Bulletin of the Faculty of Education,
Nagoya University (Educational Psychology)
1985, Vol. 32, 21 – 47.

文記憶のネットワーク活性化説に関するコンピュータ・シミュレーション

——S-O-V 型文について——*

都築誉史[1)]

I 問題と目的

Collins & Loftus (1975) と Collins & Quillian (1969) は，意味記憶内の記憶表象がネットワーク構造から成ると仮定し，ネットワーク表象内の情報の伝達と操作は，活性化の拡大 (spread of activation) プロセスによって媒介されるとする仮説を提出した。文記憶に関する研究では，Anderson (1976, 1983) や Hayes-Roth (1977) らによって同様の理論が主張され，実験的検討とモデル化がなされてきている。本研究では，ネットワーク構造から成る記憶表象と活性化の拡大プロセスとを仮定する記憶理論を，「ネットワーク活性化説」と総称することにする。ネットワーク活性化説は人間の記憶の仕組みを，構造とプロセスの両面から把握しようとする新たな枠組みを提供していると考えられる。Hayes-Roth & Hayes-Roth (1977) は，文記憶のネットワーク活性化説を次のようにまとめている。

(1) 記憶された各々の単語は，ネットワークにおける単一のノードとして表象され，ノード間の関係は適切なラベルを付与したリンクによって示される。

(2) 単語の意味は，その単語と他の単語との相互関係から引き出される。意味的に関連した単語は，そうではない単語よりも，ネットワーク内で近接して連結されている。

(3) 文は，単語ノードを基本単位としたネットワークとして表象され，高次のノードは，それが統合しているネットワーク自体を表象する。

(4) 言語的な刺激はネットワーク表象に活性化を引き起こし，活性化は，活性化したノードに連結した全てのリンクに沿って並列的に拡大する。

(5) 活性化した各々のノードから拡大する活性化の総量は，どの時点でも制限されている。

(6) 記憶表象における各々のリンクの強度は，伝播した活性化の量と頻度に関する増加関数であり，最後の活性化からの経過時間に関する減少関数である。

このように，Hayes-Roth & Hayes-Roth (1977) の理論は，単語ノードがネットワーク表象の基本単位であると仮定している点が特徴的であり，理解しやすく，実験的検討やモデル化に適している。従来，文記憶のネットワーク活性化説に関する研究では，ネットワーク構造と活性化プロセスにおける多数のパラメータを同時に操作するため，コンピュータ・シミュレーションの技法がしばしば用いられてきた。(例えば，Anderson, Kline, & Beasley, 1979; Kieras, 1981; Miller, 1981)。Miller (1981) は，Hayes-Roth & Hayes-Roth (1977) の研究をもとに，主語，動詞，目的語から成る簡単な文 (S-V-O型文) の記憶事態に対応したコンピュータ・シミュレーションを報告している。

本稿では，文記憶に関するネットワーク活性化説の有効性を検討するため，Miller (1981) のモデルを参考にシミュレーション・プログラムを作製し，実際にコンピュータ・シミュレーションを実行した結果について報告する。本研究のモデルについて述べる前に，次節では参考にしたMillerモデルについて説明する。

II Miller (1981) のモデル

Miller モデルは，ネットワーク活性化説をふまえたS-V-O型文の記憶に関するシミュレーション・モデルであり，符号化と検索の2段階から成るが，ともに共通の活性化システムを介して実行される。以下では，活性化システム，符号化プロセス，検索プロセスの3点に分けてMiller モデルを説明し，最後にその問題点についても言及する。

