of points that minimizes the variance of observations about that plane. Each dimension is then weighted and the orthogonal points on the plane are translated (using the weights) into a two dimensional space. Considerable complexity can be represented in a single two dimensional graphical display using multiple correspondence analysis.

The output of a correspondence analysis is a two dimensional (euclidian) graph on which co-occurring observations appear “in the same direction” from the origin. The strength of that relationship (called the inertia) is shown by the distance from the origin. Because there are multiple, possibly correlated, variables impacting the sequence of observations emanating from the origin, the location of points is ordinal, not nominal. For example, two observations appear in the same direction from the origin, the second twice as far from the origin as the first; this can be interpreted to mean that the second observation co-occurs with the dimension more often than the first observation. It is not necessarily valid to interpret that relationship to mean that the second observation co-occurs twice as frequently as the first, however.

4. Test #1 above deals specifically with the identification of issues in EBS transcripts. That test does not address the identification of positions or arguments. Further, test #1 does not provide evidence as to whether or not issues can be distinguished from positions and arguments. Because issues, positions and arguments were specifically identified in the IBIS transcripts provided by Corporate Memory Systems, Inc. they provided an opportunity to test the applicability of content analytic techniques for making such distinctions. Each of the IBIS transcript statements was content analyzed, tagged according to its predetermined status as an issue, position or argument and then the transcripts were correspondence analyzed in order to identify category patterns that might exist.
5. The transcripts from student training sessions were content analyzed and coded for relevance to the topic (relevant or not relevant) and then the transcripts were correspondence analyzed in order to identify category patterns that might exist.

## 2.2 The Software

LexNet is an inference engine, developed as part of this dissertation, which applies procedural content analysis knowledge to a corpus of text. The output of LexNet is a

list of content analysis categories determined to be referenced in the text corpus and the frequency of those references. LexNet is modelled after, and nearly replicates, an existing software system, the General Inquirer III [53]. The rules and data used to perform the content analysis procedures are stored in a relational knowledge base, also developed for this dissertation. The rules and data themselves, collectively referred to as the Harvard IV Psycho-Sociological Dictionary [55], were developed elsewhere for use by the General Inquirer and were converted for storage in the developed knowledge base. The relational knowledge base management system (DBMS) is described in Appendix A, LexNet in Appendix B, the knowledge base in Appendix C, the translation of the Harvard IV dictionary in Appendix D and the verification of the LexNet inference engine in Appendix E. Source code for the relational knowledge base management system, the knowledge base, the LexNet inference engine and several utility programs, including conversion from the General Inquirer format, is in Appendix G (digital). The rules and data must be purchased from ZUMA[68]. All source code is stored on a standard high density DOS format diskette in ASCII text format. The source code was developed for the Berkeley Software Distribution (BSD)<sup>®</sup>, version 4.3, of UNIX<sup>®</sup> operating system and is written in ANSI standard 'C'. Portions of the source code were generated by FLEX[57], a lexical analyzer generator.

Several modifications to the LexNet software were required in order to produce the measures used in this study. Specifically, options were added to `lexnettrans` to do the following: print a binary list of substantive categories present in a statement (similar to the `i` option); print the total number of sentences carrying each tag (similar to the General Inquirer TALLY program) across a set of documents; print the evaluative categories present in each statement; and print a list of all the categories present in a document set. Additionally, short programs were written to perform a bitwise exclusive OR between the tags assigned to two statements and to produce an indicator matrix of categories present in a document (as required for correspondence analysis).

---

BSD is a registered trademark of the University of California, Berkeley, California. UNIX is a registered trademark of AT&T Bell Laboratories, Homedale, New Jersey.

### 2.3 Experts

Three experienced facilitators were asked to confirm the findings of these simple content analysis techniques. Two of the experts conduct brainstorming sessions as a routine part of their employment. The third uses brainstorming routinely with university and corporate managers in his role as management a consultant. Each of the three has facilitated dozens of brainstorming sessions. Two of the experts were female. One is a full professor of management, one was completing a Masters thesis in Psychology and the third has a Bachelor of Science in Psychology.

The experts received a set of seven electronic brainstorming transcripts (described below), each with three sets of questions (also described below). The sets of questions were constructed to elicit whether or not the human experts agreed or disagreed with the categorizations which the computer had assigned to the transcript statements. For test #1 the experts received a list of categories (issues) identified as present in the transcript by the computer and were asked whether they agreed or disagreed that each issue was actually discussed in the transcript. For test #2 they received a list of (supposedly) evaluative statements found in the transcript by the computer and were asked whether they agreed or disagreed that the statement was, in fact, evaluative. For test #3 they received a set of statement pairs from the transcript that were identified by the computer as similar and were asked whether they agreed or disagreed that the statements were similar. Tests #4 and #5 did not require expert opinions.

### 2.4 Transcripts

Three sources of EBS transcripts were utilized: Collaborative Technologies Corporation of Austin, TX (now OmniQuest of Dallas, TX) provided a number of actual EBS transcripts — primarily from marketing discussions about the VisionQuest® product. Of the transcripts provided, several were eliminated because they contained large amounts of technical jargon that would not be recognized by the Harvard IV-4 dictionary entries. Other transcripts were eliminated because their topic was not one that allowed the raising of issues (for example, “Propose new names for this product”). The remaining transcripts were used in tests #1-3, described below. Seven transcripts were selected:

1. **MBA Success** What are the critical success factors for the MBA program?
2. **Quality Definition** What quality means to your organization.
3. **Customer Service** Definition of customer service.
4. **Product Benefits** What are the benefits provided by [product name]?
5. **Business Problem** Perceptions of major problems [company name] faces today.
6. **Staff Evaluation** A performance evaluation of [person name] performed by peers.
7. **Business Prospect** You're trying to get someone interested in [product name]. Enter two to three sentences that capture what you want to say.

Each of these seven transcripts was processed using LexNet and the Harvard IV dictionary to find high frequency category tags (issues), high evaluative scores and low difference scores (high similarity). The statements with high (low) scores were collected to create the questionnaire, described below, that the experts responded to.

Although not technically brainstorming transcripts, three IBIS transcripts were provided by Corporate Memory Systems, Inc. of Austin, TX. Each statement in these transcripts was identified as an issue, position or argument by the the author of the statement when the transcript was generated. Unfortunately, one of the three transcripts contained a very high number of technical acronyms and other jargon. The two remaining IBIS transcripts were used in test #4 described below. The two IBIS topics were:

1. **IBIS 1** Develop a strategy for a utility company.
2. **IBIS 2** How to comply with the Clean Air Act.

Finally, two transcripts were selected from several dozen produced by student groups being introduced to electronic brainstorming software. These transcripts were particularly problematic. Unlike “real” brainstorming sessions, student brainstorming sessions frequently deteriorate into informal discussions — typically about sex, violence and the university faculty. In many cases, the amount of such chatter significantly exceeds the substantive content of the discussion. Student sessions also seem to contain more personal references and “street language” which has evolved since the Harvard IV-4 dictionary was developed. These characteristics make student brainstorming sessions poor representatives of “real” brainstorming, as it occurs in business. However, these transcripts do contain high percentages of non-topic related material, which makes them ideally suited for test #5 discussed below. The two transcript topics were:

1. **MBA Success Factors** *also used above* What are the critical success factors for the MBA program?
2. **President Debate** Which [1992] presidential candidate did you consider best and why?

## 2.5 Questionnaire

Each of the experts received a questionnaire containing a total of 156 questions referring to the seven brainstorming transcripts. Appendix F contains the complete questionnaire used in Study #2. The questionnaire used in Study #1 was the same except that it did not contain the demographic questions, it used a five point Likert scale rather than a three point scale, and it did not contain written instructions (the author was able to meet extensively with each expert in Study #1). Each transcript was accompanied by a questionnaire consisting of three parts:

1. Section 1 — The top five issues identified by test #1 were listed along with a five point Likert scale ranging from ‘Strongly Agree’ to ‘Strongly Disagree’.

2. Section 2 — All statements identified by test #2 as being evaluative were repeated along with a five point Likert scale ranging from ‘Strongly Agree’ to ‘Strongly Disagree’.
3. Section 3 — All statements with a calculated similarity value (test #3) of less than 0.75 were listed along with a five point Likert scale ranging from ‘Strongly Agree’ to ‘Strongly Disagree’. The value of 0.75 was chosen arbitrarily so as to reduce the work load on the experts.

Descriptive statistics regarding the level of agreement between the experts and the heuristics developed here are presented below. Because the number of validators is quite small, exacerbated by the fact that Likert scales do not necessarily produce normally distributed results, statistical analyses beyond standard descriptive techniques could not be used meaningfully.

## 2.6 Results

The pilot studies described above suggested that high levels of rater agreement would be unlikely — and this turned out to be the case. Figure 6 shows the frequency with which each expert in Study #1 registered ratings of “Strongly Agree,” “Agree,” “Disagree,” “Strongly Disagree” and “Not Sure.” Expert number one is the dissident of the group, showing a preference for “Disagree.” Expert number three, on the other hand, seems to “Strongly Agree” with the system’s output. Expert number two falls between these two, choosing “Agree” most often. While this may seem to be problematic for this study, the effect of rater differences can be controlled using correspondence analysis.

Such differences in ratings could result from a number of causes. The most insidious of these is the possibility that the experts did not understand the directions. In fact, one of the experts pointed out such a problem with the instructions for part two (evaluative statements) — this problem will be described in detail shortly. The instructions for parts one and three were very straightforward and none of the experts indicated any misunderstanding or confusion regarding the instructions for those parts.



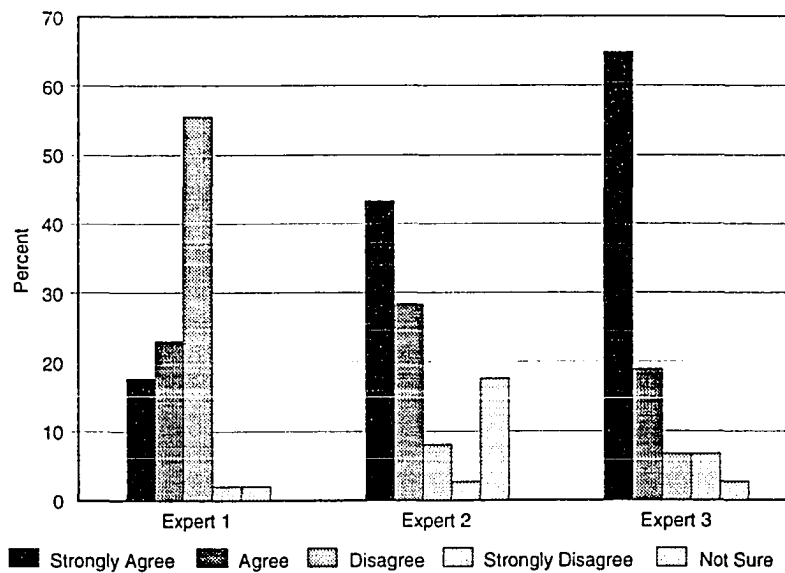


Fig. 6. Frequency of ratings by expert.

It became clear after discussions with the experts that the strength of an expert's agreement or disagreement was largely a function of the expert's temperament and that to "Strongly Agree" with a computer generated categorization is not substantively different than to "Agree". A three point scale was determined to be more appropriate to the type of judgement that the experts were being asked to make. For the remainder of the analysis of Study #1 (and all of the data collection and analysis of Study #2) a three point scale was adopted. That is, "Agree" and "Strongly Agree" were treated as like responses. Figure 7 shows the rate of agreement by expert, using a three point Likert scale, for Study #1.

It is reasonable to expect that content analysis might perform differently on different transcripts since those transcripts represent different topics. Figure 8 shows the two level ratings for each of the seven transcripts. Significant observations are that the greatest amount of agreement was achieved on transcripts five, seven and two — enticing a sales prospect, critical success factors for the MBA program and benefits

provided by a product. Agreement was lower on the topics: business problems facing a company, peer performance review, definition of customer service and a definition of quality. The peer performance review had the highest level of disagreement.

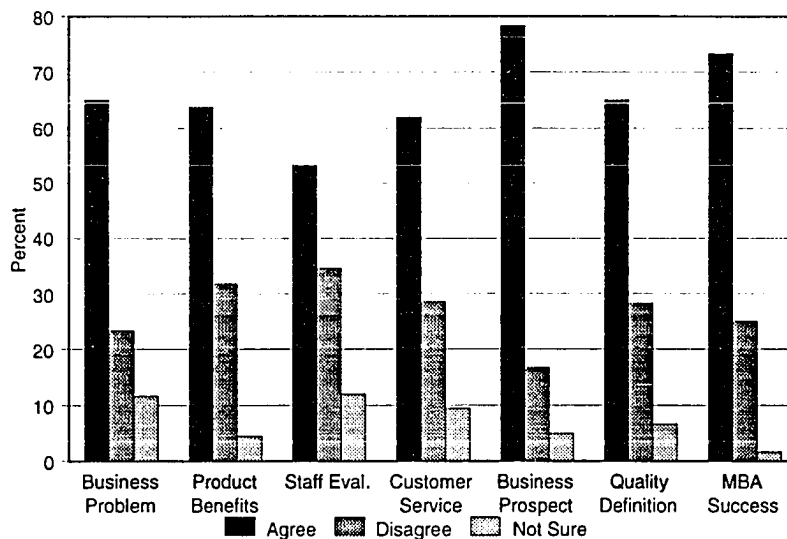


Fig. 8. Frequency of combined ratings by transcript.

Also worthy of note is the fact that, on a transcript basis, agreement is low. In no case is agreement greater than eighty percent, a level which is considered “typical” for acceptability of expert systems output [6].

It is also likely that the content analytic might produce different results depending upon the type of test (issues, evaluative statements or similar statements) being performed. Figure 9 shows excellent agreement with the identification of issues, considerably lower agreement on the similarity of statements and very low agreement on evaluative statements. The phrase “evaluative or judgemental” can be interpreted

to refer to evaluation in general or to meta-level evaluation. The peer performance review transcript intentionally elicits evaluative statements of the first kind. The rule of brainstorming that restricts evaluative statements is, however, referring to the second kind. The expert who pointed out this ambiguity chose the latter definition, while the other experts (and apparently the Harvard IV-4 dictionary) chose the former.

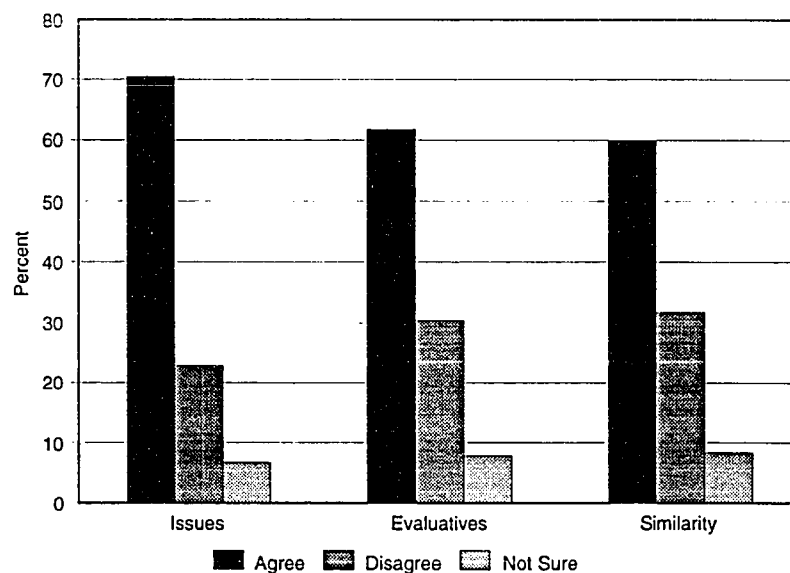


Fig. 9. Frequency of combined ratings by part.

A closer examination of the transcripts reveals that meta-level evaluative statements were quite rare. Only one statement from all seven transcripts clearly qualifies (there are several arguable cases). From the transcript on critical success factors for the MBA program, the statement “I agree with that about evaluating out [sic] efforts before knocking the program” did receive a high evaluative score — but no higher than a number of other statements.

At least one expert felt that the questionnaire was ambiguous with regard to finding evaluative statements because it did not distinguish between evaluative statements made as part of the group process and evaluative statements made *about* the group process. It appears that the Harvard IV-4 dictionary does not (perhaps cannot) make such a distinction either.

Continuing to look at greater levels of detail leads to the examination of part  $\times$  transcript combinations. Such an examination is warranted because it has been demonstrated that performance of the content analytic technique varies considerably depending upon part and to a lesser degree upon transcript.

Frequency counts, or graphs based on such counts as used above, become difficult to interpret with this number of variables (21 transcripts each with three part combinations with three experts and two rating levels). Correspondence analysis can be used to reduce the cognitive load of interpreting these findings. Recall that correspondence analysis is a graphical, multivariate technique in which direction from the origin, proximity and distance from the origin all have meaning (absolute distance between points is not interpretable, however).

As a demonstration of the technique, Figure 10 shows a correspondence analysis of expert  $\times$  transcript  $\times$  part  $\times$  rating. Which is to say it combines all of the frequency-based analysis described thus far into a single graphical presentation. The substantive conclusions (with two minor exceptions) drawn from the frequency data are supported by the correspondence analysis. There is a significant difference between the two approaches, however. Unlike frequency counts, correspondence analysis is a multivariate technique. When interpreting one variable on the correspondence analysis graph, the effects of the other variables in the model have been compensated

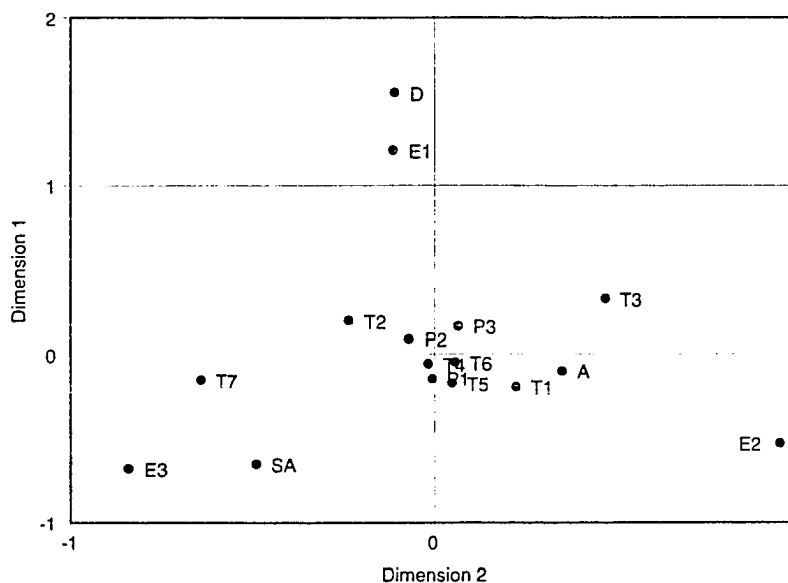


Fig. 10. Correspondence analysis (expert  $\times$  transcript  $\times$  part  $\times$  rating).

for — this results in two minor differences in the interpretation of the data presented so far.

The primary dimension (the  $y$  axis) in Figure 10 is dominated by the rating categories; Strongly Agree is well below zero, Disagree is well above, Agree is between the two (Strongly Disagree occurs so infrequently as not to contribute to any axis). The secondary dimension (the  $x$  axis) is dominated by the differences between experts; E3 is well below zero, E2 is well above and E1 is between the two. Because the experts are independent of one another, there is no meaningful interpretation of the secondary dimension. The proximity of experts to ratings corresponds with the observed rating patterns; E3 lies closest to SA, E1 lies closest to D and E2 is closest to A (although that relationship is not as strong as the other two). This conclusion is substantively identical to that reached using simple frequency counts. Because correspondence analysis is a multivariate technique, however, it shows that the relationship between experts and their voting patterns is consistent across all transcripts and tests.

Consistent with the frequency counts, transcripts seven and five fall on the “Agree” side of the origin and transcript three falls on the “Disagree” side. Transcript two and one, however, have changed places — transcript two falls on the “Disagree” side and transcript one on the “Agree” side. A closer look at the frequency data obviates why this is the case. Although transcript one has a lower level of agreement than transcript two, it also has a lower level of disagreement (caused by “Not Sure” ratings). Correspondence analysis controlled for this effect. A similar situation occurs with questionnaire parts. Part one corresponds with agreement, but once all other factors are accounted for, parts two and three correspond with disagreement.

Because distances from the origin represent the strengths of relationships in correspondence analysis, it is clear that the dominant factors in this analysis are the experts and the level of agreement. By combining the ratings as described earlier the impact of these categories can be decreased, allowing other factors to contribute more to the differentiation on the graph.

One anecdotal observation can be made from the correspondence analysis that would not be readily apparent from the frequency counts is the proximity of the points T7, SA and E3. The interpretation of this observation is that expert number three strongly agreed with the system on transcript seven much more often than the other experts. A review of the raw ratings shows that expert three registered sixteen SA ratings on transcript seven compared with nine and five for experts two and one, respectively.

Figure 11 shows the correspondence analysis of experts  $\times$  transcript\_parts  $\times$  combined ratings. Level of agreement is still the primary dimension and expert is still the secondary dimension. Expert number 2 is now considerably more similar (along

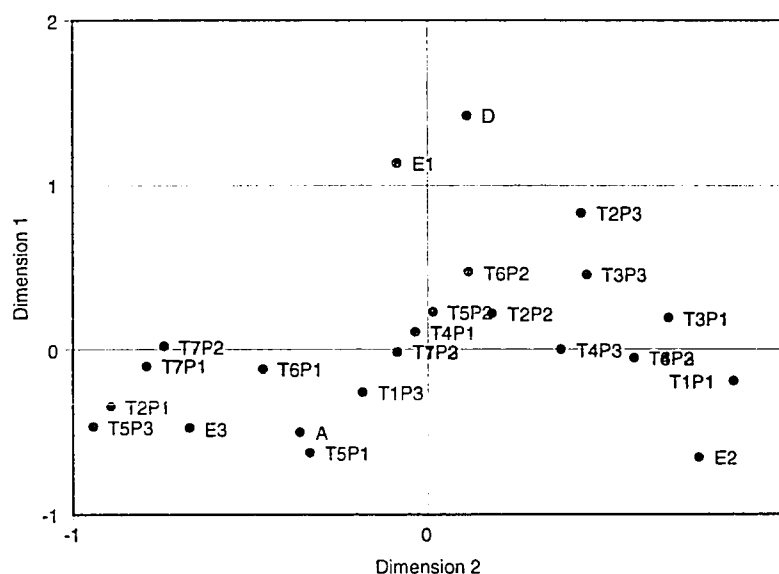


Fig. 11. Correspondence analysis (expert  $\times$  transcript\_part  $\times$  combined rating).

the  $y$  axis) to expert three.

Examination of the graph quickly shows the transcript\_parts which produced the greatest agreement (T5P1, T5P3, T2P1, T1P3, T1P1) and those which produced the greatest disagreement (T2P3, T6P2, T3P3, T5P2, T2P2, T3P1). As expected, agreement is predominantly in parts one and three and in transcript five; disagreement is predominantly in part two, transcripts two and three. Interestingly, the position of T7P1 and T7P2 shows that these are the exact areas where expert three departed greatly from the opinions of the other experts; T7P3 was a mixed bag.

A question by question discussion of the transcripts would be tedious and is unnecessary. Nevertheless, a few key points are worthy of a brief mention.

The substitution (to protect the unwitting) of the phrase "Product X" caused the ECON category to be over represented. The ECON category was consistently a source of disagreement.

Several statements were so short (and therefore had so few tags assigned) that they were identified as similar simply because they contain so little information. For

example the statements “Benchmark for quality” and “Get Well” simply because “benchmark” is not defined in the dictionary and both “well” and “quality” refer to making improvements. The analysis of more lengthy statements usually resulted in more favorable results. For example, the statements “Providing departments with meaningful analysis of operations as well as useful recommendations for improvement” and “Value of information provided to departments” were correctly identified as highly similar.

The high level of agreement on part three of transcript seven is partially due to the fact that this transcript contained a greater number of exact duplicate statements — which are, by definition, very similar.

Figure 12 shows a cumulative measure of the rate of agreement for several levels of the similarity (difference) measure. Consistent with the interpretation above, disagreement is lowest in the .4 – .5 range. Lower scores are dominated by very short statements. Agreement drops significantly when the measure is above .5.

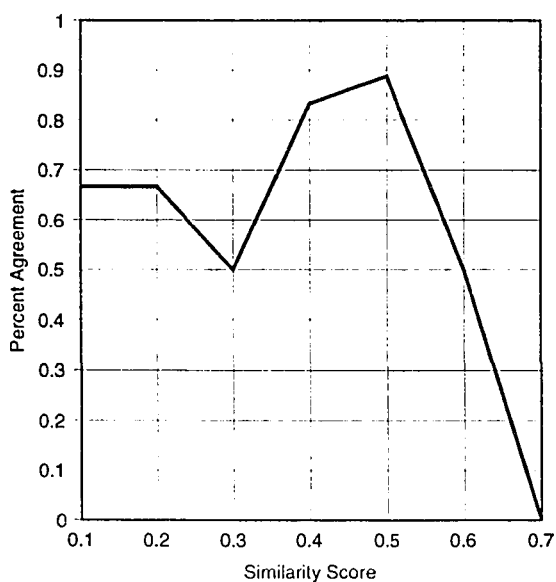


Fig. 12. Cumulative percent agreement on similarity by score.



Table 2 shows a summary of the content analysis procedures by transcript and part for study #1.

Table 2. Agreement frequency summary for study #1.

Transcript	Issues	Evaluatives	Similarity
MBA Success	60.00	66.67	73.33
Quality Definition	83.33	58.33	41.67
Customer Service	63.33	60.00	61.11
Product Benefits	73.33	53.33	60.00
Business Problem	76.67	73.33	66.67
Staff Eval.	53.33	60.00	50.00
Business Prospect	83.33	60.00	86.67

The results for tests four and five were disappointing. The correspondence analysis of the issue based information systems transcripts and the “unrelated” statements coding failed to identify any dominant dimensions. In fact, all the data points formed a tight cluster, centered at the origin meaning that there was no real difference between the statements – as measured by the content analytic categories identified.

### 3 Study #2

#### 3.1 Transcripts

The transcripts used in Study #2 were identical to the first seven transcripts described for Study #1. The two IBIS transcripts used for test #4 and the two student transcripts used for test #5 in the first study were not used because those tests yielded poor results in study #1.

#### 3.2 Procedure

The procedure followed in Study #2 was nearly identical to the procedure followed for the first three tests in Study #1. Tests four and five in study #1 produced poor results and were not repeated in Study #2. Differences between the Study #1 and

Study #2 procedures were with respect to the experts and the questionnaires and are described below.

### 3.3 Experts

Twenty faculty members and graduate students in the social sciences were individually recruited to complete the questionnaire. Of those twenty, eleven returned the questionnaire. This response rate is low considering that the experts had each previously agreed in person to complete the questionnaire, providing further evidence that the task was neither trivial nor simple. Of the eleven that responded, eight completed the questionnaire, one completed all but a single page, one completed a single page, and one did not attempt it at all. The second to the last included a note indicating that the expert did not understand the technical jargon and slang in the transcripts and could not decipher the typographical errors. The last sent a note apologizing for not completing any part of the questionnaire.

### 3.4 Questionnaire

The questionnaires used in Study #2 were the same as those used in Study #1 except that written instructions were included, a three point Likert scale was used rather than a five point scale, and a short set of demographic questions were included. The questionnaire, in its entirety is reproduced in Appendix F.

### 3.5 Results

Of the twenty original experts solicited, twelve were women. Of the nine usable responses, six were submitted by women. Not only did more women complete the (lengthy) questionnaire, but they agreed with the system findings slightly more often than the three male respondents (60% vs. 51%).

All of the respondents fell into three age categories: 26–30, 31–35 and 41–45 (N=2,4,2 respectively). The youngest respondents disagreed more often (37% vs. 22% vs. 24%) and were unsure less often (7% vs. 18% vs. 18%) than either of the other two age groups.

Most (five) of the respondents reported their area of expertise as sociology, one reported educational psychology, one cognitive psychology, one international finance and one did not respond to that question. Again, all of the respondents were either graduate students or faculty in psychology, sociology, educational psychology or clinical psychology.

Seven of the respondents reported that the highest degree they had earned was a Master's degree, the other two reported that they had completed a Doctoral degree. Respondents with Master's degrees were more likely to agree (58% vs. 51%) with the system categorizations and less likely to be unsure (13% vs 23%) than were respondents holding Doctorates.

All of the respondents indicated at the time that they were recruited that they were familiar with the process of brainstorming. Three reported that they had never participated in a brainstorming session, three had participated in fewer than ten brainstorming sessions and three indicated that they had participated in at least ten sessions. Figure 13 shows the rates of agreement by experience as a participant in brainstorming sessions. Interestingly, the most likely to disagree (by a greater than 10% margin) were those who had participated from one to nine times.

Conversely, respondents who had facilitated brainstorming sessions from one to nine times were slightly more likely to agree with the system (62% vs. 55% vs. 56%) and less likely to disagree with the system (22% vs. 27% vs. 27%).

Figure 14 shows the frequency of each rating for each expert. In general, these experts were more consistent than the experts in study #1. In every case, the experts agreed with the system findings more often than they disagreed. Experts two and three are the farthest outliers of the group; two showing a strong preference for Disagree over Not Sure and three choosing Not Sure more frequently than any other expert.

