Volume 3

Walter de Gruyter · Berlin · New York
1979

Text Processing
Textverarbeitung

Papers in Text Analysis and Text Description

Beiträge zu Textanalyse und Textbeschreibung

Edited by
Wolfgang Burghardt
and Klaus Hölker

Walter de Gruyter · Berlin · New York
1979

SHELDON KLEIN
John F. Aeschlimann
David F. Balsiger
Steven L. Converse
Claudine Court
Mark Foster
Robin Lao
John D. Oakley
Joel Smith

Automatic Novel Writing: A Status Report*

Abstract

Programmed in FORTRAN V on a Univac 1108, the system generates 2100 word murder mystery stories, complete with semantic deep structure, in less than 19 seconds.

The techniques draw upon the state of the art in linguistics, compiler theory, and micro-simulation. The plot and detailed development of events in the narrative are generated by a micro-simulation model written in a specially created, compiler-driven simulation language. The rules of a simulation model are stochastic (with controllable degrees of randomness) and govern the behavior of individual characters and events in the modelled universe of the story. This universe is represented in the form of a semantic deep structure encoded in the form of a network – a directed graph with labelled edges, where the nodes are semantic objects, and where the labelled edges are relations uniting those objects. The simulation model rules implement changing events in the story by altering the semantic network. Compiler or translator-like production rules are used to generate English narrative discourse from the semantic deep structure network (the output might be in any language). The flow of the narrative is derived from reports on the changing state of the modelled universe as affected by the simulation rules.

Nodes of the semantic network may be atoms, classes, or complex predicates that represent entire subportions of the network. Atom nodes and relations are linked to expression lists that may contain lexical stems or roots that

* Portions of this research were sponsored by National Science Foundation Grant No. GS-2595, and by the Wisconsin Alumni Research Foundation.
Abridged version of University of Wisconsin Computer Sciences Department Technical Report No. 183, July 1973, University of Wisconsin, Madison, Wisconsin 53706, USA.

are available for insertion into trees during the generation process. (Low level transformations convert the roots into appropriately inflected or derived forms. High level transformations mark the tree for application of the low level ones.) These expression lists may also contain semantic network expressions consisting of objects and relations which may themselves be linked to expression lists, thereby providing the generator with recursive expository power. An atom node may also function as a complex predicate node with status that may vary during a simulation.

Class nodes may refer to lists of object nodes, and the complex-predicate nodes can be linked to pointers to sub-portions of the network that includes themselves, allowing them to be recursively self-referential. (This would permit generation of sentences such as "I know that I know that – (sentence)").

We are also testing a natural-language meta-compiling capability – the use of the semantic network to generate productions in the simulation language itself that may themselves be compiled as new rules during the flow of the simulation. Such a feature will permit one character to transmit new rules of behavior to another character through conversation, or permit a character to develop new behavior patterns as a function of his experiences during the course of a simulation. This feature, combined with the complex-predicate nodes helps to give the system the logical power of at least the 2nd order predicate calculus.

Theoretical motivations include an interest in modelling generative-semantic linguistic theories, including case grammar and presuppositional formulations. The dynamic time dimension added to the semantic deep structure by the simulation makes it possible to formulate more powerful versions of such theories than now exist.

Table of Contents

1.0. *Introduction*

The novel writer described herein is part of an automated linguistic tool so powerful and of such methodological significance that we are compelled to claim a major breakthrough in linguistic and computational linguistic research. What is emerging is a system for modelling human linguistic and social behavior through time, including the transmission of language and complex patterns of social behavior across generations, through the mediation of language, and according to the dictates of any generative semantic linguistic theory currently in existence, including the case grammar of Fillmore, the presuppositional model of Lakoff, and the 1972 semantic theory of Katz, as well as theories of far greater power than any heretofor suggested.

The key components are a compiler driven simulation language system that manipulates events in the form of a semantic deep structure network notation, and which has the power of at least the 2nd order predicate calculus, and a linguistic generative system that can map the semantic deep structure notation into any natural language using grammars within the framework of a variety of linguistic theories, and which can also generate productions in the language of the simulation system itself, providing a natural language meta-compiler capability.

The novel writer described here is a particular application and testing of the more general system in progress. While the computer generated stories contained in the appendix are in English, they might as easily have been produced in any natural language without alteration of the simulation rules or the semantic deep structure. The simulation system that generated the plot can be used to generate any kind of human behavior, within any time scale, with any level of detail, and all within the framework of any theoretical model of behavior that a researcher may care to formulate.

For the novel writer, the simulation language was used to describe the potential behavior of a set of characters in a partially random set of situations. The deterministic aspects guarantee a murder story within the context of a weekend houseparty, arising from possible motives of greed, anger, jealousy or fear. The particular murderer and victim may vary with the random number source and with the particular specification of character traits prior to the generation. The motives for murder arise as a function of events during the course of the generation of the story.

The rules of the simulation model are stochastic, with controllable degrees of randomness, and govern the behavior of individual characters in the modelled universe of the story. This universe is represented in the form of a semantic deep structure that is encoded in the form of a network, a directed graph with labelled edges, where the nodes are semantic objects and where the labelled edges are relations uniting those objects.

The simulation rules alter events in the universe as a function of the passage of time. As the simulation progresses, the newly created events serve as the semantic deep structure input to a generative device that uses compiler or translator like rules to generate discourse in the selected natural language. The flow of the narrative is derived from successive reports on the changing state of the modelled universe.

Much of the semantic, behavioral and presuppositional information can be incorporated in the behavioral simulation rules as well as in the semantic deep structure network. The rules and the deep structure are intimately related in a number of ways. As indicated, the rules can alter the universe, and yet the rules themselves can be represented in the semantic deep structure; and the rules can be used to generate sentences in the simulation language itself, thus permitting the modification of old behavior patterns or the creation of new ones. The ability to partition the semantic deep structure into static and dynamic components, coupled with the higher order predicate calculus power permits the formulation of behavioral linguistic theories and models more powerful than any currently in existence.

In the balance of this paper we shall briefly cite relevant literature and then proceed to a discussion of the system in its novel writing aspect. The appendix includes a complete listing of the simulation language program that generated our several 'novels', and a sample story, length 2100 words, produced by the program complete with semantic deep structure and English text. We also include interesting passages from three other versions of the murder mystery derived from the same basic simulation program.

We note here that the novel writing system, which is operational on a Univac 1108 computer, uses approximately 75,000 words of storage space, of which 35,000 is required for the control mechanisms of the simulation system, 20,000 for the simulation language compiler and 20,000 for the discourse generation component. Approximately 50% of this space is used for data structures. The program generates 2100 word stories, complete with semantic deep structure descriptions as well as text, in less than 19 seconds. The system is programmed in FORTRAN V.

2.0. *Historical Background and Related Research*

The direct antecedents of this research arise from a three-fold base: our work on dependency approximation to semantic networks in discourse generation and inference making, Klein & Simmons, 1963, Klein 1965 a & b, Klein et al., 1966; our work on automatic grammatical inference, Klein, 1967, Klein et al., 1967, 1968, Klein & Kuppin, 1970, Klein & Dennison, 1971, Klein, 1973; and our research on computer simulation of group language behavior integrating all the above topics, Klein, 1965 c, 1966, Klein et al., 1969, and Klein, 1972. The first publication on our simulation language in conjection with a story producting discourse generator is described in Klein et al., 1971.

Other work involving automated semantic networks includes that of Quillian, 1966, Schank 1969, 1972, Schank & Rieger, 1973, Mel'chuk, 1970 (the list is non-exhaustive).

Work involving variants of the 1st order predicate calculus as part of the semantic base component in natural language generative models includes, McCawley, 1968, Bach & Harms, 1968, Lakoff, 1969, Green & Raphael, 1968, Coles, 1968, & Petöfi, 1973 (the list is not exhaustive).

Work involving natural language compiling into semantic representations, inference languages or simulation languages includes (in addition to our own) Kellogg, 1968, Heidorn, 1972, Simmons (in preparation), as well as Green & Raphael, *ibid* and Coles, *ibid* (again the list is not exhaustive).

3.0. *Semantic Network & Discourse Generation System*

The following explication is quoted from Klein, 1973, pp. 3–11:

Semantic Network

The semantic network consists of objects and relations linking those objects. The object nodes and relations have no names in themselves, only numbers. But they are linked to lexical expression lists that contain lexical variants as well as other expression forms. In examples of semantic network representations of deep structures bracketed lexical items selected from the associated lexical lists are provided with the objects and relations for convenience in reading. As an example consider the discourse:

"The man in the park broke the window with a hammer."
"John knows that."

The deep structure network representation might resemble:

```
0 (man) –  R (break; –1) – 0 (window)
|          |
R (in)     R (with)
|          |
0 (park)   0 (hammer)
```

(where the –1 represents a time earlier than present)

But the actual representation of the semantic deep structure is more subtle and has properties not obvious in this example illustration. The network is actually composed of semantic triples. A semantic triple can consist of any sequence of 2 or 3 objects and relations. Every object in the system has a unique number or address. Every triple in the system also has a unique number and is also associated with its time of creation. The network is actually stored in the form of a hash table, wherein the actual semantic network is implied and computable rather than overtly listed. The time of creation of each triple makes the application of tense transformations easy: the simulation system maintains a clock representing 'now'. Accordingly the relative time sequence among deep structure triples is readily computable, and serves as data for generation of surface structure expression of tense, etc. The actual representation of this sentence is closer to:

1. 0 (man) – R (break, – time) – 0 (window)
   R (break, – time) – R (with) – 0 (hammer)
2. 0 (man) – R (in) – 0 (park)

where the second triple in 1. is not actually listed separately; multi-place predicates are indexable through the primary triple.

It is worth repeating that the objects and relations are actually numbered locations with links to other objects and relations. They contain no associated content expression form other than what appears on their lexical expression lists that are also linked to them. However, a lexical expression list may contain other data than just pointers to lexical stems in a dictionary. These items include semantic triples that are not in the network (for expression of idomatic type structures) *and pointers to triples that are in the network.*

The objects and relations in these triples have their own links to their own lexical expression lists. The lexical expression list of an object or a relation may contain pointers to triples in the network that include triples of which it is a member.

Consider now the second sentence of the sample discourse:

"John knows that".

encoded in the semantic network as,

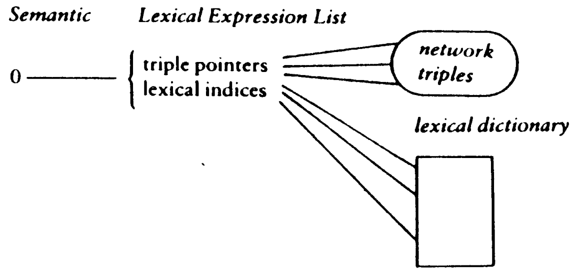
3. 0 (John) – R (know) – 0 (that)

The 0 (that) is a complex predicate object. Its lexical expression list contains pointers to semantic triples 1 and 2. The representation could be self-referential; if the lexical expression list of 0 (that) contained a pointer to triple 3, the network would represent a message approximating:

"John knows that he knows that the man in the park broke the window with a hammer."

This feature helps to give the system the logical power of the 2nd order predicate calculus (at least). Complex logical predications are represented with such predicate nodes linked by logical connective relations. Thus the statement, *if A then B,* where A and B are complex bodies of semantic discourse representing large portions of the semantic network, is represented simply as, 0 (A) – R (implication) – 0 (B), where 0 (A) and 0 (B) each point to lists of semantic triples that may also be of the same time– predications linking

predicate objects that have pointers to triples on their lists. (Always these lists may contain self-referential pointers – serving to justify the claim that the system has the power of at least the 2nd order predicate calculus.) (Other logical devices involving classes of objects and quantifiers are associated with the simulation language manipulates and modifies the semantic network.)
A final schematic of the relevant data structures:



Generative Rules: surface structure // semantic network

The phrase structure rules in the system are part of more complex rules that compile the semantic deep structure network from surface structure – and which also serve the function of generating surface structure from the network. The general form of such a rule is: phrase structure rule // canonical form of semantic triple, where the phrase structure rules are of the usual sort, where linked mappings between nodes in the right half of the phrase structure rules and elements in the network specification are indicated. Strictly speaking the network specification need not be limited just to a semantic triple, as will be seen in the section on inference of rules. Some examples of rules:

S → NP VP // O – R

VP → V NP // R – O

NPP → adj NPP // O – R (attribute) – O

Note that items may occur on either side of the // marks that are not linked to items on the opposite side.

Full comprehension of these rules can best be obtained through an example of generation of surface structure from deep structure. Generalized mechanisms for context sensitive rules and transformations are part of the model. But they are of a type more basic and primitive than in most exisiting linguistic generative models. They can represent more complex types of transformations when properly combined.

A Generation Example

Assume a grammar containing the following surface // semantic rules:

1. S → NP VP // 0 – R
2. NP → NP PP // 0 – R
3. NP → Det NPP // 0
4. NPP → adj NPP // 0 – R – 0
5. NPP → terminal
6. VP → VPP PP // R – R
7. VPP → V NP // R – 0
8. VPP → terminal
9. V → terminal
10. PP → prep NP // R – 0
11. prep → terminal

Assume that the semantic deep structure triple set to be used in the generation is:

0 (man) – R (ride) – 0 (bicycle)
R (ride) – R (in) – 0 (park)
0 (man) – R (is) – 0 (tall)

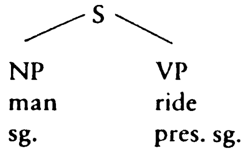
The overlap of various objects and relations in more than one triple is known to the generator by various link markings. The time associated with each triple is also part of the data. A starting symbol S is selected. A prior selective mechanism has placed the triple representing the main predication of the sentence at the top of the triple list. The generative component inspects all S rules whose right hand network description is of the same canonical form as that of the first semantic triple. Here the condition is not satisfied by the only S rule, 1. The triple is then broken into two overlapping parts, 0 (man) – R (ride) and R (ride) – 0 (bicycle). The S rules are then inspected for matches with the fractioned canonical forms. The first matches rule 1. At this point lexical stems are selected from the lexical expression lists associated with the objects and relations in the matched triple fraction. A selected lexical item is tentatively assigned to the node indicated by the link in the syntactic // semantic rule. Grammatical information associated with the lexical item in the dictionary indicates whether or not it can serve as the head of a construction dominated by the node under which it was selected. In this case:

S
NP VP
man ride
sg. pres.

Lexical Dictionary

	NP	VP	PP	ADJ
man	1	1	0	1
ride	1	1	0	0

A bit vector in the dictionary indicates the applicability of a particular node. Note that both *man* and *ride* could serve as nouns or verbs. The grammar also marks the forms when appropriate for application of low level transformations at a later stage. If *man* were selected as a stem to fill a slot defined by an adjective node, ADJ, it would at this time be marked for later application of a transformation that would add *-ly* to it. If the lexical dictionary should prevent the selection of a form, an alternate from the lexical expression list is tried. If none on the list are acceptable, another surface // semantic rule is selected to express the semantic triple. Number for objects is indicated directly in the lexical expression list associated with the particular object (some objects may be inherently plural, as in the case of objects that represent classes). As soon as the lexical items are selected and accepted (the stage in the preceeding diagram), a test for applicability of a high level transformation is made. This transformation uses as its index information that never becomes more complex than the subtree indicated in the above diagram – "a nuclear family tree" – a parent node and its immediate descendents. Often, as in this case, the lexical items are not relevant to the transformation, that here marks the VP with the same number as the NP.

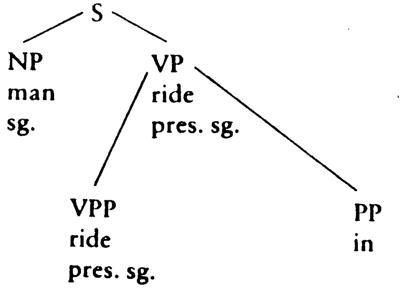


Low level transformations that operate only on terminals and their immediate parent nodes will actually convert the stems to the appropriate words at the end of the generation process. The transformation markings supplied by the high level transformations are carried with the lexical items and may serve as part of the data for defining the applicability of other high level transformations. This breaking up of the transformational component into two types of limited environment primitive operations permits extremely rapid transformational generation and parsing algorithms. The complex labor of searching for applicable environments common to most other automated transformational systems is avoided.

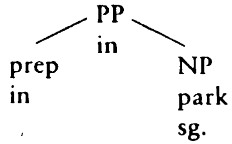
Tense information is obtained from the time marking of the triple. The simulation system maintains a clock, and the relative time order of the triples

in the deep structure generation list can be computed, so that the proper items may be marked for application of transformations handling tense.

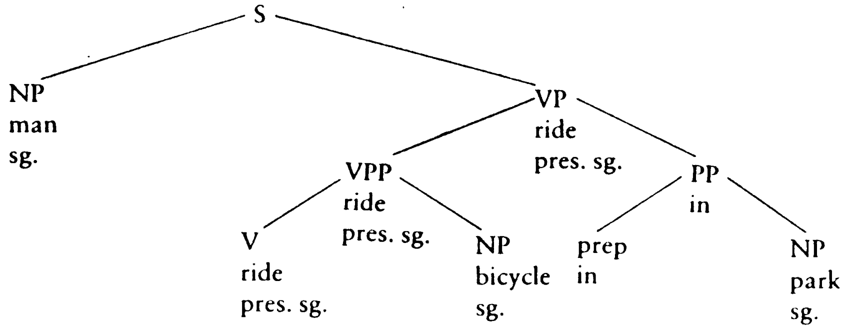
Continuing the generation process, the system saves the remainder of the first triple and skips to the second because of a special link between their relations indicating simultaneity. No VP rule matches the second triple, and it is split into the fractions R (ride) – R (in) and R (in) – R (park). The first fraction matches rule 6. After lexical item in is selected, the tree appears as:



The second triple fraction matches rule 10, yielding after lexical selection:

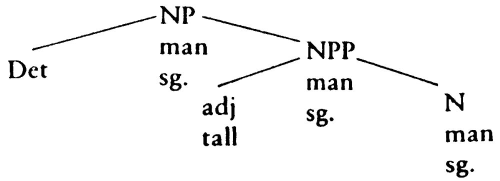


At this point, the second fraction of the first triple is matched against rule 7, and, after lexical selection, the entire tree appears as:

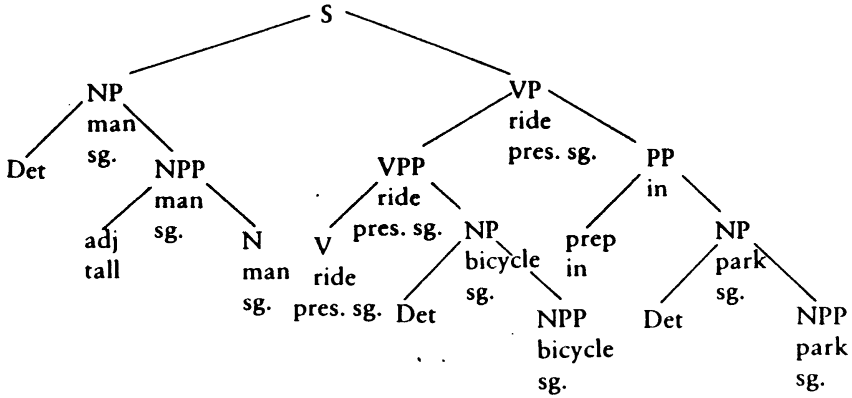


No rule matches the remaining triple 0 (man) – R (is) – 0 (tall). Rule 2 matches the first fraction, but the lexical list for the relation R (is) contains no item acceptable as a PP node descendant. Accordingly, rule 3 is selected. At this point a high level transformation marks the Det for conversion to an appropriate form at the final stage. (If the lexical item had been a proper noun, the Det node would have been marked for deletion.)

At this point rule 4 applies to the entire, unfractionalized, remaining triple, yielding the subtree:



At this point rule 3 is applied to the NP nodes dominating *bicycle* and *park*. The resultant tree is:



The final, low level transformations are applied, yielding the sentence:

"The tall man rides the bicycle in the park"

Note that the semantic triple set might have generated more than one sentence to express the content – either by deliberate stylistic design, or because the rules might not have perimitted a grammatically correct construction incorporating the entire semantic structure.

In addition to the features described in the preceeding quoted excerpt, we note that the current system makes use of production rules that refer to subclasses of relations. While such subclassification is not logically necessary for the mapping of semantic triples into surface structure, it does increase the speed of generation through the elimination of wasted effort in matching semantic triples with inappropriate rules. In the novel writer data base, for example, there are categorizations of relations into prepositional and non-prepositional types (among others) and a coding logic that permits a retreat to a more general categorization upon failure to find a match in the grammar for a particular subcategory.

There are also relations having a numeric logical typing. Such a relation may be used to select a lexical expression item as a function of its current numeric value. For example a numeric relation signifying "affection" may

vary on a scale of plus or minus 3, where plus 3 might be linked to the lexical item "adore" and minus 3 to the item "loathe". In between values link to less extreme terms. The value of such a relation can change dynamically in a simulation as a function of events – accordingly, the appropriate lexical expression of the changing relation follows automatically.