---

* 本研究の要旨は，日本認知科学会第1回大会において発表した。本研究を進めるにあたり御指導いただきました村上 隆助教授に，深く感謝致します。

1) 名古屋大学大学院教育学研究科博士課程（後期）
（指導教官 村上 隆助教授）

1　活性化システム

ノードを相互に連結し，ネットワークを構成している各々のリンクは，活性化を媒介するパイプラインに相当する。リンクは常に複線であり，それぞれ方向が限定されている。各リンクは，リンク活性化量，リンク強度，リンク伝達時間といった属性をもつ。まず，リンク活性化量は，各々のリンクが有する活性化の程度を量的に示している。また，リンク強度は，2つの単語ノード間の一時的な関連度を意味し，活性化がリンクを伝播するたびに強化される。一方，リンク伝達時間は，各リンクを活性化が移行するために必要な理論的所要時間を示し，リンク活性化量が大きくリンク強度が強いほど速いと仮定されている。そして，活性化の拡大は，ネットワーク内の全てのリンク伝達時間を比較し，その中で最小の所要時間を有するリンクを発見することによって実行される。

活性化のパイプラインに相当するリンクに対して，各々のノードは活性化の貯蔵タンクに相当する。リンクと同様に，各ノードはノード活性化量をもつ。ある時点で活性化しているノードの集合は，「活性化ノードリスト」と呼ばれ，人間が同時に意識できる情報量の限界に対応して最大値が制限されている。つまり，ある時点では活性化ノードリストに含まれるノードだけが活性化しており，他のノードは活性化状態にないと仮定されている。しかしながら，活性化の進行に伴って，様々なノードが活性化ノードリストに新しく現われたり，逆に，リストから消失したりする過程が繰り返される。

このように，活性化に関しては多くの変数が用いられているが，最も重要なのはリンク強度であり，ある時点におけるノード間の関連度を示す属性として，ネットワーク表象の特性を要約する意味をもつ。

2　符号化プロセス

符号化プロセスを説明する前に，Miller モデルで用いられている記憶表象について整理しておく必要がある。Miller モデルは，Hayes-Roth & Hayes-Roth (1977) の研究に基づいているため，高次のノードと一部の定義語ノード以外，ネットワーク表象は単語ノードを基本単位に構成されている。この高次のノードは「符号化文脈ノード」と呼ばれ，学習文自体に対応して文の記憶表象を統合する役割をもつ。また，Miller モデルの意味ネットワークは，Fillmore (1968) の格文法をふまえた上で，国語辞典から引用した定義語のネットワークとして構成されており，文脈に従って役割が決定される不特定ノードをも含んでいる。さらに，Miller モデルでは意味記憶と作業記憶とが区別され，別の構造として扱われている点も特徴的である。

次に，符号化プロセスを順を追って説明する。S-V-O型の学習文を入力すると，文の各単語に対応した3つの作業記憶ノードと，文自体を表わす符号化文脈ノードが生成される。文の入力と同時に投入された総活性化量は，活性化ノードリストに含まれるノード間で均等に配分され，さらに，各ノードから発するリンク間で分配される。そして，リンク活性化量とリンク強度から各々のリンク伝達時間が算出され，ネットワーク内で伝達時間が最小のリンクを発見することによって，連続的なシミュレーション時間の経過とともに活性化が拡大してゆく。

符号化の初期段階では，比較的単純なネットワーク内で活性化のサイクルが繰り返されるが，活性化のサイクルが進むにつれて，意味ネットワーク内の定義語ノードへと活性化は拡大する。Miller (1981) は，前者が人間の記憶プロセスにおける維持的リハーサルに対応し，後者が精緻化リハーサルに相当するとみなしている。意味ネットワーク内の定義語ノードが活性化されると，それに対応した作業記憶ノードが新たに生成され，意味ネットワーク内の関係を複製した形で相互に連結される。さらに，生成された作業記憶ノードは，リンクを介して符号化文脈ネットワークに統合される。

前述のように，Miller モデルでは活性化ノードリストに含まれるノードの最大数を制限しており，活性化ノード数が最大値をこえた場合には，活性化量が最小のノードが非活性化され，そのノード活性化量は0となる。従って，活性化サイクルの進行とともに，ネットワーク内の総活性化量は次第に減少してゆく。これに対して，Miller モデルでは，一定時間ごとに活性化量を外部から新たに再投入するという手続きが用いられている。そして，あらかじめ設定した処理時間が経過すると，符号化プロセスは終了する。