In order to confirm this apparently high level of agreement between the experts, Cohen's kappa [7] was calculated for all pairwise combinations of experts to determine the reliability of their agreement. Note that this is a measure of the level of agreement, adjusted for chance agreements, between the experts — not between the

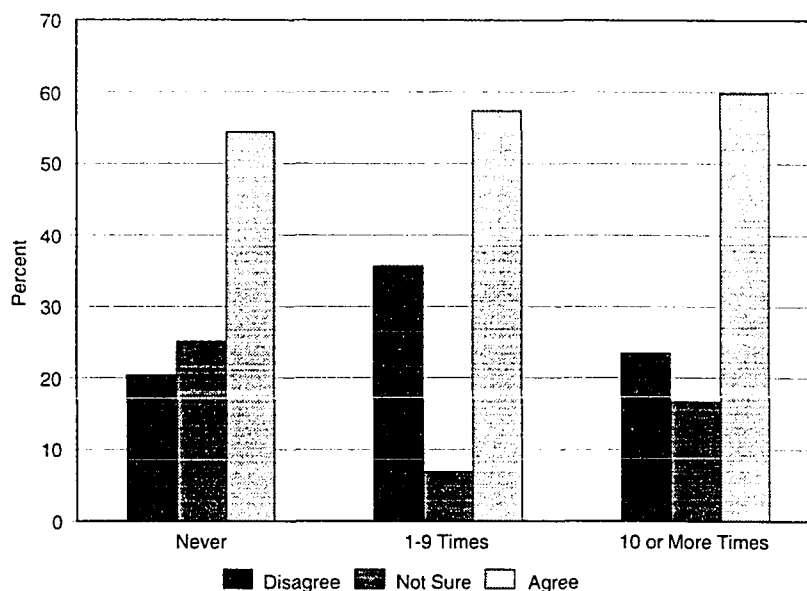


Fig. 13. Frequency of ratings by experience as a participant.

expert and the system. In order to evaluate the reliability of the system findings compared with experts it would have been necessary for the experts to develop content analytic categorizations of the transcript statements independently from the system categorizations. The pilot studies showed clearly that the cognitive load of such a task makes it unreasonable (if not impossible).

Table 3 shows the calculated kappa values for each pairwise comparison of experts. The standard errors of the kappa values ranged from 0.047 to 0.065 which creates a 95% confidence range about the kappa values of approximately plus or minus 0.13. Interpretation of Cohen's kappa is straightforward. A value of 0.0 indicates precisely the amount of agreement between two experts as would be expected by chance, values between 0.0 and -1.0 indicate more disagreement between experts than would be expected by chance, and values between 0.0 and +1.0 indicate greater agreement between experts than would be expected by chance. Values greater than

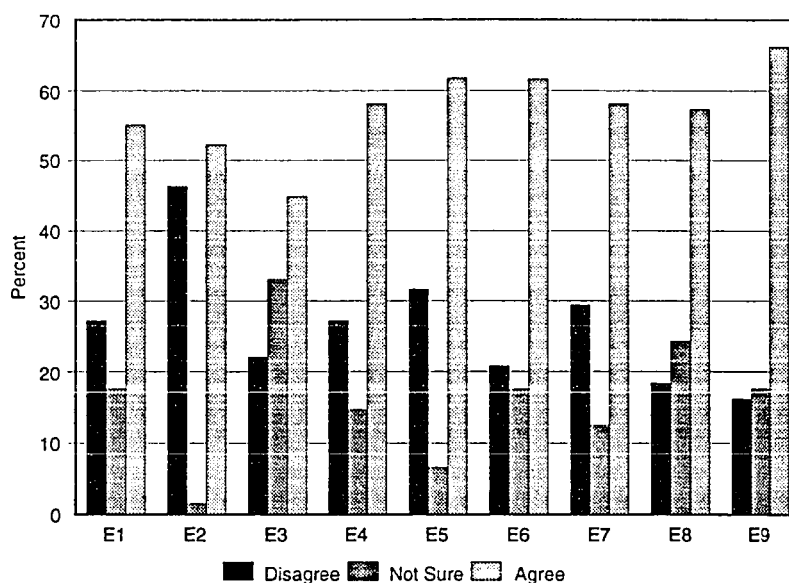


Fig. 14. Frequency of ratings by expert.

0.75 indicate strong agreement; above 0.4 good agreement; and below 0.4 poor agreement [65]. Even allowing for the 95% confidence range, no pair of experts achieve more than “good” agreement.

Figure 15 shows the agreement frequencies for study #2 by transcript. As in study #1, the greatest agreement was with the categorizations for the Business Prospect and MBA Critical Success Factors transcripts and the greatest disagreement

Cohen’s kappa is calculated as

$$\kappa = \frac{po - pc}{1 - pc}$$

where  $po$  is the observed proportion of agreement and  $pc$  is the expected proportion of agreement by chance which is the product of the sum of the proportions summed across the row times the proportions summed down the column. The standard error is given by

$$\sigma\kappa \cong \sqrt{\frac{po(1 - po)}{N(1 - pc)^2}}$$

Table 3. Cohen's kappa for each pair of experts.

Expert	Expert							
	2	3	4	5	6	7	8	9
1	0.298	0.051	0.268	0.315	0.231	0.183	0.192	0.182
2		0.172	0.434	0.465	0.175	0.306	0.211	0.279
3			0.127	0.210	0.076	0.331	0.217	0.103
4				0.458	0.238	0.194	0.227	0.448
5					0.258	0.315	0.282	0.279
6						0.089	0.087	0.076
7							0.222	0.307
8								0.214

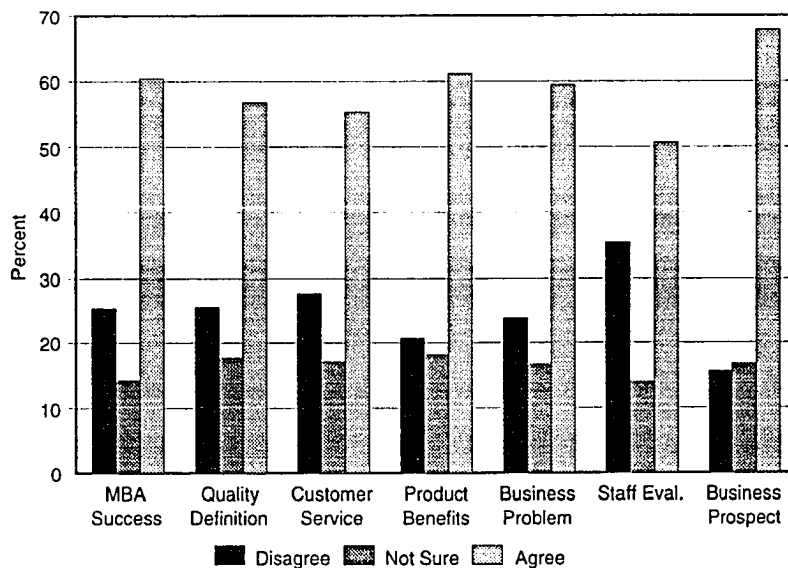


Fig. 15. Frequency of ratings by transcript.

was with the categorizations of the Staff Evaluation transcript. In general the study #2 experts seem a bit more conservative than do the experts in study #1 — with no aggregate agreement greater than 70%.

Figure 16 shows the agreement frequencies for study #2 by questionnaire part (issues, evaluatives, similarities). The results for issues and evaluatives are similar

to those found in study #1 — slightly more than 70%. For the similarities part of the questionnaire, however, the results are quite different from those found in study #1 — study #1 found 32% disagreement, in study #2 the experts disagreed with the system nearly 50% of the time.

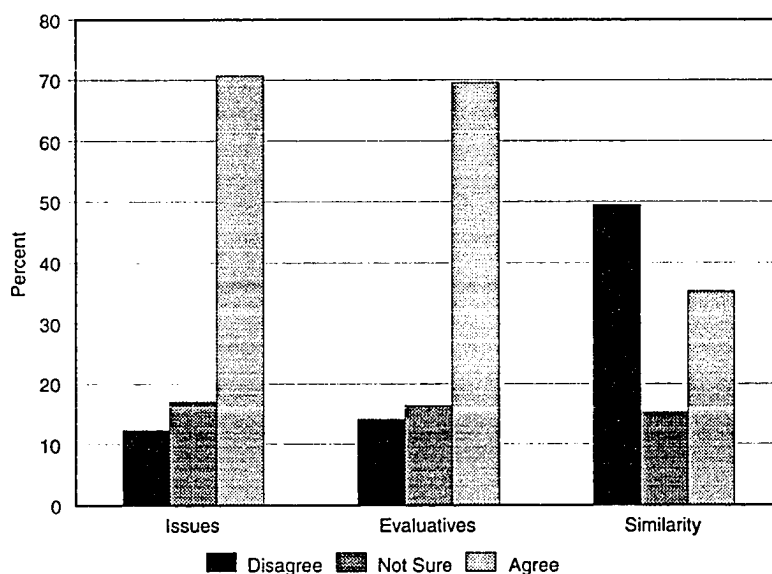


Fig. 16. Frequency of ratings by part.

Table 4 summarizes the frequency of agreement for each transcript  $\times$  questionnaire part for study #2.

A correspondence analysis was again performed on the levels of agreement by part, transcript, expert and question. No dominant axis appeared as it did in study #1. The interpretation of this is that the experts in study #2 were more consistent with each other than were the experts in study #1. The absence of a dominant axis signifies that the remaining variability in the co-occurrence of expert responses cannot

Table 4. Agreement frequency summary for study #2.

Transcript	Issues	Evaluatives	Similarity
MBA Success	76.67	55.56	46.67
Quality Definition	75.56	60.00	30.00
Customer Service	57.30	68.89	40.74
Product Benefits	80.90	63.89	31.75
Business Problem	71.11	71.11	24.44
Staff Eval.	61.25	86.67	23.33
Business Prospect	71.11	80.00	48.89

be explained by any single attribute or combination of attributes provided. For this reason the results of the correspondence analysis are not shown for study #2.

The similarity test performed in study #1 showed that only near duplicate statements could be reliably identified as 'similar.' In as much as the level of agreement with the system findings for similar statements was even lower in study #2, there was no need to repeat that test.



## SUMMARY AND CONCLUSION

The intent of this dissertation research was to provide an in-depth analysis of the types of messages that are transmitted during electronic brainstorming sessions, to evaluate empirically the usefulness of one framework for categorizing these messages, and to improve the level of computer support which can be provided to groups engaged in idea generation tasks.

Although the results from the issue identification phase of this study are quite promising, reaching on several occasions the 80% mark suggested by Clancey and Shortliffe [6], they cannot be interpreted as a pure success. When faced with the option to agree or disagree with the statement that, for example, a definition of customer service has a great deal to do with communications, the obvious answer is to agree. If, however, a user were attempting to search an organizational memory for discussions about customer service, it is unlikely that you would specify the search term "communications."

The problem is one of *level of granularity* in the domain of the group. True to its design, the Harvard IV-4 dictionary does an admirable job of categorizing a broad cross-section of everyday speech. Electronic brainstorming is seldom used for "everyday" discussions. Typically the domain of discussion in an EBS session will be much more narrowly focused. One possible solution to this problem would be to incorporate domain specific dictionaries that override a broad based dictionary like the Harvard IV.

The use of content analysis to identify similar statements will also benefit from more detailed domain data. Particularly in situations where the statements made by the participants are reasonably complete thoughts (as opposed to one or two words), this technique deserves consideration.

The results obtained for identification of evaluative and unrelated statements are less encouraging. Anyone who has participated in an EBS session (or read the transcripts in this study) will agree that these statements exist and that they can be identified; unfortunately it does not appear that content analysis is the right tool

for this identification. The author believes that these problems are the result of the “common sense knowledge” problem. Evaluative and unrelated statements tend to contain euphemisms, inside jokes, sarcasm, and (as noted earlier) personal references. Unless the domain in which the system is to operate is defined to include extensive knowledge about the discussants and their relationships to one another, this must be called “common sense.”

This study has made several steps toward validating the MMEBS presented earlier — issues are present in EBS transcripts and they can be identified automatically. Further, restatements and paraphrases exist in EBS sessions and can be identified automatically with some success. The current study did not produce evidence that positions, arguments or unrelated statements are present in EBS sessions, but there is still some face validity to those claims as well as evidence from the original and ongoing research by Conklin and others. Cohen’s kappa, used to measure the reliability of the human experts judgement of the system output shows that it is difficult to attain agreement even among experts as to whether or not categories are represented in transcript statements. This is consistent with the pilot study findings that inter-rater agreement was low. It should be noted that the experts in study #1 disagreed with one another even more than the experts in study #2 did.

A cautious affirmation of the primary research question can be provided — the MMEBS model can be (at least partially) implemented via content analytic tools to produce a plausible and useful categorization of electronic brainstorming thoughts. Issues can be identified at a low granularity and restatements can be identified provided they are near duplicates or they are long enough to make an inference about.

This research provides modest support for the contention that the MMEBS model is appropriate for the classification of EBS thoughts. More importantly, none of the findings of this research suggest that the MMEBS is an inappropriate model for such classification. One possible addition to the model would be a node type called *vote*. As noted earlier, EBS sessions are usually only one phase in a more complex decision making process. Very often EBS sessions are interspersed with evaluative

sessions. Inevitably, these evaluative sessions involve the registering of votes of solidarity or opposition. While the EBS session and the evaluative session are fairly easy to distinguish, they are clearly related by the content of the transcripts. Inclusion of a *vote* node would make the MMEBS model a more complete representation of the business decision making process.

The simple tag tallying procedures performed by the content analytic tools used in this study seem to be too shallow to capture the **relationships** between the model nodes. It appears that there are no ‘content only’ clues as to the presence of positions and arguments within a transcript. Assuming that positions and arguments do, in fact, exist in these transcripts, then the relationships between the statements made, not just their content must determine the presence or absence of a position or argument. Considerably more complex tools will be required to detect, organize and structure these relationships. Specifically, the unit of text considered as a unit can no longer be a sentence, but must be something larger. More complex rules will need to be developed to discover these relationships between statements.

Finally, it can be concluded from this study that the tools developed here can be applied with similar success to other EBS transcripts. However, as the domain of the EBS session become more narrowly focused, the performance of the system will decrease. This argues strongly in favor of domain specific dictionaries.

Perhaps the single most important conclusion that can be drawn from this study is that free form EBS sessions, while valuable as a creativity stimulator, are perhaps not the best choice for discussions that do not benefit from creativity (for example, the peer performance review) and that more structured techniques like the IBIS method may be more appropriate when the value of organizational memory is high or structured record keeping is desirable. Brainstorming is an inherently unstructured process, not a Swiss Army knife management tool. By design there are very few restrictions on participants in brainstorming — any addition of structure *a priori* may result in a reduction in the free flow of ideas that brainstorming is intended to generate.

Still, there exists a broad range of applications for group processes, both structured and unstructured. As these group decision support tools gain popularity in mainstream business organizations, the demand for trained facilitators, or surrogates thereof, will most certainly increase. Content analysis is one tool that can be applied to help bring order to otherwise unordered data.

## 1 Contributions

Of primary interest to this research was the identification of issues, positions, arguments, remarks (evaluative and metadiscussion), and clarifications (restatements). Issues are the only node type which can be promoted to the status of topic and are, therefore, the most likely to be pertinent in future EBS sessions; positions and arguments are the most likely candidates for continued interest in an evaluative process; clarifications/restatements need to be evaluated as a single entity (perhaps later treated separately); and remarks are the most likely to violate the guidelines of brainstorming. By identifying EBS ideas falling into these five categories, a significant portion of the organization needed in an EBS session could be achieved. The remaining node types (decisions, queries, and evidence) can be explicitly identified at the time of generation. That is, in order for group members to create one of these node types, they must (and can easily) identify the node or participant toward which the new node is directed.

There are four contributions from this dissertation to the theoretical state of the art in electronic brainstorming:

- A structural model and methodology for categorizing EBS thoughts. This research provides some support for the MMEBS model — particularly with regards to the existence of issues and restatements and to a lesser degree for evaluative statements.
- An evaluation of the MMEBS model as a tool for structuring transcripts of free form EBS sessions. Although this research did not irrefutably validate the MMEBS, neither did it invalidate the model.

- Empirical evidence that the facilitator's role can, in part, be supported by technology.
- A simple measure of substantive similarity between EBS messages.

It has been shown that portions of the MMEBS model can be applied to electronic brainstorming transcripts after the fact and that certain group tasks are better suited to this *post hoc* categorization than are others.

Particularly in spatially dispersed EBS sessions (something impossible under the constraints of manual brainstorming), it is not feasible to require a human facilitator. While video conferencing is a possible means for providing human facilitation to distant group participants, the technology is extremely expensive, both to acquire and to operate. Some systems (e.g., Plexsys<sup>®</sup>, SAMM<sup>®</sup>) have opted not to support spatially dispersed groups for this reason, others (e.g., VisionQuest<sup>®</sup>) support groups across distances, requiring users to provide their own mechanism for facilitation. Human facilitators are not always appropriate in sensitive EBS sessions. Although facilitators need to remain detached from the biases of the group members, using a non-group member may raise security issues. The ability to provide semi-automated support for even small amounts of facilitation effort can be quite valuable to certain groups.

The usefulness of EBS tools to serve as a meeting memory has been described by Nunamaker, Vogel and Konsynski [40]. They cite the following benefits: ability to review the full text of previous discussions, reduced need for backtracking in order to bring new group members "up to speed," reduced likelihood of overlooking problems and misunderstandings, and improving the understanding of the interrelated nature of issues. "Being able quickly and effectively to link related information to make it useful to participants and facilitators remains a continuing challenge" ([40] p. 149). This research directly addresses this challenge. (Valicich, Vogel, and Nunamaker [59] have proposed one mechanism by which this meeting memory can be scanned. The technique investigated in this dissertation provides support for a variety of alternative methods, including natural language processing and automated review of previous

group processes. By content analyzing the messages passed in one EBS session, the system will be able to identify, “quickly and effectively,” previous EBS sessions addressing similar or related issues, further reducing the possibility of overlooking important considerations.

Brainstorming is based on the theory that evaluative statements are undesirable in the generative phase [45]. Connolly, Jessup and Valacich [9] suggest that the effects of evaluative comments may be diminished in EBS. In any case, it may be desirable to identify evaluative statements. By identifying these statements, the originator of the EBS session can decide whether or not to allow such statements to be transmitted. The EBS environment offers an advantage over its manual counterpart in this regard. In manual brainstorming the only way to limit the transmission of evaluative statements is to discourage their expression. In EBS however, another alternative exists — allow participants to express evaluative statements at will, but delay transmission of those messages until the group has entered an evaluative process. In this way no information is lost and the potentially detrimental effects of evaluative statements are avoided. A third option would be to identify a designated “gatekeeper” for evaluative messages. This person could, in real time, decide whether or not a statement identified by the system as evaluative should be transmitted or delayed. This research concludes that the type of statement that Osborn did not feel should be expressed during brainstorming is rather rare in electronic brainstorming session, only perhaps because the variety of tasks to which EBS is applied was not anticipated. Still, evaluative statements *do* exist in EBS sessions. Content analysis was shown *not* to be an effective tool for identifying these.

Using the measure of similarity proposed and evaluated herein, the EBS system can cluster messages having a high degree of similarity together, allowing the system to report evaluative totals for groups of similar messages. While this dissertation has not directly addressed evaluative processes, the output of EBS is often the input to such processes. Certainly, this similarity clustering is imperfect, but any assistance in this area may save human facilitators considerable time and effort.

Finally, this research strengthens the argument for individualized interfaces to group process tools. Through two pilot studies involving non-experts and a study using expert facilitators the most consistent observation has been that each person “sees” something different in the process. Current EBS systems rely heavily on the WYSIWIS (What You See Is What I See) concept. This concept may be relaxed on four dimensions: where things appear, when things appear, who sees what, and how things appear when they are displayed [51]. Automated identification of communications components may be used to support any or all of these forms of communication filtering. For example, participants may specify that they are not qualified to deal with issues pertaining to production. Using this information, the system could be programmed to filter production related messages thereby reducing the likelihood of communication overload.

In addition to the theoretical contributions described above, there are two specific applied contributions:

- A public domain knowledgebase management system that accommodates variable length fields and keys.
- A much needed modernization of a popular content analysis tool, including a design and implementation of a knowledge base schema, an inference engine and several utilities that will ease the task of creating and maintaining content analysis dictionaries.

## 2 Limitations

Although Bales’ model and the IBIS model are both well established and tested theories of group interaction, they are by no means the only possible model that might be drawn upon. For example, McGrath [33] also has a well established model of group interaction.

As noted earlier, EBS is used for myriad tasks that manual brainstorming never was. This dissertation examined only a small subset of the possible kinds of EBS

transcripts; specifically those without technical jargon that the Harvard IV dictionary was not designed for. It is entirely reasonable to expect that content analysis would perform much better on transcripts from highly specialized fields in which a well defined vocabulary is used — given a proper dictionary. For example, a medical dictionary could be developed with relative ease because the vocabulary used in that field is unambiguous and precise.

Perhaps the greatest limitation of this study was the fact that the cognitive load on experts is so great and the task so unstructured that a double blind evaluation was not possible. In order to gain credibility as a tool for this task a double blind content analysis test must be defined. Such a test would have to be contrived in such a way as to reduce the cognitive load on the experts significantly. Unfortunately, such a contrived scenario may not be generalizable.

Finally, the fact that the suitability of only one dictionary was examined is a limitation of this study. Although the Harvard series is the most widely cited set of dictionaries it is by no means the only dictionary. Lasswell's value dictionary was designed specifically for identifying evaluative and emotional responses and might have performed better on that test. Given the low frequency of evaluative statements in the tested transcripts, even a dramatic improvement in performance might not have been valuable, however.

### 3 Future Research

LexNet itself can be improved in a number of ways. The addition of an interactive editor would make modifications to the dictionaries much easier. The morphological transformations should be eliminated entirely in favor of including each morphological form explicitly in the dictionary. The morphological transformation routines were originally chosen in order to conserve disk and memory resources — these resources are no longer in such short supply. The benefit from explicitly handling each morphological form would be a reduction in the number of incorrectly tagged word forms. The current system often incorrectly identifies undefined signs as



morphological transformations of defined signs, resulting in the assignment of tags. Because such tag assignments can trigger other rules elsewhere in the sentence it is preferable for the sign to remain untagged rather than to be mistagged. The CONTEXT programming language should be reexamined in light of recent advances in the field of semantic restrictions; as noted earlier, the CONTEXT language contains a number of constructs which are redundant and no longer useful. Finally, combining LexNet with WordNet [35] would expand the capabilities of both systems considerably.

WordNet<sup>®</sup> is an advanced online lexical dictionary that contains nearly 100,000 words, idioms and collocations. Unlike other online dictionaries, WordNet contains considerable information about the relationships between terms in the lexicon. For example, the Harvard IV-4 dictionary contains tag assignments for several dozen animals; each is assigned the tag ANI. In contrast, WordNet contains definitions for several hundred animals and also contains relational pointers which reflect the fact that each of these “inherits” from the term *animal*. By assigning the tag ANI to the term *animal* and modifying the LexNet system to search WordNet for inheritance (hyponymy), every one of the animals defined in WordNet can be disambiguatable (to some extent). Similarly, LexNet could be made to search WordNet for other relationships to assist in the disambiguation process. At least one attempt has been made to date to incorporate a disambiguation system into WordNet [60]. Unfortunately, that attempt was not successful. LexNet is based on a proven methodology that can be incorporated with WordNet fairly easily.

Perhaps the greatest potential for future research falls in the area of dictionary development. This research has clearly shown that while the Harvard IV-4 is an excellent general purpose disambiguation dictionary, it lacks information about specific problem domains that are common to business problem solving. In order for the development of a business lexicon dictionary to be justified, it must have application

---

WordNet is a trademark of Princeton University, Princeton, New Jersey.

in a repetitive, but unstructured group process. A generalized candidate consideration process (or peer performance review process), as is used by almost every major corporation seems to have many of the desired characteristics.

Finally, electronic brainstorming is apparently used for a variety of tasks that manual brainstorming would never be considered for. This is not surprising, given that it is such a new tool. Still, brainstorming (either electronic or manual) is an unstructured process that is not necessarily appropriate in a broad group of business decision-making processes. It seems that what is called for is a variety of tools that add particular kinds of structure to the general EBS process for specific tasks.

## REFERENCES

- [1] R. Bales, A Set of Categories for the Analysis of Small Group Interaction, *American Sociological Review* (1950) 257-263.
- [2] R. Bales, *Interaction Process Analysis: A Method for the Study of Small Groups*, Cambridge, MA: Addison-Wesley Publishing, 1950.
- [3] C. Beard, *The Development and Validation of an Expert System for Academic Advisement*, Unpublished Doctoral Dissertation, College Station, TX: Texas A&M University, 1991.
- [4] B. Berelson and S. DeGrazia, Detecting Collaboration in Propaganda, *Public Opinion Quarterly* (1947) 244-253.
- [5] B. Berelson, *Content Analysis in Communications Research*, New York, NY: Free Press, 1952.
- [6] W. Clancey and E. Shortliffe, *Readings in Medical Artificial Intelligence: The First Decade*, Reading, MA: Addison-Wesley Publishing, 1984.
- [7] J. Cohen, Weighted Kappa: Nomial Scale Agreement With Provision for Scaled Disagreement or Partial Credit, *Psychological Bulletin* (1968), 70(4), 213-222.
- [8] J. Conklin and M. Begeman, gIBIS: A Hypertext Tool for Exploratory Policy Discussions, *ACM Transactions on Office Information Systems* (1988) 303-331.
- [9] T. Connolly, L.M. Jessup and J.S. Valacich, Effects of Anonymity and Evaluative Tone on Idea Generation in Computer-Mediated Groups, *Management Science*, 36, June (1990) 689-703.
- [10] G. Easton, *Group Decision Support System versus Face-to-Face Communication for Collaborative Group Work: An Experimental Investigation*, Unpublished Doctoral Dissertation, Tucson, AZ: University of Arizona, 1988.
- [11] J. Fellers, *The Effect of Group Size and Computer Support on Group Idea Generation for Creativity Tasks: An Experimental Evaluation Using a Repeated Measures Design*, Unpublished Dissertation, Bloomington, IN: Indiana University, 1989.

- [13] B. Gallupe, *The Impact of Task Difficulty on the Use of a Group Decision Support System*, Unpublished Doctoral Dissertation, Minneapolis, MN: University of Minnesota, 1985.
- [14] G. Gerbner, O. Holsti, K. Krippendorff, W. Paisley and P. Stone, *The Analysis of Communication Content: Developments in Scientific Theories and Computer Techniques*, New York, NY: Wiley and Sons, 1969.
- [15] Michael J. Greenacre, *Theory and Applications of Correspondence Analysis*, London: Academic Press, 1984.
- [17] P. Gray and L. Olfman, *The User Interface in Group Decision Support Systems*, *Decision Support Systems* (1989) 119–137.
- [18] A. Grey, D. Kaplan and H. Lasswell, *Recordings and Context Units — Four Ways of Coding Editorial Content*, in Lasswell, H., Leites, N. and Associates, *Language of Politics*, Cambridge, MA: MIT Press, 1965.
- [19] P. Harmon and B Sawyer, *Creating Expert Systems for Business and Industry*, New York: John Wiley & Son, 1990.
- [20] C. Hick, J. Rush and S. Strong, *Content Analysis in: E. Dym (ed), Subject and Information Analysis*, New York, NY: Marcel Dekker Inc., 1985.
- [21] Starr R. Hiltz and Murray Turoff, *Structuring Computer-Mediated Communication Systems to Avoid Information Overload*, *Communications of the ACM*, 28, July (1985) 680–689.
- [22] O. Holsti, *Content Analysis for the Social Sciences and Humanities*, Menlo Park, CA: Addison-Wesley Publishing, 1969.
- [23] P. Jackson, *Introduction to Expert Systems*, Reading, MA: Addison-Wesley Publishing, 1990.
- [25] E. Kelly and P. Stone, *Computer Recognition of English Word Senses*, Amsterdam: North Holland, 1975.
- [26] E. Kerr and S. Hiltz, *Computer-Mediated Communications Systems: Status and Evaluation*, New York, NY: Academic Press, 1982.
- [27] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology*, Beverly Hills, CA: Sage Publications, 1980.

- [28] W. Kunz and H. Rittle, *Issues as Elements of Information Systems*, Working Paper 131, Institute of Urban and Regional Development, Berkeley, CA: University of California, 1970.
- [29] F. Lewis, *Facilitator: A Microcomputer Decision Support Group for Small Groups*, Unpublished Doctoral Dissertation, Tucson, AZ: University of Arizona, 1985.
- [30] R. Mason, *Measuring Information Output: A Communication System Approach*, *Information and Management* (1978) 219-234.
- [31] A. McCartt and J. Rohrbaugh, *Evaluating Group Decision Support System Effectiveness: A Performance Study in Decision Conferencing*, *Decision Support Systems* (1989) 243-253.
- [32] J. McDermott, *R1: The Formative Years*, *AI Magazine*, 2(2), (1981), 21-29.
- [33] J. McGrath, *Groups: Interaction and Performance*, Edgewood Cliffs, NJ: Prentice-Hall Inc., 1984.
- [35] George A. Miller, R. Beckwith, C. Fellbaum, D. Gross and K. Miller, *Five Papers on Wordnet*, Technical Report, Princeton, NJ: Cognitive Science Laboratory, Princeton University, 1994.
- [36] C. Moore, *Group Techniques for Idea Building*, Beverly Hills, CA: Sage Publications, 1987.
- [37] J. Namenwirth and R. Bibee, *Speech Codes in the Press*, *Journal of Communications* (1975) 50-63.
- [40] J. Nunamaker, D. Vogel and B. Konsynski, *Interaction of Task and Technology to Support Large Groups*, *Decision Support Systems* (1989) 139-152.
- [41] J. Nunamaker, D. Vogel, A. Heminger, B. Martz, R. Grohowski and C. McGoff, *Experiences at IBM with Group Support Systems: A Field Study*, *Decision Support Systems* (1989) 183-196.
- [42] Ogilvie, D., *Procedures for Improving the Interpretation of Tag Scores: The Case of Windle in: P. Stone, D. Dunphy, M. Smith and D. Ogilvie (eds.), The General Inquirer: A Computer Approach to Content Analysis*, Cambridge, MA: MIT Press, 1966.
- [43] R. O'Keefe, O. Balci and E. Smith, *Validating Expert System Performance*, *IEEE Expert* (1987), 2(3), 81-89.