Other features include the listing on generation or change stack of deleted triples and the possibility of marking the lexical expression list pointers with plural transformation markers. This last feature is for semantic nodes whose logical status is always plural, such as nodes that represent classes and whose lexical expression lists only contain pointers to terms descriptive of the entire class. (The dictionary only contains singular stems – hence the pointers to the dictionary connected to such nodes must receive prior plural marking.)

4.0. *Highlights of the Simulation Language*

A detailed description of an early version of the simulation language is contained in Klein, Oakley, Suurballe & Ziesemer, 1971. The basic function of the simulation component is to modify the semantic deep structure network as a function of stochastic behavioral rules that are evaluated in reference to an internal timekeeping mechanism.

A rule consists of two parts, a series of actions and a series of conditions for the implementation of those actions. The conditions are in the form of logical queries about the current state of the modelled universe as represented in the semantic network. Satisfaction or non-satisfaction of the various conditions contribute, either negatively or positively, to a cumulative probability of implementing the action list. A random number source is consulted after the conditions have been evaluated. If the preferred random number is less than or equal to the computed cumulative probability, the action list is implemented. The process can be made deterministic or random with any desired degree of control through manipulation of the probability parameters. Deterministic control is obtained by assigning very high values, such as plus or minus 10, to certain conditions because the range of the random number source is 0 to 1 (a value of 1 or greater indicates certainty and a value 0 or less is absolute rejection).

An internal clock mechanism determines the time of evaluation of groups of rules. Each group has a frequency of evaluation associated with it, and this frequency may be altered by action of some other rule. It may be increased or decreased or, in fact, temporarily or permanently turned off or *disabled*. A disabled rule may be reactivated.

There is also a directed sequence of evaluation through groups of rules in addition to the frequency factor. This sequence may be altered dynamically as a function of the actions of various rules.

The language also permits the use of classes of nodes in its actions and tests, and can also allow variables over those classes, as well as dynamic modification of class membership. There are both subscripted and unsubscripted classes and the subscripted class notation permits a class intersection logic in rules with class variables. For example, a subscripted class FRIENDS (X), where X is a node name or another class name, can function as part of a logical construct in rule condition evaluation expression or action lists.

We present next a grammar of the rules in BNF phrase structure notation, a description of the action types, and a series of examples and notes. The material should help the reader follow the murder mystery simulation program in the appendix, Section 7.6.

Grammar of the rules

⟨single-valued field⟩ ::= ⟨node name⟩ | ⟨loop-variable name⟩
⟨multiple-valued field⟩ ::= ⟨subrule-variable name⟩ | ⟨general class reference⟩
| PICK (⟨multiple-valued field⟩)
⟨specific class reference⟩ ::= ⟨unsubscripted-class name⟩
| ⟨subscripted-class name⟩ (⟨single-valued field⟩)
⟨general class reference⟩ ::= ⟨specific class reference⟩
| ⟨subscripted-class name⟩ (⟨multiple-valued field⟩)
⟨general node field⟩ ::= ⟨single-valued field⟩ | ⟨multiple-valued field⟩
⟨unary op⟩ ::= NOT | FLOAT | ABS | ENTIER | – | +
⟨binary op⟩ ::= ** | * | / | MOD | + | – | EQ | NE | LT | LE | GT | GE | AND | OR
⟨LENGTH function⟩ ::= LENGTH (⟨multiple-valued field⟩)
⟨CLOCK function⟩ ::= CLOCK
⟨relation DUR function⟩ ::= DUR (⟨relation name⟩)
⟨subrule DUR function⟩ ::= DUR (⟨general node field⟩ ⟨relation name⟩
⟨general node field⟩)
⟨constant⟩ ::= ⟨number⟩ | ⟨duration⟩
⟨relation field operand⟩ ::= ⟨relation name⟩ | ⟨LENGTH function⟩
| ⟨CLOCK function⟩ | ⟨relation DUR function⟩
| ⟨constant⟩
⟨relation field subfactor⟩ ::= ⟨relation field operand⟩
| (⟨relation field expression⟩)
⟨relation field factor⟩ ::= ⟨relation field subfactor⟩
| ⟨unary op⟩ ⟨relation field factor⟩
⟨relation field expression⟩ ::= ⟨relation field factor⟩
| ⟨relation field expression⟩ ⟨binary op⟩ ⟨relation field expression⟩
⟨subrule-variable definition⟩ ::= ⟨subrule-variable name⟩ ⟨multiple-valued field⟩
⟨sentence node field⟩ ::= ⟨general node field⟩ | ⟨subrule-variable definition⟩
⟨sentence⟩ ::= (⟨sentence node field⟩ ⟨relation field expression⟩
⟨sentence node field⟩)

⟨subrule operand⟩ ::= ⟨sentence⟩ | ⟨LENGTH function⟩ | ⟨CLOCK function⟩
| ⟨subrule DUR function⟩ | ⟨constant⟩
⟨subrule subfactor⟩ ::= ⟨subrule operand⟩ | (⟨subrule expression⟩)
⟨subrule factor⟩ ::= ⟨subrule subfactor⟩ | ⟨unary op⟩ ⟨subrule factor⟩
⟨subrule expression⟩ ::= ⟨subrule factor⟩
| ⟨subrule expression⟩ ⟨binary op⟩ ⟨subrule expression⟩
⟨option field⟩ ::= ⟨empty⟩ | , ⟨option characters⟩.
⟨option characters⟩ ::= {zero or more option characters}
⟨true-false number field⟩ ::= ⟨empty⟩ | ⟨number⟩ , ⟨number⟩
⟨subrule action field⟩ ::= ⟨empty⟩ |: ⟨action list⟩
⟨subrule⟩ ::= ⟨true-false number field⟩ ⟨option field⟩:
⟨subrule expression⟩ ⟨subrule action field⟩
⟨subrule list⟩ ::= ⟨empty⟩ | ⟨subrule list⟩ ⟨subrule⟩

Description of actions

I. ACTIONS affecting the network
I–1. Set triples in the network
where *triple:* OBJECT (0) RELATIONSHIP (R) OBJECT (0)
Forms: A. 0 R 0
B. 0 R = X 0
C. 0 R
D. 0 R = X
FORM OF TRIPLE DEPENDS ON RELATIONSHIP TYPE:
A. is transitive or intransitive relation, B. is numeric or quantitative intransitive, C. is attribute relation, D. is quantitative attribute relation or numeric attribute relation
I–2. To delete triples in the network
Form: 0 'NOT' R (0)
I–3. To modify numeric relationships in the network
Form: 0 R ± X (0)
I–4. To set secondary triples in the network
*INSERT (TRIPLE) (SECONDARY TRIPLE) ...
Secondary triples are modifiers of primary triples and are transparent to the network, being accessable only through the primary triple which it modifies. The form of a secondary triple is arbitrary with the restriction that the second argument is a relationship and the number of arguments ≤ 3.
I–5. To delete secondary triples from the network
*DELETE (TRIPLE) (SECONDARY TRIPLE)
NOTE: replace all references to ⟨NODE⟩ by ⟨GENERAL NODE FIELD⟩

II. ACTIONS affecting classes
II–1. To add nodes to a class

*ADD ⟨NODE⟩'TO' ⟨CLASS⟩: adds all members of ⟨GENERAL NODE FIELD⟩ to ⟨CLASS⟩

*MOVE ⟨NODE⟩ 'TO' ⟨CLASS⟩: the contents of ⟨CLASS⟩ is replaced by ⟨GENERAL NODE FIELD⟩

II–2. To remove nodes from a class

*REMOVE ⟨NODE⟩ 'FROM' ⟨CLASS⟩

II–3. To remove *all* entries from a class

*ERASE ⟨class⟩

III. ACTIONS affecting lexical items

III–1. To add lexical triples at run-time where the lexical triples are arbitrary combinations of 0's and R's $\leq$ 3 entries.

*LEXTRP (arbitrary triple) ... 'TO' ⟨NODE⟩ | ⟨RELATION⟩

III–2. To move lexical triples from one node or relation to another at run time

*LEXADD ⟨NODE⟩ | ⟨RELATION⟩ ... 'TO' ⟨NODE⟩ | ⟨RELATION⟩

IV. ACTIONS affecting predicate nodes

IV–1. To insert pointers to network triples to the predicate list of a node.

*DISCADD (triple) ... 'TO' ⟨NODE⟩

this action will also create triples which do not already exist in the network

IV–2. To clear the list of pointers to network triples of a node

*DISCLEAR ⟨NODE⟩

V. Actions to control the scheduling of groups of rules

V–1. To activate a group

*ENABLE ⟨GROUP NAME⟩ IN ⟨DURATION⟩

V–2. To de-activate a group

*DISABLE ⟨GROUP NAME⟩

VI. Miscellaneous Actions

VI–1. To print a list of all triples with a specified node as the subject

*DUMP ⟨NODE⟩

VI–2. To control the printing of trace messages in the

A. *TEST ABCDE = 1000

ABCD and E are optinal trace types, the number to the right of = is a maximum line count for the number of traces to be printed.

B. *TSTOP ABC

Turns off the traces specified.

C. *TSTART AB

Turns specified traces on or back on.

VI–3. To print a message

*PRINT ⟨PRINT ARGUMENT⟩

VI–4. To terminate simulation

*END

Examples and notes:

Assume in the following examples that the names below have these associations:

Node names:	JOHN MARY GEORGE SUE BEDROOM	
Relation names:	(A):	HAPPY SAD
	(I):	LIKES LOVES IN HATES DISLIKES
	(NI):	AFFECTION
Class names:	unsubscripted:	PEOPLE ROOMS
	subscripted:	FRIENDS ( )
		ENEMIES ( )
		ADJACENT ( )
Loop-variable names:		PERSON ROOM X Y
Subrule-variable names:		P Q R

General notes:

(a) Input cards are read between columns 1 and 72; 73–80 are ignored.

(b) Free format. Blanks can be used freely except in the following cases. Blanks must not appear (1) within numbers, durations, or reserved words; (2) anywhere in an option field; (3) between trace characters.

(c) Names must start with a letter, followed by letters or digits to any length. However, only the first 8 characters are saved. Thus, LOOPNAME1 and LOOPNAME2 would be taken as the same variable by the system.

(d) Relations can be of the following types:

A:	attribute (normal)
I:	normal intransitive
T:	transitive
NA:	numeric attribute (with synonym list)
NI:	numeric intransitive (with synonym list)
QA:	quantitative attribute (no synonym list)
QI:	quantitative intransitive (no synonym list)

(1) ⟨multiple-valued field⟩:

P
FRIENDS (GEORGE)
ADJACENT (ROOMS)
PICK (PEOPLE)
ENEMIES (PICK(FRIENDS(Q)))
PEOPLE

The PICK function returns a single node, chosen randomly, from its argument. Multivalued subscripts implies concatenation of the specified subscripted classes.

(2) ⟨specific class reference⟩: PEOPLE
FRIENDS (PERSON)
ADJACENT (BEDROOM)

(3) ⟨general node field⟩: JOHN MARY PERSON
P PEOPLE ENEMIES (PICK(Q))
PICK(PEOPLE)
ADJACENT(ROOMS)

(4) ⟨unary op⟩:
The FLOAT operator operates on arguments of type logical, giving 1.0 for TRUE and 0.0 for FALSE. The ENTIER operator truncates the fractional part of a number (e.g., ENTIER(14.23) = 14.0).

(5) ⟨binary op⟩:
The symbols =, ≠, <, <=, >, >= can be used as synonyms for the relational operators EQ, NE, LT, LE, GT, and GE.

(6) ⟨LENGTH function⟩: LENGTH (PEOPLE)
LENGTH (ADJACENT(ROOMS))
Returns a number equal to the number of nodes in its argument.

(7) ⟨CLOCK function⟩:
Returns a number which corresponds to the time of day, i.e., from OHOM to 23H59M.

(8) ⟨relation DUR function⟩: DUR (LIKES)
DUR (IN)
DUR (HAPPY)
This function occurs inside a sentence. (S DUR(R) 0) returns a number equal to the length of time this triple has been in the network. The relation name must be of a non-numeric relation. If the triple does not exist, a run-time error is printed and 0.0 is returned.

(9) ⟨subrule DUR function⟩: DUR (JOHN LIKES MARY)
DUR (PERSON IN R)
Returns a number equal to the length of time a triple has been in the network. The relation name must be non-numeric. While multiple-valued fields are allowed in the syntax, they must contain only a single value at execution time of a DUR function, or else a run-time error will result. Note that no subrule-variable updating ever occurs in a subrule DUR function. If the specified triple is not in the network, an error is printed out, and 0.0 is returned.

(10) ⟨relation field expression⟩: LIKES
LIKES AND NOT (HATES OR DISLIKES)
DUR (LIKES) GT 1 H OR DUR (LOVES) GT 30 M
LENGTH (P) GT O AND DUR (LIKES)/1 H*.001 LE DUR (LOVES)
ABS (AFFECTION*.003) + FLOAT (LIKES)/10.

Relation field expression can be either of type logical or type numeric. A relation name that is numeric or quantitative (ie, NA, NI, QA, or QI) is taken as a numeric operand. Other types (A, I, or T) all are assumed to be logical operands (except within a DUR function). The type of the relation expression determines what type of result the enclosing sentence will return, either a logical value or a numeric value. The operators have specified precedences not explicitly implied in the grammar, and checks are made for correct operand types.

(11) ⟨sentence⟩: (PERSON LIKES OR LOVES P. PEOPLE)
(JOHN AFFECTION MARY)
(X DUR (LIKES) Y)
(GEORGE DUR (LOVES) LT 1W SUE)
(FRIENDS (X) AFFECTION LT O Y)
(X HAPPY OR NOT SAD)
(MARY HAPPY AND LIKES JOHN)
All these sentences return a logical result except the second and third ones. If the relation expression in a sentence yields a numeric value, the subject and object fields of the sentence must be single-valued, or else an error will result.

(12) ⟨option field⟩:
An optional field which specifies the options to be in effect. Currently used options are:

S Synchronous group flag. Used in the option field of a $GROUP statement to flag a group as synchronous. Eg, $GROUP's NEWS: 1H/ON; defines a group which will be executed at hour intervals, on the hour.

O Optimization flag. (Sentences with side effects are not necessarily executed in the subrules, depending on the results of previous logical results).

C Current cycle flag. Allows sentences to test for triples which have been set true during the current time cycle. (Otherwise these are not available till a later time cycle, ie, they act as if they weren't there during the same time cycle).

An option field specified on a $GROUP, $LOOP, $RULE, or $SWITCH statement is in effect for all subrules within its scope, unless explicitly overridden by an option field at a lower level.

(13) ⟨subrule⟩:
.2,0: (PERSON LIKES OR LOVES P. PEOPLE) AND (P IN ROOM);
– 10,0, C: (X NOT IN HOUSE) OR (Y NOT IN HOUSE);

```
-.1, +.2: (P. PEOPLE LIKES X) AND (Y LIKES P):
                              *MOVE P TO TEMPCLS,
                              *ADD X TO TEMPCLS;
,OC:      (X AFFECTION MARY) *0.1 + .2;
:         CLOCK/24H + FLOAT (CLOCK LT 5H);
```

Execution of a subrule returns a number (ie, probability) and optionally specifies an action list to be unconditionally executed. Options in effect for this subrule are either explicitly stated, or are gotten from the last option field in effect (e.g., the enclosing $RULE). A "probability" of +10 or −10 means "abort the subrule list" and return either a TRUE or FALSE for the rule.

(14) ⟨action list⟩:
A list of one or more actions, separated by commas. Actions can either add or delete triples from the network, or perform a control action such as manipulating classes, enabling or disabling groups, or specifying trace or print parameters.

(15) ⟨branch field⟩: RULE1
$NEXT PERSON
$NEXT X
$ENDGROUP

A statement label gives the statement to branch to. A rule can branch anywhere within a group, including out of a loop into an outside loop, but not within a non-enclosing loop. The $NEXT format says to get the next value for a loop variable (equivalent to flowing into an $ENDLOOP statement for that loop). A branch to $ENDGROUP terminates the execution of the group, though it does *not* disable the group (a *DISABLE action is the only thing which can do this).

(16) ⟨$RULE statement⟩:
Basic unit of the language. The cumulative total of the subrule probabilities is tested against a random number which is generated. If the random number (between 0 and 1) is less than or equal to the cumulative total, the rule evaluates TRUE, and the action list of the rule is executed. If not, then it evaluates to FALSE and no actions in the rule's action list are executed. If a branch part is specified, the TRUE or FALSE result also tells where to branch to. E.g., $RULE, C ABC: T($NEXT X) X LIKES Y, *ADD Y TO FRIENDS(6);
−.2,0: (X HATES OR DISLIKES Y);
.4,0: (P. FRIENDS(X) LIKES Y) AND (X LOVES P);

(17) ⟨$SWITCH statement⟩:
This is exactly the same as a $RULE statement except that an action list cannot be specified in the main part of the statement (ie, subrule action lists are still allowed). This statement is used only for branching purposes.

(18) ⟨$LOOP statement⟩:
The specified loop variable will take on all values in the associated multiple-valued field, one at a time. One pass through the loop is made for each different value the loop variable takes on. Note that the values in this multiple-valued field are *saved* on loop entry, and even if the values of this field change during the execution of the loop, this will have no effect on the order or number of loop passes made. E.g., $LOOP, OC : X.FRIENDS (Y7)

(19) ⟨statement list⟩:
This is defined such that any $LOOP statement must have a matching $ENDLOOP statement. Such loops can be nested (currently to a maximum level of 10 only), and can contain other types of statements.

(20) ⟨$GROUP line⟩:
Identifies the start of a group, gives its time increment, and specifies wheter the group is enabled initially or not. The time increment of a group says how often that group will be executed if it is enabled. The "synchronous flag" on a group requires execution of the group only at even multiples of the specified increment. A group can disable or enable any other group, including itself. A group cannot be executed if it is disabled.

(21) Subrule-variables:
These are local variables that can take on a list of values and get updated within a rule. Any subrule variable defined inside a given rule is unknown outside that rule and therefore cannot be referenced. However, the contents of a particular subrule can be saved in a class by a control action in a subrule action list.
The initial definition of a subrule variable creates a copy of the values (ie, nodes) in the specified multiple-valued field. As the subrule-variable is referenced, values in the variable may be deleted. In fact, the only values that are allowed to remain in a subrule-variable are those which make the sentence return TRUE as a result. (If the sentence returns a numeric result (instead of logical), subrule variables within it are *not* updated, and an error occurs if a subrule variable contains more than one value.) E.g.,

(P. PEOPLE LIKES OR LOVES X)

After evaluation, P will contain all those nodes in the class PEOPLE that either LIKES or LOVES X. If no one likes or loves X, P will be set to empty and FALSE returned. If at least one value in P makes the sentence TRUE, then the sentence will return TRUE.

5.0. *Novel Writer Features and Futures*

The data base for the murder mystery simulation is rather simple and skeletal. A very small grammar was used with only a few transformations. The lexical expression lists contain only a limited selection of variants for the semantic nodes and relations. Some errors in the grammar codes of some dictionary items remain.

Our goal was to test the entire system. It is capable of operating with a vastly more sophisticated data structure. Also, not all features of the simulation language were exploited in the murder mystery program. The predicate node device was not used. Text involving productions such as "George knows that John loves Mary", were derived from exploitation of the same secondary triple device that handled expressions of the type, "John broke the window with a hammer." The reason: while the simulation language can dynamically add semantic triple list pointers to nodes and relations, the code for adding the indicated triples to the change stack is not fully implemented. The final implementation of this code will permit easy generation of direct discourse, e.g. constructions such as "John said, '(sentence$_1$, sentence$_2$ ... sentence$_n$)'"

5.1. *Style Control*

While some effort was made to control a few facets of style in the current simulation, most possibilities remain to be exploited. We have found that the simulation language itself can be exploited as a style control device. Various constructs in the rules indicate which triples may be combined into a single sentence according to a sequencing logic. Also, the repetition of the same action by several characters at the same time is usually expressed by a pronoun such as "They ..." or "Everyone ..." even though each individual action is separately tabulated in the semantic network. To achieve this a special "They" node was created in combination with a "They" class. Several individuals performing the same action in the same time period are assigned temporarily to the "They" class, and output makes use of a triple signifying the action with the "They" node functioning as the subject. Special commands such as UNLST and LST alternately block and unblock the generation of uninteresting or repetitious semantic triples. This blocking is occasionally introduced as a random device to vary the output.

A crude and not always successful device is used to control the use of definite and indefinite articles. For the first occurence of some nodes on the change stack "a" is selected – in successive productions "the" is used. (This tabulation holds for all succeeding time frames.) The device collapses where the simulation program data structure has apportioned only a single class type node for several objects (out of laziness or for economy).

Weighted probabilistic selection of syntactic rules is a device that, although not used in the current system, was actually successfully tested in an automatic essay paraphrasing and style control system described in Klein, 1965 a & b.

Narration from the point of view of particular characters is another possibility, and is perhaps most interestingly implemented with the addition of private semantic universes (see section 5.2.).