3　検索プロセス

テスト文の検索は，符号化プロセスで生成された学習文の符号化文脈ネットワークに基づいて実行される。このネットワークに沿ったテスト文の走査は，符号化時と同一の活性化システムによってなされる。検索プロセスでは，必要なリンクが全て発見されてイエス反応が出力されるか，または，所定の停止基準によってノー反応が出力されるまで活性化のサイクルが続けられる。

まず，イエス反応は，テスト文の主語，動詞，目的語に対応する3つの作業記憶ノードから，同一の符号化文脈ノードへ至る3本のリンク全てを，活性化が伝播した時点でなされる。一方，ノー反応は，テスト文に対応する3つの作業記憶ノードから，異なる符号化文脈ノード

へのリンクが活性化され，矛盾するリンク強度の合計が基準値をこえた場合になされる。また，一定時間たっても反応に必要なリンクが全て発見されない場合にも，ノー反応が出力される。

Miller モデルは，検索プロセスが終了した時点でテスト文に対する正誤反応，反応時間，確信度を出力する。正誤反応については既に説明したが，反応時間は，検索の開始から終了までの理論的な経過時間によっており，この値は前述のように，最小リンク伝達時間の逐次的な和として定義されている。また，確信度は，出力された反応を支持するリンク強度の合計から，その反応と矛盾するリンク強度を減じた値がとられている。以上，Miller（1981）による文記憶のシミュレーション・モデルを説明してきたが，次にこのモデルの問題点について述べる。

4　Miller モデルの問題点

Miller モデルは，①パラメータの数が多すぎる，②活性化量を外部から再投入するのは不合理である，③意味ネットワークの構造が複雑すぎる，といった3点から批判できる。

第1に，Miller モデルでは処理プロセスにおける諸変数の他に，多数のパラメータが用いられている。順にあげると，処理時間の制限（符号化時と検索時），投入する活性化の量，活性化を再投入する時間間隔，活性化ノードリストに含まれるノードの最大数，リンク強度の強化を特定化する値，検索停止の基準となる矛盾するリンク強度値などである。一般に，パラメータ数が必要最小限であるほどよいモデルだとされている。これに対してMiller モデルにおけるパラメータ数の多さは，活性化システムの複雑さと相まって，特定のパラメータの操作に対応したモデルの挙動や出力結果の予測を非常に困難にしている。

第2に，Miller モデルでは処理プロセスの途中で，一定時間ごとに活性化量を処理システムの外部から再投入しているが，こうした手続きは人間の情報処理プロセスと比較した場合，適切であるとは言い難い。Miller モデルでは，人間が同時に意識できる情報量の限界に対応づけて活性化ノードリストの最大値を制限したため，ネットワーク表象内の総活性化量は，活性化サイクルの進行とともに急激に減衰する。活性化の再投入はその対応策であるわけだが，人間の情報処理プロセスに対応するパラメータを1つ導入したために，人間の情報処理とは必ずしも一致しないパラメータをもう1つ採用したことになり，モデル構成における説得力を欠いている。つまり，活性化量の再投入は恣意的な操作であり，モデルを必要以上に複雑化していると考えられる。

第3に，意味ネットワーク構造の問題がある。Miller モデルの意味ネットワークは，具体的な定義語ノードを基本単位に構成されているが，かなり複雑であり，もとの単語は2水準で階層的に定義され，文脈に従って特定の単語が代入される不特定ノードも含まれている。Miller（1981）がこのように複雑な意味ネットワークを用いた主な理由は，同じモデルによってさらに複雑な文記憶事態をシミュレートしようとしたためであった。この目的のために，シミュレーションに用いた8つの文に含まれる24単語全てに関して，個別に意味ネットワークが設定された。しかしながら，簡単な文記憶モデルとしては，Miller モデルほど具体的な意味ネットワークを用いなくても，さらに抽象化した単一の意味ネットワーク構造で充分だと考えられる。