- [45] A.F. Osborn, *Applied Imagination*, Scribner's, New York, NY, rev. 1963.
- [46] J. Rawlinson, *Creative Thinking and Brainstorming*, New York, NY: John Wiley and Sons, 1981.
- [47] H. Rittle, *APIS: A Concept for an Argumentative Planning Information System*, Working Paper 324, Institute of Urban and Regional Development, Berkeley CA: University of California, 1980.
- [48] K. Rosengren, *Advances in Content Analysis*, Beverly Hills, CA: Sage Publications, 1981.
- [50] R. Steeb and S. Johnston, *A Computer-Based Interactive System for Group Decision Making*, *IEEE Transaction Systems, Man, and Cybernetics* (1981).
- [51] M. Stefik, D. Bobrow, G. Foster, S. Lanning and D. Tartar, *WYSIWIS Revised: Early Experiences with Multiuser Interfaces*, *ACM Transaction, Office Information Systems* (1987) 147-186.
- [53] P. Stone, D. Dunphy, M. Smith and D. Ogilvie, *The General Inquirer: A Computer Approach to Content Analysis*, Cambridge, MA: M.I.T. Press, 1966.
- [55] Philip J. Stone, Dexter C. Dunphy and Robert Philip Weber, *The Harvard IV-3 & 4 General Inquirer Dictionaries*, Mannheim, Federal Republic of Germany: ZUMA The Center for Surveys, Research and Methodology, 1987.
- [57] University of California, Berkeley, *flex(1)*, Version 2.4, *BSD Programmer's Manual*, 4.3-Reno Berkley Distribution, 1993.
- [59] J. Valicich, D. Vogel and J. Nunamaker, *A Semantic Guide Interface for Knowledge Base Supported GDSS*, *Transactions on Decision Support Systems*, June (1988).
- [60] Ellen Vorhees, *Using WordNet to Disambiguate Word Senses for Text Retrieval*, *Proceedings of the ACM-SIGIR Conference*, Pittsburgh, PA: Association for Computing Machinery, 1993.
- [61] R. Weber, *Society and Economy in the Western World System*, *Social Forces* (1981) 1130-1148.
- [62] R. Weber, *Measurement Models for Content Analysis, Quality and Quantity* (1983) 127-149.

- [63] R. Weber, Computer-Aided Content Analysis: A Short Primer, *Qualitative Sociology* (1984) 126–147.
- [64] R. Weber, *Basic Content Analysis*, Beverly Hills, CA: Sage Publications, 1990.
- [65] L. Wilkinson, *SYSTAT: The System for Statistics*, Evanston IL: SYSTAT Inc., 1990.
- [66] I. Zigurs, *The Effect of Computer Based Support on Influence Attempts and Patterns in Small Group Decision Making*, Unpublished Doctoral Dissertation, Minneapolis, MN: University of Minnesota, 1987.
- [67] I. Zigurs, *Interaction Analysis in GDSS Research: Description of an Experience and Some Recommendations*, *Decision Support Systems* (1989) 233–241.
- [68] C. Züll, R. Weber and P. Mohler, *Computer-Aided Text Classification for the Social Sciences: The General Inquirer III*, Mannheim, Federal Republic of Germany: ZUMA The Center for Surveys, Research and Methodology, 1989.

### **Supplementary Sources Consulted**

- [1] W. N. Francis and H. Kucera, *Frequency Analysis of English Usage: Lexicon and Grammar*, Boston, MA: Houghton Mifflin Company, 1982.
- [2] J.C. Giarratano, *CLIPS Version 6.0*, Lyndon B. Johnson Space Center, Software Technology Branch: NASA, 1993.
- [3] Richid Jernigan IV, *Metalbase, version 4.0*, Norris, TN: PO Box 827, 1992.
- [4] George A. Miller, Claudia Leacock, Randee Teng and Ross T. Bunker, *A Semantic Concordance*, Princeton, NJ: Cognitive Science Laboratory, Princeton University, 1994.
- [5] Zvi Namenwirth and Robert Philip Weber, *The Lasswell Value Dictionary*, Mannheim, Federal Republic of Germany: ZUMA The Center for Surveys, Research and Methodology, 1987.
- [6] NeXT Computer, Inc., *IndexingKit*, Redwood City, CA: NeXT Computer, Inc., 1990.

- [7] D. O'Leary, Verifying and Validating Expert Systems: A Survey, in: P. Watkins and L. Eliot (eds.), *Expert Systems in Business and Finance: Issues and Applications*, Chichester, NY:John Wiley & Sons, 1993.
- [8] SAS Institute Inc., *SAS/STAT User's Guide, Version 6, Fourth Edition, Volume 1*, Cary, NC: SAS Institute Inc., 1989.
- [9] A. Stevens, *C Database Development*, Portland, OR: Management Information Source Inc., 1987.
- [10] Philip J. Stone and Robert Philip Weber, *The General Inquirer Applications Programs*, Mannheim, Federal Republic of Germany: ZUMA The Center for Surveys, Research and Methodology, 1987.
- [11] University of California, Berkeley, *OPEN(2), BSD Programmer's Manual*, 4.3 Berkley Distribution, 1980.
- [12] University of California, Berkeley, *dbopen(3), BSD Programmer's Manual*, 4.4 Berkley Distribution, 1993.



## APPENDIX A

### THE DBMS(3) SYSTEM

#### A.1 Introduction

DBMS(3)<sup>†</sup> uses the binary tree (B-Tree) engine distributed by the University of California at Berkeley in the Berkeley Software Distribution (BSD) 4.4 called `dbopen(3)` [58]. The `dbopen(3)` library stores arbitrary, associated key/data pairs in a sorted, balanced tree structure. The key/data pairs are arbitrary because their contents are treated as sequences of bytes without any assumptions as to the contents or meaning of those bytes. The key/data pairs are associated in that the key value, alone, can be used to retrieve the data value. `Dbopen(3)` solves many of the matters concerning the physical storage of data but does not provide a broad logical structure for maintaining relationships between data — that is accomplished by DBMS(3).

DBMS(3) consists of two parts: a set of data structures which, when taken together, form a database schema; and a set of library functions that operate on those structures. Each will be briefly described, in turn.

Each database is represented by a single data structure called a *schema*. The schema describes one or more *tables* which will contain a number of data records (called *tuples*). Each tuple consists of a number of *fields*. Tables are accessed via *keys*, which define a logical order for record retrieval. Each of these, except tuple, has an associated data structure in DBMS(3) — Appendix A contains the ‘C’ language descriptions of these structures. Tuples exist in one of two forms in the DBMS(3) system: either as a user (application programmer) defined ‘C’ data structure, or as an arbitrary key/data pair. The user need not be concerned with this arbitrary

---

<sup>†</sup> It is customary for UNIX documentation titles to include a number which identifies the type of system being documented. The identifier (3) is reserved for ‘C’ programming libraries.

data representation, as the system automatically converts between the two formats as needed.

## A.2 Schema Specification

Specifying a database schema is accomplished by declaring a series of 'C' data structures. Each data structure has a corresponding datatype definition: DBMSFIELD, DBMSKEY, DBMSTABLE, and DBMSSHEMA.

### DBMSFIELD

**fieldName** may be of any length, may contain spaces (although that is ill-advised) and is case-insensitive.

**format** must be one of the following manifest constants:

**DBMS\_BYTE** a one-character integer.

**DBMS\_SHORT** a short integer.

**DBMS\_USHORT** an unsigned short integer.

**DBMS\_LONG** a long integer.

**DBMS\_ULONG** an unsigned long integer.

**DBMS\_FLOAT** a floating-point value.

**DBMS\_DOUBLE** a double-precision floating-point value.

**DBMS\_CHAR** a fixed number of characters (string).

**DBMS\_VARCHAR** a variable number of characters (string). Fields of type VARCHAR may be used in keys (indices) or in any way that other fields are used, but all fields of this type must be specified *last* in the array of fields which describes a table (see the example below).

**printFunction** is the name (address) of a function that will print data of the datatype specified for this field. This function must accept as its sole argument an untyped (void) pointer. The following functions, corresponding to the above formats, are provided:

```
void printByteField(void *dataPointer);
void printShortIntField(void *dataPointer);
void printUShortIntField(void *dataPointer);
void printLongIntField(void *dataPointer);
void printULongIntField(void *dataPointer);
void printFloatField(void *dataPointer);
void printDoubleField(void *dataPointer);
void printCharField(void *dataPointer);
```

**comparator** is the name (address) of a function that compares two fields of this datatype, returning an integer less than, equal to, or greater than zero if the first field is logically less than, equal to, or greater than the second, respectively. The function must accept exactly two untyped (void) pointers. The following functions, corresponding to the above formats, are provided:

```
int byteCompare(void *data1, void *data2);
int shortIntCompare(void *data1, void *data2);
int uShortIntCompare(void *data1, void *data2);
int longIntCompare(void *data1, void *data2);
int uLongIntCompare(void *data1, void *data2);
int floatCompare(void *data1, void *data2);
int doubleCompare(void *data1, void *data2);
int charCompare(void *data1, void *data2);
```

**size**, **DBTOffset**, **structOffset** are all calculated during the initialization process and can be set to zero.

## DBMSKEY

Data in the database is always accessed in the logical order of a specified key. Each table must have at least one key (the *primary* key). Each *foreign* key (a key to some other table) should also be defined. Keys consisting of more than one field (*compound* keys) are supported. Additionally, keys may be defined to produce a sorting order in which the database tuples will be processed.

**keyName** may be of any length, may contain spaces (although that is ill-advised) and is case-insensitive.

**fieldNames** is an array of character string pointers representing the names of the fields that comprise the key, in order.

**allowDups** may be set to the manifest constant **YES** to indicate that duplicate key values should be allowed, or **NO** to indicate that they should not. Attempts to insert records with duplicate keys will fail if this value is set to **NO**. It is permissible to define the primary key for a table to allow duplicate keys, but this may produce undesirable results as attempts to retrieve such records using any key other than the primary key will always return the first record matching the key value.

size, keyNumber, fields, cursor, bTree are all calculated during the initialization process and can be set to zero or **NULL**.

## **DBMSTABLE**

A **DBMSTABLE** is little more than an array of fields and an array of keys:

**tableName** may be of any length, may contain spaces (although that is ill-advised) and is case-insensitive.

**fields** is the name (address) of an array of **DBMSFIELDS**s.

**keys** is the name (address) of an array of **DBMSKEYS**s.

**dataFileName** is a character string that will be used to name the files which contain the data and keys for this table. The primary key and data will be stored in a file named *dataFileName.dat* and each subsequent key will be stored in a file with the same name, but the extension will be '001' for the first key, '002' for the second key, and so forth.

**numFields**, **recordSize**, **structSize** are all calculated during the initialization process and can be set to zero.

## **DBMSSCHEMA**

Finally, a **DBMSSCHEMA** is little more than an array of tables:

**tables** is the name (address) of an array of **DBMSTABLES**s.

**path** is a character string representing a path to the disk directory where the table files which make up this database are/should be stored. This path may be relative or absolute, but must exist.

**dataBaseName** is a character string that will be displayed whenever this schema is initialized. If set to **NULL**, then nothing will be displayed.

**copyright** is a character string that will be displayed whenever this schema is initialized. If set to **NULL**, then nothing will be displayed.

## Example

Figure 17 shows the definition of a table called WordForms which has six fields and two keys (a primary key and one other). The WordForms table is then shown in the array of tables which make up the final schema. The corresponding 'C' data structure definition is:

```
struct wordForm
{
    unsigned long wordFormNumber;
    char wordFormPosition;
    unsigned long wordNumber;
    char partOfSpeech;
    char probability;
    char *gloss;
};

typedef struct wordForm WORDFORM;
```

A complete listing of the LexNet schema is shown in Appendix C.

### A.3 Application Interface

Once the schema is completed, the database can be accessed logically using the following functions:

#### **initDataBase**

Initializes the database, calculates various values needed internally by the system. This function must be called before any other DBMS(3) function calls. The function accepts four arguments: a pointer to the database schema; a character string containing the name of the disk directory in which the database exists (or should be created); a character string containing the title of the database; and a character string containing a copyright message for the database. The first argument is required, the remaining arguments may be **NULL**. If the second argument is **NULL**, then the database will be accessed/created in the current directory. The third and fourth arguments are displayed on the screen.

```

/* **** define the fields **** */
DBMSFIELD wordFormFields[]={
    {"WordFormNumber",DBMS_ULONG,printLongIntField, longIntCompare,
     0,0,0},
    {"WordFormPosition",DBMS_BYTE,printByteField, byteCompare,
     0,0,0},
    {"WordFormWordNumber",DBMS_ULONG,printLongIntField,
     longIntCompare, 0,0,0},
    {"WordFormPartOfSpeech",DBMS_BYTE,printByteField, byteCompare,
     0,0,0},
    {"Probability",DBMS_BYTE,printByteField, byteCompare,
     0,0,0},
    {"WordFormGloss",DBMS_VARCHAR,printCharField, charCompare,
     VAR_LEN,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}};

/* **** define which fields make up each key **** */
char *wordFormKeyFields[]={"WordFormNumber",NULL};
char *wordFormWordKeyFields[]={"WordFormWordNumber",
    "WordFormPosition",NULL};

/* **** define the keys **** */
DBMSKEY wordFormKeys[]={
    {"WordFormKey",wordFormKeyFields,YES,0,0,NULL,NULL,NULL},
    {"WordFormWordKey",wordFormWordKeyFields,NO,0,0,NULL,NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}};

/* **** define all the tables**** */
DBMSTABLE lNetDictTables[]={
    {"DictionaryWords",dictWordFields,dictWordKeys,"dictword",0,0,0},
    {"DictionaryTags",dictTagFields,dictTagKeys, "dicttag",0,0,0},
    {"WordForms",wordFormFields,wordFormKeys, "wordform",0,0,0},
    {"SearchTags",searchTagFields,searchTagKeys, "srctag",0,0,0},
    {"SearchWords",searchWordFields,searchWordKeys, "srchword",0,0,0},
    {"Rules",ruleFields,ruleKeys, "rule",0,0,0},
    {"TagsApplied",tagsApplyFields,tagsApplyKeys, "tagsaply",0,0,0},
    {NULL,NULL,NULL,NULL,0,0,0}};

/* **** finally, define the schema **** */
DBMSSCHEMA lNetDictSchema={lNetDictTables,NULL,NULL,NULL};

```

Fig. 17. A portion of a database schema.

**openTable**

Opens a specific table in the database and all of its underlying indices. This function should be called for each table prior to inserting, retrieving, updating, or deleting records from that table. The function accepts three arguments: a pointer to an initialized database schema; a character string containing the name of the table to be opened; and an integer representing one or more flags which indicate the mode in which the table should be accessed (e.g. exclusively or shared access, read-only or read-write, etc.) — see `open(2)` [56] for an itemized description of all acceptable modes.

**insertRecord**

Inserts a 'C' data structure into the specified table. The function accepts three arguments: a pointer to an initialized database schema; a character string containing the name of the table into which the new record should be inserted; and a pointer to the 'C' data structure to insert.

**retrieveRecord**

Returns a pointer to a 'C' data structure which meets the specified restrictions. The function accepts six arguments: a pointer to an initialized database schema; a character string containing the name of the table from which the record should be retrieved; a character string containing the name of the key which specifies the logical order of retrieval; two 'C' data structures of the type being retrieved which, together, specify the minimum and maximum acceptable key values, respectively; and an integer specifying logical operation which may be any of the following:

**R.FIRST** Retrieve the minimum record logically equal to or greater than the specified minimum key value.

**R.NEXT** Retrieve the next logical record. This operation should be preceded by an **R.FIRST** or **R.CURSOR** operation.

**R.PREV** Retrieve the previous logical record. This operation should be preceded by an **R.LAST** or **R.CURSOR** operation.

**R\_LAST** Retrieve the maximum logical record less than or equal to the specified maximum key value.

**R\_CURSOR** Retrieve the logical record that matches the minimum key value.

All fields in the minimum and maximum key value structures, except those that form the key itself, are ignored. If there is no record in the specified table that meets the range restrictions as specified by the minimum and maximum key value and the requested operation, then this function returns **NULL**.

#### **deleteRecord**

Removes all records from the specified table that fall within the minimum and maximum key restrictions, inclusively. This function accepts five arguments, corresponding to the first five arguments of the `retrieveRecord` function described above.

#### **updateRecord**

Similar to the `insertRecord` function, except that a matching record in the database will be replaced, if it exists. This function accepts three arguments, corresponding to the arguments of the `insertRecord` function described above.

#### **closeTable**

Closes the specified table. Frees all associated resources. This function accepts two arguments: a pointer to an initialized database schema; and a character string containing the name of the table to be closed. Normal application termination will automatically close all open tables.

When efficiency is a consideration, the following functions may also be used: **insert\_Record**, **retrieve\_Record**, **delete\_Record**, **update\_Record**. These functions are identical to their non-underscored counterparts with two exceptions: instead of a pointer to a database schema and a table name, they accept a single pointer to a table — this saves having to locate the appropriate table in the schema; also, they do not synchronize the database on disk with the internal memory buffers.



Use of these functions greatly enhances performance, but at the cost of some complexity and risk. First, the user must maintain a pointer to the desired table structure. Such a pointer is returned by the `getTableWithName` function, which accepts a pointer to a database schema and a character string containing the name of a table. And second, there is a possibility that a system failure could cause the loss of some database changes not yet synchronized to disk. The user can force immediate synchronization using the `syncTable` function, which accepts a pointer to a database schema and a character string containing the name of the table to synchronize, or the `sync_Table` function which accepts a pointer to a database table as its only argument.

#### A.4 Programmer Notes

The DBMS(3) library of functions performs two major tasks: conversion of ‘C’ data structures into arbitrary “database thangs”<sup>†</sup> (DBT) and *vice versa*, and maintenance of keys (or indices). The library provides two high level functions, `cStructToDBTData` and `DBTDataToCStruct`, to perform the former. The latter is performed implicitly. Figure 18 shows the relationship between ‘C’ data structures and DBT data elements.

##### A.4.1 Format Conversions

Each DBT is composed of two elements: a *size* element and a *data* element. The size element simply contains the actual size of the data element. The following discussion, therefore, concentrates primarily on the management of the data element. There are two differences between the way that data is stored in a ‘C’ data structure and the way the same data is stored as “arbitrary data.” First, arbitrary data can not contain pointers to other data — all the data must be stored contiguously and, second, there is no need to align data elements on “word boundaries” in a DBT data element as there is in ‘C’ data structures.

---

<sup>†</sup> “Database thang” is a term chosen by the authors of `dbopen(3)` for lack of a more suitable term [58].

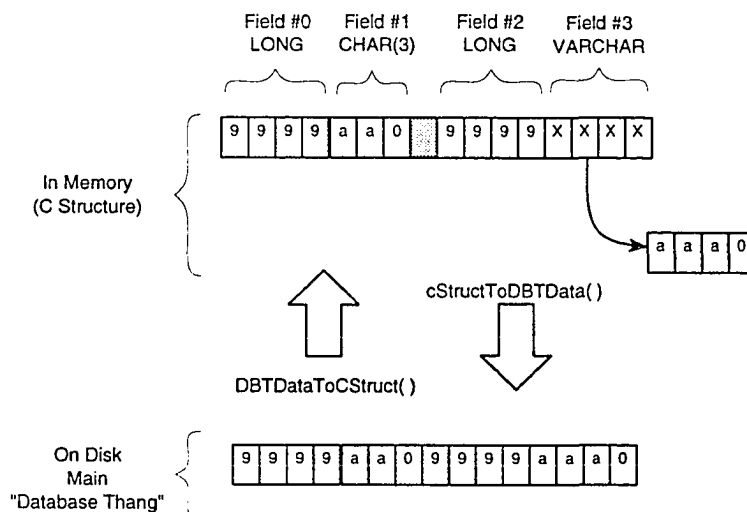


Fig. 18. Conversion between DBT data elements and 'C' data structures.

### `cStructToDBTData`

The only datatype currently supported by DBMS(3) which is stored in a 'C' data structure as a pointer is the VARCHAR datatype (a variable length character string). The `cStructToDBTData` function dereferences these pointers into the DBT that it constructs. Because such dereferenced data is (by definition) of an undetermined length, references to every field which follows a field of type VARCHAR must be calculated at run-time. It is for this reason that all VARCHAR fields must be specified last in the table definition. This function requires two arguments: a pointer to a DBMSTABLE structure and a (void) pointer to the 'C' data structure to be converted. The 'C' data structure is assumed to be the type associated with the specified table. The return value is a (void) pointer to a newly allocated DBT data element. The calling function is responsible for deallocating this memory when the DBT is no longer needed.

## DBTDataToCStruct

Most computers store certain types of data at memory locations that are even multiples of 2, 4, 8, etc. For example, the BSD Unix operating system used to develop DBMS(3) aligns all data except fixed length character arrays (fixed length strings) to even numbered memory locations. This improves the efficiency of the computer's memory management. The boundary alignments for a particular computer are calculated by the `initDataBase` function (see DBMS(3) Application Interface); the `DBTDataToCStruct` function then uses this information to "unpack" the DBT data when creating a 'C' data structure. This improves the portability of DBMS(3) data files. The `DBTDataToCStruct` function requires two arguments: a pointer to a `DBMSTABLE` structure and a (void) pointer to the DBT data element to be converted. The DBT data element is assumed to have been retrieved from the specified table. The return value is a (void) pointer to a 'C' data structure of the type associated with the specified table. The calling function is responsible for deallocating this memory when the structure is no longer needed.

There are several functions that are used in the process of converting to and from arbitrary DBT data elements. Their arguments follow a consistent pattern:

**theTable** a pointer to a `DBMSTABLE` data structure;

**theField** a pointer to a `DBMSFIELD` data structure;

**theKey** a pointer to a `DBMSKEY` data structure;

**theStruct** a (void) pointer to a 'C' data structure which is assumed to be of the type associated with **theTable**; and

**theData** a (void) pointer to an arbitrary DBT data element which is assumed to have been retrieved from **theTable**.

The conversion support functions are:

### calcFieldSizeFromDBT

Returns the actual size of a field stored in a DBT data element.

**calcStructSizeFromDBT**

Returns the actual size of a field stored in a 'C' data structure, dereferencing if necessary.

**calcDBTSize**

Returns the total size of a DBT large enough to contain the data in a 'C' data structure, dereferencing if necessary.

**calcKeySizeFromDBT**

Returns the cumulative size of all the fields which make up the specified key, given a DBT data element.

**calcDBTDataSize**

Returns the cumulative size of all the fields which make up the data element associated with a specified key. For the primary key, this will equal the size of a DBT large enough to hold the entire 'C' data structure. For all other keys, this will equal the size of the primary key (which may be variable).

**calcKeySizeFromStruct**

The complementary function to **calcKeySizeFromDBT**.

**calcFieldSizeFromStruct**

The complementary function to **calcFieldSizeFromDBT**.

**calcDBTFieldPtr**

Returns a (void) pointer into the specified arbitrary DBT data element which points to the actual beginning of the stored data for the specified field.

**calcStructFieldPtr**

Returns a (void) pointer into the specified 'C' data structure, dereferencing as needed, which points to the actual beginning of the stored data for the specified field.

#### A.4.2 Key Maintenance

Dbopen(3) stores data in arbitrary key-data pairs. Both the key and the data portions of this pair are stored as DBTs (“DataBase Thangs”). A DBT key identifies its data pair partner. This idea is abstracted to another level by DBMS(3).

Keys (indices) are used by DBMS(3) to identify specific records or groups of records with a table, to establish ranges of records to be operated upon, and to specify the order in which records are to be processed. In other words, keys in DBMS(3) are to tables as keys in dbopen(3) are to data elements. Duplicate keys are allowed. One key (the first key defined for a given table) is considered the *primary* key and is assumed to identify uniquely each record in that table. Duplicate primary keys are allowed but make little sense because accessing records in the table by any key other than the primary key will always result in retrieval of only one of the duplicate entries. See DBMS(3) Schema Specification for details on defining keys. Figure 19 shows the relationship between the primary key DBT and all other key DBTs in DBMS(3). Logically, the data portion of a non-primary key DBT points to a primary key which, in turn, points to the record to be accessed.

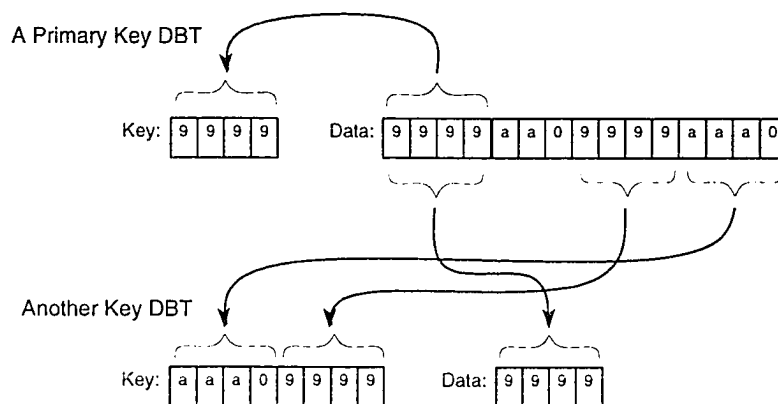


Fig. 19. Primary versus other key DBTs.

The following functions are used to construct the key-data pairs (arguments follow the convention described above):

### **getKeyValueFromDBT**

Returns a (void) pointer to arbitrary data which represents the key value for the specified key, constructed from the specified DBT. The DBT is assumed to have been constructed from a 'C' data structure of the type associated with the specified table. The calling function is responsible for deallocating this memory when the key value is no longer needed.

### **getPrimaryValueFromDBT**

A "cover" for **getKeyValueFromDBT** for key number zero.

### **getKeyValueFromStruct**

The complementary function to **getKeyValueFromDBT**.

Duplicate keys may be specified in **dbopen(3)**, but that mechanism is not adequate when more than one key is being specified for the same data element (as it can be in **DBMS(3)**). Therefore, **DBMS(3)** checks for duplicate key values before inserting any of the key-data pairs associated with a table record. The function **duplicateKeyError** returns zero (the manifest constant **FALSE**) if there are no conflicting duplicate keys in the specified DBT, or one (the manifest constant **TRUE**) if any key for this table disallows duplicate keys and the specified DBT would create such a duplicate.

Because the **DBMS(3)** functions have been designed in this way, a record insertion operation can be considered to be simply a series of key insertion operations (one of which happens to be the primary key whose data element is the actual table record). Thus, the **insertRecord** and **deleteRecord** functions are little more than calls to **insertKey** and **deleteKeysForRecord** respectively.

#### A.4.3 Debugging Functions and Global Variables

Several functions are available for printing key-data pairs. Their usefulness is primarily in debugging the **DBMS(3)** code. **PprintAKeyDataPair** accepts two DBTs, a

pointer to a `DBMSTABLE` and a pointer to a `DBMSKEY`. It calls `pprintAKeyDBT` and `pprintADataDBT` in turn. Each of those functions then calls `pprintAKey` and `pprintAData`. Finally, these functions call the function specified (in the schema) for each field which comprises the key data element or the data data element, respectively. Each field value is separated by a colon, which produces a readable, if not attractive printout of the complete key-data pair.

Two global variables are used by `DBMS(3)`. The first is `watchDBMS`. If `watchDBMS` is non-zero, then `pprintAKeyDataPair` will be called immediately before each record insertion and immediately after each retrieval. Other informational messages will also be displayed, indicating each operation that `DBMS(3)` is performing. The other global variable, `theCurrentKey`, is intended for internal use only. Because all table insertions are handled by a single set of functions, it is not possible to declare to `dbopen(3)` exactly which function should be called for key comparison. The solution to this problem was to always have `dbopen(3)` call the `compareKeys` function and then add logic to that which determines, at run-time, which comparisons should be performed. In order for it to make this determination, `compareKeys` must know which key (and therefore which table) is being operated upon. The global variable `theCurrentKey` is used to track this information. `TheCurrentKey` is set so that it points to the appropriate key before each `dbopen(3)` function call.

## A.5 Future Development

There are a number of ways in which `DBMS(3)` might be improved. This section briefly describes some of these possible improvements and enhancements.

Addition of a 'serial number' data type — each table should maintain its own serial number. The serial number should be incremented each time that a record is added to the table. Any field whose type is specified as serial and whose value is zero should automatically be set to the next available serial number when the record is added. This serial number data type should be very useful for primary keys. The `insertRecord` function should return the serial number of the inserted record.

Error handling should be centralized. Currently, each function is responsible for completely handling any possible errors. These error processing routines are somewhat redundant and are input/output dependent (i.e. they depend on direct access to the terminal which precludes use in windowed environments). Most functions will currently terminate the application (after presenting an appropriate error message); a more elegant error recovery mechanism should be implemented. More meaningful return values from most functions might be added.

The DBMS(3) library was intentionally designed with Structured Query Language (SQL, a standard language for describing and manipulating relational databases) syntax in mind. Although not implemented in this version, it is not difficult to imagine defining the schema by specifying SQL 'create' statements rather than declaring 'C' data structures. These SQL statements could be embedded directly in the 'C' source code, as the schema currently is, or they could be executed independently to produce a schema of the current form. A further extension would be to incorporate a small SQL interpreter for DBMS(3) databases to perform *ad hoc* queries.