Addition of a complex network searching component will permit the system to add rich contextual detail to events. For example, where now a change stack may contain just some bare facts about recent changes, a network searching device could seek paths between nodes in apparently unrelated triples, and, if paths exist, add them to the change stack as linking background information.

It should also be possible to have different characters produce discourse in varying styles and dialects as a function of sociolinguistic context. The techniques are implicit in the following discussion of private universes.

5.2. *Private Semantic Universes for Individual Characters*

The ability to provide individual characters in a simulation with private semantic networks, personalized grammars, and even personalized behavioral simulation rules can be achieved with only mildly clever systems programming techniques. The operating system on the Univac 1108, and operating systems of perhaps all 3rd and 4th generation computers have system commands to facilitate a restart capability – that is, the ability to store on disc the current state of a program at specified intervals during execution so that in the event of system failure, the program may be restarted at the point of the last execution of a "store on disc command", without the necessity of starting the program from the beginning.

To implement private universes for individual characters, it is only necessary to add an executive program that will treat each private universe as the total universe when it is resident in core storage, and to save it on disc with a unique name when it is ready to process another character's private universe. The existence of core-resident buffers for communication between private universes is assumed.

5.3. *Simulation of Simulations: Look-Ahead, Planning, Time Travel and Dreams*

Implementation of the private universe capability permits some fascinating possibilites: An individual character could be made to resort to his own look-ahead simulation of events in order to evaluate decision making criteria about the implication of current actions on future events. This would require a private simulation using the data and rules of a private universe. The outcome or outcomes could serve as data to compute probabilities of courses of action for the private individual's actual, simulated real world behavior. Of course introspective, look-ahead simulation need not give accurate results, only hypothetical predictions based on the private rules of a private universe. Naturally, such a universe might contain models of other characters and their private universe. The device also lends itself to the modelling of dream behavior.

For those readers with an interest in science fiction fantasy, we note that this device can be used to model time travel stories, with all conceivable paradoxes. Essentially, it is necessary that the rules permit a private character to treat his introspective look-ahead (or look-back) as serious reality rather than speculation. In the case of travel into the past, all the other characters must take the look-back seriously also.

5.4. *Semantic Parsing*

The private universe concept makes it interesting to allow communication between modelled characters directly via conversational interaction. Of course sophisticated semantic parsing techniques are required. A great deal of work in this area has been attempted by numerous researchers. Although we have not implemented such programs in this system, preliminary study suggests that it will permit semantic parsing logic many times more powerful than any in programs currently in existence. The reason: we own the universe of discourse, a universe where all the subtities of behavior, motivation and context over complex time intervals are all available as data for resolution of the ambiguity that always plagues development of sophisticated semantic parsers.

5.5. *Linguistic and Behavioral Learning: Self-Modifying Behavior and Natural Language Meta-Compiling*

The use of this system for modelling speech communities, language learning and language transmission in conjunction with sociolinguistic models has been explored in detail in Klein, 1965 c, 1966, 1972 and Klein et al. 1969.

The transmission and learning of complex, non-verbal behavioral patterns is also possible using the same mechanisms of the system. Simulation rules may also have a representation in the semantic deep structure network of private individuals. Also, the semantic deep structure may be used to generate sentences and texts (rules and rule groups) in the simulation language itself. The system already has the ability to compile dynamically and add to the simulation new rules that might be generated during the flow of a simulation. It thus becomes possible for characters to modify their own behavior rules in response to private introspection and look-ahead, or in response to verbal and non-verbal behavior of others.

The simulation rules governing rule generating behavior may themselves be modified and generated by the same mechanisms, providing the system with a natural language, meta-compiler capability.

6.0. *Significance for Linguistics, Sociolinguistics and the Behavioral Sciences in General*

We dare to say that Linguistic Theory has no future that is not linked to a computer based experimental methodology. Contemporary linguistic theoretical science has many brilliant theorists in the position analagous to that of a great mathematician attempting to formulate the methodology of long division using roman numerals.

The system described here, with its potential development, provides a means of expressing and testing a vast range of theoretical linguistic models in conjunction with a vast range of sociological and psychological behavioral models, all within the framework of a common, efficient, dynamic time-

oriented notation. The implication is that, for the first time, it will be possible to test heretofor untestable theories of language and language related behavior in psychological, sociological and historical contexts.

7.0. *Appendix*

The semantic deep structure model, as reflected in the choice of nodes, relations and mappings has been more or less arbitrary and experimental, even deliberately inconsistent. The function of the system is independent of the choice of semantic units. One may substitute any scheme according to the dictates of any theory. However, preliminary results suggest that any number of semantic deep structure components will all work nicely, and that the usual arguments for economy or elegance that are to be found in linguistic literature are not necessarily valid in this system. We sense the possibility of proof that such arguments are really functions of the particular notational devices used. A basic principle in computational work is that there is an economy trade between static storage space versus computation time. The non-computational models of linguistic theorists ignore this fact in their proposals and arguments for models of human language behavior.

7.1. *Surface Structure//Semantic Network Production Rules*

Logically, the system need not be limited to semantic 3-tuples and binary phrase structure rules, although such a convention has been used in this version.

0	= object,	sub 1 = that
R	= any relation	
RA	= attribute (adj)	
RV	= verb,	sub 1 = start, stop
RP	= prep	
RS	= possessive	
RADV	= adverb,	sub 1 = adv before verb

PMAP positionally defines mappings between PTYPE triple fragments and the phrase structure rule portions. E.g. in rule 1, the 0 is linked to the NP and the RV is linked to the VP; in rule 4, the first 0 is linked to NPP, the RS is linked to nothing and the second 0 is linked to PNP. PSUB positionally lists relation type subscripts in parallel fashion. PTRANS indicates high level transformation mapping information associated with each rule:

1. = carry down bit vector (null trans.)
2. = OR (logical) bit vectors of new nodes
3. = set infinitive bits for both words
4. = set participle bit for second word
5. = set objective case bit for second word

	GRAMMAR				PTYPE			PMAP			PSUB			PTRANS
1	S	→	NP	VP	0	RV		1	2	0	0	0	0	2
2	S	→	NP	AP	0	R		1	2	0	0	0	0	2
3	NP	→	ART	NPP	0			2	0	0	0	0	0	2
4	NP	→	PNP	NPP	0	RS	0	2	0	1	0	0	0	1
5	NPP	→	N		0			1	0	0	0	0	0	1
6	NPP	→	ADJ	NPP	0	RA		2	1	0	0	0	0	1
7	NPP	→	NPP	MOD	0	RP		1	2	0	0	0	0	1
8	NPP	→	NPP	MOD	0	RV		1	2	0	0	0	0	1
9	VP	→	V		RV			1	0	0	0	0	0	1
10	VP	→	VP	VP	RV	RV		1	2	0	1	0	0	4
11	VP	→	VP	VP2	RV	RV		1	2	0	0	0	0	1
12	VP	→	VP	THAT2	RV	0		1	2	0	0	1	0	1
13	VP	→	VP	NP	RV	0		1	2	0	0	0	0	5
14	VP	→	VP	MOD	RV	RA		1	2	0	0	0	0	1
15	VP	→	VP	MOD	RV	RP		1	2	0	0	0	0	1
16	VP	→	ADV	VP	RV	RADV		2	1	0	0	1	0	1
17	VP	→	VP	ADV	RV	RADV		1	2	0	0	0	0	1
18	MOD	→	PART		RV			1	0	0	0	0	0	1
19	MOD	→	ADJ		RA			1	0	0	0	0	0	1
20	MOD	→	PREP		RP			1	0	0	0	0	0	1
21	MOD	→	PREP	NP	RP	0		1	2	0	0	0	0	5
22	MOD	→	PART	NP	RV	0		1	2	0	0	0	0	5
23	MOD	→	ADJ	THAT2	RA	0		1	2	0	0	1	0	1
24	MOD	→	ADV	ADJ	RA	RADV		2	1	0	0	0	0	1
25	MOD	→	ADJ	VP2	RA	RV		1	2	0	0	0	0	1
26	MOD	→	ADJ	VP2	RA	RP		1	2	0	0	0	0	1
27	AP	→	IS	MOD	R			2	0	0	0	0	0	2
28	VP2	→	TO	VP	RV			2	0	0	0	0	0	3
29	VP2	→	PREP	NP	RP	0		1	2	0	0	0	0	5
30	VP2	→	PREP	MOD	RP	RV		1	2	0	0	0	0	1
31	THAT2	→	THAT	S	0			1	0	0	1	0	0	1
32	PNP	→	NP	POS	0			1	0	0	0	0	0	1

7.2. *Transformations*

As indicated earlier, the system obtains its ability to model a variety of linguistic models, and at the same time a great speed of execution, by decomposing transformational operations into primitive components at several stages. Indications for applications of the transformational fragments are marked and tabulated throughout the generation process. Some of the transformation types themselves give directions for computing and assigning the transformational markings to the growing generation tree (as in section 7.1.).

Ultimately, every terminal element is associated with a bit vector indicating applicable low level transformations as assigned during the various stages of generation. The method avoids complex tree search *after* phrase structure generation, and in comparison with other automated transformational generation systems obtains thereby what may be a 100 to 1 speed advantage.

High Level Transformation Codes (non pronoun)

1. noun sing.
2. noun plural
3. adjectival form
4. prepositional form
5. adverbial form
6. participial form
7. verb (present sing.)
8. verb (present plural)
9. verb (past sing.)
10. verb (past plural)

High Level Transformation (pronoun)

1. subjective case
2. objective case

Low Level Transformation Codes

1. NULL
2. add "will"
3. add "s"
4. add "ing"
5. add "d"
6. add "ly"
7. add "y"
8. delete 1 character and add "ies"
9. add "ed"
10. delete 1 character, add "ing"
11. delete 2 character, add "en"
12. add "es"
13. add "er"
14. add "ings"
15. add "ers"

There are other kinds of high level discourse type transformations not listed here. Of special interest is the one in the form of a special triple of the form MX QQ (n): combine the next (n) head triples with the one preceding. It can be found in the simulation commands and on change stacks.

7.3. *Dictionary*

Lines 3–8 are patterns for setting grammar symbol bits in the dictionary.

The word TYPE delimits classes of words.

The line following TYPE sets bits in the dictionary bit vector (article/no article, pronoun, etc.). For example: line 179 – bit 2 is set for all words in that class for 'no article'; in line 266, bits 2 and 9 are set for all words in that class for 'no article', 'pronoun'.

The lines with pattern types (N, V, PREP, ADJ, ADV, PART) indicate which patterns of grammar bits to set. For example: line 12, line 14 – for word "BE" all bits of pattern PART (line 8) and all bits of pattern V (line 4) will be set. Thus, "BE" is an allowable choice for V, VP, MOD, AP, VP2, or PART when matching in grammar rules.