Ⅲ　CPSモデル

今回作製した文記憶のシミュレーション・モデルは，便宜上，「CPSモデル」と名付ける。この略称は“Constructive Processing of Sentences”というMiller（1981）の論文の題目に基づいている。CPSモデルは先に述べたMillerモデルの問題点を修正し，より簡単な形に整理されている。まず，CPSモデルではパラメータを，符号化プロセスの処理時間と活性化ノードリストの最大値の2つに限定した。前者は，提示された学習文に対する被験者の心的な処理時間に対応しており，後者は前述のように，被験者が一時的に意識できる情報量の限界に対応している。また，本モデルでは処理プロセスの始めにだけ活性化量を投入し，処理プロセスの途中で活性化量の再投入は行わない。さらに，検索の停止基準は，符号化文脈ネットワークにおけるテスト文の3単語の確認という唯一のルールに限定した。

CPSモデルのシミュレーション・プログラムは，FACOM (1976), Siklóssy (1976), Winston & Horn (1981) 等を参考に，名古屋大学大型計算機センターのLISP1.5を用いて作製した。Miller モデルの実際のプログラムを参照できなかったので，今回作製したプログラムは，Miller（1981）の論文に対する筆者なりの解釈に基づいている。CPSモデルの具体的なプログラムリストは，付録として本稿の最後に示した。以下ではMiller モデルと同様に，活性化システム，符号化プロセス，検索プロセスの順にCPSモデルを説明する。ただし，既に Miller モデルについて詳述したので，CPSモデルの説明は，プログラムの出力結果を提示することにより必要最小限に止める。