DBMS(3) was always intended as a single-user product, but extension into a multi-user environment is not unthinkable. Although not explicitly stated, the `dbopen(3)` documentation implies that concurrency and transaction processing may be supported in future releases. Any transaction processing mechanism integrated with `dbopen(3)` could easily, if not transparently, be used by DBMS(3). Record locking would, however, need to be modified in a manner similar to the DBMS(3) handling of duplicate keys.

The DBMS(3) schema could be stored in a file, rather than in the source code itself. Two advantages would accrue from this change: the schema for a database could be changed without *necessarily* recompiling all associated applications; and a convenient location for storing serial number and record locking information would be created.

Finally, the DBMS(3) library could be combined with a data entry library which handles multiple representations of fields, a variety of editing control options, and support for multiple simultaneous data entry windows. Data entry screens could



be defined using structures similar to those which define the database schema. This combination, while ambitious, would result in a near commercial quality Database Management Environment.

## APPENDIX B

### LEXNET

#### B.1 Function Overview

##### **addDictWord**

This function accepts a single argument — a pointer to a character string. The string is duplicated, the duplicate is converted to lower case and the dictionary is searched. If the word is found in the dictionary, then its *wordNumber* is returned; otherwise a new `dictionaryWord` record is inserted into the database. This new word will have no word forms and no rules associated with it. The *wordNumber* of the newly added `dictionaryWord` is returned. The memory passed to this function is not affected in any way and may be deallocated by the calling function.

##### **getDictWordNumber**

This function is identical to **addDictWord** except that a new `dictionaryWord` is *not* inserted if the word does not already exist in the dictionary. In such cases, this function returns zero.

##### **getDictWordName**

This function is the complement of **getDictWordNumber**. Given a *wordNumber*, this function finds a word with that number, duplicates its root text form and returns a pointer to the newly allocated memory. The calling function is responsible for deallocating this memory.

**getTagWithName** This function returns the *tagNumber* associated with a particular *tagName*. Its only argument is a pointer to the character string containing the name to find. Zero is returned if the tag name is not in the dictionary.

**getDictTagName**

This function is the complement of **getTagWithName**. Given a *tagNumber*, this function finds a tag with that number, duplicates its name and returns a pointer to the newly allocated memory. The calling function is responsible for deallocating this memory.

The remaining functions print the dictionary in human readable form. **PrintDictionary** will print every word in the dictionary in alphabetical order. It calls **printADictWord** which calls **printAWordForm** and **printARule**. **PrintTags** is also provided, but not particularly useful.

## B.2 Detailed Memory Representation of a Sentence

**MEMSENTENCE**

**lexemes** A pointer to an array of MEMLEXEMEs.

**currentLexeme** A pointer to the element of the *lexemes* array which is currently being disambiguated.

**numLexemes** The number of elements in the *lexemes* array.

**numAmbiguousLexemes** the number of lexemes in the sentence which still need to be disambiguated.

**MEMLEXEME**

**isAmbiguous** Set to one of the manifest constants **YES** or **NO** to signify whether or not there are still rules to be tried for this lexeme.

**wordNumber** The *dictWordNumber* from the dictionary database.

**specialTags** A pointer to an array of tag numbers as described above.

**wordForms** A pointer to an array of MEMWORDFORMs.

**rules** A pointer to an array of MEMRULEs.

**currentRule** A pointer to the element of the *rules* array which is currently being fired (or which was deferred during the last attempt to fire).

**wordFormApplied** A pointer to the element of the *wordForms* array which has been identified by the rules as being the appropriate assignment for this lexeme.

**rawText** A pointer to a character string containing the lexeme as it was read from the input file.

- rootText** A pointer to a character string containing the lexeme after morphological transformation and reduction to lower case.
- numSpecialTags** The number of elements in the *specialTags* array.
- numWordForms** The number of elements in the *wordForms* array.
- numRules** The number of elements in the *rules* array.
- flag** A flag used during the disambiguation process to detect an unresolvable deadlock situation in which the disambiguation of one lexeme depends upon the disambiguation of a second which, in turn depends upon the disambiguation of the first. The deadlock breaking mechanism is described in detail in Disambiguation.

## MEMWORDFORM

- wordFormNumber** The *wordFormNumber* from the dictionary database.
- wordFormPosition** The position of this word form among the word forms associated with this lexeme. When rules apply a particular word form to a lexeme, this number is used to identify the specific word form. As noted earlier, these position numbers are sequential, but not necessarily consecutive.
- probability** The probability with which this word occurred in the original test corpus for the GI — not used by LexNet.
- tagsApplied** A pointer to an array of tag numbers which are applied to a lexeme when this word form is identified.
- numTagsApplied** The number of elements in the *tagApplied* array.

## MEMRULE

- ruleNumber** The *ruleNumber* from the dictionary database.
- rulePosition** The position of this rule among the rules associated with this lexeme. When rules perform a SKIP operation, this number identified the rule that should be tested next. As noted earlier, these position numbers are sequential, but not necessarily consecutive.
- searchStartOrigin and searchEndOrigin** One of the manifest constants K, B, E, or C as described in the previous section.
- searchStartOffset and searchEndOffset** The number of lexemes (either positive or negative) to move from the origin. An origin/offset combination exactly identifies the first and last lexeme to be examined by this rule.
- ruleType** One of the rule types TOR, TORK, TAND, TANDK, TADJ, TSAME, TSAMEK, TSAMEM, TSAMEMK, WOR, WORK, WAND, WANDK, WADJ, SUPV, or GOTO as described in the previous section.

**trueAction** One of the action types APPLY, DELID, SKIP, or NEXT as described in the previous section.

**trueArgument** The *position* of the word form to be applied for the APPLY and DELID action types, the *position* of the next rule to test for the SKIP action type, or zero for the NEXT action type.

**falseAction** One of the action types APPLY, DELID, SKIP, NEXT, or TRANS as described in the previous section.

**falseArgument** The *position* of the word form to be applied for the APPLY and DELID action types, the *position* of the next rule to test for the SKIP action type, zero for the NEXT action type, or a *dictWordNumber* for the TRANS action type.

**searchTerms** A pointer to an array of items which this test searches for — either tag numbers or word numbers, depending upon the rule type. This pointer is not used for rules of type SUPV or GOTO.

**numSearchTerms** The number of elements in the *searchTerms* array.

## APPENDIX C

## THE LEXNET KNOWLEDGE BASE SCHEMA

```

#include <lnetdict.h>

/* ****dictionary words**** */

DBMSFIELD dictWordFields[]=
{
    {"DictWordNumber",DBMS_ULONG,printLongIntField,
     longIntCompare,0,0,0},
    {"KeyText",DBMS_VARCHAR,printCharField, charCompare,
     VAR_LEN,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

char *wordNumberKeyFields[]={"DictWordNumber",NULL};
char *wordTextKeyFields[]={"KeyText",NULL};

DBMSKEY dictWordKeys[]=
{
    {"DictWordNumberKey",wordNumberKeyFields,NO,0,0,NULL,
     NULL,NULL},
    {"WordTextKey",wordTextKeyFields,NO,0,0,NULL,NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****dictionary tags**** */

DBMSFIELD dictTagFields[]=
{
    {"DictTagNumber",DBMS_ULONG,printLongIntField,
     longIntCompare,0,0,0},
    {"IsMarker",DBMS_BYTE,printByteField, byteCompare,
     0,0,0},
    {"TagName",DBMS_VARCHAR,printCharField, charCompare,
     VAR_LEN,0,0},
    {"Description",DBMS_VARCHAR,printCharField, charCompare,
     VAR_LEN,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

```

```

char *tagNumberKeyFields[]={ "DictTagNumber",NULL};
char *tagNameKeyFields[]={ "TagName",NULL};

DBMSKEY dictTagKeys []=
{
    {"TagNumberKey",tagNumberKeyFields,NO,0,0,NULL,NULL,NULL},
    {"TagNameKey",tagNameKeyFields,NO,0,0,NULL,NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****word forms**** */

DBMSFIELD wordFormFields []=
{
    {"WordFormNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"WordFormPosition",DBMS_BYTE,printByteField, byteCompare,
        0,0,0},
    {"WordFormWordNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"WordFormPartOfSpeech",DBMS_BYTE,printByteField,
        byteCompare,0,0,0},
    {"Probability",DBMS_BYTE,printByteField, byteCompare,
        0,0,0},
    {"WordFormGloss",DBMS_VARCHAR,printCharField, charCompare,
        VAR_LEN,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

char *wordFormKeyFields[]={ "WordFormNumber",NULL};
char *wordFormWordKeyFields[]={ "WordFormWordNumber",
    "WordFormPosition",NULL};

DBMSKEY wordFormKeys []=
{
    {"WordFormKey",wordFormKeyFields,YES,0,0,NULL,NULL,NULL},
    {"WordFormWordKey",wordFormWordKeyFields,NO,0,0,NULL,
        NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****tags applied**** */

DBMSFIELD tagsApplyFields []=
{
    {"WordFormNumber",DBMS_ULONG,printLongIntField,

```

```

        longIntCompare,0,0,0},
{"TagNumber",DBMS_ULONG,printLongIntField, longIntCompare,
    0,0,0},
{NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

char *wordFormAndTagFields[]={"WordFormNumber","TagNumber",NULL};
char *formKeyFields[]={"WordFormNumber",NULL};
char *tagKeyFields[]={"TagNumber",NULL};

DBMSKEY tagsApplyKeys[]=
{
    {"WordFormAndTagKey",wordFormAndTagFields,NO,0,0,NULL,
        NULL,NULL},
    {"WordFormKey",formKeyFields,YES,0,0,NULL,NULL,NULL},
    {"TagKey",tagKeyFields,YES,0,0,NULL,NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****rules **** */

DBMSFIELD ruleFields[]=
{
    {"RuleNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"WordNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"RulePosition",DBMS_BYTE,printByteField, byteCompare,
        0,0,0},
    {"SearchStartOrigin",DBMS_SHORT,printShortIntField,
        shortIntCompare, 0,0,0},
    {"SearchEndOrigin",DBMS_SHORT,printShortIntField,
        shortIntCompare, 0,0,0},
    {"SearchStartOffset",DBMS_BYTE,printByteField,byteCompare,
        0,0,0},
    {"SearchEndOffset",DBMS_BYTE,printByteField,byteCompare,
        0,0,0},
    {"RuleType",DBMS_BYTE,printByteField,byteCompare, 0,0,0},
    {"TrueAction",DBMS_BYTE,printByteField,byteCompare,0,0,0},
    {"TrueArgument",DBMS_SHORT,printShortIntField,
        shortIntCompare,0,0,0},
    {"FalseAction",DBMS_BYTE,printByteField,byteCompare,0,0,0},
    {"FalseArgument",DBMS_SHORT,printShortIntField,
        shortIntCompare,0,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}
}

```



```

};

char *ruleKeyFields[]={"RuleNumber",NULL};
char *ruleWordKeyFields[]={"WordNumber","RulePosition",NULL};

DBMSKEY ruleKeys[]=
{
    {"RuleKey",ruleKeyFields,NO,0,0,NULL,NULL,NULL},
    {"RuleWordKey",ruleWordKeyFields,NO,0,0,NULL,NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****search tags**** */

DBMSFIELD searchTagFields[]=
{
    {"SearchTagRuleNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"SearchTagTagNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"SearchTagPosition",DBMS_BYTE,printByteField,
        byteCompare,0,0,0},
    {NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

char *searchTagKeyFields[]={"SearchTagRuleNumber",
    "SearchTagPosition",NULL};
char *searchTagTagKeyFields[]={"SearchTagTagNumber",NULL};

DBMSKEY searchTagKeys[]=
{
    {"SearchTagKey",searchTagKeyFields,NO,0,0,NULL,NULL,NULL},
    {"SearchTagTagKey",searchTagTagKeyFields,YES,0,0,NULL,
        NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* ****search tags**** */

DBMSFIELD searchWordFields[]=
{
    {"SearchWordRuleNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},
    {"SearchWordWordNumber",DBMS_ULONG,printLongIntField,
        longIntCompare,0,0,0},

```

```

        {"SearchWordPosition",DBMS_BYTE,printByteField,
         byteCompare,0,0,0},
        {NULL,DBMS_NULL,NULL,NULL,0,0,0}
};

char *searchWordKeyFields[]={"SearchWordRuleNumber",
                             "SearchWordPosition", NULL};
char *searchWordWordKeyFields[]={"SearchWordWordNumber",NULL};

DBMSKEY searchWordKeys[]=
{
    {"SearchWordKey",searchWordKeyFields,NO,0,0,NULL,NULL,NULL},
    {"SearchWordWordKey",searchWordWordKeyFields,YES,0,0,NULL,
     NULL,NULL},
    {NULL,NULL,0,0,0,NULL,NULL,NULL}
};

/* **** define all the tables**** */

DBMSTABLE lNetDictTables[]=
{
    {"DictionaryWords",dictWordFields,dictWordKeys, "dictword",
     0,0,0},
    {"DictionaryTags",dictTagFields,dictTagKeys, "dicttag",
     0,0,0},
    {"WordForms",wordFormFields,wordFormKeys, "wordform",
     0,0,0},
    {"SearchTags",searchTagFields,searchTagKeys, "srctag",
     0,0,0},
    {"SearchWords",searchWordFields,searchWordKeys, "srchword",
     0,0,0},
    {"Rules",ruleFields,ruleKeys, "rule",0,0,0},
    {"TagsApplied",tagsApplyFields,tagsApplyKeys, "tagsaply",
     0,0,0},
    {NULL,NULL,NULL,NULL,0,0,0}
};

/* **** finally, define the schema **** */

DBMSSCHEMA lNetDictSchema={lNetDictTables,NULL,NULL,NULL};

```

## APPENDIX D

### HARVARD IV'S NEW LEASE ON LIFE

The General Inquirer is an automated content analysis tool developed in the mid 1960s. It takes as input a text corpus to be analyzed and a "dictionary" of content analysis rules and categories. It processes the text corpus by attempting to match the patterns defined by the rules with the patterns observed in the text. When a match is found the system assigns one or more category tags to the text. The current distribution of the General Inquirer (the General Inquirer III) is not significantly different from the original application developed nearly thirty years ago.

The General Inquirer III distribution contains several computer programs written in PL/I (with one routine in optimized binary code) and three disambiguation dictionaries in text (EBCDIC) format. The distribution is available, for a modest fee, on magnetic tape from the Center for Surveys, Research and Methodology (ZUMA), Mannheim, FRG. It is intended for research purposes only. Specifically, the computer programs [54] included in the distribution are:

**TEXTREAD** Prepares the raw text for input to TAGGER.

**TAGGER** Performs the disambiguation.

**RETRIEVE** Performs simple queries on TAGGER output.

**CONNECT** Transforms TAGGER output into any of three formats:

1. Text with embedded sense numbers.
2. Frequency counts, by sentence, for specified tags.
3. Binary strings, one per sentence, in which each character indicates the presence or absence of a tag.

**TALLY** Produces tag frequency counts on a per document basis.

**PEEL** Reformats TAGGER format dictionary entries in human readable format.

**PARSER** Reformats human readable dictionary entries in TAGGER format.

**DICTMERG** Combines two TAGGER format dictionaries.

**DOCUMENT** Lists an entire dictionary in one of several formats.

These programs are designed to run under either the IBM MVS or VM operating systems, and sample Job Control Language (JCL) is provided for each. The original PL/I program code was written during 1960's, when batch processing of punched cards was the normal input procedure. Every effort was made in the original programs to minimize memory usage and maximize computational performance. For these reasons, ZUMA makes the following statements in their introductory overview: "...the system should not be considered portable..." and "the system is not particularly user-friendly" ([68], p. 3). Finally, users are cautioned "...not to modify the software in any way..." ([68], p. 1).

The dictionaries included in the distribution are:

**LVD [38]** The Lasswell Value Dictionary is an outgrowth of the Namenwirth Political Dictionary which was a modification of early Harvard Dictionaries ([37], [53]). It contains tag assignments for more signs than the Harvard Dictionaries, but fewer disambiguation rules.

**HIV-3 [55]** The Harvard Fourth Dictionary in its third major release. It operationalizes the category scheme described in Stone, Dunphy, Smith and Ogilvie (1966), but includes the disambiguation rules added during the early 1970's.

**HIV-4 [55]** The most recent release of the Harvard Dictionary has had a number of entries and categories modified to increase the reliability and validity of the HIV-3 Dictionary.

The three dictionaries are generally incompatible since the LVD and HIV-3 do not contain the most recent set of changes to the disambiguation rules.

Early work with the General Inquirer III distribution motivated the author to rewrite the application programs and redesign the distributed dictionaries to take advantage of more modern computer application design methods. Specifically, the applications were rewritten in the 'C' programming language, and the data storage was redesigned into a relational database. These two changes should significantly improve the portability and flexibility of the dictionaries and accompanying tools.

The intention of the General Inquirer application rewrite was to generate a *reasonably* faithful implementation of the logical structure embodied in the General Inquirer. The resulting application (LexNet) is not a translation of the PL/I code (the author does not know the PL/I programming language), but performs as the GI design documents say that it *should* perform — a reverse engineered GI, if you will. LexNet is said to be a *reasonably* faithful rewrite because there are many minor design issues that are not explicitly described in the GI documentation (in general it is quite complete and well written, however). This document will not describe those aspects which are well documented by Züll, Weber and Mohler [68] or Kelly and Stone [25], but will focus on the differences between those design documents and the LexNet implementation and the unique features of LexNet.

An early design decision was to represent the disambiguation dictionaries in a relational database format. Several implementations were evaluated including, but not limited to: the NeXT IndexingKit [39], Clips [16], MetalBase [24], CDATE [52] and several commercial database management systems. Each of these was deemed unsuitable for one or more of the following reasons: cost, support for variable length data fields, performance, or portability. The final decision was to write a portable database management system that supports variable length data fields. Appendix A contains a detailed description of the DBMS(3) function library. Appendix F (digital) contains complete source code to that library.

#### D.1 The LexNet Knowledge Base

The GI's TAGGER program is actually an interpreter which applies a knowledge base (called a content analysis dictionary) to a source document. The knowledge base is a collection of short computer programs written in a programming language called *CONTEXT*, developed by G. Heil [25]. The reader is directed to Kelly and Stone [25] and Züll, Weber and Mohler [68] for a complete description of the *CONTEXT* language. What follows is a brief overview that should be sufficient to understand the operation, if not the logic behind, programs written in the *CONTEXT* language.

Each program consists of three parts: the word itself, a list of the different senses (or word forms) that the word may assume, and a set of heuristic rules which can be applied to the sentence containing this particular word in order to determine which of the word forms is most appropriate. The definition of word forms is preceded by the delimiter 'TAGS:' and the disambiguation heuristics by 'RULES:'. A CONTEXT program ends with the delimiter 'END;'. Figure 20 shows the CONTEXT program for the word *love* as stored for use by the GI and will be used as an example in this section. Figure 21 shows the same program as stored by LexNet. The similarity in appearance is intentional, but superficial.

```

M LOVE=
TAGS: (1%48)SUPV.INTREL.AFFIL.PSTV.PSV.
      VERB TO HAVE AFFECTION OR STRONG LIKING FOR,
      ESPECIALLY FOR ONE OF OPPOSITE SEX
(2%30)NOUN.EMOT.AFFIL.PLEASUR.PSTV.PSV.NOUN
      THE AFFECTION
(3%12)MODIF.AFFIL.EMOT.PSTV.PSV.PLEASUR.
      IDIOM-ADJ 'IN LOVE'--ENAMORED
(4% 4)MODIF.EMOT.AFFIL.PLEASUR.PSTV.PSV.
      ADJ 'LOVING'--FEELING OR SHOWING LOVE
(5% 1)SUPV.INTREL.AFFIL.STRNG.ACTV.
      IDIOM-VERB 'MAKE LOVE'
(6% 1)NOUN.AFFIL.INTREL.
      IDIOM-NOUN 'LOVE LIFE'--ONE'S SEXUAL OR ROMANTIC
      RELATIONS
(7% 5)HANDELS.
      IDIOM-VERB 'FALL IN LOVE'--HANDLED BY "FALL" AND
      "FELL"
RULES: ( 8)TOR(K+0,K+0,APLY(1),,ED.)
      ( 9)TOR(K+0,K+0,APLY(4),,ING.)
      (10)TSAMEM(K-1,K-1,APLY(2),,DET.PRE.)
      (11)TOR(K-1,K-1,APLY(2),,EVAL.DIM.)
      (12)TOR(K+1,K+1,APLY(1),,DET.PRON.HU.)
      (13)TOR(K-1,K-1,,18,DET.PREP.)
      (14)WOR(K+1,K+1,DELID(6),,LIFE.)
      (15)WOR(K-1,K-1,DELID(3),,IN.)
      (16)WOR(K-1,K-1,DELID(5),APLY(2),MAKE.MADE.)
      (18)TOR(K-1,K-1,APLY(1),APLY(2),DEF1.MOD.TO.LY.HU.)END;

```

Fig. 20. The Harvard IV-4 CONTEXT program for the word *love*.

TAGS:  
 (1%48) verb to have affection or strong liking for,  
           especially for one of opposite sex  
           pstv.psv.intrel.affil.supv.  
 (2%30) noun the affection  
           pleasur.pstv.psv.emot.affil.noun.  
 (3%12) idiom-adj 'in love'--enamored  
           pstv.psv.pleasur.affil.emot.modif.  
 (4%4) adj 'loving'--feeling or showing love  
           pleasur.pstv.psv.emot.affil.modif.  
 (5%1) idiom-verb 'make love'  
           strng.actv.intrel.affil.supv.  
 (6%1) idiom-noun 'love life'--one's sexual or romantic  
           relations  
           affil.intrel.noun.  
 (7%5) idiom-verb 'fall in love'--handled by "fall" and  
           "fell"  
           handels.

RULES:  
 (8) TOR(K+0,K+0,APPLY(1),NEXT(0),ed.)  
 (9) TOR(K+0,K+0,APPLY(4),NEXT(0),ing.)  
 (10) TSAMEM(K-1,K-1,APPLY(2),NEXT(0),det.pre.)  
 (11) TOR(K-1,K-1,APPLY(2),NEXT(0),eval.dim.)  
 (12) TOR(K+1,K+1,APPLY(1),NEXT(0),det.pron.hu.)  
 (13) TOR(K-1,K-1,NEXT(0),SKIP(18),det.prep.)  
 (14) WOR(K+1,K+1,DELID(6),NEXT(0),life.)  
 (15) WOR(K-1,K-1,DELID(3),NEXT(0),in.)  
 (16) WOR(K-1,K-1,DELID(5),APPLY(2),make.made.)  
 (18) TOR(K-1,K-1,APPLY(1),APPLY(2),def1.mod.to.ly.hu.)

Fig. 21. The LexNet CONTEXT program for the word *love*.

Each word is represented by its root text (explained in Morphological Transformations) and a single letter (H, M, L, K) which indicates the frequency with which this word appeared in the original sample used to build the Harvard dictionaries. This information is used by the GI to optimize memory usage — it serves no other useful purpose and is not preserved in LexNet.

Each word form consists of a set of tags and an optional text gloss describing the word form (called a 'comment' by the authors of the GI). Word forms in the Harvard dictionaries also often contain a percentage representing how often that particular word form appeared in the original test corpus used to develop them. This information is preserved in, but not used by, LexNet. In Figure 20, the word '*love*' has seven

defined forms. Four are idioms and the remaining three are split across part of speech boundaries (verb, noun and adjective).

Finally, the rules section contains an ordered set of rules of 16 possible types. There are two special rule types (GOTO and SUPV) that will be described shortly. The remaining fourteen rule types fall into two broad categories: those that search the sentence for other words and those that search the sentence for category tags. Thus, each of these rules consists of a range of words in the sentence to examine, a list of terms (either words or tags) to search for, an action to perform if the rule succeeds and an action to perform if the rule fails.

#### D.1.1 Rule Types

**TOR** Any of the specified tags appearing in the search range.

**TORK** Same, except that the keyword (the word we are disambiguating) should be skipped if it is in the search range.

**TAND** All of the tags must appear in the range, in order.

**TANDK** Same except that the keyword may intervene.

**TADJ** All of the tags must appear in the range, on adjacent words.

**TSAME** All the tags must appear on the same word.

**TSAMEK** Same except the keyword may intervene.

**TSAMEM** The first, but none of the other tags must appear on a single word in the search range.

**TSAMEMK\*** Same except the keyword may intervene.

**WOR** Any of the specified words appearing in the search range.

**WORK\*** Same except that the keyword may intervene.

**WAND** All of the specified words must appear in the search range, in order.

**WANDK\*** Same except that the keyword may intervene.

**WADJ** All of the specified words must appear in the search range, adjacent to one another.



In the GI, each rule type also implies a specific order in which the search range should be tested — from beginning to end or from the keyword toward each end — as a means of optimizing performance. This distinction is not maintained in LexNet, making the rule types marked with an asterisk (\*) identical to their keyword inclusive counterparts. All of the rule types are preserved in LexNet for purposes of debugging and comparison.

The GOTO rule type is used primarily for morphological transformations (e.g., *kept* as the past tense of *keep*). It indicates simply that a particular word form should be applied and processing should continue with the specified (replacement) word.

The SUPV rule type is actually a macro for nine tests of the other types. These nine tests are frequently employed to differentiate between noun and verb forms of words. The SUPV rule type is described in detail by Züll, Weber and Mohler [68].

Figure 20 shows that ten rules are needed to disambiguate the word *love*; six of type TOR, three of type WOR and one of type TSAMEM. Note that neither the rules nor the word forms are necessarily numbered consecutively — although they are (must be) numbered sequentially.

### D.1.2 Search Ranges

The range of words in the sentence that this rule should examine is given in four parts: two origins plus two offsets. An origin may be one of four points in the sentence: the keyword (K), the beginning of the sentence (B), the end of the sentence (E), or the last word which matched the previous test (C). An offset is simply a number of words ('lexemes' is more accurate, because punctuation marks are treated as separate units), either positive or negative to move from the offset.

Range definitions with negative offsets from the beginning or positive offsets from the end of a sentence are invalid constructs. This creates a potential point of confusion in that the category tag E which signifies the end of a sentence is assigned to the last true word in a sentence. This word may be (in fact, usually is) followed by various punctuation. The same is true for the beginning of the sentence although

punctuation preceding the first word of a sentence is the exception rather than the rule.

The K origin is by far the most common, and C the least common. The word *are* demonstrates the use of the E and C origins. Figure 22 shows the Harvard IV-4 CONTEXT program for the word *are*. Rule number six determines whether or not the sentence ends with a question mark. Rules seven and eight work together, first determining if the sequence of words is *are being* and then, if that is true, whether either of the two words after *being* is in the past tense.

```

H ARE=
TAGS:  (1%70)SUPV.VERB.BE.
        VERB USED AS COPULA CONNECTING SUBJECT TO PREDICATE
        ADJECTIVE OR NOMINATIVE, OR TO CONNOTE EXISTENCE
        (ESPECIALLY WITH 'THERE . . .')
      (2% 8)SUPV.VERB.BE.
        VERB USED AS AUXILIARY TO FORM SIMPLE PROGRESSIVE
      (3%22)SUPV.VERB.BE.PASSIVE.
        VERB USED AS AUXILIARY TO FORM PASSIVE
      (4% 1)SUPV.VERB.BE.PASSIVE.
        VERB USED AS AUXILIARY TO FORM PASSIVE PROGRESSIVE
RULES: ( 5)TOR(K+1,K+1,, 7,DET.PREP.DEG.)
      ( 6)TOR(E-0,E-0,,APLY(1),Q.)
      ( 7)WOR(K+1,K+1,, 9,BEING.)
      ( 8)TOR(C+1,C+2,APLY(4),,ED.)
      ( 9)TSAMEM(K+1,K+2,,13,ED.BE.)
     (10)TOR(C+0,C+0,,APLY(3),EMOT.EVAL.)
     (11)WOR(C+1,C+1,APLY(3),APLY(1),BY.)
     (13)TOR(K+1,K+2,APLY(2),APLY(1),ING.)END;

```

Fig. 22. The Harvard IV-4 CONTEXT program for the word *are*.

### D.1.3 Actions

All rule types, except the GOTO type, have two actions that might be performed: one to perform if the test succeeds and another to perform if it fails (rules may also be deferred, which will be discussed later). There are four kinds of actions:

**APPLY**

Applies the specified word form to the word and stops further processing of the keyword (it is completely disambiguated).

**DELID**

Applies the specified word form to the word, stops further processing of the keyword and treats all words from the keyword to the word that matched the test as a single lexeme. GI keeps the words separate but removes all tags from the other words in the idiom; LexNet physically joins them, forming a single lexeme. This difference increases the readability of LexNet output and seldom, if ever, effects the disambiguation process.

**SKIP**

Continue processing with the specified rule. GI stores this simply as a rule number while LexNet explicitly identifies this action type.

**NEXT**

Do nothing, continue with the next rule. This is a default condition (no action) in GI; it is explicitly identified in LexNet. The NEXT action can be thought of as a special case of the SKIP action.

The GOTO rule type always performs exactly two actions: a word form is applied, and processing is transferred to another dictionary entry. in LexNet notation, the word form to apply is stored in the *trueAction* field and the special action type **TRANS** appears in the *falseAction* field. The argument to a **TRANS** action is the *dictWordNumber* of the dictionary entry that replaces the current lexeme. Thus, the **TRANS** action only appears in the *falseAction* field of GOTO rules. The GI allows the word form identifier to be zero, LexNet continues this questionable practice even though every dictionaryWord entry in LexNet has at least one word form (which may have zero or more tags associated with it).

Rule number eight in figure 20, then, can be read as “Search the sentence for any matching tag (TOR), starting with the keyword (K+0) and ending with the keyword

(K+0), if found then apply word form number 1 and stop processing else continue with the next rule, search for the tag(s) ED.”