The lines following pattern types indicate transformations to be associated with all words in the class. For example: in line 25, noun sing. transformation is TRANS # 1 on word 0; noun pl. transformation is TRANS# 3 on word 0. Stem alternates are listed with their associated transformations. Word 0 = main entry Word 1 = 1st stem, Word 2 = 2nd stem, etc. For example: in lines 15–16, V present sing. is TRANS # 1 on stem 1 (null trans on "is"); V past sing. is TRANS #1 on stem 2 (null trans on "are").

```
00001
00002
00003   N      N      NP     NPP
        THAT2  THAT   PNP
00004   V      V      VP     MOD
        AP     VP2
00005   PREP   PREP   AP     MOD
        VP2
00006   ADJ    ADJ    AP     MOD
00007   ADV    ADV
00008   PART   PART
00009   BEOF
00010   TYPE
00011
00012   PART
00013       0  4
00014     V
00015       1  1  2  1
00016       3  1  4  1
00017      WORD   2BE
00018      STEM   2IS
00019      STEM   3ARE
00020      STEM   3WAS
00021
00022   TYPE
00023
00024
N
00025    0  1  0  3
00026   WORD  13BILLIARD ROOM
00027   WORD   9FOOTPRINT
00028   WORD   6NEPHEW
00029   WORD   4GAME
00030   WORD  12HANDKERCHIEF
00031   WORD  14SECRET PASSAGE
00032   WORD   5STAIN
00033   WORD   6STRAND
00034   WORD   6THREAD
00035   WORD  12TENNIS COURT
00036   WORD   3PUB
00037   WORD   6AFFAIR
00038   WORD   3BAR
00039   WORD   6BEATLE
```



```
00040   WORD   4CARD
00041   WORD   6CHANCE
00042   WORD   4CLUB
00043   WORD   9COMPANION
00044   WORD   6COOKIE
00045   WORD   6CORNER
00046   WORD   6DETAIL
00047   WORD   5HOTEL
00048   WORD  12INTERMISSION
00049   WORD  10INVITATION
00050   WORD   7MORNING
00051   WORD   5MOVIE
00052   WORD   4PARK
00053   WORD   4ROCK
00054   WORD   4SONG
00055   WORD   9TELEPHONE
00056   WORD   7THEATER
00057   WORD   4HALL
00058   WORD   8CORRIDOR
00059   WORD   4HAND
00060   WORD   5HOUSE
00061   WORD   9INSPECTOR
00062   WORD   9DETECTIVE
00063   WORD   4JAIL
00064   WORD   3JAW
00065   WORD   4CHIN
00066   WORD   5JEWEL
00067   WORD   4JOKE
00068   WORD   7KITCHEN
00069   WORD   6DAGGER
00070   WORD   4MAID
00071   WORD   7MISTAKE
00072   WORD   5ERROR
00073   WORD   6MOTIVE
00074   WORD   4NECK
00075   WORD   9NIGHTGOWN
00076   WORD   4NOSE
00077   WORD   4NOTE
00078   WORD   5NOVEL
00079   WORD   5NURSE
00080   WORD   5OTHER
00081   WORD   6PARLOR
00082   WORD   5PIANO
00083   WORD   6PILLOW
00084   WORD   7CUSHION
00085   WORD   5PLACE
00086   WORD   3COP
00087   WORD   7PARTNER
00088   WORD   4ROOM
00089   WORD   7SERVANT
00090   WORD   5SHIRT
00091   WORD   5STAIR
00092   WORD   7STOMACH
00093   WORD   3SUN
00094   WORD   4TIME
00095   WORD   4YARD
00096   WORD   5TRUTH
00097   WORD   7WEEKEND
00098   WORD   6WINDOW
00099   WORD   6ENCORE
00100   WORD   7EVENING
00101   WORD   4DOOR
00102   WORD   5FLOWER
00103   WORD  11FINGERPRINT
00104   WORD  11PAPERWEIGHT
00105   WORD   4SHOE
00106   WORD   4VASE
00107   WORD  10ACCUSATION
00108   WORD   6BATHROOM
00109   WORD   7BEDROOM
00110   WORD   4BOOK
00111   WORD   9PAPERBACK
00112   WORD   6BOTTLE
00113   WORD   6BUTLER
00114   WORD   9BREAKFAST
00115   WORD   6BUTTON
00116   WORD  13CANDLE
                HOLDER
00117   WORD   9CARD GAME
00118   WORD  12CONVERSATION
00119   WORD   4TALK
00120   WORD   4COOK
00121   WORD   6CORPSE
00122   WORD   5DIVAN
00123   WORD   9DAVENPORT
00124   WORD   5CRIME
00125   WORD  12CROQUET GAME
00126   WORD   4DAWN
00127   WORD   7SUNRISE
00128   WORD   3DAY
00129   WORD  11DINING ROOM
00130   WORD   8DRAWER
00131   WORD   5GARDEN
00132   WORD   5PISTOL
00133   WORD  12DRAWING
                ROOM
00134   WORD  11GREEN HOUSE
00135   WORD   3GUN
00136   TYPE
00137
00138   N
00139    0  1  0  1
00140   WORD   4HAIR
00141   WORD   7JEWELRY
00142   WORD   5BLOOD
00143   WORD   4GORE
00144   WORD   4MILK
00145   WORD   5MONEY
```

```
WORD   3MUSIC
WORD   5TRASH
WORD   7REVENGE
WORD   4JUNK
WORD   7WEATHER
WORD   4FOOD
TYPE
N
  0  1  0  11
WORD   8BARMAN
WORD   3MAN
WORD   9POLICEMAN
WORD   5WOMAN
TYPE
N
  0  1  0  12
WORD   3ASH
WORD   5COUCH
TYPE
N
  0  1  0  8
WORD   3SKY
WORD   8ACTIVITY
WORD   4BODY
WORD  11FUNNY STORY
WORD   7LIBRARY
WORD   5PARTY
WORD   5BELLY
WORD   5STUDY
TYPE
 2
N
  0  1  0  3
WORD   5CIGAR
WORD   8HAVANA
WORD   4CLUE
WORD   4HINT
WORD   3BED
WORD   7FASHION
WORD   4FEAR
WORD   6SUPPER
WORD   6DINNER
TYPE
 2
N
  0  1  0  12
WORD   8BUSINESS
WORD   5LUNCH
TYPE
 2
N
  0  1  0  8
WORD   5STOGY
TYPE
 2
N
  0  1  0  1
WORD   8POLITICS
WORD   3TEA
WORD   5VODKA
WORD  11INFORMATION
WORD   4PORT
WORD   8JEALOUSY
WORD   5SHERRY
WORD   6COFFEE
WORD   4SODA
WORD   7WHISKEY
WORD   4HUME
WORD   7HEATHER
WORD   6MAGGIE
WORD   9BILLIARDS
WORD   6TENNIS
WORD   9AFFECTION
WORD   7PASSION
WORD   6MONDAY
WORD   7TUESDAY
WORD   8THURSDAY
WORD   9WEDNESDAY
WORD   9SOMETHING
WORD   8ADULTERY
WORD   3FUN
WORD  10GOOD, NIGHT
WORD   5CLIVE
WORD   5CATHY
WORD   9CATHERINE
WORD  14LADY CATHERINE
WORD   5CHESS
WORD   7CROQUET
WORD   8DR. HUME
WORD  20DR. BARTHOLO-
          MEW HUME
WORD   5JAMES
WORD   6BRIDGE
WORD   4JOHN
WORD  11JOHN BUXLEY
WORD  11LADY BUXLEY
WORD   9LADY JANE
WORD   4JANE
WORD  21LADY BUXLEY-S
          BEDROOM
WORD  11LORD EDWARD
WORD   6EDWARD
WORD   9MARION
WORD   3FLORENCE
WORD   6RONALD
WORD   6FRIDAY
WORD   6SUNDAY
WORD   8SATURDAY
WORD   8TEA TIME
WORD   6NO BODY
WORD   4WHAT
WORD   3WHO
WORD   8EVERYONE
WORD   4ONCE
WORD   6NO ONE
WORD   7SOMEONE
WORD   2IT
WORD   4THAT
TYPE
  2  9
N
  0  1  1  1
WORD   4THEY
STEM   4THEM
TYPE
N
  0  1  1  3
WORD   5KNIFE
STEM   5KNIVE
TYPE
PREP
  0  1
WORD   4WITH
WORD   2TO
WORD   4THRU
WORD   2ON
WORD   3OFF
WORD   2OF
WORD   4INTO
WORD   2IN
WORD   3FOR
WORD   2UP
WORD   2AT
WORD   5ABOUT
WORD   4FROM
WORD   2BY
WORD   6DURING
WORD   7AGAINST
WORD   4NEAR
TYPE
PREP
  0  1
ADV
  0  1
WORD   4DOWN
TYPE
ADJ
  0  1
WORD   4KIND
WORD  10UNPLEASANT
WORD   9WONDERFUL
WORD  10INTERESTED
WORD   6FRIGID
WORD   9INNOCENT
WORD   3BIG
WORD   4COLD
WORD   5MUSTY
WORD   6SMALL
WORD   8PLEASANT
WORD   6SINGLE
WORD   4WARM
WORD   8VALUABLE
WORD   5UPSET
WORD  10UNFAITHFUL
WORD   7UNAWARE
WORD   6STUPID
WORD   6STRONG
WORD   5SORRY
WORD   4RICH
WORD   6PRETTY
WORD  12PORNOGRAPHIC
WORD   4POOR
WORD   4NICE
WORD  10MISLEADING
WORD   3MAD
WORD   4LONG
WORD   4LAST
WORD   5HEAVY
WORD   8HANDSOME
WORD   5HAPPY
WORD   3FAT
WORD   8BRIGHT
WORD   8FRAGRANT
WORD   4COOL
WORD   4UGLY
WORD   5SWEET
WORD   4DEEP
WORD   4EVIL
WORD   4GOOD
WORD   6AFRAID
WORD   4DARK
WORD  13BLOOD THIRSTY
WORD   6ASLEEP
WORD   6CLEVER
WORD   5BRAVE
WORD   7IDIOTIC
WORD   4DUMB
WORD   5SMART
WORD  11NOT JEALOUS
```

```
WORD  8IMPOTENT
WORD  9OVERSEXED
WORD  10EASY GOING
WORD  9IRRITABLE
WORD  7VIOLENT
WORD  12IMPOVERISHED
WORD  10WELL TO DO
WORD  9BRILLIANT
WORD  7JEALOUS
WORD  9BEAUTIFUL
TYPE

ADJ
 0  9
PART
 0  4
V
 0  3  0  1
 0  9  0  3
WORD  5SCREW
WORD  6HAPPEN
WORD  5OFFER
WORD  6RECALL
WORD  8REMEMBER
WORD  6SIGNAL
WORD  5SHOCK
WORD  4RUIN
WORD  4YAWN
WORD  4YELL
WORD  4WANT
WORD  4WAIT
WORD  4TALK
WORD  8THREATEN
WORD  7SUSPECT
WORD  7SUGGEST
WORD  5START
WORD  7STAGGER
WORD  7SMOTHER
WORD  6SCREAM
WORD  5SCOFF
WORD  6RETURN
WORD  7PRETEND
WORD  5POINT
WORD  3PAY
WORD  3OWN
WORD  4OPEN
WORD  7MENTION
WORD  4LOOK
WORD  5LAUGH
WORD  4KICK
WORD  4JOIN
WORD  6INSULT
WORD  7INHERIT
WORD  5GROAN
WORD  5GREET
WORD  4LAST
WORD  7WHISPER
WORD  5FAINT
WORD  5ENJOY
WORD  6COMMIT
WORD  5CHEAT
WORD  6ARREST
WORD  4CALL
WORD  3ASK
WORD  6AWAKEN
WORD  6ATTACK
WORD  5COVER
WORD  8BETRAY
WORD  5ENTER
WORD  6ACCOST
WORD  9BLACKMAIL
EORD  6FOLLOW
WORD  4CALM
WORD  5FLIRT
WORD  7FLATTER
WORD  10COMPLIMENT
WORD  6GOSSIP
TYPE

ADJ
 0  5
PART
 0  10
V
 0  3  0  1
 0  5  0  5
WORD  6IGNORE
WORD  4TIRE
WORD  6AROUSE
WORD  6RELATE
WORD  6ENRAGE
WORD  8SURPRISE
WORD  4WAVE
WORD  8STRUGGLE
WORD  5SOLVE
WORD  5SNORE
WORD  5SMOKE
WORD  5SMILE
WORD  5SERVE
WORD  6SEDUDE
WORD  6RESUME
WORD  6REMOVE
WORD  7PREPARE
WORD  4MOVE
WORD  6ARRIVE
WORD  8ANNOUNCE
WORD  5ARGUE
WORD  9INTRODUCE
WORD  6INVITE
WORD  6NOTICE
WORD  5PHONE
WORD  7RECEIVE
WORD  7DECEIVE
WORD  7SHUFFLE
WORD  4LIKE
WORD  7DESPISE
WORD  4LOVE
WORD  5CURSE
WORD  6DECIDE
WORD  5GRATE
WORD  7EXAMINE
WORD  6ACCUSE
WORD  8CONVINCE
WORD  5CHOKE
WORD  8COLLAPSE
WORD  12CONGRATU-
        LATE
WORD  4FIRE
WORD  4HATE
WORD  7DISLIKE
TYPE

ADJ
 0  9
PART
 0  4
V
 0  12  0  1
 0  9  0  9
WORD  5WATCH
WORD  7DISCUSS
WORD  5DRESS
WORD  7DEPRESS
WORD  9EMBARASS
WORD  4WASH
WORD  7UNDRESS
WORD  5TOUCH
WORD  5SMASH
WORD  5SLASH
WORD  6SEARCH
WORD  7SCRATCH
WORD  4PUSH
WORD  4KISS
WORD  6CARESS
WORD  7CONFESS
TYPE

PART
 1  4
V
 0  3  0  1
 1  9  1  9
WORD  4STOP
STEM  5STOPP
WORD  4GRAB
STEM  5GRABB
WORD  4STAB
STEM  5STABB
WORD  4TRIP
STEM  5TRIPP
WORD  3RIP
STEM  4RIPP
WORD  5PANIC
STEM  6PANICK
TYPE

PART
 0  4
V
 0  3  0  1
 0  1  0  1
WORD  4HURT
WORD  4BEAT
WORD  4READ
TYPE

PART
 0  4
V
 0  3  0  1
 0  5  0  5
WORD  8OVERHEAR
WORD  4HEAR
WORD  5AGREE
TYPE

PART
 1  4
V
 0  3  0  1
 0  1  0  1
WORD  3HIT
STEM  4HITT
WORD  4QUIT
STEM  5QUITT
TYPE

PART
 0  4
V
 0  3  0  1
 1  1  1  1
WORD  5BRING
STEM  7BROUGHT
WORLD 4SING
STEM  4SANG
```

```
WORD   3EAT
ATEM   3ATE
WORD   4FEEL
STEM   4FELT
WORD   5BREAK
STEM   5BROKE
WORD   5BLEED
STEM   4BLED
WORD   4FIND
STEM   5FOUND
WORD   3SEE
STEM   3SAW
WORD   5SHOOT
STEM   4SHOT
WORD   4SINK
STEM   4SANK
WORD   5SNEAK
STEM   5SNUCK
WORD   5STEAL
STEM   5STOLE
WORD   4TELL
STEM   4TOLD
WORD   5THROW
STEM   5THREW
WORD   4WEAR
STEM   4WORE
WORD   3SAY
STEM   4SAID
WORD   4MEET
STEM   3MET
WORD   4KNOW
STEM   4KNEW
WORD   4DRAW
STEM   4DREW
WORD   4KEEP
STEM   4KEPT
WORD   5THINK
STEM   7THOUGHT
TYPE
PART
  0  4
V
  0  12  0  1
  1  1  1  1
WORD   5CATCH
STEM   6CAUGHT
TYPE
PART
  0  10
V
  0  3  0  1
  1  1  1  1
WORD   4COME
STEM   4CAME
WORD   6WRITE
STEM   5WROTE
WORD   4TAKE
STEM   4TOOK
WORD   4MAKE
STEM   4MADE
WORD   4RISE
STEM   4ROSE
WORD   5LEAVE
STEM   4LEFT
WORD   4GIVE
STEM   4GAVE
WORD   5AWAKE
STEM   5AWOKE
WORD   7FORGIVE
STEM   7FORGAVE
TYPE
ADJ
  1  5
PART
  0  4
V
  1  3  0  1
  1  5  1  5
WORD   5MARRY
STEM   6MARRIE
WORD   3TRY
STEM   4TRIE
WORD   3CRY
STEM   4CRIE
WORD   5CARRY
STEM   8CARRIE
WORD   4DENY
STEM   5DENIE
TYPE
PART
  2  4
V
  0  3  0  1
  1  1  1  1
WORD   3WIN
STEM   3WON
STEM   4WINN
WORD   3GET
STEM   3GOT
STEM   4GETT
WORD   3RUN
STEM   3RAN
STEM   4RUNN
WORD   3SIT
```



```
STEM   3SAT
STEM   4SITT
TYPE
PART
  0  10
V
  1  1  0  1
  2  1  2  1
WORD   4HAVE
STEM   3HAS
STEM   3HAD
TYPE
ADJ
  2  1
PART
  1  4
V
  0  3  0  1
  0  5  0  5
WORD   3DIE
STEM   2DY
STEM   4DEAD
TYPE
N
0  1  0  3
PART
  1  4
V
  0  3  0  1
  1  9  1  9
WORD   4PLAN
STEM   5PLANN
TYPE
N
  0  1  0  3
PART
  0  4
V
  0  3  0  1
  1  1  1  1
WORD   4FALL
STEM   4FELL
TYPE
N
  0  1  0  3
PART
  0  4
V
  0  3  0  1
  0  9  0  9
WORD   8QUESTION
WORD   4WALK
WORD   4HEAD
TYPE
  2
N
  0  1  0  1
PART
  0  4
V
  0  3  0  1
  0  9  0  9
WORD   6POISON
TYPE
PART
  0  4
V
  0  12  0  1
  1  1  1  1
ADJ
  2  1
WORD   2GO
STEM   4WENT
STEM   4GONE
TYPE
PART
  0  10
V
  0  3  0  1
  1  1  1  1
ADJ
  2  1
WORD   4HIDE
STEM   3HID
STEM   6HIDDEN
TYPE
N
  1  13  1  15
PART
  1  4
V
  0  3  0  1
  0  5  0  5
WORD   4LOVE
STEM   3LOV
TYPE
N
  0  13  0  15
ADJ
```

```
  0  9
PART
  0  4
V
  0  3  0  1
  0  9  0  9
WORD   6MURDER
WORD   4KILL
WORD   4PLAY
TYPE

N
  0  1  0  3
WORD   8*1MURDER
TYPE

N
  2  1  2  3
PART        T
  1  4
V
  0  3  0  1
  0  5  0  5
WORD   3LIE
STEM   2LY
STEM   4LIAR
TYPE

N
  0  1  0  3
ADJ
  0  7
WORD   5SMELL
WORD   5CLOUD
TYPE

N
  0  1  0  1
ADJ
  0  7
WORD   4RAIN
WORD   4LUST
WORD   4WIND
TYPE
 2
N
  0  1  0  1
ADJ
  0  7
WORD   5GREED
TYPE

N
  0  1  0  12
ADJ
  0  7
WORD   3SEX
WORD   5GROUCH
TYPE

N
  0  1  0  3
ADJ
  0  6
WORD   6FRIEND
WORD   6COWARD
TYPE

N
  0  1  0  1
ADJ
  1  7
WORD   5ANGER
STEM   4ANGR
TYPE

ADJ
  0  1
ADV
  1  6
WORD   6GENTLE
STEM   4GENT
TYPE

ADJ
  0  1
ADV
  0  6
WORD   5CLOSÉ
WORD   5USUAL
WORD   6CASUAL
WORD   7CAREFUL
WORD   5QUIET
WORD   4LOUD
WORD   4SOFT
WORD   4WILD
WORD   4WEAK
TYPE

ADJ
  0  1
ADV
  0  1
WORD   4VERY
WORD   4ALSO
WORD   6ALWAYS
WORD   5AGAIN
WORD   4WELL
```



```
WORD   4OVER
WORD   4SACK
WORD   5EARLY
WORD   4AWAY
TYPE

N
  0  1  0  3
ADJ
  1  7
WORD   3SUN
STEM   4SUNN
TYPE

PART
  0  4
V
  0  3  0  1
  1  1  1  1
N
  0  1  0  3
ADJ
  2  1
WORD   5DRINK
STEM   5DRANK
STEM   5DRUNK
TYPE

N
  0  4  0  14
WORD   9*1SMOTHER
```