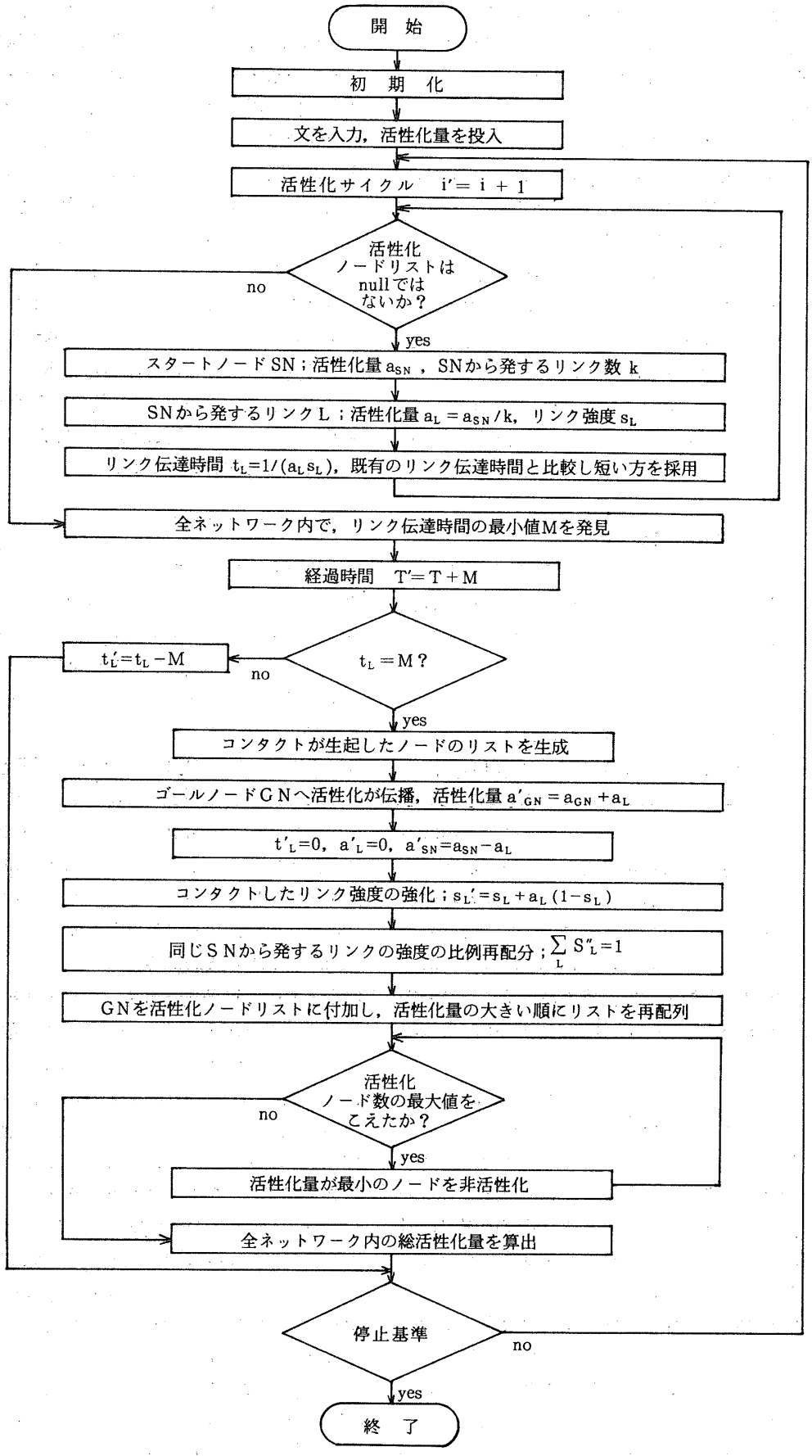


図1　ＣＰＳモデルの活性化システムの概略

1　活性化システム

図1はCPSモデルの活性化システムの概略を，フローチャートの形に整理したものである。活性化の1サイクルは，活性化ノードリストに属するノードSNの活性化量を，SNから出ているリンク間で均等に分配することによって始まる。SNから発するリンクの1つをLとすると，そのリンク活性化量 a_L とリンク強度 s_L[2] から，Lにおけるリンク伝達時間 t_L が算出される。t_L は s_L が強いほど，また，a_L が大きいほど速いと仮定できるので（Anderson, 1976; McClelland, 1979; Miller, 1981），最も単純に次式によって定義する。

$$t_L = 1 / (a_L \, s_L)$$

なお，新たに算出した t_L は，リンクLが既に有しているリンク伝達時間と比較され，小さい方の値が採用される。

こうした処理が，活性化ノードリストに属するノード全てについて実行されると，活性化システムはネットワーク内の全てのリンクのリンク伝達時間を比較し，最小所要時間Mを発見する。この時，リンク伝達時間がMであったリンクは，出発点のノード（スタートノードSN）から到達点のノード（ゴールノードGN）へと活性化を伝播したとみなし，そのリンクとGNとは「コンタクトした」と言うことにする。

コンタクトが生ずると，活性化システムは以下の3つの処理を実行する。第1に，GNの活性化量はコンタクトしたリンクの活性化量だけ増加し，SNの活性化量はその分減少する。第2に，コンタクトしたリンクの強度 s_L は，Miller（1981）に従い，次の関数によって強化される。

$$s_L' = s_L + a_L (1 - s_L)$$[3]

さらに，強化されたリンク強度は，SNから発する全てのリンク間で，リンク強度の合計が1.0となるように比例再配分される。第3に，コンタクトが生起したGNは活性化ノードリストに加えられ，そして，ノード活性化量が大きい順に活性化ノードリストが再配列される。活性化ノードリストの最大値は制限されているので，その結果として，活性化量が小さいノードはリストから除外され，非活性化される。一方，リンク伝達時間が最小値Mよりも長く，コンタクトが生じなかったリンクに関しては，リンク伝達時間からMを引いた値が新たに付与される。

以上が活性化の1サイクルだが，活性化システムは所定の停止基準に達するまでこのサイクルを反復し，ネットワーク表象内で活発な情報のやりとりがなされる。リンク強度は各々の活性化サイクルにおいて，コンタクトによる強化と，スタートノードを共有する別のリンクの強化と比例再配分に伴う強度の減少とを繰り返し，その時点におけるネットワークの状態を要約する意味をもつ。