The knowledge base (collection of CONTEXT programs) is stored in a text file for use by the GI. LexNet stores the knowledge base in a relational database using the DBMS(3) library. Figure 23 shows the design for the relational database, as an Entity-Relationship Diagram (ERD).

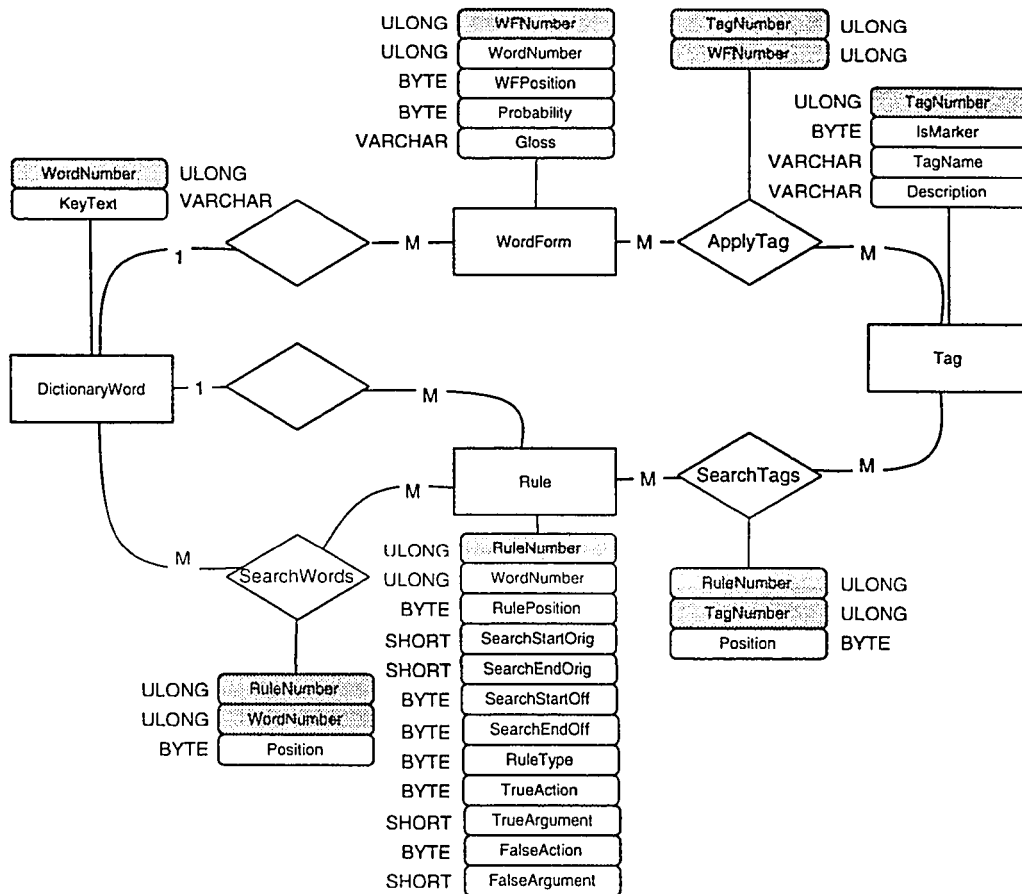


Fig. 23. The Entity-Relationship Diagram for the LexNet knowledge base.

Conventions for drawing ERDs vary considerably. Specific examples will clarify the notation used here. There is a Many-to-Many relationship between Rules and DictionaryWords, which is to say that each rule may search for several different words and each word may be the target of many different rule searches. Conversely, there is a One-to-Many relationship between Rules and DictionaryWords representing the fact that each rule is used to disambiguate only one word, but that it may require many rules to completely disambiguate a particular word. Titles have not been given to relationships (represented as diamonds) that will not become part of the database schema. Primary keys are shown in darkened attribute boxes.

The major functions used to access LexNet dictionaries are described in Appendix B along with a description of the dictionary conversion `lexnetconv` process. Appendix C shows the complete schema for a LexNet Dictionary. It is a direct translation of the design shown in Figure 23 to the schema format required by DBMS(3). Complete source code to the LexNet system is in Appendix G (digital).

#### D.1.4 Utilities

Two utility programs are provided with the dictionary library: `showword` and `xref`. The `showword` program accepts as its sole argument a word to be looked up in the dictionary. If that word exists in the dictionary, then it is printed in a format identical to that shown in Figure 21. The `xref` program produces a cross reference for either a word or a category tag. Its specification is:

```
xref [ -tw ] [ tag|word ]
```

If a word is being cross referenced, then the output of `xref` will show all the other word entries in the dictionary which search for the existence of the specified word. For example, the command

```
xref -w about
```

yields the following output

Rules:	
Word: time	Rule: 47
Word: mill	Rule: 5
Word: mad	Rule: 9
Word: feel	Rule: 19
Word: bring	Rule: 14
Word: business	Rule: 11
Word: come	Rule: 14
Word: hear	Rule: 14
Word: bother	Rule: 9
Word: judgment	Rule: 4
Word: argument	Rule: 4

The output for a tag cross reference is similar but includes a listing of all word forms that apply the specified tag — this output is usually quite lengthy.

## D.2 The Disambiguator

The application `lnettrans` serves as the inference engine in the disambiguation process. Each sentence to be disambiguated passes through several more or less distinct processes: lexical analysis, morphological transformation, and finally through a multi-pass disambiguation engine.

### D.2.1 User's Guide

Disambiguation of text files is greatly simplified using LexNet (compared to using the GI). The user needs only to decide which outputs are desired from the disambiguation process. The options are:

- x Output the “raw” sentence, after lexical analysis. Each lexeme is separated by a space and unprocessable characters are removed.
- s Output the sentence (lexeme by lexeme) after morphological transformation but before disambiguation, showing all special tags assigned, possible word forms and rules that might be tested.
- i Output a string of binary digits representing the presence or absence of each category tag (in tagNumber order).
- f Output the “root” sentence, after disambiguation with word form assignments attached to each lexeme.

a Output the sentence (lexeme by lexeme) showing all tags (special and word form) assigned to each lexeme.

The following forms of output are also available, but are useful primarily for debugging. These options generally create a great deal of output.

b Output a description of the current sentence whenever the **breakLock** function is called. The **breakLock** function is described in the Disambiguation section.

t Output a description of the current lexeme whenever a test is performed. This option generates voluminous output!

r Output the result of each rule test.

Specification of the **i**, **b**, **t** and/or **r** options automatically implies specification of the **x** option.

The chosen options are specified on the command line followed by a series of file names to be processed. If no file names are provided, keyboard input is processed (note that [Ctrl][d] generates an end-of-file character on most keyboards). All output is displayed to the console, although it can be redirected to a file. The technical specification for the command line is:

```
lnettrans [ -xsifabtr ] [file] ...
```

For example, to process the files `bra01.text` and `bra02.text` in the current directory, producing both a series of binary digits representing the presence or absence of each category tag and a listing of the lexemes with their tag assignments, and storing the output in a file named `listing1.output`, the following command would be used:

```
lnettrans -ia bra01.text bra02.text > listing1.output
```

This yields output similar to that shown in Figure 24 (minor reformatting was necessary):

## D.2.2 Lexical Analysis

With one exception, breaking English sentences down into individual lexemes is a fairly straightforward process. Words are contiguous sequences of the letters A–Z, upper or lower case. Words containing hyphens are considered to be intentionally hyphenated (the hyphen is retained) unless the hyphen is followed by a carriage return. In that case, the word is treated as if it had been split — the hyphen and the carriage return are removed and the two parts of the word joined. Whitespace (tabs, spaces and carriage returns) are ignored, except that they separate words. Punctuation (semicolon, colon, comma, parenthesis, and quotation marks) are treated as non-terminating punctuation. Periods, exclamation points and question marks are considered terminal punctuation. Numbers may have a leading negative and may contain a decimal point. Any number preceded by a dollar sign is considered to be a reference to money. Money values greater than \$999,999,999 will be identified correctly, but will be truncated.

The lone exception to this straightforward lexical analysis is the treatment of abbreviations. Exactly how abbreviations are handled by the GI is not described in the documentation. The method employed here seems to reasonably mimic the actual performance of the GI, and in some cases performs better than the GI.

A word followed immediately by a period may signify either the end of a sentence, or an abbreviation. To distinguish these two cases LexNet first looks the word up in the dictionary with the period still attached because some common abbreviations (like Mr., Mrs., etc. [,etc.]) are specifically defined there. If the lookup succeeds, then the word together with the trailing period are considered to be an abbreviation and LexNet continues reading the rest of the sentence. If the word before the period is exactly one character in length, then it is treated as an abbreviation. Otherwise, the period is removed and the word is again looked up in the dictionary. If the word





(*sans* period) is found in the dictionary, then the period is treated as a terminal punctuation. If the word is not found in the dictionary, but it is less than four characters in length, then it is treated as an abbreviation. Otherwise the period is treated as a terminal punctuation mark.

### D.2.3 Memory Representation

As each lexeme is identified in the input file, a 'C' data structure of type *MEMLEXEME* is created. The structure of lexemes in memory is similar but not identical to the storage arrangement in the dictionary database. Figure 25 shows the various 'C' data structures employed, and their relationships to one another.

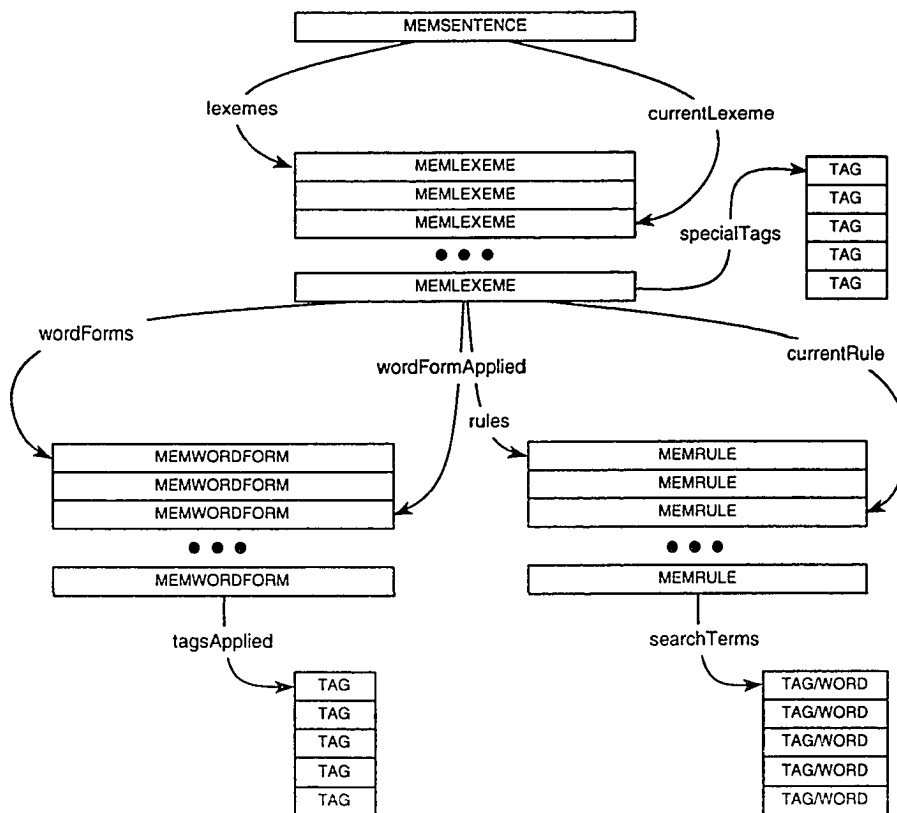


Fig. 25. Memory representation of a sentence.

At any given moment, there is one and only one MEMSENTENCE structure. Its most important elements are two pointers: one pointing to the array of MEMLEXEMEs which constitute the sentence, and the other pointing to a specific lexeme in that array — the lexeme which is currently being disambiguated.

Each MEMLEXEME contains five interesting pointers: (1) to an array of MEMWORDFORMs, (2) to the specific word form that this lexeme has been identified with after disambiguation, (3) to an array of MEMRULEs, (4) to the rule that is currently being tested, and (5) to an array of special tag numbers that have been applied to this lexeme, but which are not associated with a particular word form. For the most part, these special tags are markers; that is, the information they carry is not likely to be of interest to the researcher, but is useful during the disambiguation process to follow. Special tags are applied either by the morphological transformation process (described in the next section) or by a rule of type GOTO.

Each MEMWORDFORM also carries a pointer to an array of tag numbers. In this case they are the tags which should be applied to the lexeme if the rules determine that this word form is the correct one.

Finally, each MEMRULE carries a pointer to an array of items to search for. This array contains either tag numbers or word numbers, depending upon the rule type. This arrangement is quite different from the way that this same information is stored in the dictionary database. It is possible (however unlikely due to the disparity in the absolute number of tags in the dictionary compared with the number of words defined) for two rules to have identical search term arrays, but be referring to different data (because one rule searches for words and the other for tags). This arrangement simplifies the memory representation considerably, thereby making the application code more easily understood.

A detailed description of the in-memory representation of a sentence appears in Appendix B.

#### D.2.4 Dictionary Conversion

The `lnetconv` program converts dictionaries from the text file format required by the GI to the relational format used by LexNet. The program takes three parameters,

in order. First is the name of the file containing tags (called the “SPEC1” file in GI terminology). The second parameter is the name of the file containing the rule definitions (called the “SOURCE” file in GI terminology). And the final parameter is the name of the directory where the LexNet database should be stored — this directory must exist.

The `lnetconv` program first reads the file containing tags, reporting any duplicates that are encountered. The Harvard IV-4 dictionary contains five such duplicates: HUMAN & HU, HAVE & HAV, FEMALE & FEM, BODYPRT & BODY, and ANIMAL & ANI. Multiple names for the same tag category are not allowed in LexNet, so these duplicates must be resolved before processing can continue. For this dissertation, the duplicates were removed using a text editor with search and replace capabilities. All references to the tag HUMAN in the file containing the Harvard IV-4 rules were changed to refer to the tag HU, HAVE to HAV, etc.

Having removed all duplicate tags from the source files, processing by `lnetconv` continues with the “SOURCE” file. This phase of the conversion process is performed by a plain, if somewhat byzantine, lexical analyzer written in the programming language flex [57].

Unlike the text format version of the Harvard IV-4 dictionary, LexNet is a closed system. That is, all words that are referenced by the dictionary rules exist in the dictionary. As rule search terms are read into the database, if the search term is a word (as opposed to a tag) and the word is not in the dictionary, then it is added. This does not cause a problem if a true definition of the word is later encountered because the `addDictionaryWord` function described earlier simply returns the existing `dictionaryWord`'s number. If, however, a word is genuinely defined more than once in the text dictionary, then only the word forms and rules defined last are retained. This situation occurs only once with the Harvard IV-4 dictionary; with the word *awoke*. Fortunately, the second definition was the preferred one.

Similarly, LexNet does not allow the same tag to be applied to a particular word form more than once; this occurs sixteen times in the text version of the Harvard IV-4. `Lnetconv` prints a warning message when these are encountered, ignores the

duplicate, and continues processing. The order of tags applied by word forms is not stored, but the order of search terms is (for purposes of TSAMEM testing).

#### D.2.5 Morphological Transformation

As each lexeme is identified in the input file it is looked up in the dictionary and if that look up fails, then it is morphologically transformed until either its root form is found in the dictionary or no further transformations can be performed, in which case it is considered “leftover” or undefined. This suffix removal process is described in some detail by Kelly and Stone [25] and the reader is directed there for a description of the suffix removal process. In addition to suffix removal, LexNet performs several other simple disambiguations during this process.

The morphological transformation process, in addition to finding the root form of a word, applies certain category tags to the words transformed. The real work of the morphological transformations is performed by the function `findRoot`. Before attempting suffix removal, the following lexemes are identified: single letter words (which obviously can not have a suffix removed), punctuation marks, money, numbers, references to years (any integer between 1800 and 2000 is arbitrarily categorized as both a number and a year reference), and contractions.

The GI assigns the tag `EST` to all words ending with `ER` or `EST` despite the existence of a distinct `ER` suffix tag. This practice occasionally leads to incorrect disambiguation of other lexemes. Because of the high frequency with which the `ER` suffix occurs in the lexicon (more than 18,500 times in the Brown [12] corpus) and the large number of rules that test for this suffix (79 rules in the Harvard IV-4 [55]), LexNet distinguishes the two suffix tags. It is worth noting that LexNet and the GI will not always disagree on tests that search for the `ER` tag for two reasons: several of these tests also search for the `EST` tag, and there are a number of word forms which apply the `ER` tag independently of the suffix removal process (e.g., *better*).

The GI tags all words ending in `'S` with the tags `GEN`, `DET`, `'S`, `BE`, `VERB`, and `SUPV`. A quick search of the Brown corpus [12] turned up 5680 occurrences of the `'S` suffix; 472 of these were *it's*, *he's* or *she's*. An examination of the first 100 occurrences

of 'S in that corpus identified 97 occurrences of possession (GEN and DET) and 3 cases of contraction (BE, VERB, and SUPV). Of the three cases of contraction, two were *it's* and one was *he's*. If contractions were treated as separate lexemes, then a fairly simple set of rules could be written to correctly distinguish between these two forms.

While the algorithmic approach to morphological transformation is more conservative of storage space, it also has several drawbacks when compared with an explicit definition of each and every morphological form. First, there are several cases in which the algorithm simply does not work; hence the need for a GOTO rule type. Second, words that are not in the dictionary at all may get incorrectly tagged as having a suffix; this seems to happen most frequently with last names ending with *er*. Third and finally, the algorithmic approach is computationally more expensive than an explicit approach.

Three other operations are performed as words are read from the input file. First, any lexeme that has no rules associated with it is assigned its first word form. Second, any lexeme that has only one rule and that rule is of type *GOTO* will have that rule processed. Third, a heuristic is applied to alter the processing of the word *to*, or more precisely a word following the word *to*. This heuristic is described in more detail in Verification.

#### D.2.6 Disambiguation

Once an entire sentence has been loaded into memory, all root forms have been identified, and all trivial disambiguation has been completed, the primary disambiguation process begins. Tests involving words (e.g., rules of type WOR, WAND, etc.) can always be resolved completely. Rules which search for tags, however, may be deferred. That is, each tag test has three possible outcomes: the test succeeds with certainty, the test fails with certainty, or the outcome of the test depends upon how other words in the sentence (still to be processed) are disambiguated.

The process followed by the GI to resolve these deferred tests, called "Forward Tagging Logic," is described as follows by Kelly and Stone ([25], p. 99):

- 1) If tag occurs on at least 90% of senses of test word, then consider match made. If it does not appear on any of the senses, consider no match made. Otherwise flag test and defer till next pass.
- 2) On next pass, if word is still not disambiguated, keep flag up for further passes until:
  - a) forward word is resolved or
  - b) a pass is made in which none of the flagged words are resolved.
- 3) Call breaklock routine to resolve first flagged word in sentence as follows:
 

If the percentage of senses containing the tag is greater than 80% then consider a match made.

Else if the percentage is less than 20%, then consider no match to be made.

Else if there are more tests to be made after a negative outcome, consider no match made.

Else if there are more tests to be made after a positive outcome, consider a match made.

Else if we are in a SUPV routine and a positive match assigns sense 1, then consider a match made.

Else consider no match made.
- 4) After a breaklock is made, all other flagged tests are retried to see if any can be resolved. When another pass is made without further improvement, step three is used again.

LexNet uses a very similar, but not identical approach. Figure 26 shows the logic process used by LexNet. Processing begins with the first ambiguous lexeme in the sentence. The first untried rule of that lexeme is tried. If the test succeeds or fails with 100% certainty, then the appropriate action is taken (either a word form is applied and processing moves to the next ambiguous word or some other rule is tested). If, however, the result of the test can not be determined with certainty, then

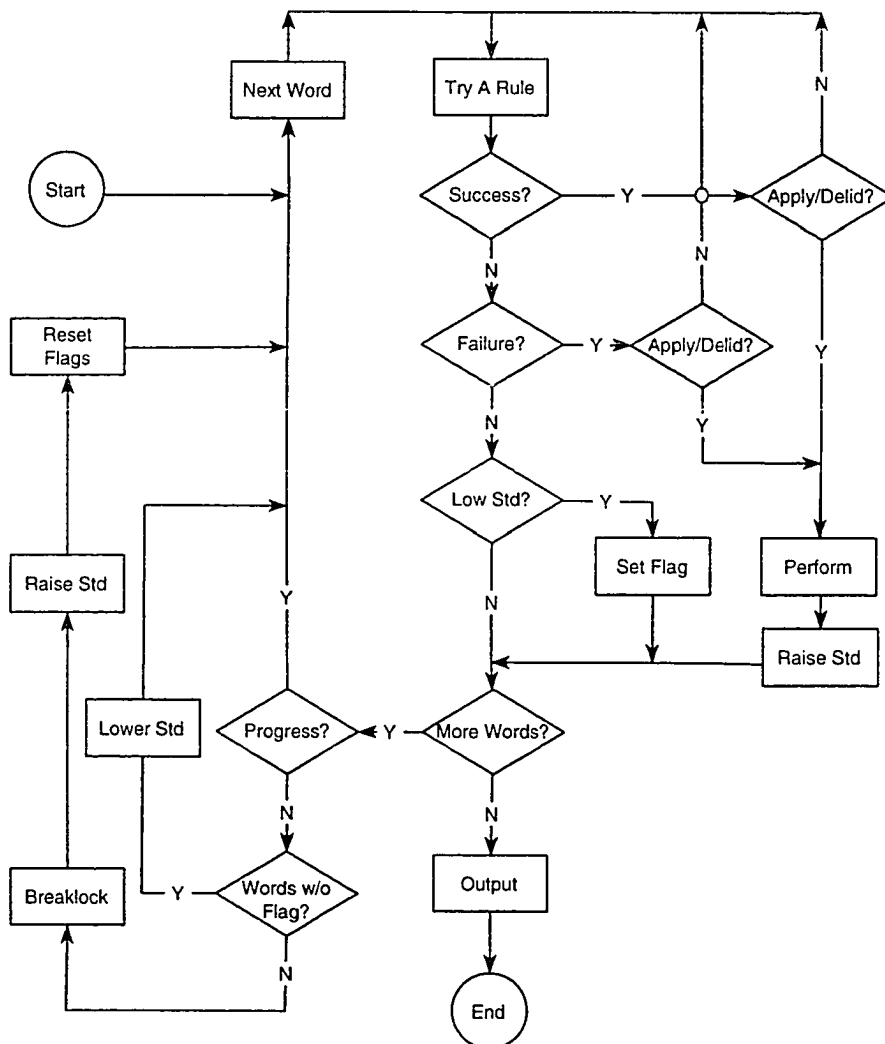


Fig. 26. The LexNet disambiguation process.

the rule is deferred and the next ambiguous lexeme is tried. This process continues until no rules on any of the remaining ambiguous lexemes can fire.

When this happens, LexNet “lowers its standards.” Processing continues as before, except that the test succeeds if at least 90% of the word forms for this word will apply the tag in question. The test fails if less than 10% apply the tag. LexNet’s



standards remain low until a rule fires; at which time the standards are raised again. When a lexeme has been unsuccessfully tested using the lower standard, a flag is set for that lexeme. This process continues until an unsuccessful attempt has been made to disambiguate every remaining ambiguous lexeme using the lower standard. If there are still ambiguous lexemes, then the **breakLock** function is called for the first ambiguous lexeme in the sentence.

The **breakLock** function follows the logic of step 3) described above for the GI. After the **breakLock** function has been used to disambiguate one lexeme, the standard is again raised and all flags are removed from ambiguous lexemes. Which is to say that processing continues, not with step 3) as described above, but with step 1) described above (as modified). In summary, LexNet generally follows the “Forward Tagging Logic” of the GI, except that it keeps the threshold for success high longer and it reinstates that high threshold more often. These minor modifications allow a greater number of disambiguations to occur with certainty.

#### D.2.6.1 The Tests

Each of the fifteen defined tests (the type GOTO is always unconditional) is defined individually. With the exception of SUPV, which has no arguments, each test is called with an integer starting position, an integer stopping position, a pointer to an array of search terms and the number of elements in that array. The start and stop positions must already have been tested to be sure that they are valid in the current sentence.

The word tests are all reasonably straightforward loop operations. The tag tests rely primarily on a function called **hasTag** which accepts a pointer to a MEMLEX-EME and a tag number. This function returns one of the manifest constants **YES**, **NO**, or **MAYBE** depending upon whether the tag can be detected with certainty on the specified lexeme. Its operation is as follows:

If the tag has been assigned to the lexeme as a special tag, then return **YES**.

If the lexeme has been disambiguated and the tag is assigned by the appropriate word form, then return **YES**.

If the lexeme has been disambiguated, then return **NO**.

For each rule that has not yet been tried for the specified lexeme, determine if the successful outcome would result in the tag being applied.

For each rule that has not yet been tried for the specified lexeme, determine if the failure outcome would result in the tag being applied.

If every remaining possible outcome would apply the tag, then return **YES**.

If none of the remaining outcomes would apply the tag, then return **NO**.

If the tag is one that is always (or almost always) applied during the morphological transformation step, then return **NO**.

Otherwise return **MAYBE**.

### D.3 Verification

As each of the test functions (**TOR**, **WAND**, etc.) was developed, it was tested with a variety of input patterns. Similarly, the rule firing mechanism was tested using well documented [25] sentence disambiguations like “Sally kept up with John on the hike,” “Jerry kept it up too long” and the more difficult “But rather – just like my relative, he grew rather upset.” Once satisfied that the results (not necessarily the procedure) from LexNet were identical to the GI, the “real world” verification could commence.

Figure 27 shows a trace of the disambiguation of the sentence *Sally kept up with John on the hike*. The trace was generated using the **Inettrans** program and specifying the following options: output the sentence after morphological transformation but before disambiguation (s), output information about each lexeme as tests are performed on it (t), output the sentence after disambiguation showing the tags assigned to each lexeme (a), and output the result of each rule test (r). Input was taken

directly from the keyboard console (resulting in the “Document # 0” identifier). The output was abbreviated to save space.

The first pass performs morphological transformations on the lexemes of the sentence until either the lexeme is found in the dictionary or it is determined that the lexeme cannot be morphologically transformed to yield a lexeme in the dictionary. All of the lexemes in this (simple) sentence except *kept* are disambiguated by the first pass. The lexemes *up*, *with*, *on*, and *the* are invariant in the dictionary; *hike* is not found in the dictionary; and *Sally* is incorrectly identified as a morphological form of the lexeme *sale*. This last assignment demonstrates why explicitly entering all morphological forms in the dictionary should be preferred over the use of a morphological transformation function. In a more complex sentence the incorrect assignment of economic tags to *Sally* might have resulted in additional incorrect assignments elsewhere in the sentence. Finally, the morphological transformation routine has correctly identified *kept* as the past tense of *keep*.

The disambiguater must determine which of the eight senses (forms) of the lexeme *keep* in the Harvard IV dictionary is most correct in this particular sentence. Actually, two of the senses are parts of idioms which are handled by the rules for other lexemes in those idioms which leaves six senses to be distinguished here. Twelve rules exist in the Harvard IV dictionary for *keep* and they are processed in order. The first rule (numbered 10 in the program output) examines the four lexemes following *keep* to determine if any of them are the lexeme *from*. If *from* were found in one of those positions then processing would continue with the second rule (11). *From* does not appear in this sentence however, so processing continues with the fourth rule (13). Rule 13 tests to determine if either of the two lexemes following *keep* has been assigned the morphological tag *ing*. This rule also fails to fire, causing processing to continue with rule 16. Rule 16 tests the two lexemes following *keep* to determine if either of them is the lexeme *up*. This rule fires successfully and therefore processing continues with rule 17. Rule 17 examines the lexeme immediately following *keep* to determine whether or not it is *up*, which it is. The action associated with success of rule 17 is **DELID** (the idiom action). The correct interpretation of **DELID** is to

===== Document # 0 Sentence # 1 =====

Raw: Sally      Root: sale      (1) Ambiguous: NO

Tags:

(1%0) econ.econ\*.means.noun.

Rules:

Special Tags:ly b

Word Form Tags: econ econ\* means noun

Raw: kept      Root: keep      (0) Ambiguous: YES

Tags:

(1%79) supv.actv.persist.

(2%4) supv.actv.hostile.intrel.power.strng.

(3%9) supv.actv.try.

(4%2) supv.complt.strng.

(5%0) supv.handels.actv.percv.strng.

(6%1) means.noun.

(7%1) handels.

(8%0) handels.

Rules:

(10) WOR(K+1,K+4,NEXT(0),SKIP(13),from.)

(11) TOR(C+1,C+1,APPLY(2),NEXT(0),ing.)

(12) TOR(K+1,K+1,APPLY(2),NEXT(0),det.pron.hu.)

(13) TOR(K+1,K+2,NEXT(0),SKIP(16),ing.)

(14) WOR(K+1,K+1,DELID(1),APPLY(1),on.)

Fig. 27. Trace of a simple disambiguation

- (16) WOR(K+1,K+2,NEXT(0),SKIP(19),up.)
- (17) WOR(K+1,K+1,DELID(3),NEXT(0),up.)
- (18) WAND(K+1,K+2,DELID(3),NEXT(0),it.up.)
- (19) TOR(K+0,K+0,NEXT(0),SKIP(22),ing.)
- (20) WANDK(K-1,K+1,APPLY(6),NEXT(0),in.with.)
- (21) TOR(K-1,K-1,APPLY(5),NEXT(0),det.)
- (22) WOR(K+1,K+3,APPLY(4),APPLY(1),promise.appointment.word.engagement.)