7.4. *Nodes, Relations and Classes*

The input data for the nodes contains a listing of node names followed by a lexical expression list. Numbers separated by spaces indicate the following:

0 = singular
1 = Plural
2 = Singular, but definite article even on 1st occurence
3 = plural, and always associated with a definite article

Note that this information is eventually passed on to both high level and low level transformation components; other devices may also determine number at later stages.

Three pieces of information are associated with the relation input in addition to the specification of the lexical expression list. The letter codes indicate logical type:

A = attribute (normal)
T = transitive
NI = numeric intransitive: with lexical expression list
QA = quantitative attribute (no lexical expression list)
I = normal intransitive
NA = numerical attribute (with lexical expression list)

'Transitive' and 'intransitive' here refer to logical transitivity as opposed to syntactic transitivity. E.g. "if A R B and BRC, then ARC" implies that R is transitive. The first number following the letter code represents the relation type:

3 = general class
2 = attribute class
4 = prepositional class
6 = adverbial type
5 = possessive

These are *not* grammar codes, but rather devices for speeding up selecting of rules for generation. The designations as preposition, adverb, etc. are arbitrary; they actually represent a higher order semantic classification. The third number represents an additional subclass marking for partition of the class specified by the 1st digit.

The class listing contains the class names followed by a listing of elements; the listing may be empty or include both nodes and other class names.

```
GXGT CS*0BRL.ABSSFC
    $LIMITS       START = 19W3D10H* END = 20W2D14H;
    %
SILLRDRM 0 = 'BILLIARD ROOM';
    %    ******************   NODES   ******************
    %
    $NODES;
    THAT 0 = 'THAT';
    MX 0 = 'THAT';
    LST 0 = 'THAT';
    ULST 0 = 'THAT';
    ACCUSATION 2 = 'ACCUSATION';
    ACTIVITIES 1 = 'ACTIVITY';
    ADULTRY 0 = 'ADULTERY';
    ASHES 1 = 'ASH';
    BATHROOM 2 = 'BATHROOM';
    BED 0 = 'BED';
    BEDROOM 2 = 'BEDROOM';
    BILLIARDS 0 = 'BILLIARDS';
    BILLRDRM 0 = 'BILLIARD ROOM';
    BLOOD 2 = 'BLOOD' 'GORE';
    BOOK 0 = 'BOOK' 'PAPERBACK';
    BOOKS 1 = 'BOOK';
    BOTTLE 2 = 'BOTTLE';
    BREAKFAST 0 = 'BREAKFAST';
    BRIDGE 0 = 'BRIDGE';
    BUSINESS 0 = 'BUSINESS';
    BUTLER 0 = 'BUTLER' 'CLIVE';
    BUTTON 0 = 'BUTTON';
    CANDLHOLD 0 = 'CANDLE HOLDER';
    CARDGAME 2 = 'CARD CAME';
    CATHY 0 = 'CATHY' 'CATHERINE' 'LADY CATHERINE';
    CHESS 0 = 'CHESS';
    CIGARS 1 = 'CIGAR' 'STOGY' 'HAVANA';
    CLUES 1 = 'CLUE' 'HINT';
    CLUE1 0 = 'CLUE';
    CLUE2 0 = 'CLUE';
    COFFEE O = 'COFFEE';
    COMPANION 0 = 'COMPANION';
    CONVERSATION 0 = 'CONVERSATION';
    CONVERTNS 1 = 'CONVERSATION';
    COOK 0 = 'COOK' 'MAGGIE';
    CORPSE 2 = 'CORPSE' 'BODY';
    COUCH 0 = 'COUCH' 'DIVAN' 'DAVENPORT';
    CRIME 2 = 'CRIME';
```



```
    CROQGAME 2 = 'CROQUET GAME';
    CROQUET 0 = 'CROQUET';
    DAWN 2 = 'DAWN' 'SUNRISE';
    DAY 2 = 'DAY';
    DINER 0 = 'LUNCH';
    DININGRM 0 = 'DINING ROOM';
    DOOR 2 = 'DOOR';
    DRAWER 0 = 'DRAWER';
    DRAWERS 1 = 'DRAWER';
    DRAWINGRM 2 = 'DRAWING ROOM';
    DRHUME 0 = 'DR. HUME' 'DR. BARTHOLOMEW HUME' 'HUME';
    ENCORE 0 = 'ENCORE';
    EVENING 2 = 'EVENING';
    EVERYONE 0 = 'EVERYONE';
    FALL2 0 = 'FALL';
    FASHION 2 = 'FASHION';
    FEAR 0 = 'FEAR';
    FLOWERS 3 = 'FLOWER';
    FOOD 2 = 'FOOD';
    FOOTPRINT 0 = 'FOOTPRINT';
    FPRINTS 3 = 'FINGERPRINT';
    FRIDAY 0 = 'FRIDAY';
    GAME 0 = 'GAME';
    GARDEN 0 = 'GARDEN';
    GOODNIGHT 0 = 'GOOD NIGHT';
    GOODTIME 0 = 'FUN';
    GREED 0 = 'GREED';
    GREENHS 0 = 'GREEN HOUSE';
    GUN 0 = 'GUN' 'PISTOL';
    HAIR 0 = 'HAIR';
    HALL 2 = 'HALL' 'CORRIDOR';
    HANDKERCHIEF 0 = 'HANDKERCHIEF';
    HANDS 1 = 'HAND';
    HEAD 2 = 'HEAD';
    HOUSE 0 = 'HOUSE';
    INFORMATION 2 = 'INFORMATION';
    INSPECTOR 0 = 'INSPECTOR' 'DETECTIVE';
    JAIL 2 = 'JAIL';
    JAMES 0 = 'JAMES';
    JAW 2 = 'JAW' 'CHIN';
    JEALOUSY 0 = 'JEALOUSY';
    JEWELS 1 = 'JEWEL' 'JEWELRY';
    JOHNBUX 0 = 'JOHN' 'JOHN BUXLEY';
    JOKE 0 = 'JOKE' 'FUNNY STORY';
    KITTCHEN 2 = 'KITCHEN';
    KNIFE 0 = 'KNIFE' 'DAGGER';
    LADYBUX 0 = 'LADY BUXLEY';
    LADYJANE 0 = 'JANE' 'LADY JANE';
    LBROOM 2 = 'LADY BUXLEY S BEDROOM';
    LIAR 0 = 'LIE';
    LIBRARY 0 = 'LIBRARY';
    LORDED 0 = 'LORD EDWARD' 'EDWARD';
    LOVER 0 = 'LOVE';
    MAID 0 = 'MAID' 'HEATHER';
```

```
MARION 0 = 'MARION';
MEN 3 = 'MAN';
MILK 0 = 'MILK';
MISTAKE 0 = 'MISTAKE' 'ERROR';
MONEY 2 = 'MONEY';
MOTIVE2 2 = 'MOTIVE';
MURDER 2 = '*1MURDER';
MURDERER 2 = 'MURDER' 'KILL';
MUSIC 0 = 'MUSIC';
NECK 2 = 'NECK';
NEPHEW 0 = 'NEPHEW';
NIGHTGOWN 0 = 'NIGHTGOWN';
NOONE 0 = 'NO ONE' 'NOBODY';
NOSE 2 = 'NOSE';
NOTE 0 = 'NOTE';
NOVEL 0 = 'NOVEL';
NURSE 0 = 'FLORENCE';
ONCE 0 = 'ONCE';
OTHERS 3 = 'OTHER';
PAPERWT 0 = 'PAPERWEIGHT';
PARLOR 0 = 'PARLOR';
PARTNER2 0 = 'PARTNER';
PARTY 0= 'PARTY';
PIANO 2 = 'PIANO';
PILLOW 0 = 'PILLOW' 'CUSHION';
PLACE 0 = 'PLACE';
PLAN 2 = 'PLAN';
PLAYER2 0 = 'PLAY';
POISON 2 = 'POISON';
POLICE 3 = 'POLICEMAN' 'COP';
POLITICS 0 = 'POLITICS';
PORT 0 = 'PORT';
QUESTNS 1 = 'QUESTION';
REVENGE 0 = 'REVENGE';
RONALD 0 = 'RONALD';
ROOM 2 = 'ROOM';
SATURDAY 0 = 'SATURDAY';
SECRETPASSAGE 0 = 'SECRET PASSAGE';
SERVANTS 3 = 'SERVANT';
SHERRY 0 = 'SHERRY';
SHIRT 0 = 'SHIRT';
SHOE 0 = 'SHOE';
SKY 2 = 'KY';
SMOTHERING 0 = '*1SMOTHER';
SOMEONE 0 = 'SOMEONE';
STAIN 0 = 'STAIN';
STAIRS 1 = 'STAIR';
STOMACH 2 = 'STOMACH' 'BELLY';
STRAND 0 = 'STRAND';
STRANDOFHAIR 0 = ;
STUDY 2 = 'STUDY';
SUN 2 = 'SUN';
SUNDAY 0 = 'SUNDAY';
SUPPER 0 = 'SUPPER' 'DINNER';
```



```
TEA 2 = 'TEA';
TEATIME 0 = 'TEA TIME';
TENNIS 0 = 'TENNIS';
TENNISCOURT 2 = 'TENNIS COURT';
THEY 3 = 'THEY';
THREAD 0 = 'THREAD';
TIME 2 = 'TIME';
TRASH 0 = 'TRASH' 'JUNK';
TRUTH 2 = 'TRUTH';
VASE 0 = 'VASE';
VODKA 0 = 'VODKA';
WALK 0 = 'WALK';
WEATHER 2 = 'WEATHER';
WEEKEND 0 = 'WEEKEND';
WHAT 0 = 'WHAT';
WHISKY 0 = 'WHISKEY';
WHO 0 = 'WHO';
WINDOW 0 = 'WINDOW';
WOMEN 3 = 'WOMAN';
YARD 2 = 'YARD';
AFFAIR 2 = 'AFFAIR';
AFFEKTION1 0 = 'AFFECTION';
BAR 2 = 'BAR';
BARMAN 2 = 'BARMAN';
BEATLES 3 = 'BEATLE';
CANTEEN 0 = 'PUB';
CARDS 3 = 'CARD';
CHANGE 0 = 'CHANGE';
CLUE 0 = 'CLUB';
COOKIES 1 = 'COOKIE';
CORNER 0 = 'CORNER';
DETAILS 1 = 'DETAIL';
DRINK1 0 = 'DRINK';
DRINKS 1 = 'DRINK';
FRIEND 0 = 'FRIEND';
HOTEL 0 = 'HOTEL';
INTERMISSION 0 = 'INTERMISSION';
INVITATION 0 = 'INVITATION';
IT 0 = 'IT';
MONDAY 0 = 'MONDAY';
MOVIE 0 = 'MOVIE';
MORNING 2 = 'MORNING';
PARK 0 = 'PARK';
PASSION 0 = 'PASSION';
ROCKS 3 = 'ROCK';
SODA 0 = 'SODA';
SOMETHING 0 = 'SOMETHING';
SONG 0 = 'SONG';
TELEPHONE 0 = 'TELEPHONE';
THEATRE 0 = 'THEATER';
THURSDAY 0 = 'THURSDAY';
TUESDAY 0 = 'TUESDAY';
WEDNESDAY 0 = 'WEDNESDAY';
%
```

```
%    ******************   RELATIONS   ******************
%
%RELATIONS;
GG NA (10) 2 0 = 'THAT';
ACCOST I 3 0 = 'ACCOST';
ACCUSE I 3 0 = 'ACCUSE';
AFFECTION NI (3) 3 0 = 'HATE'/-2.5/'DISLIKE'/-0.5/'LIKE'/2.5/'LOVE';
AGREE A 3 0 = 'AGREE';
AGREEWITH I 3 0 = ;
ANNOUNCE I 3 0 = 'ANNOUNCE';
ARGUE A 3 0 = 'ARGUE';
ARGUWITH I 3 0 =;
ARREST I 3 0 = 'ARREST';
ARRIVE A 3 0 = 'ARRIVE';
ASK I 3 0 = 'ASK';
ASKFOR I 3 0 = ;
ATTACK I 3 0 = 'ATTACK';
AWAKE A 3 0 = 'AWAKE';
AWAKEN A 3 0 = 'AWAKEN';
BEAT I 3 0 = 'BEAT';
BETRAY I 3 O = 'BETRAY';
BLACKMATL I 3 0 = 'BLACKMAIL';
BLEED A 3 0 = 'BLEED';
BREAK A 3 0 = 'BREAK';
CALL I 3 0 = 'CALL';
CALM I 3 0 = 'CALM';
CARESS I 3 0 = 'CARESS';
CARRY I 3 0 = 'CARRY';
CATCH I 3 0 = 'CATCH';
CHEAT A 3 0 = 'CHEAT';
CHOKE I 3 0 = 'CHOKE';
COLLAPSE A 3 0 = 'COLLAPSE';
COMMIT I 3 0 = 'COMMIT';
COMPLIMENT I 3 0 = 'COMPLIMENT';
CONFESS A 3 0 = 'CONFESS';
CONGRATU I 3 0= 'CONGRATULATE';
CONVINCE I 3 0 = 'CONVINCE';
COUNT QA (8) 2 0 = ;
COVER A 3 0 = 'COVER';
COVERWITH I 2 0 = ;
CRY A 3 0 = 'CRY';
CURSE I 3 0 = 'CURSE';
DECIDE A 3 0 = 'DECIDE';
DECIEVE I 3 0 = 'DECEIVE';
DENY I 3 0 = 'DENY';
DESPISE I 3 0 = 'DESPISE';
DIE A 3 0 = 'DIE';
DISCUSS I 3 0 = 'DISCUSS';
DRAW I 3 0 = 'DRAW';
DRINK I 3 0 = 'DRINK';
EAT I 3 0 = 'EAT';
ENJOY I 3 0 = 'ENJOY';
ENTER I 3 0 = 'ENTER';
EXAMINE I 3 0 = 'EXAMINE';
FAINT A 3 0 = 'FAINT';
FALL A 3 0 = 'FALL';
FALLDOWN I 3 0 = ;
FEEL I 3 0 = 'FEEL';
FEELNO A 3 0 = 'FEEL';
FEELWELL A 3 0 = ;
FIND I 3 0 = 'FIND';
FIRE A 3 0 = 'FIRE';
FLATTER I 3 0 = 'FLATTER';
FLIRT A 3 0 = 'FLIRT';
FLIRTWITH I 3 0 = ;
FOLLOW I 3 0 = 'FOLLOW';
FORGIVE I 3 0 = 'FORGIVE';
FUCK I 3 0 = 'SCREW' 'SEDUCE';
GET I 3 0 = 'GET';
GETDRESS A 3 0 = ;
GETUP A 3 0 = ;
GIVE I 3 0 = 'GIVE';
GO A 3 1 = 'GO';
GOFOR I 3 0 = ;
GOSSIP A 3 0 = 'GOSSIP';
GOSSIPNO I 3 0 = 'GOSSIP';
GOTO I 3 0 = ;
GRAB I 3 0 = 'GRAB';
GRABFOR I 3 0 = ;
GRATE A 3 0 = 'GRATE';
GREET I 3 0 = 'GREET';
GROAN A 3 0 = 'GROAN';
HAVE I 3 0 = 'HAVE';
HEADNO A 3 0 = 'HEAD';
HEADFOR I 3 0 = ;
HEAR I 3 0 = 'HEAR';
HIDE I 3 0 = 'HIDE';
HIDENO A 3 0 = 'HIDE';
HIT I 3 0 = 'HIT';
IGNORE I 3 0 = 'IGNORE';
INHERIT I 3 0 = 'INHERIT';
INSULT I 3 0 = 'INSULT';
IS I 3 0 = 'BE';
JOIN I 3 0 = 'JOIN';
KEEP I 3 1 = 'KEEP';
KICK I 3 0 = 'KICK';
KILL I 3 0 = 'KILL';
KILLEDBY I 3 0 = ;
KISS I 3 0 = 'KISS';
KNOW I 3 0 = 'KNOW';
LAST A 3 0 = 'LAST';
LAUGH A 3 0 = 'LAUGH';
LEAVE I 3 0 = 'LEAVE';
LOOK A 3 0 = 'LOOK';
LOOKFOR I 3 0 = ;
LOOKTHRU I 3 0 = ;
LOOKWELL A 3 0 = ;
```

```
MAKE I 3 0 = 'MAKE';
MENTION I 3 0 = 'MENTION';
MEET I 3 0 = 'MEET';
MOVE A 3 0 = 'MOVE';
OPEN I 3 0 = 'OPEN';
OVERHEAR I 3 0 = 'OVERHEAR';
OWN I 3 0 = 'OWN';
PANIC A 3 0 = 'PANIC';
PAY I 3 0 = 'PAY';
PLANNO A 3 0 = 'PLAN';
PLAY I 3 0 = 'PLAY';
POINT I 3 0 = 'POINT';
POISONS I 3 0 = 'POISON';
POS I 5 0 = 'BE';
PREPARE I 3 0 = 'PREPARE';
PRETEND A 3 0 = 'PRETEND';
PUSH I 3 0 = 'PUSH';
QUESTION I 3 0 = 'QUESTION';
GUIT A 3 0 = 'GUIT';
READ I 3 0 = 'READ';
RELATEDTO I 3 0 = ;
REMOVE I 3 0 = 'REMOVE';
RESUME I 3 1 = 'RESUME';
RETURN I 3 0 = 'RETURN';
RETUPNIO I 3 0 = ;
RIP I 3 0 = 'RIP';
RIPFROM I 3 0 = ;
RISE A 3 0 = 'RISE';
RUN A 3 0 = 'RUN';
SAY I 3 0 = 'SAY';
SAYTO I 3 0 = ;
SCOFF A 3 0 = 'SCOFF';
SCRATCH I 3 0= 'SCRATCH';
SCREAN A 3 0 = 'SCREAM';
SEARCH I 3 0 = 'SEARCH';
SEDUCE I 3 0 = 'SEDUCE';
SEE I 3 0 = 'SEE';
SERVE I 3 0 = 'SERVE';
SHOOT I 3 0 = 'SHOOT';
SHOOTAT I 3 0 = ;
SINK A 3 0 = 'SINK';
SIT A 3 0 = 'SIT';
SLASH I 3 0 = 'SLASH';
SMASH I 3 0 = 'SMASH';
SMILE A 3 0 = 'SMILE';
SMILEAT I 3 0 = ;
SMCKE I 3 0 = 'SMOKE';
SMOTHER I 3 0 = 'SMOTHER';
SNEAK A 3 0 = 'SNEAK';
SNORE A 3 0 = 'SNORE';
SOLVE I 3 0 = 'SOLVE';
STAB I 3 0 = 'STAB';
STAGGER A 3 0 = 'STAGGER';
START I 3 0 = 'START';
```



```
STARTNO A 3 1 = 'START';
STEAL I 3 0 = 'STEAL';
STOP I 3 0 = 'STOP';
STOPNO A 3 1 = 'STOP';
STRUGGLE A 3 0 = 'STRUGGLE';
STRUGLWITH I 3 0 = ;
SUGGEST I 3 0 = 'SUGGEST';
SURPRISE I 3 0 = 'SURPRISE';
SUSPECT I 3 0 = 'SUSPECT';
TAKE I 3 0 = 'TAKE';
TALK A 3 0 = 'TALK';
TALKABOUT I 3 0 = ;
TALKWITH I 3 0 = ;
TELL I 3 0 = 'TELL';
THINK I 3 0 = 'THINK';
THREATEN A 3 0 = 'THREATEN';
THROW I 3 0 = 'THROW';
THROWAWAY I 3 0 = ;
TOUCH I 3 0 = 'TOUCH';
TRIP I 3 0 = 'TRIP';
TRY A 3 0 = 'TRY';
UNDRESS A 3 0 = 'UNDRESS';
WAIT A 3 0 = 'WAIT';
WAITFOR I 3 0 = ;
WALKNO A 3 0 = 'WALK';
WALKIN I 3 0 = ;
WANT I 3 0 = 'WANT';
WANTNO A 3 0 = 'WANT';
WASH A 3 0 = 'WASH';
WAVE I 3 0 = 'WAVE';
WEAR I 3 0 = 'WEAR';
WHYKILL QI (15) 3 0 = ;
WISPER A 3 0 = 'WHISPER';
WISPERTO I 3 0 = ;
WRITE I 3 0 = 'WRITE';
YELL A 3 0 = 'YELL';
YELLAT I 3 0 = ;
YAWN A 3 0 = 'YAWN';
BRING I 3 0 = 'BRING';
COME I 3 0 = 'COME';
COMEWITH I 3 0 = ;
FORCAST QA (13) 2 0 = ;
GOZZIP I 3 0 = ;
HAPPENED A 3 0 = 'HAPPEN';
INTRODUCE I 3 0 = 'INTRODUCE';
INVITE I 3 0 = 'INVITE';
LIKE I 3 0 = 'LIKE';
NOTICE I 3 0 = 'NOTICE';
NUMBER QA (6) 2 0 = ;
OFFER I 3 0 = 'OFFER';
PHONE I 3 0 = 'PHONE';
RECALL I 3 0 = 'RECALL';
RECEIVE I 3 0 = 'RECEIVE';
REMEMBER I 3 0 = 'REMEMBER';
```

```
RUNINTO I 3 0 = ;
SHUFFLE I 3 0 = 'SHUFFLE';
SIGNAL I 3 0 = 'SIGNAL';
SING I 3 0 = 'SING';
SITDOWN A 3 0 = ;
WATCH I 3 0 = 'WATCH';
WIN I 3 0 = 'WIN';
%
%     *****************  ADJ  *****************
%
AFRAID A 2 0 = 'AFRAID';
ANGRY A 2 0 = 'ANGER';
AROUSED A 2 0 = 'AROUSE';
ASLEEP A 2 0 = 'ASLEEP';
ATTRACTIVE NA (3) 2 0 = 'UGLY'/-0.5/'PRETTY'/1.5/'BEAUTIFUL';
BEAUTIFUL A 2 0 = 'BEAUTIFUL';
BIG A 2 0 = 'BIG';
BLOODTHIRSTY A 2 0 = 'BLOOD THIRSTY';
BRIGHT A 2 0 = 'BRIGHT';
CLEVER A 2 0 = 'CLEVER';
CLOUDY A 2 0 = 'CLOUD';
COLD A 2 0 = 'COLD';
COOL A 2 0 = 'COOL';
COURAGE NA (3) 2 0 = 'COWARD' /0.5/'BRAVE';
DARK A 2 0 = 'DARK';
DEAD A 2 0 = 'DE';
DEEP A 2 0 = 'DEEP';
DEPRESSED A 2 0 = 'DEPRESS';
DRESSED A 2 0 = 'DRESS';
DRUNK A 2 0 = 'DRINK';
DUMB A 2 0 = 'DUMB';
EARLY A 2 0 = 'EARLY';
EMBARAED A 2 0 = 'EMBARASS';
ENRACED A 2 0 = 'ENRAGE';
EVIL A 2 0 = 'EVIL';
FAT A 2 0 = 'FAT';
FRAGRENT A 2 0 = 'FRAGRANT';
FRAID I 2 0 = 'AFRAID';
FRIENDLY A 2 0 = 'FRIEND';
GOOD NA (3) 2 0 = 'EVIL'/-3/'UNPLEASANT'/-1/'NICE'/0/'GOOD'/1/
        'KIND'/2/'WONDERFUL';
GOODS A 2 0 = 'GOOD';
GONE A 2 0 = 'GO';
GREEDY A 2 0 = 'GREED';
GROUCHY A 2 0 = 'GROUCH';
HAPPY A 2 0 = 'HAPPY';
HANDSOME NA (3) 2 0 = 'UGLY'/0.5/'HANDSOME';
HEAVY A 2 0 = 'HEAVY';
HIDDEN A 2 0 = 'HIDE';
HURT A 2 0 = 'HURT';
INTERESTED A 2 0 = 'INTERESTED';
INNOCENT A 2 0 = 'INNOCENT';
IQ NA (150) 2 0 = 'IDIOTIC'/75/'STUPID' 'DUMS'/99/'SMART'/127/
                      'BRILLIANT';
```



```
JEALOUS NA (3) 2 0 = 'NOT JEALOUS'/0.5/'JEALOUS';
KILLED A 2 0 = 'KILL';
LONG A 2 0 = 'LONG';
LOUDLY A 2 0 = 'LOUD';
MAD A 2 0 = 'MAD';
MADAT I 2 0 = ;
MARRIED A 2 0 = 'MARRY';
MISLEADING A 2 0 = 'MISLEADING';
MUSTY A 2 0 = 'MUSTY';
NICE A 2 0 = 'NICE';
PLEASANT A 2 0 = 'PLEASANT';
POOR A 2 0 = 'POOR';
PORNOG A 2 0 = 'PORNOGRAPHIC';
PRETTY A 2 0 = 'PRETTY';
RAYNY A 2 0 = 'RAIN';
RELATED A 2 0 = 'RELATE';
RICH A 2 0 = 'RICH';
RUINED A 2 0 = 'RUIN';
SEXDRIVE NA (4) 2 0 = 'FRIGID'/-4/'IMPOTENT'/0.5/'LUST'/1.8/
         'OVERSEXED';
SEXY A 2 0 = 'SEX';
SINGLE A 2 0 = 'SINGLE';
SHOCKED A 2 0 = 'SHOCK';
SMALL A 2 0 = 'SMALL';
SMELLY A 2 0 = 'SMELL';
SORRY A 2 0 = 'SORRY';
STRONG A 2 0 = 'STRONG';
STUPID A 2 0 = 'STUPID';
SUNNY A 2 0 = 'SUN';
SURPRISE A 2 0 = 'SURPRISE';
TIRED A 2 0 = 'TIRE';
UNAWARE A 2 0 = 'UNAWARE';
UNFAITHFUL A 2 0 = 'UNFAITHFUL';
UPSET A 2 0 = 'UPSET';
VALUABLE A 2 0 = 'VALUABLE';
VIOLENT NA (3) 2 0 = 'EASY GOING'/0.5/'IRRITABLE'/0.5/'VIOLENT';
WARM A 2 0 = 'WARM';
WEAK A 2 0 = 'WEAK';
WEALTH NA (3) 2 0 = 'IMPOVERISHED'/-2.5/'POOR'/0.5/WELL TO DO'/2.5/
                              'RICH';
WINDY A 2 0 = 'WIND';
XX A 2 0 = 'WELL';
%
%     *****************  PREP  *****************
%
ABOUT I 4 0 = 'ABOUT';
AGAINST I 4 0 = 'AGAINST';
AT I 4 0 = 'AT';
BY I 4 0 = 'BY';
DOWN I 4 0 = 'DOWN';
DURING I 4 0 = 'DURING';
FOR I 4 0 = 'FOR';
FROM I 4 0 = 'FROM';
IN I 4 0 = 'IN';
```

```
NEAR I 4 0 = 'NEAR';
OF I 4 0 = 'OF';
OFF I 4 0 = 'OFF';
ON I 4 0 = 'ON';
THRU I 4 0 = 'THRU';
TO I 4 0 = 'TO'
UP A 4 0 = 'UP';
WITH I 4 0 = 'WITH';
INTO I 4 0 = 'INTO';
%
%     ***************** ADV *****************
%
AGAIN A 5 0 = 'AGAIN';
ALSO A 6 1 = 'ALSO';
ALWAYS A 6 1 = 'ALWAYS';
AWAY I 6 0 = 'AWAY';
BACK A 6 0 = 'BACK';
CAREFULLY A 6 0 = 'CAREFUL';
CASUALLY A 6 1 = 'CASUAL';
CLOSELY A 6 0 = 'CLOSE';
DOWNNO A 6 0 = 'DOWN';
GENTLY A 6 1 = 'GENTLE';
OVER A 2 0 = 'OVER';
QUIETLY A 6 0 = 'QUIET';
SOFTLY A 6 0 = 'SOFT';
USUALLY A 6 0 = 'USUAL';
VERY A 6 1 = 'VERY';
WEAKLY A 6 0 = 'WEAK';
WELL A 6 0 = 'WELL';
WILDLY A 6 0 = 'WILD';
%
%     ***************** CLASSES *****************
%
$CLASSES;
BRIDGER = ;
CHASER(BILLIARUS) = BILLRORM;
CHASER(CHESS) = STUDY;
CHASER(TENNIS) = TENNISCOURT;
CHESSER = ;
CONVERSING = ;
CRCQER = ;
```