2　符号化プロセス

符号化プロセスを逐次的に説明する前に，CPSモデルにおける記憶表象についてまとめておく。本モデルでは意味ネットワークを簡略化するため，相互に関連する2つの定義語ノードから構成された抽象的で単一のネットワーク構造によって，全ての単語を規定した。ただし後述するように，今回のシミュレーションでは動詞の類義語が問題となるので，動詞の定義語の1つは類義動詞としたが，ネットワーク構造としては主語や目的語と全く同一である。一方，作業記憶ノードと符号化文脈ノードに関しては Miller モデルに準じた。

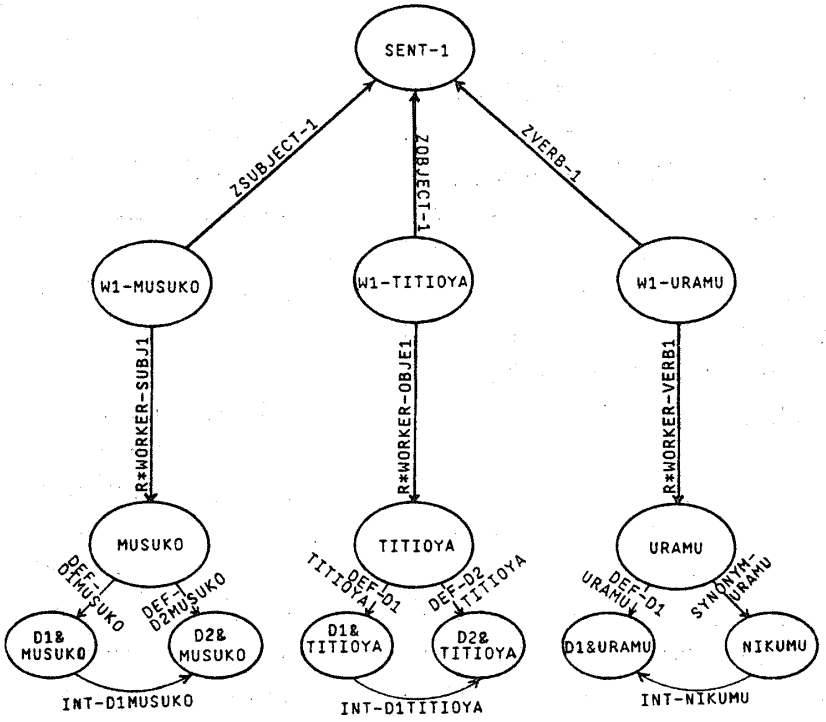
次に，ネットワーク図と出力結果とを参照しながら，符号化プロセスを概観する。CPSモデルは，活性化の拡大を媒介として符号化文脈ネットワークを逐次的に構成してゆく。本モデルで扱う符号化文脈ネットワークは単一であり，文や単語の違いに左右されないが，図を見やすくするために，例として「息子が父親をうらむ。」という学習文（文番号1）をとりあげる。まず図2は，この学習文に対する初期のネットワーク表象を示している。次の図3は，意味記憶に活性化が及んだため，作業記憶ノード間で意味ネットワーク内の相互関係が複製された，第2段階のネットワーク表象を示している。さらに図4は，全てのノードが符号化文脈ノード（SENT—1）を中心に統合され，構成が完了したネットワークを表わしている。これに対して，図5は活性化ノード数を15に設定した場合の，符号化プロセスにおける出力の一部を示している。図2～4のネットワーク表現との対応で言えば，図2のネットワークは図5の活性化サイクル1～4に相当し，図5のサイクル5で，図3のネットワーク段階に達する。さらに，図5のサイクル7で図4の符号化文脈ネットワークが完成され，それ以降は完成したネットワーク内を活性化が循環することになる。

図5の出力結果についてさらに詳しく見てゆくと，120行は最初の活性化ノードリストを示し，190行は活性化の投入と活性化サイクルの開始を示している。250行は最小のリンク伝達時間を，280行はシミュレーショ