Trying rules for keep

Trying:keep Rule:10 Current: 1 Start: 2 Stop: 5 WOR FAILURE

Trying:keep Rule:13 Current: 1 Start: 2 Stop: 3 TOR FAILURE

Trying:keep Rule:16 Current: 1 Start: 2 Stop: 3 WOR SUCCESS

Trying:keep Rule:17 Current: 1 Start: 2 Stop: 2 WOR SUCCESS

Raw: Sally           Root: sale           (1) Tags: ly b econ econ\* means noun

Raw: kept\_up        Root: keep\_up       (3) Tags: ed supv actv try

Raw: with           Root: with           (1) Tags: root prep

Raw: John           Root: john           (0) Tags: x root

Raw: on             Root: on             (1) Tags: root prep space

Raw: the            Root: the            (1) Tags: root det art

Raw: hike           Root: hike           (0) Tags: x root e

Raw: .             Root: .             (0) Tags: per punc

Fig. 27. Continued

treat the lexeme being tested (*keep*), the lexeme which caused the rule to fire (*up*), and all the lexemes which appear between them (none in this case) as a single lexeme — removing any tags previously assigned to those lexemes and adding the tags for the identified sense (3 in this case).

The second trace, shown in Figure 28 is more difficult to disambiguate — *But rather - just like my relative, he grew rather upset*. There are seven ambiguous lexemes in this sentence: *just*, *like*, *relative*, *grew*, *upset*, and *rather* (which appears twice, each in a different sense).

All of the lexemes except *relative* and *grew* are disambiguated straightforwardly. *Relative* can not be disambiguated because the first test checks the lexeme itself (*relative*) for the tag *ly*. Because *relative* does not have that tag assigned and has not been disambiguated, it is not possible to conclude with certainty whether or not the rule will fire. This may seem to be a sort of “catch-22”, but recall that *ly* might have been assigned by the morphological transformation which preceded the disambiguation process.

*Grow* (of which *grew* is a morphological form) can be only partially disambiguated. The first rule fails causing the second to be tested. Like *relative*, the rules for *grow* examine the lexeme itself for a tag. In this case, however, the tag *ing* is always assigned by the morphological transformation process and therefore it can be determined with certainty that the test lexeme does not have the sought after tag. Unfortunately, the next rule to be tested examines lexemes further to the right in the sentence which have not yet been disambiguated, resulting in deferral.

The second pass yields no progress on *relative*, but *grow* is successfully disambiguated because the lexemes to the right of it now have tags assigned to them. The third pass produces no tag assignments. Detecting that no progress has been made during a complete pass, the system “lowers it’s standards” for certainty (a process described earlier). Even with lower standards *relative* can not be disambiguated and the fourth pass also yields no progress. The system detects the condition and applies the “breaklock” rules (also described earlier). These rules cause the action branch that will lead to further tests to be selected — in this case, the **failure** branch.

===== Document # 0 Sentence # 2 =====

Raw: rather      Root: rather      (0) Ambiguous: YES

Tags:

(1%44) ly.know.negate.

(2%38) ly.quan.undrst.

(3%13) ly.virtue.

(4%3) ly.know.ovrst.

Rules:

(6) WOR(K+1,K+1,APPLY(1),NEXT(0),than.)

(7) WOR(K-1,K-2,APPLY(3),NEXT(0),would.)

(8) TOR(K-1,K-2,APPLY(3),NEXT(0),'d.)

(9) TOR(K-1,K-2,NEXT(0),SKIP(12),def1.)

(10) WOR(K-1,K-1,APPLY(3),APPLY(2),just.much.)

(12) WOR(K-1,K-1,APPLY(1),NEXT(0),but.)

(13) TOR(K+1,K+1,NEXT(0),SKIP(15),art.)

(14) TOR(K-1,K-1,APPLY(1),NEXT(0),be.)

(15) WOR(K-1,K-1,APPLY(4),NEXT(0),or.)

(16) TOR(K+0,K+0,APPLY(1),APPLY(2),b.)

Special Tags:root

Word Form Tags:

Raw: just      Root: just      (0) Ambiguous: YES

Tags:

(1%69) ly.ovrst.quan.

(2%28) ly.time\*.

Fig. 28. Trace of a complicated disambiguation

(3%1) modif.legal.pstv.virtue.

(4%0) ly.legal.pstv.virtue.

Rules:

(6) TOR(K+0,K+0,NEXT(0),APPLY(4),root.)

(7) WOR(K+1,K+1,NEXT(0),SKIP(10),like.)

(8) TOR(K-1,K-1,APPLY(1),APPLY(2),def1.mod.)

(10) TOR(K+1,K+1,APPLY(1),NEXT(0),art.to.)

(11) TOR(K+1,K+1,APPLY(2),NEXT(0),time.)

(12) TOR(K+1,K+1,NEXT(0),SKIP(15),prep.)

(13) WOR(K+1,K+1,APPLY(1),APPLY(2),for.by.)

(15) TOR(K+1,K+1,APPLY(2),NEXT(0),int.)

(16) TOR(K-1,K-1,NEXT(0),SKIP(18),ed.hav.)

(17) TSAME(K+1,K+1,APPLY(2),NEXT(0),ed.supv.)

(18) WOR(K+1,K+1,APPLY(2),NEXT(0),as.such.)

(19) WOR(K+1,K+1,NEXT(0),SKIP(21),that.)

(20) TOR(K+2,K+2,APPLY(2),NEXT(0),punc.conj1.)

(21) TSAMEM(K-1,K-1,APPLY(3),NEXT(0),det.'s.)

(22) TOR(K-1,K-1,NEXT(0),APPLY(1),be.link.)

(23) TOR(K+1,K+1,APPLY(3),APPLY(1),punc.conj1.)

Special Tags:root

Word Form Tags:

Raw: like            Root: like            (0) Ambiguous: YES

Tags:

(1%51) prep.conj.conj2.rel.

(2%47) supv.arousal.pstv.psv.afil.

Fig. 28. Continued



(3%2) emot.arousal.noun.pstv.psv.affil.

Rules:

- (4) TOR(K+0,K+0,NEXT(0),SKIP(10),root.)
- (5) TOR(K-1,K-1,APPLY(2),NEXT(0),mod.do.def1.)
- (6) TOR(K-1,K-1,NEXT(0),SKIP(8),ly.)
- (7) TOR(C-1,C-1,APPLY(2),NEXT(0),mod.do.)
- (8) TOR(K-1,K+1,APPLY(2),APPLY(1),to.)
- (10) TOR(K-1,K-1,APPLY(2),NEXT(0),def.)
- (11) TOR(K+0,K+0,NEXT(0),APPLY(2),ing.)
- (12) TOR(K+1,K+1,APPLY(1),APPLY(3),det.pron.hu.)

Special Tags:root

Word Form Tags:

Raw: relative    Root: relative    (0) Ambiguous: YES

Tags:

- (1%55) hu.kin.kin\*.noun.role.
- (2%16) modif.know.psv.undrst.weak.
- (3%27) ly.know.psv.undrst.weak.

Rules:

- (5) TOR(K+0,K+0,APPLY(3),NEXT(0),ly.)
- (6) TOR(K+0,K+0,APPLY(1),NEXT(0),s.'s.s'.)
- (7) TOR(K+1,K+1,NEXT(0),APPLY(2),conj.punc.prep.pron.supv.ly.det.)
- (8) WOR(K+1,K+1,APPLY(2),APPLY(1),to.)

Special Tags:root

Fig. 28. Continued

## Word Form Tags:

Raw: grew            Root: grow            (0) Ambiguous: YES

## Tags:

(1%34) supv.actv.incr.psv.strng.  
 (2%29) supv.incr.strng.  
 (3%22) supv.vb.verb.incr.strng.  
 (4%3) supv.actv.power.strng.work.  
 (5%11) modif.actv.incr.strng.

## Rules:

(7) WOR(K+1,K+1,DELID(2),NEXT(0),up.)  
 (8) TOR(K+0,K+0,NEXT(0),SKIP(11),ing.)  
 (9) TOR(K-1,K-2,NEXT(0),SKIP(11),det.prep.)  
 (10) TOR(K-1,K-1,APPLY(5),NEXT(0),det.prep.ly.)  
 (11) TOR(K+1,K+2,APPLY(3),NEXT(0),color.emot.dim.er.)  
 (12) TOR(K+1,K+1,APPLY(3),NEXT(0),to.ed.)  
 (13) WOR(K+1,K+2,APPLY(3),NEXT(0),old.)  
 (14) TSAMEM(K+1,K+1,APPLY(3),NEXT(0),eval.ly.)  
 (15) TOR(K+1,K+1,APPLY(4),APPLY(1),det.indef.food.)

## Special Tags:ed

## Word Form Tags:

Raw: rather        Root: rather        (0) Ambiguous: YES

## Tags:

(1%44) ly.know.negate.  
 (2%38) ly.quan.undrst.  
 (3%13) ly.virtue.

Fig. 28. Continued

(4%3) ly.know.ovrst.

Rules:

- (6) WOR(K+1,K+1,APPLY(1),NEXT(0),than.)
- (7) WOR(K-1,K-2,APPLY(3),NEXT(0),would.)
- (8) TOR(K-1,K-2,APPLY(3),NEXT(0),'d.)
- (9) TOR(K-1,K-2,NEXT(0),SKIP(12),def1.)
- (10) WOR(K-1,K-1,APPLY(3),APPLY(2),just.much.)
- (12) WOR(K-1,K-1,APPLY(1),NEXT(0),but.)
- (13) TOR(K+1,K+1,NEXT(0),SKIP(15),art.)
- (14) TOR(K-1,K-1,APPLY(1),NEXT(0),be.)
- (15) WOR(K-1,K-1,APPLY(4),NEXT(0),or.)
- (16) TOR(K+0,K+0,APPLY(1),APPLY(2),b.)

Special Tags:root

Word Form Tags:

Raw: upset      Root: upset      (0) Ambiguous: YES

Tags:

- (1%62) modif.emot.ngtv.pain.psv.weak.
- (2%3) hostile.ngtv.noun.vary.weak.
- (3%8) modif.actv.ngtv.pain.strng.
- (4%14) supv.actv.ngtv.pain.strng.
- (5%9) supv.actv.exert.hostile.ngtv.strng.

Rules:

- (7) TOR(K+1,K+1,APPLY(4),NEXT(0),def2.def4.hu.)
- (8) TOR(K+1,K+1,NEXT(0),SKIP(13),det.)

Fig. 28. Continued

- (9) TOR(C+1,C+1,APPLY(4),NEXT(0),hu.)
- (10) TOR(K+0,K+0,NEXT(0),APPLY(5),root.)
- (11) TOR(K-1,K-2,SKIP(13),NEXT(0),be.vb.)
- (12) TOR(K+2,K+2,NEXT(0),APPLY(5),punc.)
- (13) TOR(K+0,K+0,APPLY(3),NEXT(0),ing.)
- (14) TOR(K+0,K+0,APPLY(2),NEXT(0),s.)
- (15) TOR(K-1,K-1,APPLY(5),NEXT(0),to.mod.do.hav.)
- (16) TOR(K-1,K-1,APPLY(2),APPLY(1),det.)

Special Tags:root e

Word Form Tags:

Trying rules for rather

Trying:rather Rule:6 Current: 1 Start: 2 Stop: 2 WOR FAILURE

Trying:rather Rule:7 Current: 1 Start: 0 Stop: 0 WOR FAILURE

Trying:rather Rule:8 Current: 1 Start: 0 Stop: 0 TOR FAILURE

Trying:rather Rule:9 Current: 1 Start: 0 Stop: 0 TOR FAILURE

Trying:rather Rule:12 Current: 1 Start: 0 Stop: 0 WOR SUCCESS

Trying rules for just

Trying:just Rule:6 Current: 3 Start: 3 Stop: 3 TOR SUCCESS

Trying:just Rule:7 Current: 3 Start: 4 Stop: 4 WOR SUCCESS

Trying:just Rule:8 Current: 3 Start: 2 Stop: 2 TOR FAILURE

Trying rules for like

Trying:like Rule:4 Current: 4 Start: 4 Stop: 4 TOR SUCCESS

Trying:like Rule:5 Current: 4 Start: 3 Stop: 3 TOR FAILURE

Trying:like Rule:6 Current: 4 Start: 3 Stop: 3 TOR SUCCESS

Trying:like Rule:7 Current: 4 Start: 2 Stop: 2 TOR FAILURE

Trying:like Rule:8 Current: 4 Start: 3 Stop: 5 TORK FAILURE

Fig. 28. Continued

Trying rules for relative  
Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR DEFERRED

Trying rules for grow  
Trying:grow Rule:7 Current: 9 Start: 10 Stop: 10 WOR FAILURE  
Trying:grow Rule:8 Current: 9 Start: 9 Stop: 9 TOR FAILURE  
Trying:grow Rule:11 Current: 9 Start: 10 Stop: 11 TOR DEFERRED

Trying rules for rather  
Trying:rather Rule:6 Current: 10 Start: 11 Stop: 11 WOR FAILURE  
Trying:rather Rule:7 Current: 10 Start: 9 Stop: 8 WOR FAILURE  
Trying:rather Rule:8 Current: 10 Start: 9 Stop: 8 TOR FAILURE  
Trying:rather Rule:9 Current: 10 Start: 9 Stop: 8 TOR SUCCESS  
Trying:rather Rule:10 Current: 10 Start: 9 Stop: 9 WOR FAILURE

Trying rules for upset  
Trying:upset Rule:7 Current: 11 Start: 12 Stop: 12 TOR FAILURE  
Trying:upset Rule:8 Current: 11 Start: 12 Stop: 12 TOR FAILURE  
Trying:upset Rule:13 Current: 11 Start: 11 Stop: 11 TOR FAILURE  
Trying:upset Rule:14 Current: 11 Start: 11 Stop: 11 TOR FAILURE  
Trying:upset Rule:15 Current: 11 Start: 10 Stop: 10 TOR FAILURE  
Trying:upset Rule:16 Current: 11 Start: 10 Stop: 10 TOR FAILURE

Trying rules for relative  
Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR DEFERRED

Trying rules for grow  
Trying:grow Rule:11 Current: 9 Start: 10 Stop: 11 TOR SUCCESS

Trying rules for relative  
Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR DEFERRED

Trying rules for relative  
Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR DEFERRED

Fig. 28. Continued

Trying rules for relative

Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR DEFERRED

Trying rules for relative

Trying:relative Rule:5 Current: 6 Start: 6 Stop: 6 TOR FAILURE

Trying:relative Rule:6 Current: 6 Start: 6 Stop: 6 TOR FAILURE

Trying:relative Rule:7 Current: 6 Start: 7 Stop: 7 TOR SUCCESS

Trying:relative Rule:8 Current: 6 Start: 7 Stop: 7 WOR FAILURE

Raw: But	Root: but	(1) Tags: root b conj conj1 undrst
Raw: rather	Root: rather	(1) Tags: root ly know negate
Raw: -	Root: -	(0) Tags: dash punc
Raw: just	Root: just	(2) Tags: root ly time*
Raw: like	Root: like	(1) Tags: root prep conj conj2 rel
Raw: my	Root: my	(1) Tags: root det gen self singp
Raw: relative	Root: relative	(1) Tags: root hu kin kin* noun role
Raw: ,	Root: ,	(0) Tags: comma punc
Raw: he	Root: he	(1) Tags: root def1 pron def male other thrdp
Raw: grew	Root: grow	(3) Tags: ed supv vb verb incr strng
Raw: rather	Root: rather	(2) Tags: root ly quan undrst
Raw: upset	Root: upset	(1) Tags: root e modif emot ngtv pain psv weak
Raw: .	Root: .	(0) Tags: per punc

Fig. 28. Continued

The disambiguation process proceeds uneventfully thereafter. In this sentence, the breaklock rules result in the correct sense of *relative* being selected.

Since the intent of LexNet is to more or less replicate the behavior of the GI, the verification process was (apparently) straightforward. All that was necessary was to process a large volume of text using the GI, process the same text corpus using LexNet, and verify that the two produce the same results. The text corpus that was chosen was the well known “Brown Million Word Sample of the English Language” [12]. This text sample is widely considered to be representative of the American lexicon [34]. In fact the same corpus, in an earlier form, was sampled for the original verification of the Harvard dictionaries [25].

The entire Brown corpus was obtained in Standard Generalized Markup Language (SGML) format. The SGML format uses special character sequences to delimit paragraphs, headings, sections, etc. in a document. Neither the GI nor LexNet are programmed to interpret these special character sequences, so they were removed using a series of filters written using the standard (and unattributable) UNIX utilities Stream Editor (`sed(1)`), Simple Text Formatter (`fmt(1)`) and Translate Characters (`tr(1)`). These filters performed the following functions: remove markers for the beginning and end of documents, paragraphs, and sentences; remove headings; change open and close quotation marks (“ and ”) to standard quotation marks ("); remove sentence number markers; split each line to less than 80 characters (the GI requires input in 80 column punch card format); convert to uppercase (the GI does not recognize lower case characters); and prepend the filename to each line (the GI requires an “identifier” on each “card”) — these filename identifiers were removed before processing with LexNet.

The entire Brown corpus was then transferred to an Amdahl 5090 mainframe computer and processed through the TEXTREAD, TAGGER and CONNECT programs. This processing required 17.36 CPU minutes over a period of 6.5 hours of “wall clock” time. The first 33 of the Brown corpus files were then processed using LexNet on a NeXT 68030 workstation. On average, each file required 3 minutes of

CPU time (“wall clock” time varied from 4 minutes upward, depending on the number of other tasks the workstation was performing). Extrapolating this figure to the full Brown corpus (500 files) yields an estimate of 35 CPU hours to disambiguate the entire corpus. Considering the difference in CPU speed, the fact that the dictionary is not stored in memory by LexNet, and that in LexNet every effort was made to increase portability and readability, while the GI was optimized for performance, these times are quite respectable.

The listing files produced by the GI and LexNet were then compared, using a short ‘C’ program. The first pass on the first file produced 130 differences. Of these differences, 99 were produced either directly or indirectly by the morphological transformation process. In every case except the tagging of words ending in *er* with the tag EST, LexNet was made to conform to the result produced by the GI. Of the remaining 31 differences, 20 involved the word *to* when followed by a verb (e.g., *to face*, *to place*, etc.). The GI correctly identified the fact that *to* was being used as an infinitive, but LexNet identified it as a preposition.

In each case, the activation of rules was traced through the sentence manually — the conclusion was that LexNet was activating the rules correctly. This dilemma, should LexNet be modified to produce the correct classification (attempting to be consistent with the GI) or should it be left unmodified so that it accurately reflects the rules established by the Harvard dictionary, was resolved using a simple heuristic. Prior to the commencement of the disambiguation process proper, but after all morphological transformations, each word following *to* is tested to determine whether or not it has a verb form. If a verb form exists for the word, then *to* is tagged as an infinitive. The heuristic resolved 20 of these differences in the first Brown corpus file, creating zero mismatches. In the second Brown corpus file, this heuristic creates 2 mismatches (one in favor of LexNet, one in favor of the GI) but corrects 32 (which seems a reasonable tradeoff).

This left 11 unresolved differences between the GI and LexNet output in the first file and 10 more in the second (including the 2 mismatches created by the *to* heuristic. Appendix D shows these 21 differences along with their analysis.

Each difference is documented as follows:



The difference itself is defined, showing the word form identified by the GI and the wordform identified by LexNet.

Mismatch: Sentence= 8 GI= GREAT(3) LN= great(2)

The Harvard IV-4 entry for the mismatched lexeme is given (rules that are never tested and word forms that are not considered have been deleted to conserve space).

```
great:
  TAGS:
    (1%63) adj of more than ordinary size, extent, number,
    degree, importance, eminence
    quan.pstv.strng.ovrst.eval.modif.
LN*   (2%9) adj 'greater'
    quan.pstv.strng.ovrst.eval.modif.
GI    (3%7) adj 'greatest'
    quan.pstv.strng.ovrst.eval.modif.
  RULES:
SUCCESS (7) TOR(K+0,K+0,APPLY(2),NEXT(0),er.)
        (8) TOR(K+0,K+0,APPLY(4),NEXT(0),ly.)
        (9) TOR(K+0,K+0,APPLY(3),NEXT(0),est.)
```

To the left of the word forms has been inserted 'LN' or 'GI' to highlight the word forms identified by each system. An asterisk is placed next to the actual (correct) word form used in the sentence.

To the left of each rule is the the result: "SUCCESS" or "FAIL" indicating how LexNet (not the GI) fired the rules. Note that the complete set of rule firings may have taken several passes through the sentence.

The last part of each entry is the set of final tag assignments given by the GI (not LexNet) so that the success or failure of each rule can be confirmed. Blank space and unreferenced output has been deleted to conserve space.

```
** DOCUMENT 1 *** SENTENCE      8
   16:  ACHIEVE          SUPV ROOT COMPLT PSTV STRNG ACTV
   17:   GREAT          MODIF EST COMP OVRST EVAL QUAN PSTV STRNG
   18:  EFFICIENCY     NOUN ROOT ABS ABS* VIRTUE PSTV STRNG
```

In this case, it can be shown that the morphological transformation in the GI incorrectly assigned the tag EST to the word *greater* and that caused rule 7 to fail when it should have succeeded. In this particular case, the difference would not effect any substantive conclusions about the text or any downstream disambiguations because senses 2 and 3 both assign the same set of tags. In other cases, differences in tag assignments can chain react to produce different interpretations.

Several other mismatches deserve special mention here. In sentence 23, document 1, the following mismatch occurs:

```
Mismatch: Sentence= 23 GI= THAT(2) LN= that(1)
that:
      TAGS:
LN*   (1/52) conj "he saw that he must go," "it is certain that
he will go," "the fact that he will go is
evident"
      conj2.conj.
GI    (2/36) pron "that is his mother," "points that are made"
      impers.indef.pron.
      RULES:
SUCCESS (6) TOR(K+1,K+1,NEXT(0),SKIP(10),punc.)
FAIL    (7) TOR(K-1,K-1,APPLY(2),NEXT(0),prep.)
??      (8) TOR(C-1,C-1,APPLY(1),APPLY(2),prep.conj2.)
** DOCUMENT 1 *** SENTENCE 23
      7: RECOMMEND SUPV ED STRNG COMFORM
      8: THAT PRON ROOT INDEF IMPERS
      9: : PUNC
     10: FOUR DET ROOT NUMB CARD QUAN
```

Rule 6 succeeds because *that* is followed by punctuation; rule 7 is therefore tested. Rule 7 fails because *that* is not preceded by a preposition; rule 8 is tested next. Rule 8 is not interpretable because the range specifier indicates that the test should be performed on the lexeme preceding the lexeme that matched the last test. Unfortunately, the last test did not match any lexeme. LexNet maintains the last matched word pointer throughout the processing of the sentence. The word preceding that last match was, indeed, a preposition. In this particular case LexNet chose the correct word form — but this result was quite accidental.

Sentence 55, document 1 demonstrates a case in which both LexNet and the GI select incorrect word forms. This same pattern of tests is repeated in sentence 57.

Sentence 60, document 1 is illustrative.

```

Mismatch: Sentence= 60 GI= OPEN(5) LN= open(4)
open:
    TAGS:
    (1%36) adj-adv-noun not closed - exposed, accessible,
frank, public, in the open
    qual.pstv.modif.
    (2%3) adv"openly"--publicly, in the open
    pstv.virtue.ly.
    (3%5) verb to becomeopen
    work.supv.
LN    (4%39) verb to render open
    work.actv.supv.
GI*   (5%3) verb to begin, commence, inaugurate
    begin.actv.supv.
RULES:
FAIL  (9) TOR(K+0,K+0,NEXT(0),SKIP(18),root.)
      (13) TOR(K+2,K+3,APPLY(5),APPLY(4),com.coll.)
FAIL  (18) TOR(K+0,K+0,APPLY(2),NEXT(0),ly.)
FAIL  (19) TOR(K+0,K+0,NEXT(0),SKIP(27),ing.)
FAIL  (20) TOR(K+0,K+0,APPLY(6),NEXT(0),s.)
FAIL  (21) TOR(K-1,K-2,NEXT(0),SKIP(25),det.)
FAIL  (25) TOR(K-1,K-1,NEXT(0),SKIP(27),prep.)
FAIL  (27) TOR(K-3,K+3,APPLY(5),NEXT(0),com.time.coll.)
SUCCESS (28) TOR(K+1,K+1,APPLY(4),NEXT(0),det.pron.)
      (36) WOR(K+1,K+1,APPLY(5),APPLY(3),with.)
** DOCUMENT 1 *** SENTENCE 60
    1: VANDIV      B COMP EST X
    2: OPEN        SUPV ED BEGIN ACTV
    3: HIS         DET ROOT GEN THRDP MALE OTHER
    4: RACE        NOUN ROOT RITUAL POLIT ACTV
    5: FOR         PREP ROOT CONJ CONJ2

```

The GI identifies word form 5 for the lexeme *open*; there are precisely 3 rules for *open* which might result in that assignment:

```

(13) TOR(K+2,K+3,APPLY(5),APPLY(4),com.coll.)
(27) TOR(K-3,K+3,APPLY(5),NEXT(0),com.time.coll.)
(36) WOR(K+1,K+1,APPLY(5),APPLY(3),with.)

```

All of these tests can be conclusively proven to fail. The only way that *open* could be assigned word form 5 in this sentence would be if the word *race* at position K+2 were assigned the word form meaning “a major group of persons united by descent” which would result in the tag COLL (for collective) being assigned, allowing either rule 13 or rule 27 for *open* to fire successfully. But that will not (does not) occur because the tests for *race* recognize that the sequence *race for* occurs in the sentence. The only conclusion that can be drawn from this set of assignments is that the “forward tagging logic” used by the GI must have implicitly assigned the incorrect word form to *race* while disambiguating *open*. A careful trace of the “forward tagging

logic” rules supports this conclusion. This evidence supports the contention that the disambiguation logic implemented by LexNet is more robust than that used by the GI.

LexNet and the GI treat GOTO rules slightly differently, and that difference shows up in sentence 6, document 2. When LexNet encounters a GOTO rule, it physically replaces the lexeme’s root text with the new lexeme (the raw text remains unchanged). The GI apparently leaves the lexeme undisturbed. This only becomes an issue when rules of surrounding lexemes test for the existence of a particular word. In sentence 6, document 2, LexNet’s strategy works against it, but in sentence 16, document 2, the very same set of rules works in LexNet’s favor. Either approach is logically supportable.

Sentence 13, document 2, is the only case in which the *to* heuristic described previously works against LexNet. In general, that heuristic is designed to make LexNet conform to the GI’s behavior. On occasion, however, it causes a mismatch. In sentence 13, document 2, such a mismatch occurs and the GI makes the correct assignment. In sentence 25 of that document, however, another mismatch on the word *to* works out in LexNet’s favor.

Sentence 25 of document 2 is also interesting because it contains a grammatical error. The original sentence contains the sequence *still be to worked out* which quite obviously should have been *still to be worked out*. Had this grammatical error not existed, LexNet and the GI would have agreed on the categorization of *to be*.

Thus, after processing 3947 words, there was a total of 21 mismatches between the two systems. In 16 of those 21 cases, LexNet chose the correct word form assignment. In three of the cases the GI selected the preferred word form and in two cases, neither system chose the correct form. This evidence suggests that LexNet performs at least as well as, if not better than, the GI.

LexNet’s lexical analyzer also seems to outperform the analyzer in the GI. This is not surprising since the lexical analyzer embedded in LexNet was developed using the well tested and very robust Lex programming language. This language was not available to the authors of the GI.

As evidence of the superiority of the analyzer in LexNet, consider the sentence fragment *public schools, would reduce from 24 to 12semester hours* which appears in the second Brown corpus document. This sentence is missing a space between *12* and *semester*. LexNet (correctly) broke this garble into two separate lexemes. The GI on the other hand, was unable to process the remainder of the sentence. It produced a 'core dump' of sorts and printed the message:

```
START BAIL OUT PROCEDURE...*****..HELP!.. HELP!!...*
```

which is not very informative. This response to garbles occurs twice in document number 2. Two other kinds of (garble) sequences were not analyzable by the GI. A number of sentences contained lexemes inside square brackets (e.g., *republican leader Dirksen [Ill.]*) and others contained the sequence *~ADC* (referring to the government Aid to Dependent Children program). Neither LexNet nor the GI makes any practical sense of the tilde or square bracket symbols. LexNet simply discards the unrecognized symbols (after printing an appropriate message) and processes what it can salvage from the character sequence. The GI discards the entire character sequence (without notification). Because CONTEXT programs are highly dependent upon the the relative position of lexemes, such omissions have the potential to alter the disambiguation process substantively.

## APPENDIX E

## ANALYSIS OF GI VERSUS LEXNET OUTPUT

Mismatch: Sentence= 8 GI= GREAT(3) LN= great(2)

great:

TAGS:

(1%63) adj of more than ordinary size, extent, number,  
degree, importance, eminence

quan.pstv.strng.ovrst.eval.modif.

LN\* (2%9) adj 'greater'

quan.pstv.strng.ovrst.eval.modif.

GI (3%7) adj 'greatest'

quan.pstv.strng.ovrst.eval.modif.

RULES:

SUCCESS (7) TOR(K+0,K+0,APPLY(2),NEXT(0),er.)

(8) TOR(K+0,K+0,APPLY(4),NEXT(0),ly.)

(9) TOR(K+0,K+0,APPLY(3),NEXT(0),est.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 8

16: ACHIEVE SUPV ROOT COMPLT PSTV STRNG ACTV

17: GREAT MODIF EST COMP OVRST EVAL QUAN PSTV STRNG

18: EFFICIENCY NOUN ROOT ABS ABS\* VIRTUE PSTV STRNG

Explanation: LexNet's morphological transformation routine correctly identified the 'er' ending on 'greater'.

=====  
Mismatch: Sentence= 14 GI= THIS(1) LN= this(2)

this:

TAGS:

GI (1%65) adj-adv "this job bothers me", "itis this far"  
dem.dem1.det.

LN\* (2%35) pron "this is something else"

indef.impers.pron.