7.5. *Network and Simulation Rule Plot Specification*

The specification of the network includes the assignment of all initial conditions: numerical attributes, lexical triples, semantic triples, and a listing of relations which are logically mutually exclusive for automatic maintenance of logical consistency. This initialization of starting conditions is part of the first time frame of the simulation. Comments on the significance of groups of rules appear indented between them. The following is a small sample of the total data.


```
DETECT = DRHUME;
DRNK = COFFEE SHERRY WHISKY PORT VODKA;
ENEMY( ) = ;
EVIDENCE = ;
FEMALE = LADYBUX NURSE MAID COOK CATHY LADYJANE MARION;
FIGHTER = ;
FINDER =;
GAMES = CHESS TENNIS BILLIARDS;
GUESTS = LADYBUX NURSE;
HEAVYOBJ = PAPERWT CANDLHOLD;
INTERUPT = ;
INVITED = JOHNBUY DRHUME JAMES MARION RONALD CATHY LORDED
LADYJANE;
KILLER = ;
KLUES = STRANDOFHAIR FOOTPRINT THREAD HANDKERCHIEF STAIN
ASHES
        SECRETPASSAGE;
LOC = HALL PARLOR DRAWINGRM GREENHS LIBRARY DININGRM STAIRS
LBROOM
GARDEN BATHROOM TENNISCOURT BILLRDRM YARD;
LOSER = ;
MALE = BUTLER DRHUME RONALD JOHNBUX JAMES LORDED;
MEAL = ;
MOTIVE = ;
MROOM = ;
OBJECT = BOOK VASE SHOE HEAVYOBJ;
PARTNER(JAMES) = RONALD;
PARTNER(RONALD) = JAMES;
PLASE = PARK MOVIE HOTEL GARDEN TENNISCOURT;
PLAYED = ;
PLAYER = ;
POSKILLR = ;
POSVICTM( ) = ;
READER = ;
RELATIVE(JOHNBUX) = LADYBUX;
RELATIVE(LADYBUX) = JOHNBUX;
RELATIVE(BUTLER) = JAMES;
RELATIV(JAMES) = BUTLER;
PENDEVOUS = ;
RENDM = ;
RETIRED = ;
RETIRING = ;
SERVANT = COOK BUTLER MAID;
SPOUSE(CATHY) = RONALD;
SPOUSE(JAMES) = MARION;
SPOUSE(LADYJANE) = LORDED;
SPOUSE(LORDED) = LADYJANE;
SPOUSE(MARION) = JAMES;
SPOUSE(RONALD) = CATHY;
TOPIC = FASHION POLITICS TENNIS BUSINESS THEATRE MUSIC FLOWERS
BOOKS
        CHESS;
VICTIM = ;
WEAPON = ;
```

```
TALKING = GUESTS;
TEMP = ;
WAKE = GUESTS INVITED;
WANTED ( ) =;
WINNER = ;
PEOPLE = WAKE SERVANT;
%
                    % *****************  NET-
WORK  *****************
    %
    %               INITIALIZE PERSONALITY CHARACTERISTICS NOT TO
    %               BE DESCRIBED IN OUTPUT.
    $NETWORK:
                    LADYBUX COURAGE = 2,
                    LADYBUX VIOLENT = 1,
                    JOHNBUX IQ = 100,
                    JOHNBUX COURAGE = -1,
                    DRHUME WEALTH = -2,
                    DRHUME VIOLENT = 3,
                    DRHUME AFFECTION = -1 LORDED,
                    DRHUME AFFECTION = -1 RONALD,
                    DRHUME AFFECTION = 1 LADYBUX,
                    LORDED IQ = 100,
                    LORDED COURAGE = 1,
                    LORDED MARRIED,
                    LORDED AFFECTION = 1 DRHUME,
                    LADYJANE WEALTH = 3,
                    LADYJANE IQ = 100,
                    LADYJANE VIOLENT = -1,
                    LADYJANE MARRIED,
                    RONALD IQ = 110,
                    RONALD VIOLENT = -1,
                    RONALD JEALOUS = 1,RONALD MARRIED,
                    CATHY IQ = 100,
                    CATHY WEALTH = 2,
                    CATHY MARRIED,
                    JAMES COURASE = 2,
                    JAMES MARRIED,
                    MARION COURAGE = 2,
                    MARION MARRIED,
                    BUTLER VIOLENT = -1,
                    NURSE IQ = 100,
                    MAID COURAGE = -2,
                    COOK IQ = 100,
                    COOK COURAGE = 2,
                    SUN FORCAST = 15;
    %
    %               DEFINE COMPOUND RELATIONS IN TERMS OF
    %               INDIVIDUAL RELATIONS.
    %
    *LEXTRP (GO FOR) TO GOFOR;
    *LEXTRP (MAD AT) TO MADAT;
    *LEXTRP (GET UP) TO GETUP;
    *LEXTRP (GAME OF CROQUET) TO CROQGAME;
```



```
*LEXTRP (YELL AT) TO YELLAT;
*LEXTRP (FEEL WELL) TO FEELWELL;
*LEXTRP (FLIRT WITH) TO FLIRTWITH;
*LEXTRP (COVER WITH) TO COVERWITH;
*LEXTRP (GRAB FOR) TO GRABFOR;
*LEXTRP (ASK FOR) TO ASKFOR;
*LEXTRP (FALL DOWN) TO FALLDOWN;
*LEXTRP (WISPER TO) TO WISPERTO;
*LEXTRP (WALK IN) TO WALKIN;
$RULS:              *MOVE BRIDGER TO TEMP;
$LOOP:              P.PICK(TEMP);
$RULE:              F(L1)
                    *REMOVE P FROM TEMP,
                    *INSERT (P CHEAT)(CHEAT AT BRIDGE);
     :              - (P GOOD)/3;
$LOOP:              Q.PICK(TEMP);
$RULE:              T(END)
                    *INSERT (Q SEE THAT)(P CHEAT),
                    Q ACCUSE P,
                    *INSERT (P SAY THAT)(Q IS LIAR);
-10,.25:            (P EGL SPOUSE(Q));
     :              - (P AFFECTION Q)/8;
$ENDLOOP:
$ENDLOOP:
$RULE L1:           CARDGAME GOODZ;
-10,.08:            (CARDGAME GOODZ);
$SWITCH:            F($ENDGROUP);
-05.-10:            NUM(TALKING) GT O;
$LOOP:              P.PICK(BRIDGER);
$LOOP:              Q.PICK(TALKING);
$RULE:              ($ENDGROUP)
                    *REMOVE P FROM BRIDGER,
                    *ADD P TO TALKING,
                    *REMOVE Q FROM TALKING,
                    *ADD Q TO BRIDGER,
                    P LEAVE CARDGAME,
                    *INSERT (Q TAKE PLACE)(PLACE POS P);
$ENDLOOP:
$ENDLOOP:
$RULE END:          BRIDGER NOT PLAY BRIDGE,
                    THEY NOT PLAY BRIDGE,
                    *DISABLE BRIDGING,
                    *ADD BRIDGER TO TALKING,
                    *ERASE BRIDGER,
                    CARDGAME OVER;
$ENDGROUP;
%
%
$GROUP CROQING:     10M/OFF;
$SWITCH.C:          T(END);
10,0:               (ACTIVITIES EQL INTERUPT);
     :              .1;
$LOOP:              P.CROQER;
$SWITCH:            T(END);
```

```
10,-10:              (P EQL INTERUPT);
$ENDLOOP:
$RULE:               CROQGAME GOOD2;
-10,.08:             (CROQGAME GOOD2);
$LOOP:               P.PICK(CROQER);
$RULE:               *REMOVE P FROM CROQER,
                     *ADD P TO TALKING,
                     P NOT GOTO YARD.
                     P NOT PLAY CROQUET,
                     P WEAVE CROQGAME;
-10,.15:             NUM(CROQER) EQ 2;
$ENDLOOP:
$LOOP:                  P.PICK(TALKING);
$RULE:                  ($ENDGROUP)
                        *REMOVE P FROM TALKING,
                        *ADD P TO CROQER,
                        P GOTO YARD,
                        P JOIN CROQGAME;
-10,.15:                NUM(CROQER) EQ 6;
$ENDLOOP;
$RULE END:              CROQER NOT GOTO YARD,
                        THEY NOT GOTO YARD,
                        CROQER NOT PLAY CROQUET,
                        THEY NOT PLAY CROQUET,
                        *DISABLE CROQING.
                        *ADD CROQER TO TALKING,
                        *ERASE CROQER,
                        CROQGAME OVER;
*ENDGROUP;
%
% **************** SECTION D ****************
%
%
%          SECTION D CONTROLS THE LOVERS' TRYSTS. THESE
%          MAY OCCUR ONCE IN THE AFTERNOON AND/OR ONCE AT
%          NIGHT AFTER EVERYONE HAS GONE TO BED.
%
%
%          GROUP STARTWALK CHOOSES THE PARTICIPANTS FOR A
%          TRYST AND HAS THEM STOP THEIR CURRENT
%          ACTIVITIES. A TRYST WILL OCCUR ONLY IF THERE
%          IS AT LEAST ONE COUPLE READY TO GO. IF THERE
%
           IS MORE THAN ONE POSSIBLE COUPLE. A SINGLE
%
           COUPLE IS SELECTED RANDOMLY.
%
           THESE CANDIDATE COUPLES ARE ONES WHO HAVE
%
           ENGAGED IN PREVIOUS FLIRTATIONS.
$GROUP STARTWALK:       1H/OFF;
$RULE:                  T($ENDGROUP)
                        *DISABLE STARTWALK;
10,-10:                 NUM(RENDEVOUS) EQ 0;
```



```
$LOOP:               P.PICK(RENDEVOUS);
$RULE:               *DISABLE STARTWALK.
                     *REMOVE P FROM RENDEVOUS,
                     *MOVE P TO REMDM,
                     *ADD P TO INTERUPT,
                     *ADD WANTED(P) TO INTERUPT,
                     *ADD CHASER(P) TO INTERUPT,
                     *ENABLE REND IN 20M;
$RULE:               T($ENDGROUP)
                     *ENABLE AFTERN IN 15M;
10,-10:              CLOCK EQ 15H;
$RULE:               *ENABLE NIGHT IN 15M;
$ENDLOOP;
$ENDGROUP;
%
%                 GROUP AFTERN BEGINS AN AFTERNOON TRYST.
%
$GROUP AFTERN:        1H/OFF;
$LOOP:               P.RENDM;
$LOOP:               W.WANTED(P);
$LOOP:               C.CHASER(P);
$RULE:               *DISABLE AFTERN,
                     *MOVE GREENHS TO MROOM,
                     *MOVE WINDOW TO WEAPON,
                     *MOVE HOUSE TO MOTIVE,
                     *INSERT(W DECIDE)(DECIDE GOFOR WALK),
                     W.SMILEAT P,
                     *INSERT (P SEE THAT)(W GOTO GARDEN),
                     P FOLLOW W,
                     *INSERT (C SEE THAT)(P FOLLOW W),
                     *INSERT (C THINK THAT) (P AFFECTION = 3 W),
                     W WALKIN GARDEN,
                     C FOLLOW P,
                     P MEET W;
$ENDLOOP;
$ENDLOOP;
$ENDLOOP;
$ENDGROUP;
%
%                    GROUP NIGHT BEGINS A NIGHT-TIME TRYST.
%
$GROUP NIGHT:        1H/OFF;
$LOOP:               P.RENDM;
$LOOP:               W.WANTED(P);
$LOOP:               C.CHASER(P);
$RULE:               *DISABLE NIGHT,
                     *MOVE LIBRARY TO MROOM,
                     *MOVE DOOR TO WEAPON,
                     *MOVE BED TO MOTIVE,
                     P AWAKE,
                     P GETUP;
$RULE:               *INSERT (P THINK THAT)(SPOUSE(P) ASLEEP);
10,-10:              (P MARRIED);
$RULE:               *INSERT (P PLANNS)(PLANNO MEET W),
```

```
                                P ENTER HALL,
                                W GETUP,
                                W GOTO HALL,
                                C KNOW PLAN,
                                *INSERT (C DECIDE)(DECIDE FOLLOW THEY),
                                ULST XX,
                                *INSERT (C DECIDE)(DECIDE FOLLOW P),
                                *INSERT (C DECIDE)(DECIDE FOLLOW W),
                                LST XX;
$ENDLOOP;
$ENDLOOP;
$ENDLOOP;
$ENDGROUP;
%
%        GROUP REND CONTROLS THE ACTUAL TRYST ITSELF.
%        THERE IS ALWAYS AN OBSERVER INVOLVED. HIS
%        (OR HER) ACTIONS DEPEND ON HIS RELATIONSHIP TO
%        THE OTHER TWO AND ON HIS OWN PERSONALITY.
%
$GROUP REND:                    1H/OFF;
$RULE:                          *DISABLE REND;
$LOOP:          P.RENDM;
$LOOP:          W.WANTED(P);
$LOOP:          C.CHASER(P);
$RULE:          *REMOVE P FROM INTERUPT,
                *REMOVE C FROM INTERUPT,
                *REMOVE W FROM INTERUPT;
$RULE:          P KISS W;
      :         .5;
$RULE:          W CARESS P;
      :         .5;
$RULE:          W KISS P;
      :         .5;
$RULE:          ULST XX,
                P GOTO MROOM,
                W GOTO MROOM,
                C FOLLOW P,
                C FOLLOW W,
                LST XX,
                THEY GOTO MROOM,
                C FOLLOW THEY,
                W UNDRESS,
                P FUCK W;
$LOOP:          S.SPOUSE(P);
$RULE:          *INSERT (C SEE THAT)(P FUCK W);
      :         .5;
$RULE:          P COMMIT ADULTRY,
                *ADD P TO POSVICTM(S),
                *ADD S TO POSKILLR,
                S WHYKILL = 2 P,
                *ADD W TO POSVICTM(S),
                S WHYKILL = 3 W;
10,-10:         (P MARRIED);
$ENDLOOP;
```



```
$LOOP:          S.SPOUSE(W);
$RULE:          W COMMIT ADULTRY,
                *ADD S TO POSKILLR,
                *ADD W TO POSVICTM(S),
                S WHYKILL = 2 W,
                *ADD P TO POSVICTM(S),
                S WHYKILL = 3 P;
$ENDLOOP;
$SWITCH:        T(L1);
10,-10:         (C EQL SPOUSE(P)) OR (C EQL SPOUSE(W));
$RULE:          *ADD P TO POSKILLR,
                P WHYKILL = 5 C,
                *ADD C TO POSVICTM(P),
                *INSERT (C DECIDE)(DECIDE BLACKMAIL P),
                *INSERT (P LEAVE MROOM)(LEAVE WITH W),
                C ACCOST P,
                C BLACKMAIL P;
$RULE:          T(L5)
                *INSERT (C THREATEN)(THREATEN TELL SPOUSE(P));
10,-10:         (P MARRIED);
$RULE:          *INSERT(C THREATEN)(THREATEN TELL SPOUSE(W));
$RULE L5:       MADAT C;
      :         .5;
$RULE:          *INSERT (P THREATEN)(THREATEN KILL C):
      :             (P VIOLENT)/9 + .5;
$RULE:              P AFRAID,
                    *INSERT (P AGREE)(AGREE PAY C);
      :             (P WEALTH)/8 + .5;
$RULE:              (L3)
                    P AFFECTION = -3 C;
%
$RULE L1:           C ENRAGED;
      :             (C JEALOUS)/7 + .6;
$RULE:              F(L2)
                    C ENTER MROOM,
                    C YELLAT P;
      :             (C VIOLENT)/8 + .5;
$RULE:              C CRY;
10,-10:             (C EQL FEMALE);
$RULE:              *INSERT (C THREATEN)(THREATEN KILL P);
      :             (C VIOLENT)/7 + .5;
$RULE:              W EMBARASD;
      :             .7;
$RULE:              W CRY;
      :             .5;
$RULE:              (L3)
                    *INSERT(SPOUSE(C) ASK C)(ASK FORGIVE SPOUSE(C));
      :             .7;
%
$RULE L2:           C LOOKTHRU WEAPON,
                    MX QO = 1,
                    C HIDDEN;
$RULE:              C MADAT SPOUSE(C);
      :             (C JEALOUS)/8 + .5;
```

```
$RULE:          *INSERT (C WANT THAT)(C KILL W);
    :           (C VIOLENT)/10 + .4;
$RULE:          *INSERT (C WANT THAT)(C KILL P);
    :           (C VIOLENT)/10 + .4;
$RULE:          C CRY;
.8,-10:         (C EQL FEMALE);
$RULE:          C UPSET,
                C DEPRESSED;
$RULE L3:       EVERYONE GOTO MOTIVE,
                ULST XX,
                C GOTO MOTIVE,
                P GOTO MOTIVE,
                W GOTO MOTIVE,
                LST XX,
                *REMOVE P FROM INTERUPT,
                *REMOVE C FROM INTERUPT,
                *REMOVE W FROM INTERUPT;
$RULE:          T($ENDGROUP)
                *ADD P TO TALKING,
                *ADD W TO TALKING,
                *ADD C TO TALKING;
10,-10:         CLOCK GT 3H;
$RULE:          *ADD P TO RETIRED,
                *ADD W TO RETIRED,
                *ADD C TO RETIRED;
$ENDLOOP;
$ENDLOOP;
$ENDLOOP;
$ENDGROUP;
%
% ****************  SECTION E  ****************
%
%               SECTION E CONTROLS THE ACTUAL COMMITING OF THE
%               MURDER.
%
%
%               GROUP DOKILL RANDOMLY SELECTS A KILLER FROM
%               AMONG THE KILLER's POTENTIAL VICTIMS. THE
%               WAY THE MURDER IS COMMITTED IS DETERMINED BY
%               THE MOTIVE FOR THE CRIME. THERE ARE SIX
%               POSSIBLE MOTIVES AND SIX CORRESPONDING
%               MODUS OPERANDI.
%
$GROUP DOKILL:      1H/OFF;
$SWITCH:            T(L1);
10,-10:             NUM(POSKILLR) EQ 0;
$LOOP:              K.PICK(POSKILLR);
$LOOP:              V.PICK(POSVICTM(K));
$RULE:              *ENABLE FINDING IN 1H,
                    *DISABLE DOKILL,
                    *REMOVE V FROM RETIRED,
                    *REMOVE V FROM PEOPLE,
                    *REMOVE V FROM SUESTS,
                    *REMOVE V FROM SERVANT,
```



```
                *REMOVE V FROM MALE,
                *REMOVE V FROM FEMALE,
                *MOVE K TO KILLER,
                *MOVE V TO VICTIM;
$SWITCH:        T(KSA);
10,-10:         (K WHYKILL V) EQ 2;
$SWITCH:        T(KSL);
10,-10:         (K WHYKILL V) EQ 3;
$SWITCH:        T(KB);
T10,-10:        (K WHYKILL V) EQ 5;
$SWITCH:        T(KR);
10,-10:         (K WHYKILL V) EQ 6;
$SWITCH:        T(KBP);
10,-10:         (K WHYKILL V) EQ 7;
$SWITCH:        T(KLB);
10,-10:         (K WHYKILL V) EQ 9;
$RULE:          ($ENDGROUP)
                *PRINT '****ERROR: NO MOTIVE',
                *PRINT K,
                *PRINT V,
                *END;
%
%               STABBING YOUR SPOUSE FOR ADULTRY.
%
$RULE KSA:      *INSERT (K KNOW THAT)(V COMMIT ADULTRY),
                K ENRAGED,
                *INSERT (K MADAT V)(MADAT VERY),
                *INSERT (K DECIDE)(DECIDE STAB V),
                DAY IS SUNDAY,
                TIME IS DAWN,
                *INSERT (V AWAKEN)(AWAKEN EARLY),
                *INSERT (V DECIDE)(DECIDE GOFOR WALK),
                *INSERT (V GETUP)(GETUP QUIETLY),
                *INSERT (V THINK THAT)(K ASLEEP);
$RULE:          ULST XX;
    :           .4;
$RULE:          V OETDRESS.
                LST XX;
$RULE:          V GOTO GARDEN,
                K FOLLOW V;
$RULE:          V SEE K;
    :           .7;
$RULE:          *INSERT (K HAVE KNIFE)(KNIFE LONG);
    :           .7;
$RULE:          *INSERT (K WAVE KNIFE)(WAVE WILDLY),
                K STAB V,
                MX QQ = 1,
                V SCREAM;
$RULE:          *INSERT (KNIFE SINK)(SINK DEEP);
    :           .7;
$RULE:          *INSERT (V STRUGGLE)(STRUGGLE WEAKLY);
    :           .7;
$RULE:          V HIT K;
    :           .6;
```

```
$RULE:          ($ENDGROUP)
                *INSERT (K SLASH V)(SLASH AGAIN),
                *INSERT (K SAY THAT)(V BETRAY K),
                V COVERWITH BLOOD,
                MX QQ = 1,
                V DIE,
                K HIDE KNIFE,
                K RETURNTO BEDROOM,
                *INSERT (K WASH)(WASH OFF BLOOD),
                *MOVE JEALOUSY TO MOTIVE,
                *MOVE GARDEN TO MROOM,
                *MOVE KNIFE TO EVIDENCE,
                *MOVE KNIFE TO WEAPON;
%
%                   SHOOTING YOUR SPOUSE's LOVER.
%
$RULE KSL:      *MOVE GUN TO WEAPON,
                *MOVE JEALOUSY TO MOTIVE,
                *MOVE LIBRARY TO MROOM,
                *INSERT (K KNOW THAT)(V FUCK SPOUSE(K)),
                K AFFECTION = -3 V,
                K WANT REVENGE,
                *INSERT (K DECIDE)(DECIDE KILL V),
                K WRITE NOTE,
                *INSERT (V GET NOTE)(NOTTE FROM K),
                V MEET K,
                DAY IS SUNDAY,
                TIME IS DAWN,
                K GETUP,
                K GOTO LIBRARY,
                V GOTO LIBRARY,
                *INSERT (V THINK THAT)(K UNAWARE);
$RULE:          *INSERT (V SEE THAT)(K UPSET).
                *INSERT (V TRY)(TRY CALM K);
        :       .7;
$RULE:          *INSERT (K SAY THAT)(V EVIL)(V STEAL SPOUSE(K));
        :       .7;
$RULE:          *INSERT (K POINT GUN)(POINT AT V),
                V SEE GUN;
$RULE:          F(K1)
                V ATTACK K,
                *INSERT (V HIT K)(HIT IN STOMACH),
                *INSERT (V TRY)(TRY GRAB GUN);
        :       (V VIOLENT)/8 + .5;
$RULE:          K HIT V;
        :       .6;
$RULE:          K STRUGLWITH V;
        :       .7;
$RULE:          K KEEP CUN;
$RULE K1:       K SHOOT V,
                *INSERT (V STAGGER)(STAGGER BACK),
                V DIE,
                K HIDE GUN,
                K LOOKFOR NOTE,
```



```
                NOTE GONE,
                K RETURNTO BEDROOM;
$RULE:          T($ENDGROUP)
                *MOVE NOTE TO EVIDENCE;
        :       .65;
$RULE:          ($ENDGROUP)
                *MOVE GUN TO EVIDENCE;
%
%               HITTING SOMEONE OVER THE HEAD WITH A HEAVY
%               OBJECT FOR BLACKMAILING YOU.
%
$LOOP K9:       H.PICK(HEAVYOBJ);
$RULE:          *MOVE H TO EVIDENCE,
                *MOVE H TO WEAPON,
                *MOVE FEAR TO MOTIVE,
                *MOVE HALL TO MROOM,
                V BLACKMAIL K,
                *INSERT (K MADAT V)(MADAT VERY),
                *INSERT (K AFRAID)(AFRAID OF V),
                *INSERT (K DECIDE)(DECIDE KILL V),
                DAY IS SUNDAY,
                TIME IS DAWN,
                K GETUP,
                K GOTO HALL,
                MX QQ = 1,
                HALL DARK,
                K HIDENO,
                K HAVE H,
                *INSERT (V AWAKEN)(AWAKEN EAPLY),
                *INSERT (V EARLY)(EARLY USUALLY),
                V GOFOR WALK,
                K WAITFOR V,
                K SURPISE V,
                *INSERT (K HIT V)(HIT WITH H),
                *INSERT (V GROAN)(GROAN WEAKLY),
                V DIS;
```

7.6. *Sample Murder Mystery Texts*

We offer a 2100 word story, complete with semantic deep structure, generated in under 19 seconds. We also offer selected murder scenes from other runs that used different random number sequences and/or different character trait specification for Dr. Hume. (In some runs he was made very lustful and evil.) The change stack listing does show all triple linkages that are tabulated by the system.