2）リンク強度の初期値は，同一ノードから発するリンク数をkとおくと，1／k。

3）リンク強度の初期値を s_0，n回目のコンタクト後のリンク強度を s_n，リンク活性化量をaとすると，一般形は，

$$s_n = 1 - (1 - s_0)(1 - a)^n$$



(ENCODING SENTENCE NO.1)
(ENCODING SENTENCE)=(MUSUKO TITIOYA URAMU)

図2　第1段階の符号化文脈ネットワークの例[4)]

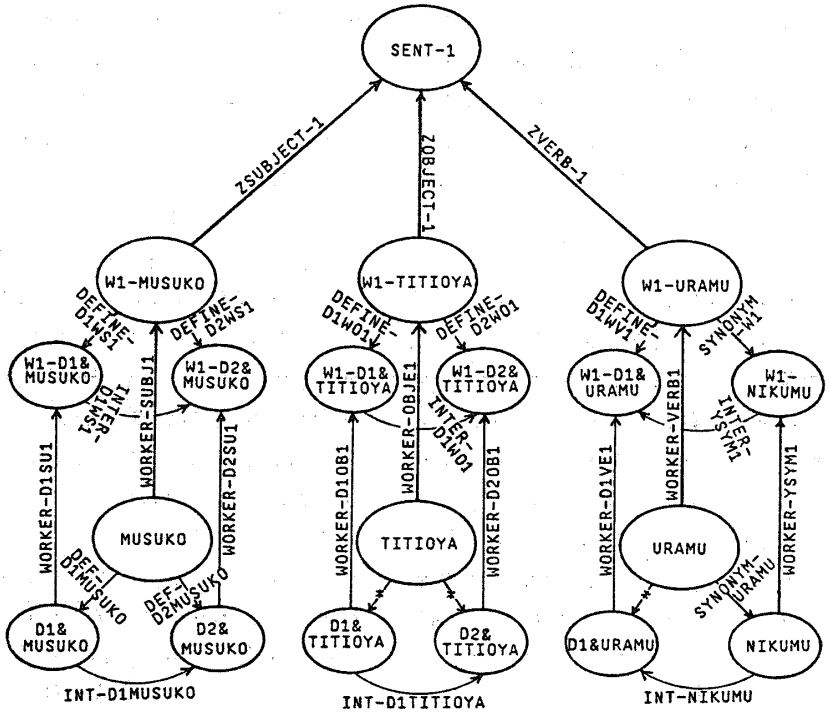


図3　第2段階の符号化文脈ネットワークの例[5)]