RULES:

SUCCESS (3) TOR(K+1,K+1,APPLY(2),NEXT(0),s.punc.conj.art.pron.  
prep.s'.supv.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 14

1: " QUOTE

2: THIS DET ROOT B DEM DEM1

3: IS SUPV ROOT VERB BE

4: ONE PRON ROOT DEF DEF4

=====  
Mismatch: Sentence= 23 GI= THAT(2) LN= that(1)  
that:

TAGS:

LN\* (1%52) conj "he saw that he must go," "it is certain that  
he will go," "the fact that he will go is  
evident"

conj2.conj.

GI (2%36) pron "that is his mother," "points that are made"  
impers.indef.pron.

RULES:

SUCCESS (6) TOR(K+1,K+1,NEXT(0),SKIP(10),punc.)  
FAIL (7) TOR(K-1,K-1,APPLY(2),NEXT(0),prep.)  
?? (8) TOR(C-1,C-1,APPLY(1),APPLY(2),prep.conj2.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 23

7:	RECOMMEND	SUPV ED STRNG COMFORM
8:	THAT	PRON ROOT INDEF IMPERS
9:	:	PUNC
10:	FOUR	DET ROOT NUMB CARD QUAN

Explanation: the C origin in rule 8 is undefined because no word  
matched the previous test. The last word that DID pass a test  
(a test on a different lexeme) WAS preceded by a preposition.

=====  
Mismatch: Sentence= 47 GI= HAVE(1) LN= have(2)  
have:

TAGS:

GI (1%36) verb possess, experience, engage in, cause to  
happen

actv.hav.rel.verb.supv.

LN\* (2%7) verb to be compelled or under obligation to  
do something--"have to"

need.power.weak.psv.hav.mod.verb.supv.

RULES:

FAIL (7) TOR(K+1,K+1,NEXT(0),SKIP(11),det.pron.)  
SUCCESS (11) TOR(K+1,K+2,NEXT(0),SKIP(13),to.)  
SUCCESS (12) WOR(K+1,K+1,APPLY(2),NEXT(0),got.to.)  
(13) TOR(K+1,K+1,NEXT(0),SKIP(16),supv.ed.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 47

6:	WOULD	SUPV ROOT VERB MOD ED
7:	HAVE	SUPV ROOT VERB HAVE REL ACTV
8:	TO	SUPV ROOT VERB TO
1	9:	FACE SUPV ROOT PERCV ACTV STRNG

=====  
Mismatch: Sentence= 55 GI= ISSUE(3) LN= issue(1)  
issue:

## TAGS:

LN (1%91) noun a point in question or a matter that is  
in dispute

pfreq.legal.actv.com.know.polit.noun.

\* (2%5) noun that which is printed or published and  
distributed

comnobj.object.com.noun.

GI (3%4) verb to go, pass, or flow out, come forth,  
discharge, emit

actv.exert.strng.supv.

## RULES:

FAIL (4) TOR(K+0,K+0,APPLY(3),NEXT(0),ed.ing.)  
FAIL (5) TOR(K-1,K-1,NEXT(0),SKIP(9),dem.pre.numb.)  
FAIL (9) TOR(K-1,K-1,APPLY(2),NEXT(0),time.)  
FAIL (10) TOR(K+1,K+1,APPLY(3),NEXT(0),det.com.)  
FAIL (11) TSAMEM(K+1,K+1,APPLY(3),NEXT(0),pron.def1.int.)  
SUCCESS (12) TOR(K+0,K+0,NEXT(0),APPLY(1),root.)  
FAIL (13) TOR(K-1,K-1,APPLY(3),APPLY(1),to.mod.do.neg.def.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 55

9: BOND NOUN ROOT ECON COM COMFORM AFFIL STRNG  
10: ISSUE SUPV ROOT EXERT STRNG ACTV  
11: APPROVE SUPV ED AFFIL PSTV STRNG COMFORM

=====  
Mismatch: Sentence= 55 GI= EARLY(3) LN= early(2)  
early:

## TAGS:

(1%68) adv-adjective in or during the first part of  
a period of time, course of action,  
series of events

time\*.ly.

LN\* (2%30) adv-adj earlier--comparative of 'early'.

time\*.ly.

GI (3%2) adv-adjective earliest--superlative of 'early'.

time\*.modif.

## RULES:

FAIL (4) TOR(K+0,K+0,APPLY(1),NEXT(0),root.)  
SUCCESS (5) TOR(K+0,K+0,APPLY(2),APPLY(3),er.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 55

11: APPROVE SUPV ED AFFIL PSTV STRNG COMFORM



12: EARLY           MODIF EST COMP TIME\*  
13: IN             PREP ROOT LY SPACE

Explanation: LexNet's morphological transformation routine correctly identified the 'er' ending on 'earlier'.

```
=====
Mismatch: Sentence= 57 GI= ISSUE(3) LN= issue(1)
issue:
  TAGS:
LN   (1%91) noun a point in question or a matter that is
in dispute
      pfreq.legal.actv.com.know.polit.noun.
*   (2%5) noun that which is printed or published and
distributed
      comnobj.object.com.noun.
GI   (3%4) verb to go, pass, or flow out, come forth,
discharge, emit
      actv.exert.strng.supv.
  RULES:
FAIL  (4) TOR(K+0,K+0,APPLY(3),NEXT(0),ed.ing.)
FAIL  (5) TOR(K-1,K-1,NEXT(0),SKIP(9),dem.pre.numb.)
FAIL  (9) TOR(K-1,K-1,APPLY(2),NEXT(0),time.)
FAIL  (10) TOR(K+1,K+1,APPLY(3),NEXT(0),det.com.)
FAIL  (11) TSAMEM(K+1,K+1,APPLY(3),NEXT(0),pron.def1.int.)
SUCCESS (12) TOR(K+0,K+0,NEXT(0),APPLY(1),root.)
FAIL  (13) TOR(K-1,K-1,APPLY(3),APPLY(1),to.mod.do.neg.def.)
** DOCUMENT 1 *** SENTENCE 57
    1: THE           DET ROOT B ART
    2: BOND          NOUN ROOT ECON COM COMFORM AFFIL STRNG
    3: ISSUE         SUPV ROOT EXERT STRNG ACTV
    4: WILL          SUPV ROOT VERB MOD PFREQ
=====
```

```
Mismatch: Sentence= 58 GI= THERE(2) LN= there(1)
there:
  TAGS:
LN*  (1%60) pron existential operator--'there are 2 senses'
      pron.
GI   (2%38) adv locative--in that place--'he was there'
      space.ly.
  RULES:
SUCCESS (4) TOR(K+1,K+3,APPLY(1),NEXT(0),be.)
** DOCUMENT 1 *** SENTENCE 58
```

5: SAID SUPV ROOT SAY ED PFREQ  
 6: THERE LY ROOT SPACE  
 7: ALSO MODIF ROOT LY QUAN  
 8: IS SUPV ROOT VERB BE  
 9: A DET ROOT ART

=====  
 Mismatch: Sentence= 60 GI= OPEN(5) LN= open(4)

open:

TAGS:

(1%36) adj-adv-noun not closed - exposed, accessible,  
 frank, public, in the open

qual.pstv.modif.

(2%3) adv"openly"--publicly, in the open

pslv.virtue.ly.

(3%5) verb to becomeopen

work.supv.

LN (4%39) verb to render open

work.actv.supv.

GI\* (5%3) verb to begin, commence, inaugurate

begin.actv.supv.

RULES:

FAIL (9) TOR(K+0,K+0,NEXT(0),SKIP(18),root.)

FAIL (18) TOR(K+0,K+0,APPLY(2),NEXT(0),ly.)

FAIL (19) TOR(K+0,K+0,NEXT(0),SKIP(27),ing.)

FAIL (20) TOR(K+0,K+0,APPLY(6),NEXT(0),s.)

FAIL (21) TOR(K-1,K-2,NEXT(0),SKIP(25),det.)

FAIL (25) TOR(K-1,K-1,NEXT(0),SKIP(27),prep.)

FAIL (27) TOR(K-3,K+3,APPLY(5),NEXT(0),com.time.coll.)

SUCCESS (28) TOR(K+1,K+1,APPLY(4),NEXT(0),det.pron.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 60

1: VANDIV B COMP EST X

2: OPEN SUPV ED BEGIN ACTV

3: HIS DET ROOT GEN THRD P MALE OTHER

4: RACE NOUN ROOT RITUAL POLIT ACTV

5: FOR PREP ROOT CONJ CONJ2

=====  
 Mismatch: Sentence= 78 GI= THIS(1) LN= this(2)

this:

TAGS:

GI (1%65) adj-adv "this job bothers me", "itis this far"  
 dem.dem1.det.

LN\* (2%35) pron "this is something else"

indef.impers.pron.

RULES:

SUCCESS (3) TOR(K+1,K+1,APPLY(2),NEXT(0),s.punc.conj.art.pron.  
prep.s'.supv.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 78

1: " QUOTE  
1 2: THIS DET ROOT B DEM DEM1  
3: WAS SUPV ED VERB BE

=====  
Mismatch: Sentence= 86 GI= GOT(2) LN= get(1)

get:

TAGS:

LN\* (1%29) verb to have or gain possession or control of  
something--to obtain, fetch, receive,  
acquire--includes use in past tense meaning  
"have"--"she's got brown hair";  
understand (1); overpower, injure, kill (0)  
fetch.actv.supv.

GI (2%58) verb become, move, cause to occur or be, have  
happen--"he'll get better,""we finally got  
home," "get it done," "you'll get to do it"  
verb.vb.supv.

RULES:

FAIL (12) TOR(K-1,K-1,NEXT(0),SKIP(14),hav.'s.)  
FAIL (14) WOR(K+1,K+1,DELID(4),NEXT(0),over.)  
FAIL (15) WAND(K+1,K+2,DELID(5),NEXT(0),rid.of.)  
FAIL (16) WAND(K+1,K+2,DELID(6),NEXT(0),around.to.)  
FAIL (17) TSAMEM(K+1,K+1,APPLY(2),NEXT(0),ly.det.)  
SUCCESS (18) TOR(K+1,K+2,NEXT(0),SKIP(23),det.)  
FAIL (19) TOR(K+1,K+1,NEXT(0),SKIP(23),det.ly.)  
SUCCESS (23) TOR(K+1,K+1,NEXT(0),SKIP(30),pron.hu.)  
FAIL (24) TOR(C+0,C+0,SKIP(30),NEXT(0),def1.)  
FAIL (25) TOR(C+1,C+2,APPLY(2),NEXT(0),ed.emot.ing.)  
FAIL (26) WOR(C+1,C+2,APPLY(2),NEXT(0),into.out.in.on.over.  
across.onto.through.together.)  
SUCCESS (27) TOR(C+0,C+0,NEXT(0),APPLY(1),def.hu.)  
FAIL (28) TOR(C+1,C+2,APPLY(2),APPLY(1),to.)

\*\* DOCUMENT 1 \*\*\* SENTENCE 86

14: AND CONJ ROOT CONJ1  
15: WILLIAM S X  
16: GOT SUPV ED VERB VB  
17: HIMSELF PRON THRDP OTHER MALE DEF DEF3 PFREQ SELF  
18: A DET ROOT ART  
19: PERMIT NOUN ROOT OBJECT COM LEGAL COMNOBJ POWER

20: TO SUPV ROOT VERB TO  
 21: CARRY SUPV ROOT FETCH STRNG ACTV

Explanation: at rule 24, the origin C points to the word 'HIMSELF'.

=====  
 Mismatch: Sentence= 6 GI= SINCE(1) LN= since(2)  
 since:

TAGS:

GI\* (1%52) conj-prep-adv from some past time to the present  
           time\*.conj2.ly.conj.prep.  
 LN (2%45) conj indicates causality--because  
           causal.ovrst.conj2.conj.

RULES:

FAIL (4) WOR(K-1,K-1,APPLY(1),NEXT(0),ever.)  
 FAIL (5) WOR(K+1,K+1,APPLY(1),NEXT(0),then.)  
 FAIL (6) WOR(K+1,K+1,APPLY(2),NEXT(0),there.)  
 FAIL (7) TOR(K+1,K+2,APPLY(1),NEXT(0),per.q.)  
 SUCCESS (8) TOR(K+1,K+1,NEXT(0),SKIP(10),ed.)  
 FAIL (9) TOR(K-1,K-2,APPLY(1),NEXT(0),hav.)  
 SUCCESS (10) WOR(K+1,K+6,APPLY(2),NEXT(0),is.are.am.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 6

21: BOOK NOUN S OBJECT COM COMNOBJ  
 22: " QUOTE  
 23: SINCE PREP ROOT CONJ CONJ2 LY TIME\*  
 24: TEXAS NOUN ROOT NAME POLIT ECON  
 25: WAS SUPV ED VERB BE  
 26: A DET ROOT ART

Explanation: LexNet physically replaces a lexeme when a GOTO rule is encountered ('WAS' becomes 'IS').

=====  
 Mismatch: Sentence= 13 GI= TO(2) LN= to(1)

to:

TAGS:

LN (1%61) infinitive infinitive  
           to.verb.supv.  
 GI\* (2%31) prep preposition  
           prep.

RULES:

SUCCESS (25) TSAME(K+1,K+1,APPLY(1),NEXT(0),verb.root.)  
 (26) TOR(K+1,K+2,NEXT(0),SKIP(30),det.pron.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 13

30: POCKET           NOUN S OBJECT TOOL  
 31: TO               PREP ROOT  
 32: BANK             NOUN S HUMAN COLL ECON ECON\*  
 33: ,                PUNC COMMA  
 34: INSURANCE       NOUN ROOT MEANS ECON ECON\*  
 35: AND              CONJ ROOT CONJ1  
 36: PIPELINE        X ROOT

=====  
 Mismatch: Sentence= 16 GI= SINCE(1) LN= since(2)  
 since:

TAGS:  
 GI       (1%52) conj-prep-adv from some past time to the present  
           time\*.conj2.ly.conj.prep.  
 LN\*      (2%45) conj indicates causality--because  
           causal.ovrst.conj2.conj.

RULES:  
 FAIL     (4) WOR(K-1,K-1,APPLY(1),NEXT(0),ever.)  
 FAIL     (5) WOR(K+1,K+1,APPLY(1),NEXT(0),then.)  
 FAIL     (6) WOR(K+1,K+1,APPLY(2),NEXT(0),there.)  
 FAIL     (7) TOR(K+1,K+2,APPLY(1),NEXT(0),per.q.)  
 SUCCESS (8) TOR(K+1,K+1,NEXT(0),SKIP(10),ed.)  
 FAIL     (9) TOR(K-1,K-2,APPLY(1),NEXT(0),hav.)  
 SUCCESS (10) WOR(K+1,K+6,APPLY(2),NEXT(0),is.are.am.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 16  
 12: HEAR            NOUN ING RITUAL LEGAL  
 13: ,               PUNC COMMA  
 14: SINCE           PREP ROOT CONJ CONJ2 LY TIME\*  
 15: THE             DET ROOT ART  
 16: BILL            NOUN ROOT OBJECT POLIT COM POLIT\* COMNOBJ  
 17: WAS             SUPV ED VERB BE PASSIVE  
 18: INTRODUCE      SUPV ED ACTV COMFORM  
 19: ONLY            LY ROOT QUAN UNDRST  
 20: LAST            DET ROOT NUMB ORD MODIF TIME\*

Explanation: LexNet physically replaces a lexeme when a GOTO rule is  
 encountered ('WAS' becomes 'IS').

=====  
 Mismatch: Sentence= 20 GI= OLD(4) LN= old(3)  
 old:

TAGS:  
 (1%67) adjective aged, long standing, prior,  
        used affectionately toward something or someone

known a longtime  
                   weak.time\*.modif.  
 (2%13) idiom-adj. "(n) year(s), month(s) old"--  
 phrase used to specify (n)  
                   time\*.modif.  
 LN\* (3%15) adjective "older"--comparative  
                   weak.time\*.modif.  
 GI (4%4) adjective "oldest"--superlative  
                   weak.time\*.modif.

RULES:  
 SUCCESS (5) TOR(K+0,K+0,APPLY(3),NEXT(0),er.)  
 (6) WOR(K-1,K-1,DELID(2),NEXT(0),year.month.)  
 (7) TOR(K+0,K+0,APPLY(1),APPLY(4),root.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 20  
 40: PERMIT SUPV ING INTREL POWER PSTV STRNG PSV PFREQ  
 41: OLD MODIF EST COMP TIME\* WEAK

=====  
 Mismatch: Sentence= 25 GI= BE(1) LN= be(3)

be:

TAGS:  
 GI (1%47) verb used as a copula connecting subject to  
 predicate adjective or nominative  
                   verb.be.supv.  
 (2%3) verbused as auxiliary to form progressive  
                   verb.be.supv.  
 LN\* (3%50) verb used as auxiliary to form passive  
                   passive.verb.be.supv.  
 (4%0) idiom "tobe sure"--handled by "sure"  
                   handels.

RULES:  
 FAIL (5) TOR(K+1,K+1,APPLY(1),NEXT(0),det.prep.)  
 SUCCESS (6) TOR(K+1,K+2,APPLY(3),NEXT(0),ed.)  
 (7) TOR(K+1,K+2,APPLY(2),APPLY(1),ing.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 25  
 4: STILL LY ROOT TIME\* PFREQ  
 5: BE SUPV ROOT VERB BE  
 6: TO PREP ROOT  
 7: WORK SUPV ED SOLVE ACTV  
 8: OUT

=====  
 Mismatch: Sentence= 25 GI= TO(2) LN= to(1)

to:

TAGS:  
LN\* (1%61) infinitive infinitive  
to.verb.supv.  
GI (2%31) prep preposition  
prep.  
RULES:  
(25) TSAME(K+1,K+1,APPLY(1),NEXT(0),verb.root.)  
(26) TOR(K+1,K+2,NEXT(0),SKIP(30),det.pron.)  
(27) TOR(K+1,K+1,APPLY(2),NEXT(0),det.pron.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 25  
4: STILL LY ROOT TIME\* PFREQ  
5: BE SUPV ROOT VERB BE  
6: TO PREP ROOT  
7: WORK SUPV ED SOLVE ACTV  
8: OUT

Explanation: No rules fire in LexNet. Assignment of word form 1 performed by the LexNet 'to' heuristic.

=====

Mismatch: Sentence= 30 GI= LATE(6) LN= late(1)  
late:

TAGS:  
LN\* (1%57) adj-adv "later"--at a more advancedtime  
time\*.modif.  
GI (6%2) adj "latest"--most recent, current, coming after  
all others  
time\*.modif.

RULES:  
SUCCESS (7) TOR(K+0,K+0,APPLY(1),NEXT(0),er.)  
(8) TOR(K+0,K+0,APPLY(3),NEXT(0),ly.)  
(9) TOR(K+0,K+0,APPLY(6),NEXT(0),est.)

\*\* DOCUMENT 2 \*\*\* SENTENCE 30  
25: LATE MODIF COMP EST E TIME\*

=====

Mismatch: Sentence= 37 GI= LATE(6) LN= late(1)

\*\* DOCUMENT 2 \*\*\* SENTENCE 37  
8: LATE MODIF EST COMP TIME\*

=====

Mismatch: Sentence= 39 GI= AID(2) LN= aid(1)

aid:

TAGS:  
LN\* (1%16) verb to give help  
actv.affil.pstv.strng.intrel.supv.  
GI (2%81) noun help, a helper  
pstv.virtue.actv.affil.noun.  
RULES:  
FAIL (4) TOR(K+0,K+0,APPLY(1),NEXT(0),ed.ing.)  
FAIL (5) TAND(K-1,K-1,APPLY(2),NEXT(0),det.prep.)  
?? (6) TOR(C+0,C+0,APPLY(1),APPLY(2),to.mod.pron.do.ly.)  
\*\* DOCUMENT 2 \*\*\* SENTENCE 39  
11: WOULD SUPV ROOT VERB MOD ED  
12: AID NOUN ROOT AFFIL PSTV VIRTUE ACTV  
13: MORE DET ROOT PRE PRE2 PRON LY COMP ER QUAN STRNG

Explanation: The origin C is undefined for rule 6.

=====  
Mismatch: Sentence= 61 GI= THIS(1) LN= this(2)

this:

TAGS:  
GI (1%65) adj-adv "this job bothers me", "itis this far"  
dem.dem1.det.  
LN\* (2%35) pron "this is something else"  
indef.impers.pron.  
RULES:  
SUCCESS (3) TOR(K+1,K+1,APPLY(2),NEXT(0),s.punc.conj.art.pron.  
prep.s'.supv.)  
(4) TOR(K+1,K+1,NEXT(0),APPLY(1),ly.)  
\*\* DOCUMENT 2 \*\*\* SENTENCE 61  
1: " QUOTE  
2: THIS DET ROOT B DEM DEM1  
3: IS SUPV ROOT VERB BE  
4: A DET ROOT ART



## APPENDIX F

### QUESTIONNAIRE

#### F.1 Instructions

*Important notes:*

- Your participation in this study is voluntary (and appreciated!).
- There are no penalties or rewards associated with performance or participation in this study.
- There is no compensation for participating in this study.
- You may refuse to answer any questions which make you uncomfortable.
- Your responses are anonymous – you do not need to identify yourself.

*Instructions:*

- The purpose of this study is to determine the extent to which a particular computer program can identify *issues*, *evaluative statements* and *statements which are similar to one another* in brainstorming transcripts. Approximately 20 subjects are participating in this study.
- Enclosed are: one (1) demographic survey and seven (7) brainstorming transcripts with accompanying questionnaires. Each of these questionnaires has three parts corresponding to the identification of issues, evaluative statements and similar statements, respectively.
- Please read each transcript and answer the questions in the questionnaires to the best of your ability.
- Please work independently.
- There is no time limit (others have reported that it takes between 60 and 90 minutes to complete all seven transcripts).
- Please return the completed questionnaires to: David Cheslow, BANA Department, 401 Blocker Bldg. by Wednesday, August 3, 1994.
- **Thank you very much!**

## F.2 Demographic Questions

Demographic information:

Please check the boxes and fill in the blank to best describe you and your background.

Gender:  Female  
 Male

Age:  20-25     31-35     41-45     51-55  
 26-30     36-40     46-50     56-60

Major field of study or expertise: \_\_\_\_\_

Highest degree earned:  Bachelors  
 Masters  
 Doctorate

How many times have you *participated* in brainstorming sessions?  Never  
 Fewer than 10 times  
 10 or more times

How many times have you *been a leader or facilitator* in brainstorming sessions?  Never  
 Fewer than 10 times  
 10 or more times

## F.3 MBA Success Factors

Topic: What are the critical success factors for the MBA program as you see them ?

blah.

Glad to see you could make it Mike.

Understand business practices.

top-quality faculty.

Develop leadership skills.

International Emphasis.

case studies.

time management.

confidence development.

Professors involved in Industry.

Integration of class room material to real world situations.

people management.

program to help find internships.

Better management of the program ( director and administrative staff ).

building a representation of TAMU MBA program.

Students with previous work experience.

highly integrated program.

improvement of communication abilities.

ability to apply textbook skills to the real world .

**INTERNSHIPS.**

communication between faculty members.

partnership between students and professors.

Students performance.

being able to learn each subject thoroughly.

communication skills.

cost reduction of text books.

learning to spell.

MIKE WHY WERE YOU LATE.

bring in many recruiters.

reassurance of relevance of material.

program that matches students to employers.

real learning, not upload and download (cram and purge).

not being swamped with material so that learning is impaired.

survival.

developing leadership skills. probably should follow the contingency approach.

keep classes small.

++++SLEEP++++.

INTERNSHIPS AVAILABLE TO MOST STUDENTS.

develop intellectual curiosity.

increased opportunities to meet with former graduates(not enough now).

Build international recognition.

driving on no matter how much material given.

more hands on rather than read and repeat.

beat the hell out of alabama.

alabama sucks ha-ha-ha-ha.

a PLACEMENT CENTER FOR BUSINESS DEPT OR MBA.

contacts with the real world.

less lists.

WHERE IS INDIANA.

soccer not accounting.

real world? What is the real world?

where is arizona.

more interaction with professionals in your desired field of study.

professors who don't let you slip through cracks.

attract published and quality profs.

have classes at Duddley's.

reduction in the emphasis on tests.

motivation, motivation, motivation.

more field trips.

cohort c not being treated like a redheaded stepchild.

team work.

bring in speakers from industry.

place higher value on learning.

..or treated like a birth defect..

balanced work load (Marketing--AAUUGH!!!).

would # 57 please send some motivation this way.

I am a redheaded stepchild.

big emphasis on "teaching" faculty, not on published necessarily.

insider information.

More emphasis on learning and knowledge, less on tests.

and not get caught.

must increase entrance standards.

a realization that cohort c was told to expect something it has not been given.

gaining hands-on experience by consulting for local firms.

no more 16 hour semesters.

Spoetzl Tour.

practical possibility: company simulation throughout the program; can run our own companies, using knowledge gained from classes.

Access to information on what people really do in "the real world".

amen.

cases that integrate not alienate.

Communication between students & faculty in interested field.

Spoetzl case or two.

giving students an opportunity to take field trips of companies to dallas or houston.

less info on each test if we must have them.

these cases do integrate.

no more 8 or 15 chapter tests.

International emphasis on work, rather than study.

simulations.

more integration between classes.

set up classes that would simulate the business environment.

scott mendel i love you.

use mba's to figure why private school overhead is half of public's.

Empathy of professors.

no more whining.

thanks whoever I needed the boost this morning.

lets talk about mike.

Not sympathy!!

mike who.

more professors willing to work with us in developing our skill.

opportunities for internships.

please no more whining.

I want to be like Mike.

Who's whining?

Mike, are you really engaged to your cousin?

hows Angie last yr's bana 607 T.A. doing?

clear guidelines from MPO regarding degree plans,etc.

learning to be comfortably numb.

crimson tied.

yes, less whining.

Mike, don't mind them.

my mind is BLANK.

classroom participation.

how do you make a hormone.

more computer lab hours.

dont pay her.

maybe we should all evaluate our personal efforts before bitching about the work.

if i knew what i needed i wouldn't be here.

how do you make mike moabn.

I agree with that about evaluating out efforts before knocking the program.

continue allowing lots of elective hours. Self-direction!

more emphasis on international aspect - let's all go abroad!!

David!

have guest professors from foreign universities].

Viva le France!

IA France, not IE France !



Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Academics, teaching                            |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction with them               |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Spatial relationships, near/far                |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Quantities, amounts, more or less of something |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Communication                                  |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Means to accomplish a goal, how-to             |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Power, control, authority                      |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                    |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Places, physical locations                     |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A  
       I agree with that about evaluating out efforts before knocking the program.

D    NS    A  
       yes, less whining.

D    NS    A  
       more professors willing to work with us in developing our skill.

D    NS    A  
       not being swamped with material so that learning is impaired.

D    NS    A  
       beat the hell out of alabama.

Comments/ Other evaluative statements:

---

---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} WHERE IS INDIANA.  } where is arizona.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

D	NS	A	} Develop leadership skills.  } developing leadership skills.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

D	NS	A	} no more whining.  } please no more whining.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

D	NS	A	} time management.  } people management.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

D	NS	A	} real world?  } What is the real world?
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} communication skills.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} opportunities for internships.
D	NS	A	} reduction in emphasis on tests.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} More emphasis on learning and knowledge, less on tests.
D	NS	A	} people management.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Better management of the program (director and administrative staff).
D	NS	A	} International Emphasis.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} International emphasis on work, rather than study.
D	NS	A	} improvement of communication abilities.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} More emphasis on learning and knowledge, less on tests.

Comments/ Other similar statements:

---



---

## F.4 Definition of Quality

Topic: What quality means to your organization.

A standard of excellence.

Benchmark for quality.

Facilitating change and improvement in the City.

Part of mission and vision statement.

provide customers with timely implementable recommendations.

More of a value statement than a mission statement.

Perhaps combine with ""Facilitating change and improvement....."

Quality is an aspect of the culture in our department?

which inspires each of us to do our best and to allow others to do their best.

To do our best at what?.

auditing, bringing about change, etc. .

on-time, on-target response to customer.

Providing valuable, reliable, and objective information to City Council, management.

Directed to management level of organization.

Quality statement about HOW we accomplish our mission.

Assisting CoA departments find ways to serve their customers with the fewest resources necessary to provide

the highest possible service.

Get well...taxpayers' point of view.

This statement helps to define more clearly what we mean by improvement.

Providing departments with meaningful analysis of operations as well as useful recommendations for improvement.

Value of information provided to departments.

descriptive as well as recommendations.

doing our best all the time to reach the goals of the organization.

Following government auditing standards.

Anticipating the needs of our customers.

Being proactive rather than reactive.

Supporting our conclusions with accurate, organized and factual data.

Treating customers with care, respect and honesty.

These thoughts are mainly concerned with information, customers, and goals.

Customers are a given but information and goals are things we can change, so maybe that is the inherent definition of "quality."

Presenting information as concise as possible.

Having a qualified, experienced, and trained staff to provide the best assistance to our customers.

Helping City departments understand and improve their operations, and making sure managers are accountable for assets.

Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- | D                        | NS                       | A                        |  |
|--------------------------|--------------------------|--------------------------|--|
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour     |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                        |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction with them                   |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Interrelations, connectedness                      |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Communication                                      |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Power, control, authority                          |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Economic matters, making money, buying and selling |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Social roles of people                             |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Awareness, knowledge, knowing                      |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Collectives, groups                                |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A  
         Having a qualified, experienced, and trained staff to provide the best assistance to out customers.

D    NS    A  
         doing our best all the time to reach the goals of the organization.

D    NS    A    Assisting CoA departments find ways to serve their customers with the fewest resources necessary to provide

D    NS    A    which inspires each of us to do our best and to allow others to do their best.

D    NS    A    To do our best at what?.