7.6.1. *A 2100 Word Murder Mystery Story*

The following material is a short sample of the actual computer output, plus the full text of an actual story, minus deep structure, and edited for capitalization and spacing within paragraphs.

CHANGE STACK FOR TIME 19W3D10H

1: (LADYBUX WEALTH) = 3.0000
2: (MX QQ) = 2.0000
3: (LADYBUX GOOD) = 3.0000
4: (LADYBUX IQ) = 125.0000
5: (LADYBUX SINGLE) SET AT 19W3D10H
6: (MX QQ) = 2.0000
7: (LADYBUX ATTRACTI) = −2.0000
8: (LADYBUX SEXDRIVE) = 4.0000
9: (JOHNBUX IS NEPHEW) SET AT 19W3D10H
10: (NEPHEW POS LADYBUX) SET AT 19W3D10H
11: (JOHNBUX GOOD) = −3.0000
12: (MX QQ) = 2.0000
13: (JOHNBUX WEALTH) = −2.0000
14: (JOHNBUX VIOLENT) = 1.0000
15: (JOHNBUX SINGLES) SET AT 19W3D10H
16: (MX QQ) = 2.0000
17: (JOHNBUX HANDSOME) = 3.0000
18: (JOHNBUX SEXDRIVE) = 3.0000
19: (JOHNBUX AFFECTIO LORDEO) = −2.0000
20: (JOHNBUX AFFECTIO DRHUME) = −2.0000
21: (DRHUME GOOD) = −3.0000
22: (MX QQ) = 2.0000
23: (DRHUME IQ) = 150.0000
24: (DRHUME COURAGE) = 3.0000
25: (DRHUME SEXDRIVE) = 4.0000
26: (DRHUME SINGLE) SET AT 19W3D10H
27: (MX QQ) = 1.0000
28: (DRHUME HANDSOME) = 1.0000
29: (LORDED WEALTH) = 3.0000
30: (MX QQ) = 2.0000
31: (LORDED GOOD) = 2.0000
32: (LORDED VIOLENT) = −1.0000
33: (LORDED HANDSOME) = −1.0000
34: (MX QQ) = 1.0000
35: (LORDED SEXDRIVE) = 2.0000
36: (LORDED MARRIED) SET AT 19W3D10H
37: (MARRIED TO LADYJANE) SET AT 19W3D10H
38: (LORDED AFFECTIO LADYJANE) = 1.0000
39: (LORDED JEALOUS) = −1.0000
40: (LORDED AFFECTIO JOHNBUX) = −1.0000
41: (LADYJANE AFFECTIO LORDED) = 1.0000
42: (MX QQ) = 2.0000
43: (LADYJANE ATTRACTI) = 1.0000
44: (LADYJANE JEALOUS) = 1.0000

WONDERFUL SMART LADY BUXLEY WAS RICH.
UGLY OVERSEXED LADY BUXLEY WAS SINGLE.
JOHN WAS LADY BUXLEY's NEPHEW.
IMPOVERISHED IRRITABLE JOHN WAS EVIL.
HANDSOME OVERSEXED JOHN BUXLEY WAS SINGLE.
JOHN HATED EDWARD.
JOHN BUXLEY HATED DR. BARTHOLOMEW HUME.

BRILLIANT BRAVE HUME WAS EVIL.
HUME WAS OVERSEXED.
HANDSOME DR. BARTHOLOMEW HUME WAS SINGLE.
KIND EASY GOING EDWARD WAS RICH.
OVERSEXED LORD EDWARD WAS UGLY.
LORD EDWARD WAS MARRIED TO LADY JANE.
EDWARD LIKED LADY JANE.
EDWARD WAS NOT JEALOUS.
LORD EDWARD DISLIKED JOHN.
PRETTY JEALOUS JANE LIKED LORD EDWARD.

CHANGE STACK FOR TIME 19W3D10H1M

1: (RONALD GOOD) = 2.0000
2: (MX QQ) = 1.0000
3: (RONALD WEALTH) = 2.0000
4: (RONALD MARRIED) SET AT 19W3D10H1M
5: (MARRIED TO CATHY) SET AT 19W3D10H1M
6: (MX QQ) = 1.0000
7: (RONALD SEXDRIVE) = 1.0000
8: (RONALD AFFECTIO CATHY) = 3.0000
9: (MX QQ) = 1.0000
10: (RONALD HANDSOME) = 1.0000
11: (RONALD AFFECTIO DRHUME) = 1.0000
12: (RONALD AFFECTIO JAMES) = −1.0000
13: (CATHY GOOD) = 2.0000
14: (MX QQ) = 2.0000
15: (CATHY VIOLENT) = −2.0000
16: (CATHY SEXDRIVE) = 1.0000
17: (CATHY AFFECTIO RONALD) = 3.0000
18: (MX QQ) = 2.0000
19: (CATHY ATTRACTI) = 2.0000
20: (CATHY JEALOUS) = 1.0000
21: (JAMES IS PARTNER2) SET AT 19W3D10H1M
22: (PARTNER2 POS RONALD) SET AT 19W3D10H1M
23: (JAMES AFFECTIO RONALD) = −3.0000
24: (JAMES IQ) = 80.0000
25: (MX QQ) = 2.0000
26: (JAMES GOOD) = −3.0000
27: (JAMES VIOLENT) = 3.0000
28: (JAMES MARRIED) SET AT 19W3D10H1M
29: (MARRIED TO MARION) SET AT 19W3D10H1M
30: (MX QQ) = 2.0000
31: (JAMES SEXDRIVE) = −3.0000
32: (JAMES HANDSOME) = −3.0000
33: (JAMES AFFECTED MARION) = −1.0000
34: (MX QQ) = 2.0000

CHANGE STACK FOR TIME 20W2D1H

CHANGE STACK FOR TIME 20W2D1H5M

CHANGE STACK FOR TIME 20W2D1H15M

CHANGE STACK FOR TIME 20W2D1H25M

CHANGE STACK FOR TIME 20W2D1H35M

1: NOT (DRHUME PLAY BRIDGE) SET AT 20W1D21H55M
2: NOT (JOHNBUX PLAY BRIDGE) SET AT 20W1D21H55M
3: NOT (LADYBUX PLAY BRIDGE) SET AT 20W1D21H55M
4: NOT (LADYJANE PLAY BRIDGE) SET AT 20W1D21H55M
5: NOT (THEY PLAY BRIDGE) SET AT 20W1D21H55M
6: (CARDGAME OVER) SET AT 20W2D1H35M

THE CARD GAME WAS OVER.

CHANGE STACK FOR TIME 20W2D2H

CHANGE STACK FOR TIME 20W2D2H15M

1: (JOHNBUX AWAKE) SET AT 20W2D2H15M
2: (JOHNBUX GETUP) SET AT 20W2D2H15M
3: (JOHNBUX PLANNO) SET AT 20W2D2H15M
4: (PLANNO MEET MARION) SET AT 20W2D2H15M
5: (JOHNBUX ENTER HALL) SET AT 20W2D2H15M
6: (MARION GETUP) SET AT 20W2D2H15M
7: NOT (MARION GOTO PARLOR) SET AT 20W1D20H15M
8: (MARION GOTO HALL) SET AT 20W2D2H15M
9: (JAMES KNOW PLAN) SET AT 20W2D2H15M
10: (JAMES DECIDE) SET AT 20W2D2H15M
11: (DECIDE FOLLOW THEY) SET AT 20W2D2H15M
12: (ULST XX) SET AT 20W2D2H15M
13: (JAMES DECIDE) SET AT 20W2D2H15M
14: (DECIDE FOLLOW JOHNBUX) SET AT 20W2D2H15M
15: (JAMES DECIDE) SET AT 20W2D2H15M
16: (DECIDE FOLLOW MARION) SET AT 20W2D2H15M
17: (LST XX) SET AT 20W2D2H15M

JOHN AWOKE.
JOHN BUXLEY GOT UP.
JOHN PLANNED TO MEET MARION.
JOHN ENTERED THE CORRIDOR.
MARION GOT UP.
MARION WENT TO THE HALL.
JAMES KNEW THE PLAN.
JAMES DECTDED TO FOLLOW THEM.

CHANGE STACK FOR TIME 20W2D2H20M

1: (JOHNBUX KISS MARION) SET AT 20W2D2H20M
2: (MARION KISS JOHNBUX) SET AT 20W2D2H20M
3: (ULST XX) SET AT 20W2D2H20M
4: NOT (JOHNBUX GOTO PARLOR) SET AT 20W1D20H15M
5: (JOHNBUX GOTO LIBRARY) SET AT 20W2D2H20M
6: NOT (MARION GOTO HALL) SET AT 20W2D2H15M

7: (MARION GOTO LIBRARY) SET AT 20W2D2H20M
8: (JAMES FOLLOW JOHNBUX) SET AT 20W2D2H20M
9: (JAMES FOLLOW MARION) SET AT 20W2D2H20M
10: (LST XX) SET AT 20W2D2H20M
11: NOT (THEY GOTO GREENHS) SET AT 20W1D15H20M
12: (THEY GOTO LIBRARY) SET AT 20W2D2H20M
13: (JAMES FOLLOW THEY) SET AT 20W2D2H20M
14: (MARION UNDRESS) SET AT 20W2D2H20M
15: (JOHNBUX FUCK MARION) SET AT 20W2D2H20M
16: (MARION COMMIT ADULTRY) SET AT 20W2D2H20M
17: (JAMES ENRAGES) SET AT 20W2D2H20M
18: (JAMES ENTER LIBRARY) SET AT 20W2D2H20M
19: (JAMES YELLAT JOHNBUX) SET AT 20W2D2H20M
20: (JAMES THREATEN) SET AT 20W2D2H20M
21: (THREATEN KILL JOHNBUX) SET AT 20W2D2H20M
22: (MARION EMBARASO) SET AT 20W2D2H20M
23: (MARION CRY) SET AT 20W2D2H20M
24: (EVERYONE GOTO BED) SET AT 20W2D2H20M
25: (ULST XX) SET AT 20W2D2H20M
26: (JAMES GOTO BED) SET AT 20W2D2H20M
27: (JOHNBUX GOTO BED) SET AT 20W2D2H20M
28: (MARION GOTO BED) SET AT 20W2D2H20M
29: (LST XX) SET AT 20W2D2H20M

JOHN BUXLEY KISSED MARION.
MARION KISSED JOHN.
THEY WENT TO THE LIBRARY.
JAMES FOLLOWED THEM.
MARION UNDRESSED.
JOHN BUXLEY SCREWED MARION.
MARION COMMITED ADULTERY.
JAMES WAS ENRAGED.
JAMES ENTERED THE LIBRARY.
JAMES YELLED AT JOHN.
JAMES THREATEND TO KILL JOHN BUXLEY.
MARION WAS EMBARASSED.
MARION CRIED.
EVERYONE WENT TO BED.

CHANGE STACK FOR TIME 20W2D3H

CHANGE STACK FOR TIME 20W2D4H

CHANGE STACK FOR TIME 20W2D5H

CHANGE STACK FOR TIME 20W2D6H

1: (JAMES RICH) SET AT 20W2D6H
2: (RICH VERY) SET AT 20W2D6H
3: (BUTLER WEALTH) = - 3 0000
4: (BUTLER WANT MO[illegible] ET AT 20W2D6H
5: (BUTLER RELATEDT JAMES) SET AT 20W2D6H
6: (BUTLER DECIDE) SET AT 20W2D6H
7: (DECIDE POISONS JAMES) SET AT 20W2D6H

```
8:  (BUTLER THINK THAT) SET AT 20W2D6H
9:  (BUTLER INHERIT MONEY) SET AT 20W2D6H
10: (BUTLER KNOW THAT) SET AT 20W2D6H
11: (JAMES DRINK MILK) SET AT 20W2D6H
12: (BUTLER POISONS MILK) SET AT 20W2D6H
13: (JAMES DRINK MILK) SET AT 20W2D6H
14: (JAMES GOTO BED) SET AT 20W2D6H
15: (JAMES DIE) SET AT 20W2D6H
16: (OTHERS THINK THAT) SET AT 20W2D6H
17: (JAMES ASLEEP) SET AT 20W2D6H
18: (ULST XX) SET AT 20W2D6H
19: (CATHY THINK THAT) SET AT 20W2D6H
20: (JAMES ASLEEP) SET AT 20W2D6H
21: (COOK THINK THAT) SET AT 20W2D6H
22: (JAMES ASLEEP) SET AT 20W2D6H
23: (DRHUME THINK THAT) SET AT 20W2D6H
24: (JAMES ASLEEP) SET AT 20W2D6H
25: (JOHNBUX THINK THAT) SET AT 20W2D6H
26: (JAMES ASLEEP) SET AT 20W2D6H
27: (LADYBUX THINK THAT) SET AT 20W2D6H
28: (JAMES ASLEEP) SET AT 20W2D6H
29: (LADYJANE THINK THAT) SET AT 20W2D6H
30: (JAMES ASLEEP) SET AT 20W2D6H
31: (LORDED THINK THAT) SET AT 20W2D6H
32: (JAMES ASLEEP) SET AT 20W2D6H
33: (MAID THINK THAT) SET AT 20W2D6H
34: (JAMES ASLEEP) SET AT 20W2D6H
35: (MARION THINK THAT) SET AT 20W2D6H
36: (JAMES ASLEEP) SET AT 20W2D6H
37: (NURSE THINK THAT) SET AT 20W2D6H
38: (JAMES ASLEEP) SET AT 20W2D6H
39: (RONALD THINK THAT) SET AT 20W2D6H
40: (JAMES ASLEEP) SET AT 20W2D6H
41: (LST XX) SET AT 20W2D6H
42: (BUTLER REMOVE FPRINTS) SET AT 20W2D6H
43: (BUTLER RETURN BOTTLE) SET AT 20W2D6H
```

```
JAMES WAS VERY RICH.
CLIVE WAS IMPOVERISHED.
CLIVE WANTED THE MONEY.
THE BUTLER WAS RELATED TO JAMES.
THE BUTLER DECIDED TO POISON JAMES.
CLIVE THOUGHT THAT CLIVE INHERITED THE MONEY.
CLIVE KNEW THAT JAMES DRANK A MILK.
CLIVE POISONED THE MILK.
JAMES DRANK THE MILK.
JAMES WENT TO BED.
JAMES DIED.
THE OTHERS THOUGHT THAT JAMES WAS ASLEEP.
CLIVE REMOVED THE FINGERPRINTS.
THE BUTLER RETURNED THE BOTTLE.
```

```
CHANGE STACK FOR TIME 20W2D7H
```


MURDER MYSTERY 1 by Novel Writer Simulation Program

(Edited only for capitalization and spacing within paragraphs. Original paragraphing, reflecting sequential time frames, is preserved)
Wonderful smart Lady Buxley was rich. Ugly oversexed Lady Buxley was single. John was Lady Buxley's nephew. Impoverished irritable John was evil. Handsome oversexed John Buxley was single. John hated Edward. John Buxley hated Dr. Bartholomew Hume. Brilliant Hume was evil. Hume was oversexed. Handsome Dr. Bartholomew Hume was single. Kind easy going Edward was rich. Oversexed Lord Edward was ugly. Lord Edward was married to Lady Jane. Edward liked Lady Jane. Edward was not jealous. Lord Edward disliked John. Pretty jealous Jane liked Lord Edward.

Well to do Ronald was kind. Lusty Ronald was married to Cathy. Handsome Ronald loved Catherine. Ronald liked Hume. Ronald disliked James. Easy going lusty Cathy was kind. Beautiful jealous Catherine loved Ronald. James was Ronald's partner. James hated Ronald. Evil violent James was dumb. Impotent ugly James was married to Marion. Well to do Jealous James disliked Marion. James disliked Dr. Bartholomew Hume. Unpleasant violent Marion was smart. Beautiful Marion was impoverished. Jealous oversexed Marion hated James. Marion disliked Florence.

Florence was Lady Buxley's companion. Wonderful Florence was easy going. Beautiful oversexed Florence was single. The smart unpleasant butler was lusty. Poor brave butler was single. The dumb maid was good. Pretty poor Heather was single. Ugly violent cook was single. The cook was poor.

The day was Tuesday. The weather was rainy. Marion was in the park. Dr. Bartholomew Hume ran into Marion. Hume talked with Marion. Marion flirted with Hume. Hume invited Marion. Dr. Hume liked Marion. Marion liked Dr. Bartholomew Hume. Marion was with Dr. Bartholomew Hume in the hotel. Marion was near Hume. Dr. Hume caressed Marion with passion. Hume was Marion's lover. Lady Jane following them saw the affair. Jane blackmailed Marion. Marion was impoverished. Jane was rich.

Marion phoned Jane in the morning. Marion invited Jane to go to a theater. Jane agreed. Jane got dressed for the evening. They met them in the theater. Jane introduced Lord Edward during an intermission to Marion.

The day was Wednesday. The weather was windy. Lady Jane was in the tennis court. John ran into Lady Jane. John talked with Jane. Lady Jane flirted with John Buxley. John Buxley invited Lady Jane. John liked Lady Jane. Lady Jane liked John. John Buxley was with Jane in a movie. John was near Lady Jane. Jane caressed John Buxley with passion. Lady Jane was John's lover. Cathy following them saw the affair. Cathy blackmailed Lady Jane. Jane was well to do. Lady Catherine was rich.

Lady Catherine invited Jane to play bridge. Lady Catherine told Marion to come with Lady Buxley. Jane asked them to sit down. Lady Jane brought the cards. Jane offered drinks. Lady Buxley asked for whiskey on the rocks. The others had coffee with cookies. Jane shuffled the cards. Lady Jane started a game. Marion casually signaled Lady Buxley with hands. Jane noticed it. Lady Jane suspected that they cheated. Jane watched them closely. Marion won the game with Lady Buxley. Jane was upset with Catherine. Lady Jane disliked Marion.

The day was Thursday. The weather was rainy. A small pub was on a corner. John Buxley was in the pub. John Buxley asked for whiskey on the rocks. John got a drink from the barman. John talked with Hume near the bar. Hume sang the Beatles's song. John Buxley was drunk. James said that Marion commited adultery. James thought that James was drunk. James was depressed. James left the pub. Edward said that Lady Jane commited adultery. John Buxley thought Lord Edward was drunk. Lord Edward was depressed. Lord Edward left the pub.

The day was Friday.

Lady Buxley had a big house. Lady Buxley's house had a pretty fragrant garden. A green house was in the garden. The garden was near the tennis court. The house had a big bright dining room. The house also had a pleasant parlor. A cool dark musty library was near the parlor. The time was evening. Lady Buxley gave a party. The party lasted for a weekend.

Lady Buxley talked with Florence

Marion arrived with James.

Catherine arrived with Ronald.

Edward arrived with Jane.

Dr. Hume arrived. Dr. Bartholomew Hume joined a conversation.

Catherine talked with Dr. Bartholomew Hume. Dr. Bartholomew Hume flirted with Lady Catherine. Dr. Bartholomew Hume said that Lady Catherine was beautiful. Dr. Hume wanted to seduce Catherine. Hume told a joke. Catherine laughed.

Lady Buxley talked with Ronald. Florence talked with Dr. Bartholomew Hume. Dr. Hume flirted with Florence. Dr. Bartholomew Hume flattered Florence. Florence was very aroused. Dr. Bartholomew Hume liked Florence. Florence liked Hume.

The servants went to bed.

John Buxley arrived. Lady Buxley greeted John Buxley. John joined the conversation.

John Buxley talked with Jane. John Buxley casually mentioned politics. Lady Jane discussed politics with John Buxley. Lady Jane said that the weather was nice.

Lord Edward talked with Lady Jane. Florence talked with Edward. Edward flirted with Florence. Lord Edward wanted to seduce Florence. Lord Edward smiled at Florence. Florence smiled at Lord Edward. Jane saw that Edward whispered to Florence. Lady Jane was angry. Lord Edward saw that Lady Jane was angry.

Marion talked with Lord Edward. Lord Edward flirted with Marion. Lord Edward said that Marion was beautiful. Lord Edward smiled at Marion. Edward gently touched Marion. Lord Edward whispered to Marion. Edward liked Marion. Marion liked Edward. James saw that Marion talked with Edward. Jane saw that Edward whispered to Marion. Jane was angry. Jane saw that Edward smiled at Marion.

Everyone went to bed.

The day was Saturday. The sun rose. The servants got up. The cook went to the kitchen. The cook prepared a breakfast. Clive followed the cook. Clive seduced Maggie in the kitchen.

The day was beautiful. They got up. They got dressed. They went down to the breakfast.

Florence talked with Ronald. Ronald said that Florence looked well. Florence casually mentioned business. Ronald hated conversations about business.

The breakfast was over. James talked with Lady Buxley. James casually mentioned a music. Lady Buxley discussed the music with James.

Everyone went to the parlor.

James talked with Dr. Hume. Hume argued with James. James said that Hume was idiotic. Hume threatened to hit James. Dr. Bartholomew Hume cursed James. James hit Dr. Bartholomew Hume in the nose. Dr. Bartholomew Hume tried to grab James. James pushed Hume. Hume threatened to kill James. Dr. Bartholomew Hume hit James. James hated Dr. Hume.