Comments/ Other evaluative statements:

---

---



Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} Benchmark for quality.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Get well.
D	NS	A	} Providing departments with meaningful analysis of operations as well as useful recommendations for improvement.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Value of information provided to departments.
D	NS	A	} A standard of excellence
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Benchmark for quality.
D	NS	A	} Providing valuable, reliable, and objective information to City Council, management.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Value of information provided to departments.
D	NS	A	} Directed to management level of organization.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Having a qualified, experienced, and trained staff to provide the best assistance to out customers.

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

<table border="0"> <tr> <td style="text-align: center;">D</td> <td style="text-align: center;">NS</td> <td style="text-align: center;">A</td> <td rowspan="2" style="font-size: 2em; vertical-align: middle;">{</td> <td rowspan="2" style="vertical-align: middle;">                 More of a value statement than a mission statement.                  descriptive as well as recommendations.             </td> </tr> <tr> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> </tr> </table>	D	NS	A	{	More of a value statement than a mission statement. descriptive as well as recommendations.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<table border="0"> <tr> <td style="text-align: center;">D</td> <td style="text-align: center;">NS</td> <td style="text-align: center;">A</td> <td rowspan="2" style="font-size: 2em; vertical-align: middle;">{</td> <td rowspan="2" style="vertical-align: middle;">                 provide customers with timely implementable recommendations.                  Value of information provided to departments.             </td> </tr> <tr> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> </tr> </table>	D	NS	A	{	provide customers with timely implementable recommendations. Value of information provided to departments.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<table border="0"> <tr> <td style="text-align: center;">D</td> <td style="text-align: center;">NS</td> <td style="text-align: center;">A</td> <td rowspan="2" style="font-size: 2em; vertical-align: middle;">{</td> <td rowspan="2" style="vertical-align: middle;">                 To do our best at what?.                  doing our best all the time to reach the goals of the organization.             </td> </tr> <tr> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> </tr> </table>	D	NS	A	{	To do our best at what?. doing our best all the time to reach the goals of the organization.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<table border="0"> <tr> <td style="text-align: center;">D</td> <td style="text-align: center;">NS</td> <td style="text-align: center;">A</td> <td rowspan="2" style="font-size: 2em; vertical-align: middle;">{</td> <td rowspan="2" style="vertical-align: middle;">                 Providing valuable, reliable, and objective information to City Council, management.                  Helping City departments understand and improve their operations, and making sure managers are accountable for assets.             </td> </tr> <tr> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> <td style="text-align: center;"><input type="checkbox"/></td> </tr> </table>	D	NS	A	{	Providing valuable, reliable, and objective information to City Council, management. Helping City departments understand and improve their operations, and making sure managers are accountable for assets.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D	NS	A	{			More of a value statement than a mission statement. descriptive as well as recommendations.																													
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																	
D	NS	A	{	provide customers with timely implementable recommendations. Value of information provided to departments.																															
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																	
D	NS	A	{	To do our best at what?. doing our best all the time to reach the goals of the organization.																															
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																	
D	NS	A	{	Providing valuable, reliable, and objective information to City Council, management. Helping City departments understand and improve their operations, and making sure managers are accountable for assets.																															
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>																																	

Comments/ Other similar statements:

---



---

## F.5 Customer Service

Topic: Definition of Customer Service.

involvement of the customer in making decisions in our company.

Matching the customer's expectations for assistance after the sale.

delivering on the support service expectations of the customer.

Quality service, service with a smile.

delivering more than the customer expects.

partnership with the customer in an ongoing relationship.

no surprises or disruptions in service.

This is when the customer provides you with great service.

Guaranteed.

Customer is always right.

anticipating correctly what the customer wants and delivering it.

a willingness to "go the extra mile" to MAKE SURE that the customer is happy with us.

Doing whatever is necessary to achieve total satisfaction.

Easy channels of feedback built into the product

1.E. 800#s.

When we are ready to provide whatever it takes to support the product.

Providing services beyond those we are responsible for.

All aspects of supporting the customer including hot line, bug fixes, administrative support, consulting, new releases, customer communication.

In other words, the entire organization exists to support the customer.

Listening to feedback and responding with better products at next release.

Being available when they need us.

Staying in touch with customers' changing needs -- not falling out of contact.

Proactive contact with the customer to ensure that we are satisfying their needs.

Addressing issues before they become crises.

Maintaining Technical contacts and references for referral purposes.

Maintaining customer goodwill by excellent quality of product.

Implementation of Advanced Human Communication Technologies.

Being immediately available 24-hours per day for customer inquiries.

Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Economic matters, making money, buying and selling |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Social roles of people                             |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and Interaction with them                   |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                        |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Communications                                     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Spatial relationships, near/far                    |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Power, control and authority                       |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Time, speed, urgency                               |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Submission, dependence, vulnerability              |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A  
         Listening to feedback and responding with better products at next release.

D    NS    A  
         This is when the customer provides you with great service.

D    NS    A    a willingness to "go the extra mile" to MAKE SURE that the customer is happy with us.  
     

D    NS    A  
         Doing whatever is necessary to achieve total satisfaction.

D    NS    A    Staying in touch with customers' changing needs -- not falling out of contact.  
     

Comments/ Other evaluative statements:

---

---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} Matching the customer's expectations for assistance after the sale.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} delivering on the support service expectations of the customer.

D	NS	A	} All aspects of supporting the customer including hot line, bug fixes, administrative support, consulting, new releases, customer communication.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} In other words, the entire organization exists to support the customer.

D	NS	A	} Matching the customer's expectations for assistance after the sale.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Implementation of Advanced Human Communication Technologies.

D	NS	A	} partnership with the customer in an ongoing relationship.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} In other words, the entire organization exists to support the customer.

D	NS	A	} delivering on the support service expectations of the customer.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} anticipating correctly what the customer wants and delivering it.

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A  
     

} delivering on the support service expectations of the customer.  
Implementation of Advanced Human Communication Technologies.

Comments/ Other similar statements:

---

---



## F.6 Product Benefits

Topic: What are the benefits provided by those things?

Outcomes that are real, valid.

Proof of the path (i.e., documentation).

This has potential.

More cohesive groups.

Clearer communication.

Lets group members contribute on their own schedule, at their own pace (non-face-to-face).

Provides optimal support for the different phases of a group's evolution: 7 stages that work effectively in different time and place settings.

Evolution of a Corporate Memory.

Less meetings.

More focused meetings.

Can accommodate more stakeholders per meeting.

Increased sense of individual value.

Less time spent resolving issues.

More creativity.

More time for the real work to be done.

This idea has potential.

Can avoid getting trapped by someone's personal agenda.

Has potential.

I can think about things with more concentration and can capture those.

I can start a meeting process prior to the actual scheduled time.

Ongoing creativity.

At last.

a way to keep up with the flow of ALL the group's ideas.

without losing ideas along the way.

Team members can prepare for meetings more fully.

and contribute more freely.

The velocity of the group's idea exchange and processing increases.

exponentially!

Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- | D                        | NS                       | A                        |  |
|--------------------------|--------------------------|--------------------------|--|
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Quantities, amounts, more or less of something |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Vitues, desirable outcomes                     |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Means to accomplish goals, how-to              |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Awareness, knowledge, knowing                  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction with them               |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Time, speed, urgency                           |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Interrelations, connectedness                  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Collectives, groups                            |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Change   |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A    Provides optimal support for different phases of a group's  
         evolution: 7 stages that work effectively in different time and  
place settings.

D    NS    A    Can accommodate more stakeholders per meeting.

D    NS    A    I can think about things with more concentration and can  
         capture those.

D    NS    A    and contribute more freely.

Comments/ Other evaluative statements:

---

---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} This idea has potential.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Has potential
			} More creativity
D	NS	A	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	} Ongoing creativity
			} More creativity.
D	NS	A	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	} Has potential.
			} More cohesive groups.
D	NS	A	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	} Team members can prepare for meetings more fully.
			} Lets group members contribute on their own schedule, at their own pace (non-face-to face).
D	NS	A	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	} and contribute more freely.

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} Can accomodate more stakeholders per meeting. < and contribute more freely.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

D	NS	A	} More creativity < and contribute more freely.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Comments/ Other similar statements:

---

---

## F.7 Business Problem

Topic: Perceptions of major problems company X faces today.

No dedicated R&D effort for East Region.

Technology Transfer.

Different teams have different priorities for working on the same problems.

Customers need more specialized help for their business problems and they need it in a reasonable time frame to be effective.

More training required for increasing technical skills of company X Staff and to stay on top of new technology.

We need to work more closely with the technology innovators within our region - customers who know what they want to do with computers to expand their business.

We need to find a solution to the basic problems that many customers have of not be computer literate enough to find answers to their own problems - more specialized training for customers and encouragement from management to be self supporting.

Customer Preception.

Team Communication.

We sometimes don't work together very well.

Division between LAN Systems group and Telecommunications is not in the right place.

Lack of enthusiasm amongst department employees.

Understanding the customer's needs, applying the computer technology to the FBU's business needs.

Resolution of Problems.

Don't always put the customer first.

Functions/ jobs not clearly defined.

Too much time spent on paperwork, details that do not contribute to the bottom line.

Clarifying who are our customers.

Customers expect us to stay ahead of them and be ready to give recommendations on HW & SW before they are ready to make a decision.

This boils down to no R&D.

Too much time spent hand-holding with the customers. Having to do this wastes talent.

Anybody could do this; i. e. changing passwords, telling them how to exit and save a WP document, etc.

customers don't involve us from the beginning, planning stages when they are looking at a new product, process or project.

They expect us to pick up in the middle of a project and supply them with full support!

We need to know what's going on from the beginning.

We need to have agreements between ourselves and our customers on what services and levels of services we provide.

Lack of backup in key jobs.

No standards requires more support effort.

Being able to have time to PLAN.

We automate other departments but cannot automate our own.

Teams do not interact well.

The teams work well separately, but do not interconnect to focus in on the best resolution for the customer.

No real commitment to the long term plan.

Take each other's jobs more seriously.

We are ALL important to the final product - excellent customer service.

Development of the Partnership Approach to doing business.

Lack of Service Level agreements with our customers. not project oriented.

do things haphazardly Lack of communication between company X members.

more trai.

Manager doesn't provide enough guidance on strategy and plans.

We don't try hard enough to help each other.



I don't feel I have enough time to spend coordinating.

between teams and customers.

more training across teams.

Don't recognize all of company X as a single Team.

We don't ask each other for help when we need it.

It's hard to say no when someone comes to you outside the "system" for help.

We don't have methods to prioritize requests.

We are not involved enough in the planning and budgeting of the FBU's.

Customers don't know what our procedures are.

we don't coordinate new ideas on hardware we are looking to test.

We don't have procedures.

Customers need to be brought into the company X Team We do not all share the same ideas, plans, goals.

Figuring out how to become more pro-active rather than just responding to customer requests--i.

e. assume the leadership role.

We don't formalize plans and bounce them off our customers.

We need more cross training with the business units, know their business.

company X teams don't see other company X teams as customers.

We need to have Management input as an integral part of our decision making process.

We need to shut the doors on all "new" projects and clean up our own back yard, do the "right" things for a few months.

me more pro-active rather than just responding to customer requests--i.

e. assume the leadership role.

We don't formalize plans and bounce them off our customers.

Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction with them                   |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Economic matters, making money, buying and selling |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Spatial relationships, near/far                    |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Negation, "NOT" combined with other issues         |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Social roles of people                             |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Quantities, amounts, more or less of something     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                        |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Collectives, groups                                |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Personal relationships                             |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A    We need to shut the doors on all "new" projects and clean up our own backyard, do the "right" things for a few months.

D    NS    A    It's hard to say no when someone comes to you outside the system for help.

D    NS    A    Lack of enthusiasm amongst department employees.

D    NS    A    We need to find a solution to the basic problems that many customers have of not being computer literate enough to find answers to their own problems - more specialized training for customers and encouragement from management to be self supporting.

D    NS    A    Customers need more specialized help for their business problems and they need it in a reasonable timeframe to be effective.

Comments/ Other evaluative statements:

---



---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} We need to find a solution to the basic problems that many customers have of not be computer literate enough to find answers to their own problems - more specialized training for customers and encouragement from management to be self supporting.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} Skill levels in company X do not fit the needs of the Customers, i.e. Computer person available to explain Geophysical software hardware packag.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} Too many teams.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} more training across teams.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} We don't have methods to prioritize requests.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} We don't have procedures.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} We are not involved enough in the planning and budgeting of the FBU's.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
D	NS	A	} We don't have procedures.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	}	We need to find a solution to the basic problems that many customers have of not be computer literate enough to find answers to their own problems - more specialized training for customers and encouragement from management to be self supporting.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		Customers need to be brought into the company X Team We do not all share the same ideas, plans, goals.

Comments/ Other similar statements:

---

---

## F.8 Staff Evaluation

Topic: Peer performance review rating.

The product X Users Manual has never been an impediment to release.

The quality of the product X Users Manual is high. It is kept up-to-date, readable, and useful.

The informational content of her training sessions is high.

Her delivery has improved significantly and she requires very little technical attention during the training sessions.

All of the above is true.

person A's weakness isn't in her work but in her ability to make her needs known to the people she is supporting.

She has shown steady improvement in this area, though, and I certainly wouldn't call it a problem.

More like an opportunity for improvement.

So noted.

Thanks for the feedback.

person A has done excellent work under very difficult circumstances.

It's not easy documenting a rapidly moving target like product X.

person A's products have been excellent.

Attention to detail, completeness, and desired to make an outstanding product, all while accepting limitations of reality.

I have never had any difficulties conversing with person A.

I enjoy our conversations.

Although not a "technical" person, she quickly grasps technical concepts and can apply them to her work on the manual and her training activities.

person A functions as the indispensable "alternate perspective".

Besides, she laughs at my jokes.

My initial rating for interaction with colleagues contains a 4 (four).

I assume it still does.

You are required to give me feedback on this.

Because you did not know you would be required to respond, it is 50/50 on whether or not it is fair to demand a response at this point.

Therefore, to resolve this dilemma, if I do not receive any comments regarding my score of 4 (although I greatly encourage them), I will assume the individual rating me as such has reconsidered - and I will strike the 4 from my mind - having learned nothing about myself - and will strike the 4 from my experience - as I haven't got time for the pain.

so the song goes.

Thank you.

person A.

person A is sensitive to the customers' needs and wants.

She is able to adapt quickly during training sessions depending upon the desires of the customers.

person A is still developing her training skills.

She does good preparatory work, but her actual presentation/meeting skills are not as developed.

This is pretty much the pot calling the kettle black, though.

I have no doubt she will continue to improve with practice.

Noted.

And I whole-heartedly agree.

Part of what I am doing to improve is to standardize my material - so that I use the same stuff, can note pitfalls, where to expound, etc., on the same material.

To date, every training class has been unique.

And, as you know, I am writing four courses for person B, two of which I will "try out" at Company Y in New Orleans.

That should add some boards to my character-house!

She brings a perfectionist attitude to her job and a great desire to do her very best, to produce a product that she is proud to put her name on.

person A often works extra hours.

product X was a moving target late in '90 and early '91.

She put in extra effort to capture product X in the Users Manual.

person A's professional attitude has always been positive, even during a period where a promised job appeared to be given to someone else.

person A dealt with the issue by communicating her disappointment through her supervisor and resolving the problem professionally.

person A's attitude is second to none (with the possible exception of Chris, who I'm convinced is a Martian in disguise).

Don't change a thing!

person A works hard, gets her work done, and communicates well with others.

person A is a valued employee at company X.

I certainly value her as a colleague.

Ditto.

person A's "weaknesses" are in the area of run-time people-management, which is an area that her original duties as technical writer did not exercise as much as current duties as a trainer/facilitator.

Her performance in the first area was excellent, and her performance in the latter area is good and can only continue to improve.

I'm not certain that I understand run-time people-management here - unless it is in the context of managing my training classes - or gathering together my resources.

Could you explain a little more here?

Thanks.

person A.

She takes charge of her work, requires little supervision, and does the very best job that she can.



Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction between them                |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Social roles of people                             |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                        |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Spatial relationships, near/far                    |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Quantities, amounts, more or less of something     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Women  |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Communications                                     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Economic matters, making money, buying and selling |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Time, speed, urgency                               |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Means to accomplish goals, how-to                  |

Comments/ Other issues:

---



---

Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

D    NS    A    person A dealt with the issue by communicating her  
         disappointment through her supervisor and resolving the  
 problem professionally.

D    NS    A    person A's professional attitude has always been positive, even  
         during a period where a promised job appeared to be given to  
 someone else.

D    NS    A    She brings a perfectionist attitude to her job and a great desire  
         to do her very best, to produce a product that she is proud to put  
 her name on.

D    NS    A    Therefore, to resolve this dilemma, if I do not receive any  
         comments regarding my score of 4 (although I greatly  
 encourage them), I will assume the individual rating me as such  
 has reconsidered - and I will strike the 4 from my mind - having  
 learned nothing about myself - and will strike the 4 from my  
 experience - as I haven't got time for the pain.

D    NS    A    person A's weakness isn't in her work but in her ability to make  
         her needs known to the people she is supporting.

Comments/ Other evaluative statements:

---



---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} Thanks for the feedback.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Thanks.

D	NS	A	} Thanks for the feedback.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Thanks you.

D	NS	A	} Thanks for the feedback.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} And I whole-heartedly agree.

D	NS	A	} Her delivery has improved significantly and she requires very little technical attention during the training sessions.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} She takes charge of her work, requires little supervision, and does the very best job that she can.

D	NS	A	} So noted.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} Noted.

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

<table border="0"> <tr> <td>D</td> <td>NS</td> <td>A</td> </tr> <tr> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </table>	D	NS	A	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ <p>This is pretty much the pot calling the kettle black, though.</p> <p>That should add some boards to my character-house!</p>
D	NS	A					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
<table border="0"> <tr> <td>D</td> <td>NS</td> <td>A</td> </tr> <tr> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </table>	D	NS	A	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ <p>She is able to adapt quickly during training sessions depending upon the desires of the customers.</p> <p>person A is a valued empolyee at company X.</p>
D	NS	A					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
<table border="0"> <tr> <td>D</td> <td>NS</td> <td>A</td> </tr> <tr> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </table>	D	NS	A	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ <p>person A has done excellent work under very difficult circumstances.</p> <p>person A works hard, gets her work done, and communicates well with others.</p>
D	NS	A					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
<table border="0"> <tr> <td>D</td> <td>NS</td> <td>A</td> </tr> <tr> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </table>	D	NS	A	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ <p>person A's weakness isn't in her work but in her ability to make her needs known to the people she is supporting.</p> <p>person A works hard, gets her work done, and communicates well with others.</p>
D	NS	A					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
<table border="0"> <tr> <td>D</td> <td>NS</td> <td>A</td> </tr> <tr> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> <td><input type="checkbox"/></td> </tr> </table>	D	NS	A	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ <p>You are required to give me feedback on this.</p> <p>Because you did not know you would be required to respond, it is 50/50 on whether or not it is fair to demand a response at this point.</p>
D	NS	A					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					

Comments/ Other similar statements:

---



---

## F.9 Business Prospect

Topic: You're trying to get someone interested in product X. Enter two to three sentences that capture what you want to say.

I know my target.

Product X takes me there - like an arrow.

Using product X to reach an outcome is like preparing a fine martini: place all your thoughts, ideas, comments, and dreams into product X, press the F4-Submit key, then drink the results with gusto.

The right people.

The right issues.

The right tool to get the job done.

With product X our team gets ideas out on the table quickly.

and it propels us to problem solutions.

quickly.

Has potential.

product X is to a "manual" meeting as a computer is to a child in a library.

product X brings out the best of people's ideas.

everyone's ideas.

You spend half your time in meetings.

Would you be interested in a product that doubles the value of that time?

Who: committed oriented bureaucracies.

Quality is essential to the survival of your company.

Would you be interested in a product that will help you successfully implement a quality program.

Audience: Quality-oriented organizations.

At the end of a tough decision making session, we look back and see that product X was the foundation for making it all possible.

Do you feel overwhelmed by meetings?

Would you like to reduce the frequency and duration of these meetings?

product X can alleviate meeting frustrations while improving group productivity.

That's about it.

Your best ideas come from your employees.

VisionQuest lets you tap into that enormous resource.

That's about it.

product X systematizes the old ways of gathering ideas and evaluating strategies.

product X supports any group process by adding structure, documentation, focus, and enhanced input (creativity).

You have a problem you want to solve, an outcome you want to reach.

product X allows creative teams to capture concepts and build on them more effectively than any manual approach.

I like this differentiation between product X and "manual meetings."

Computers have been used to accelerate computations, such as spreadsheet calculations, and to instantaneously bridge both distance and time with tools such as email.

Networked personal computers equipped with product X allow team members to share a work space, even if separated by time or distance.

product X collects creative information, distributes it to team members, and performs the calculations necessary for a team to assess their level of agreement.

Are your meetings fun?

If not, product X is the answer.

product X provides a template to allow you to conduct a self-assessment on Quality.

product X can improve your group and team dynamics, communication, and creativity.

Easy to use and powerful in its impact!

product X can front-end the quantitative tools for quality (like SPC, QFD) with the qualitative input necessary to begin.

product X provides a structure and vehicle for customer focus groups.

Group and team dynamics evolve effortlessly in a product X environment of creativity and purpose.

product X completes the picture when you already have the quantitative tools to begin an analysis.

Bingo!

product X adds the qualitative dimension to analysis.

product X provides enhanced group performance and participation by allowing individuals to maximize their sense of value and purpose.

If you think that teamwork is vital to your way of doing business, then you must see product X!

product X amplifies team dynamics: creativity and participation are enhanced, points of agreement and disagreement are immediately clear, and a resolution is more quickly achieved.

product X prints and preserves a record of your team's work.

product X is a simple and elegant solution to energizing business teams.

product X brings the power of the PC to the meeting room, or the power of a team session to your desktop.

product X focuses your team and significantly enhances both its productivity and the quality of its work.

You're gonna save a helluva lot of money with this product!

Using product X, you can create the map which will lead you to the outcome, find the shortest distance, reach the end of the rainbow, and capture the colors on paper.

Part 1 – Issues: Indicate your level of agreement that each item below is an issue in the transcript.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Economic matters, making money, buying and selling |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Tools  |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Tangible objects                                   |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Virtues, desirable outcomes                        |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | People and interaction with them                   |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Supportiveness, teamwork, collective behaviour     |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Spatial relationships, near/far                    |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Collectives, groups                                |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Awareness, knowledge, knowing                      |
| D                        | NS                       | A                        |  |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Technical work processes                           |

Comments/ Other issues:

---



---



Part 2 – Evaluatives: Indicate your level of agreement that each statement below is evaluative or judgemental.

D=Disagree, NS=Not Sure, A=Agree

- |                          |                          |                          |  |
|--------------------------|--------------------------|--------------------------|--|
| D                        | NS                       | A                        | product X amplifies team dynamics: creativity and participation are enhanced, points of agreement and disagreement are immediately clear, and a resolution is more quickly achieved. |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |  |
| D                        | NS                       | A                        | product X collects creative information, distributes it to team members, and performs the calculations necessary for a team to assess their level of agreement.                      |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |  |
| D                        | NS                       | A                        | product X allows creative teams to capture concepts and build on them more effectively than any manual approach.   |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |  |
| D                        | NS                       | A                        | Using product X, you can create the map which will lead you to the outcome, find the shortest distance, reach the end of the rainbow, and capture the colors on paper.               |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |  |
| D                        | NS                       | A                        | product X can alleviate meeting frustrations while improving group productivity.   |
| <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |  |

Comments/ Other evaluative statements:

---

---

Part 3 – Similarities: Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	} That's about it.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} That's about it.
D	NS	A	} product X can improve your group and team dynamics, communication, and creativity.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} product X prints and preserves a record of your team's work.
D	NS	A	} product X supports any group process by adding structure, documentation, focus, and enhanced input (creativity).
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} product X focuses your team and significantly enhances both its productivity and the quality of its work.
D	NS	A	} product X can alleviate meeting frustrations while improving group productivity.
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
			} product X can improve your group and team dynamics, communication, and creativity.

Part 3 – Similarities (continued): Indicate your level of agreement that each pair of statements below express the same thought.

D=Disagree, NS=Not Sure, A=Agree

D	NS	A	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	{ product X can improve your group and team dynamics, communication, and creativity. product X focuses your team and significantly enhances both its productivity and the quality of its work.

Comments/ Other similar statements:

---

---

## APPENDIX G

## SOURCE CODE LISTINGS (DIGITAL)

This appendix (digital) contains the source code for the relational knowledge base management system (DBMS), the content analysis knowledge base (DICTION), the inference engine (LEXNET), and three utilities (CONVERT, SHOWWORD, and XREF). All source code is stored on a standard high density DOS format diskette in ASCII text format. The source code was developed for the Berkeley Software Distribution (BSD)<sup>®</sup>, version 4.3, of UNIX<sup>®</sup> operating system and is written in ANSI standard 'C'.

Each application is stored using a common directory structure. The *src* directory contains the 'C' source code files themselves. The *obj* directory is used to store the compiler object files when the application is created. The directory is initially empty and can safely be emptied when redistributing the application. The *inc* directory contains any files which are included by more than one of the 'C' source code files. Each application directory also contains either a *bin* directory or a *lib* directory. These directories will hold the executable (binary) or library files, respectively, produced by the application compilation.

Each application directory contains a 'Makefile' which contains the instructions necessary to completely construct the application (executable or library). Each installer must edit the first few lines of the make files to reflect the location of certain files on their computer system. Specifically, the lines

```
BASE=
```

```
LOCAL=
```

```
LOCALINC=
```

```
LOCALLIB=
```

may need to be changed. `BASE` refers to the directory location of the application being compiled. `LOCAL`, `LOCALINC`, and `LOCALLIB` refer to directories that will be searched when the compiler is attempting to find files that are included into the 'C' source code files. The default values for these three values are customary directory names and need only be changed if the user does not have the access rights necessary to write to these directories on their system. The main makefile calls a second makefile located in either the *bin* or *lib* directory. This second makefile performs the linking step required to create the application. No changes are necessary in the second makefile.

## G.1 DBMS

The DBMS library performs the relational knowledge base management functions for LexNet. It is designed to be a general purpose database manager and is useful independent of LexNet.

In order to build the DBMS library, `dbopen(3)` must be installed on the computer system. `Dbopen(3)` is part of the University of California, BSD<sup>®</sup> software distribution, version 4.4. It can be obtained free of charge, subject to certain licensing restrictions by anonymous FTP. Because DBMS requires this library and all the other applications developed for this dissertation depend upon the DBMS library, it is mandatory that this library be obtained. All of the source code developed for this dissertation was developed to be portable across a variety of hardware and operating system platforms. `Dbopen(3)`, however, is restricted to UNIX<sup>®</sup> systems. This will be a barrier to porting this software system to non-UNIX platforms.

The makefile included with DBMS allows a standard "make" and a "make install". The "make" command compiles and links the library, leaving the newly created library in the *lib* directory. The "make install" command also installs the library and include files in the directories specified by the `LOCALINC`, and `LOCALLIB` directories described above.

## G.2 DICTION

The `lnetdict` library (in the `DICTION` directory) contains the LexNet knowledge base schema and the source code for the high level access functions described in Appendix B.

## G.3 CONVERT

This directory contains the source code for the LexNet inference engine (`lnettrans`). The file `lexyy.c` was generated by FLEX[57], a lexical analyzer generator. The source code file `trans.l` contains the lexical analyzer generator source code.

## G.4 CONVERT

This directory contains the source code for a conversion utility (`lnetconv`) which converts the textual rules supplied for the General Inquirer format provided by ZUMA[68] into the relational knowledge base format used by LexNet. The file `lexyy.c` was generated by FLEX[57], a lexical analyzer generator. The source code file `gilexer.l` contains the lexical analyzer generator source code.

## G.5 SHOWWORD

This directory contains the source code for a small application program which finds a given word (or tag) in the LexNet knowledge base and generates a human readable display of the information — similar to the text format used by the General Inquirer.

## G.6 XREF

This directory contains the source code for a small application program which shows all cross-references to a given word or tag stored in the knowledge base. This utility should prove invaluable to individuals who wish to create their own dictionaries.

## VITA

David A. Cheslow  
RR 3 Box 72 B  
Willis, VA 24380

(540) 789-4802  
Internet: dcheslow@genesee.freenet.org

### Education

December 1995 Doctor of Philosophy in Business Analysis and Research, Texas A&M University emphasizing management information systems.

August 1984 Master of Business Administration, Virginia Polytechnic Institute and State University emphasizing management and management science.

July 1982 Bachelor of Science, Radford University majoring in Business Administration.

### Work Experience

January 1995 to present Vice President of Software Development, Milagro Systems, Inc. of Austin, TX.

September 1992 to January 1995 Lecturer, University of Michigan at Flint.

August 1988 to August 1992 Graduate Teaching Assistant, Texas A&M University.

May 1990 Training Consultant, Group Technologies Corporation of Austin, TX.

August 1985 to July 1988 Planning Research Economist, Virginia Department of Taxation.

March 1983 to June 1984 Research Assistant, Virginia Polytechnic Institute and State University.

### Publications

Cheslow, David A., *Sources of Utility in Group Processes*, Proceedings of the Twenty First Annual Meeting of the Southeast Decision Sciences Institute, Washington D.C., February 27, 1991, 149-51.

Courtney, James F., and Cheslow, David A., *A GDSS Researcher's Workbench*, Paper presented at the Nineteenth Annual Meeting of the Hawaii International Conference on Computer Systems, Kuna, Hawaii, January, 1990.

Pearce, Stephen L., Cheslow, David A., and Workman, Michael E., *Customer Base Management: Improving Profitability Through Information Technology* (CBM), Version 3.2, Bryan TX: Stephen L. Pearce & Associates, 1991.