Dr. Hume asked Lord Edward to play chess. Edward agreed. Lord Edward went to the study with Dr. Hume. They played chess. Hume was a good player. Lord Edward played chess well.

Florence talked with John. John flirted with Florence. John wanted to screw Florence. Florence smiled at John Buxley.

James talked with John. John laughed. John Buxley said that James looked well.

Ronald talked with James. James argued with Ronald. Ronald said that James was idiotic. James threatened to hit Ronald. Ronald hit James. James kicked Ronald in the belly. Ronald groaned softly. Ronald hit James in the nose. James tried to grab Ronald. Ronald pushed James. Ronald struggled with James. James threatened to kill Ronald. James hit Ronald. Ronald hated James.

Lady Buxley talked with Florence.

The cook went to the kitchen. Maggie prepared lunch.

Ronald talked with Lady Buxley.

Clive announced lunch. Edward stopped playing chess. Dr. Bartholomew Hume stopped playing chess.

Everyone went to the dining room. Everyone sat down. Clive served the food. Lunch started.

Florence talked with Hume. Florence casually mentioned fashion. Dr. Bartholomew Hume hated the conversations about fashion.

Lunch was over. The men went to the parlor. The men smoked cigars. The women went to the drawing room. The women drank whiskey.

Everyone went to the parlor. Marion decided to go for a walk. Marion smiled at Edward. Edward saw that Marion went to the garden. Edward followed Marion. Jane saw that Edward followed Marion. Jane thought that Lord Edward loved Marion. Jane followed Lord Edward. Lord Edward met Marion.

Edward kissed Marion. Marion caressed Edward. They went to the green house. Lady Jane followed them. Marion undressed. Edward screwed Marion. Edward commited adultery. Marion commited adultery. Lady Jane was enraged. Jane entered the green house. Jane yelled at Lord Edward. Jane cried. Jane threatened to kill Lord Edward. Marion was embarassed. Lord Edward asked Lady Jane to forgive Lord Edward. Everyone went to the house.

Marion talked with John Buxley. John Buxley flirted with Marion. John Buxley gently touched Marion. Marion smiled at John. John Buxley wanted to seduce Marion. Marion wanted to seduce John Buxley. James saw that Marion talked with John. James was mad at John. James overhearing Marion was angry. Marion saw that James was upset. Marion talked with James.

The butler announced tea.

Everyone went to the garden. The butler served tea. The day was cool. The sky was cloudy. The garden was nice. The flowers were pretty. Marion complimented Lady Buxley.

Ronald talked with Marion.

Tea time was over.

Everyone went to the parlor.

The cook went to the kitchen. Maggie prepared dinner.

Dr. Hume asked Edward to play tennis. Edward agreed. Lord Edward went to the tennis court with Dr. Hume. They played tennis. Dr. Hume was the good player. Edward played tennis well.

The butler announced dinner.

Dr. Bartholomew Hume stopped playing tennis. Edward stopped playing tennis.

Everyone went to the dining room. Everyone sat down. The butler served the food. Supper started.

Marion talked with Florence. Florence argued with Marion. Marion said that Florence was idiotic.

Florence talked with Lady Buxley.

Supper was over. The men went to the parlor. The men smoked fat smelly stogies. The men drank sherry. The women went to the drawing room. The women gossiping drank coffee.

Everyone went to the parlor.

Marion talked with Jane.

James went to the library. James read the good paperback. Edward asked Ronald to play tennis. Ronald agreed. Ronald went to the tennis court with Lord Edward. They played tennis.

John suggested the game of bridge. Lady Buxley agreed. Dr. Bartholomew Hume. Jane agreed. They played bridge.

The servants went to bed. Everyone went to bed.

James stopped reading the book.

Ronald beat Lord Edward at tennis. Lord Edward stopped playing tennis. Ronald stopped playing tennis.

John Buxley cheated at bridge.

John cheated at bridge.

The card game was over.

John awoke. John Buxley got up. John planned to meet Marion. John entered the corridor. Marion got up. Marion went to the hall. James knew the plan. James decided to follow them.

John Buxley kissed Marion. Marion kissed John. They went to the library. James followed them. Marion undressed. John Buxley screwed Marion. Marion commited adultery. James was enraged. James entered the library. James yelled at John. James threatened to kill John Buxley. Marion was embarassed. Marion cried. Everyone went to bed.

James was very rich. Clive was impoverished. Clive wanted the money. The butler was related to James. The butler decided to poison James. Clive thought that Clive inherited the money. Clive knew that James drank a milk. Clive poisoned the milk. James drank the milk. James went to bed. James died. The others thought that James was asleep. Clive removed the fingerprints. The butler returned the bottle.

Ronald awakened. Ronald got up. Ronald thought that the day was beautiful. Ronald found James. Ronald saw that James was dead. Ronald yelled. The others awakened. The others ran to Ronald. The others saw James. Everyone talked. Heather called the policemen. Hume examined the body. Dr. Bartholomew Hume said that James was killed by poison.

John talked with Edward about the murder.

Edward talked with Maggie about the murder. Maggie was upset about the murder.

The cops arrived. The cops were idiotic. A detective examined the corpse. The policemen looked for hints in the bathroom. Dr. Bartholomew Hume also looked. Edward tried to calm Marion.

The policemen questioned Dr. Bartholomew Hume. The detective asked questions. The policemen searched the garden. The policemen tried to find clues. Marion cried.

Dr. Bartholomew Hume searched stairs. Hume looked for hints. Dr. Hume questioned Lady Buxley. Dr. Hume knew that Lady Buxley told the truth. Florence talked with Heather about the murder. Marion cried.

The policemen questioned Ronald. The inspector suspected Ronald. The inspector asked the stupid questions. The policemen searched the parlor. The policemen tried to find hints. Florence was upset.

Dr. Bartholomew Hume searched the dining room. Dr. Bartholomew Hume looked for hints.

The cops questioned Heather. The detective asked the stupid questions. Dr. Hume questioned Heather. Dr. Hume knew that Heather told the truth. The cops searched the tennis court. Clive talked with Ronald about the murder. The butler said that James was kind. The cook talked about the murder.

Dr. Bartholomew Hume searched the bathroom. Dr. Hume looked for clues. Marion cried.

Dr. Hume questioned Florence. Hume knew that Florence told the truth. Dr. Bartholomew Hume got information from Florence. The cops searched the bathroom. The cops found a thread. The thread was misleading clue. Lady Buxley talked with John about the murder. Lady Buxley said that James was kind. Dr. Hume was upset.

Dr. Bartholomew Hume searched the library. The cops questioned John Buxley. The detective asked the stupid questions. Hume questioned the cook. Dr. Bartholomew Hume knew that Maggie told the truth. Hume got information from the cook.

Hume went to the bathroom. Dr. Hume found the bottle Hume knew the murderer. Hume asked everyone to go to the parlor. Dr. Bartholomew Hume said that the murderer was in the room. Everyone was surprised. Everyone talked. Dr. Bartholomew Hume said that James was killed by poison. Hume said that the butler killed James. Everyone was shocked. The butler drew a pistol. Clive headed for the door. Dr. Bartholomew hume followed Clive. The butler shot at Hume. Dr. Bartholomew Hume grabbed a paperweight. Dr. Bartholomew Hume threw the paperweight at Clive. The paperweight hit Clive in the head. Clive fell. Dr. Bartholomew Hume took the gun. The policemen took Clive. Ronald congratulated Hume. Clever Dr. Hume solved the crime.

7.6.2. *Murder and Solution from Story 2*

JAMES KNEW THAT HUME SCREWED MARION.
JAMES HATED DR. BARTHOLOMEW HUME.
JAMES WANTED A REVENGE.
JAMES DECIDED TO KILL DR. HUME.
JAMES WROTE A NOTE.
DR. HUME GOT THE NOTE FROM JAMES.
HUME MET JAMES.
THE DAY WAS SUNDAY.
THE TIME WAS THE DAWN.
JAMES GOT UP.
JAMES WENT TO THE LIBRARY.
DR. BARTHOLOMEW HUME WENT TO THE LIBRARY.
HUME THOUGHT THAT JAMES WAS UNAWARE.
JAMES SAID THAT DR. BARTHOLOMEW HUME WAS EVIL.
JAMES POINTED A PISTOL AT DR. BARTHOLOMEW HUME.
DR. HUME SAW THE PISTOL.
HUME ATTACKED JAMES.
DR. BARTHOLOMEW HUME HIT JAMES IN THE BELLY.
DR. BARTHOLOMEW HUME TRIED TO GRAB THE PISTOL.
JAMES HIT HUME.
JAMES STRUGGLED WITH DR. BARTHOLOMEW HUME.
JAMES KEPT THE PISTOL.
JAMES SHOT DR. BARTHOLOMEW HUME.
HUME STAGGERED BACK.
DR. BARTHOLOMEW HUME DIED.
JAMES HID THE GUN.
JAMES LOOKED FOR THE NOTE.
THE NOTE WAS GONE.
JAMES RETURNED TO THE BEDROOM.

LADY JAME AWAKENED.
LADY JANE GOT UP.
JANE THOUGHT THAT THE DAY WAS BEAUTIFUL.
JANE FOUND DR. BARTHOLOMEW HUME.
LADY JANE SAW THAT DR. HUME WAS DEAD.
LADY JANE SCREAMED LOUD.
LADY JANE FAINTED.
THE OTHERS AWAKENED.
THE OTHERS RAN TO LADY JANE.
THE OTHERS SAW DR. BARTHOLOMEW HUME.
EVERYONE TALKED.
EDWARD CALLED THE COPS.
FLORENCE EXAMINED THE CORPSE.
FLORENCE SAID THAT DR. BARTHOLOMEW HUME WAS KILLED BY THE GUN.
THE POLICEMEN ARRIVED.
THE COPS WERE IDIOTIC.
A DETECTIVE EXAMINED THE CORPSE.
THE COPS LOOKED FOR CLUES IN THE LIBRARY.
FLORENCE ALSO LOOKED.

FLORENCE TALKED WITH THE COOK ABOUT THE MURDER.
THE COOK WAS UPSET ABOUT THE MURDER.
JAMES SAID THAT RONALD KILLED DR. HUME.
RONALD DENIED THE ACCUSATION.
RONALD SAID THAT JAMES WAS STUPID.

THE COPS QUESTIONED FLORENCE.
THE DETECTIVE SUSPECTED FLORENCE.
THE INSPECTOR ASKED QUESTIONS.
LADY CATHERINE TALKED ABOUT THE MURDER.

FLORENCE SEARCHED THE PARLOR.
FLORENCE LOOKED FOR HINTS.
FLORENCE QUESTIONED THE BUTLER.
FLORENCE GOT INFORMATION FROM CLIVE.

FLORENCE SEARCHED THE LIBRARY.
FLORENCE LOOKED FOR HINTS.
THE COPS QUESTIONED LADY JANE.

FLORENCE SEARCHED THE LIBRARY.
FLORENCE FOUND ASHES.
THE ASHES WERE VALUABLE CLUE.
THE POLICEMEN QUESTIONED RONALD.
THE INSPECTOR ASKED THE QUESTIONS.
JAMES TALKED ABOUT THE MURDER.

FLORENCE QUESTIONED MARION.
FLORENCE KNEW THAT MARION TOLD THE TRUTH.
FLORENCE GOT INFORMATION FROM MARION.

THE COPS QUESTIONED HEATHER.
THE INSPECTOR ASKED THE QUESTIONS.
THE COPS SEARCHED THE DRAWING ROOM.
THE POLICEMEN FOUND A THREAD.
THE THREAD WAS MISLEADING CLUE.
CATHERINE TALKED WITH THE BUTLER ABOUT THE MURDER.
CATHY SAID THAT DR. BARTHOLOMEW HUME WAS KIND.
THE BUTLER AGREED.
CLIVE WAS UPSET ABOUT THE MURDER.

FLORENCE WENT TO THE LIBRARY.
FLORENCE FOUND THE NOTE.
FLORENCE KNEW THE KILLER.
FLORENCE ASKED EVERYONE TO GO TO THE PARLOR.
FLORENCE SAID THAT THE MURDERER WAS IN THE ROOM.
EVERYONE WAS SURPRISED.
EVERYONE TALKED.
FLORENCE SAID THAT DR. HUME WAS KILLED BY THE PISTOL.
FLORENCE SAID THAT JAMES KILLED DR. BARTHOLOMEW HUME.
EVERYONE WAS SHOCKED.
JAMES DREW THE GUN.
JAMES HEADED FOR THE DOOR.
FLORENCE TRIPPED JAMES.

JAMES FELL.
FLORENCE STRUGGLED WITH JAMES.
THE GUN FIRED.
FLORENCE GOT THE GUN.
THE COPS TOOK JAMES TO THE JAIL.
THE POLICEMEN CONGRATULATED FLORENCE.
CLEVER FLORENCE SOLVED THE CRIME.

7.6.3. *Murder Scene from Story 3*

DR. BARTHOLOMEW HUME BLACKMAILED EDWARD.
EDWARD WAS AFRAID OF DR. HUME.
LORD EDWARD DECIDED TO KILL DR. BARTHOLOMEW HUME.
THE DAY WAS SUNDAY.
THE TIME WAS THE SUNRISE.
LORD EDWARD GOT UP.
LORD EDWARD WENT TO THE DARK CORRIDOR.
LORD EDWARD HID.
EDWARD HAD A CANDLE HOLDER.
DR. BARTHOLOMEW HUME AWAKENED EARLY.
DR. BARTHOLOMEW HUME WAS USUALLY EARLY.
DR. HUME WENT FOR THE WALK.
EDWARD WATTED FOR HUME.
LORD EDWARD SURPRISED HUME.
EDWARD HIT DR. BARTHOLOMEW HUME WITH THE CANDLE HOLDER.
DR. BARTHOLOMEW HUME GROANED WEAKLY.
DR. HUME DIED.
EDWARD RETURNED TO THE BEDROOM.

7.6.4. *Murder Scene from Story 4*

LORD EDWARD KNEW THAT LADY JANE COMMITED ADULTRY.
LORD EDWARD WAS ENRAGED.
EDWARD DECIDED TO STAB JANE.
THE DAY WAS SUNDAY.
THE TIME WAS THE SUNRISE.
JANE AWAKENED EARLY.
LADY JANE DECIDED TO GO FOR THE WALK.
JANE GOT UP QUIETLY.
JANE THOUGHT THAT EDWARD WAS ASLEEP.
JANE GOT DRESSED.
JANE WENT TO THE GARDEN.
EDWARD FOLLOWED LADY JANE.
JANE SAW EDWARD.
LORD EDWARD HAD A LONG DAGGER.
EDWARD WAVED THE DAGGER WILDLY.

LORD EDWARD STABBED JANE SCREAMING.
THE KNIFE SANK DEEP.
JANE STRUGGLED WEAKLY.
JANE HIT EDWARD.
LORD EDWARD SLASHED JANE AGAIN.
EDWARD SAID THAT LADY JANE BETRAYED LORD EDWARD.
JANE DYING COVERED WITH THE BLOOD.
LORD EDWARD HID THE KNIFE.
EDWARD RETURNED TO THE BEDROOM.
LORD EDWARD WASHED OFF THE BLOOD.

8.0. *References*

Bach, E.
1968
"Nouns and noun phrases", in: Bach, E., Harms, R.T. (eds.), *Universals in Linquistic Theory* (Chicago: Holt, Rinehart and Winston, Inc.) 90–122.

Coles, S.L.
1968
"An on-line question-answering system with natural language and pictorial input", in: *Proc. ACM 23rd Nat. Conf.* (Princeton: Brandon Systems Press).

Fillmore, C.J.
1968
"The case for case", in : Bach, E., Harms, R.T. (eds.), in: *Universals in Linquistic Theory* (Chicago: Holt, Rinehart and Winston, Inc.) 1–88.

Green, C.C., Raphael, B.
1968
"Research on intelligent question-answering systems", in: *Proc. ACM 23rd Nat. Conf.* (Princeton: Brandon System Press).

Heidorn, G.E.
1972
Natural language inputs to a simulation programming system, Report NPS-55HD72 101A, Navel Postgraduate School (Monterey California).

Katz, J.
1972
Semantic Theory (New York: Harper and Row).

Kellogg, C.H.
1968
"A natural language compiler for on-line data management", in: *Proc. AFIPS 1968 FJCC 33* (Montvale: AFIPS Press).

Klein, S.
1965a
"Automatic paraphrasing in essay format", in: *Mechanical Translation 8,* Nos. 2 and 3 combined.

Klein, S.
1965b
"Control of style with a generative grammar", in: *Language 41,* No. 4.

Klein, S.
1965c
"Some components of a program for dynamic modelling of historical change in language", in: *Preprints of Invited Papers for 1965 International Conference on Computational Linguistics* (New York).

Klein, S.
1973
Automatic inference of semantic deep structure rules in generative semantic grammars, UWCS Tech Report 180; also in: *Preprints of Papers Presented at 1973 International Conference on Computational Linguistics, Pisa, August 27-September 1.*

Klein, S.
1967
"Current research in the computer simulation of historical change in language", in: *Actes du X^e Congrès International des Linguistes, Bucharest 1967* IV (Academy of the Socialist Republic of Roumania).

Klein, S.
1972
Computer simulation of language contact models, UWCS Tech Report 167; also in: *Proceedings SECOL 8 Conference, Oct. 26–29, 1972* (Georgetown Univ.).

Klein, S., Davis, B., Fabens, W., Herriot, R., Katke, W., Kuppin, M.A., Towster, A.
1967a
AUTOLING: an automated linguistic fieldworker (Second International Conference on Computational Linguistics, August 1967, Grenoble).

Klein, S., Fabens, W., Herriot, R., Katke, W., Kuppin, M.A., Towster, A.
1968
The AUTOLING system, UWCS Tech Report 43. (Univ. of Wiscons in Computer Scl. Dept.)

Klein, S., Dennison, T.A.
1971
An interactive program for learning the morphology of natural languages, UWCS Tech Report 144; also in: *Proceedings of the 1971 International Conf. on Computational Linguistics, Debrecen, Hungary, September 1971* (Mouton and Hungarian Academy of Sciences).

Klein, S., Kuppin, M.A.
1970
An interactive, heuristic program for learning transformational grammars, UWCS Tech Report 97; also in: *Computer Studies in the Humanities and Verbal Behavior 3,* No.3, October 1970.

Klein, S., Kuppin, M.A., Meives, K.
1969
"Monte carlo simulation of language change in Tikopia and Maori", in: *Preprints of Papers Presented at the 1969 International Conference on Computational Linguistics* (Stockholm: KVAL).

Klein, S., Lieman, S.L., Lindstrom, G.D.
1966
DISEMINER: a distributional-semantics inference maker, Tech. Report of Carnegie Inst. of Tech.; also in: *Computer Studies in the Humanities in: Verbal Behavior 1,* No. 1, Jan. 1968.

Klein, S., Oakley, J.D., Suurballe, D.A., Ziesemer, R.A.
1971
A program for generating reports on the status & history of stochastically modifiable semantic models of arbitrary universes, UWCS Tech Report 142; also in: *Statistical Methods in Linguistics 8,* 1972; also in: *Proc. of 1971 Int. Conf. on Computational Linguistics, Debrecen, Hungary, September 1971* (Mouton and Hungarian Academy of Sciences).

Lakoff, G.
1969
"On generative semantics", in: Steinberg, D.D., Jakobovits, L.A. (eds.), *Semantics – An Interdisciplinary Reader in Philosophy, Linguistics, Anthropology and Psychology* (London: Cambridge Univ. Press) 232–296.

McCawley, J.D.
1968

"The role of semantics in a grammar", in: Bach, E., Harms, R.T. (eds.), *Universals in Linguistic Theory* (Chicago: Holt, Rinchart and Winston Inc.), 124–169.

Mel'čuk, I.A. Žolkovskij, A.K.
1970
"Toward a functioning 'meaning-text' model of language", in: *Linguistics* 57.

Petöfi, J.S.
1973
Towards an empirically motivated grammatical theorie of verbal texts (Bielefelder Papiere zur Linguistik und Literaturwissenschaft); also in: Petöfi, J.S., Rieser, H. (eds.), *Studies in Text Grammar* (Dordrecht: Reidel) 205–275.

Quillian, R.
1966
Semantic memory (Ph. D. Thesis, Carnegie-Mellon University, Pittsburgh; also: Cambridge, Mass.: Bolt, Beranek and Newman).

Schank, R.C.
1969
A conceptual dependency representation for a computer-oriented semantics, (Stanford Univ.: Artificial Intelligence Memo (AIM) 83).

Schank, R.C.
1972
"Conceptual dependency: a theory of natural language understanding", in: *Cognitive Psychology 3,* No. 4.

Schank, R.C., Rieger, C.J.
1973
Inference and the computer understanding of natural language (Stanford Univ.: Artificial Intelligence Memo (AIM) 197).

Simmons, R.F.
1970
Some relations between predicate calculus and semantic net representations of discourse (Austin: Univ. of Texas, Computer Science Dept, preprint).

Simmons, R.F.
1972
"Generating English discourse from semantic networks", in: *Communications of the ACM 14,* No. 10.

Simmons, R.F.
Compiling semantic networks from English sentences (in prep.).