4）図中の“R＊”は，逆方向のリンクを示す。

5）図が見にくくなるので，一部のラベル名は“＊”と略記した。

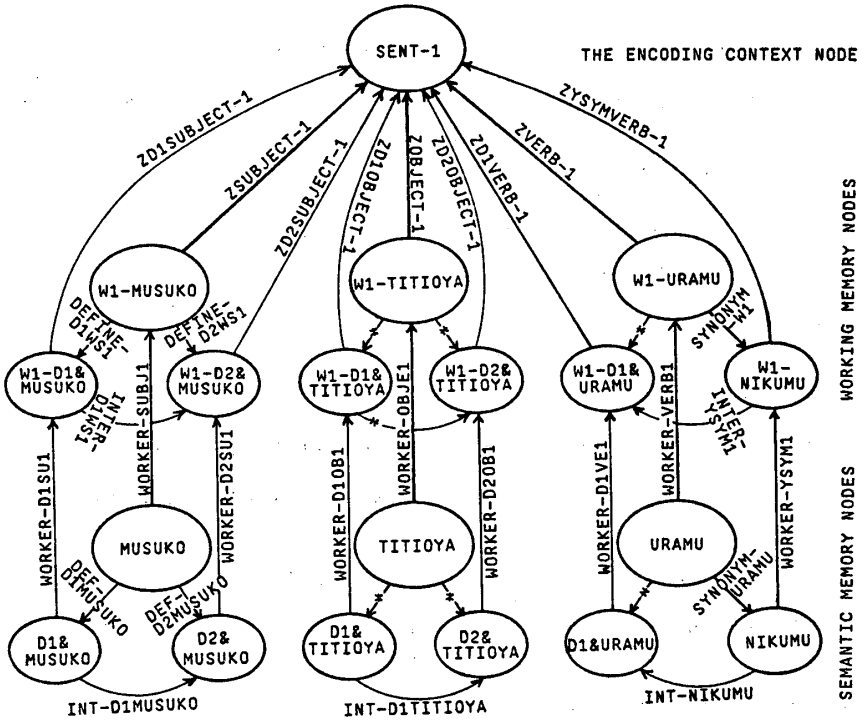


図4　完成された符号化文脈ネットワークの例

```
00010 (*** ALL VARIABLES WERE CLEARED ***)
00020
00030 (ENCODING SENTENCE 1)
00040 (ENCODING SENTENCE)=(MUSUKO TITIOYA URAMU)
00050
00060
00070 (CONSTRUCTED NODES NO1)=(MUSUKO TITIOYA URAMU W1-MUSUKO W1-TITIOYA W1-UR
00080 AMU SENT-1 NIKUMU D1&URAMU D2&TITIOYA D1&TITIOYA D2&MUSUKO D1&MUSUKO)
00090
00100 (*** NETWORK WAS RECONSTRUCTED ***)
00110
00120 (ACTIVE NODES LIST)=(W1-MUSUKO W1-TITIOYA W1-URAMU SENT-1)
00130
00140 (END OF CORRECTING ELPSTIME)
00150 (*** ALL ACTIVATION WAS CLEARED ***)
00160
00170 ################  (ELAPSED TIME) = 1  ################
00180
00190 (*** DIVIDE 15 UNITS OF ACTIVATION AMONG ACTIVE NODES ***)
00200
00210
00220
00230 (********** CYCLE 1 **********)
00240
00250 (MINIMUM TIME)=1.066666E+00
00260
00270 (ELAPSED TIME)=1.000106E+04
00280 ==============*1.066405E-01
00290
00300 (CONTACTED LINKS)=(R*WORKER-VERB1 ZVERB-1 R*WORKER-OBJE1 ZOBJECT-1 R*WOR
00310 KER-SUBJ1 ZSUBJECT-1)
00320
00330 (W1-URAMU =======* URAMU)(W1-URAMU =======* SENT-1)(W1-TITIOYA =======*
00340 TITIOYA)(W1-TITIOYA =======* SENT-1)(W1-MUSUKO =======* MUSUKO)(W1-MUSUK
```

図5－1　符号化プロセスの出力例[6]①

6）入力形式は，ENCODING ((MUSUKO TITIOYA URAMU) 1)。

```
00350 O =======* SENT-1)
00360
00370 (CONTACTED NODES)=(SENT-1 URAMU TITIOYA MUSUKO)
00380
00390 (ACTIVE NODES LIST)=(SENT-1 URAMU TITIOYA MUSUKO W1-MUSUKO W1-TITIOYA W1
00400 -URAMU)
00410
00420 (ACTIVATION)=1.500000E+01
00430
00440
00450
00460 (********** CYCLE 2 **********)
00470
00480 (MINIMUM TIME)=9.600005E-01
00490
00500 (ELAPSED TIME)=1.000202E+04
00510 ==============*2.023437E-01
00520
00530 (CONTACTED LINKS)=(R*ZSUBJECT-1 R*ZOBJECT-1 R*ZVERB-1)
00540
00550 (SENT-1 =======* W1-MUSUKO)(SENT-1 =======* W1-TITIOYA)(SENT-1 =======*
00560 W1-URAMU)
00570
00580 (CONTACTED NODES)=(W1-MUSUKO W1-TITIOYA W1-URAMU)
00590
00600 (ACTIVE NODES LIST)=(W1-MUSUKO W1-TITIOYA W1-URAMU SENT-1 URAMU TITIOYA
00610 MUSUKO)
00620
00630 (ACTIVATION)=1.500000E+01
00640
00650
00660
00670 (********** CYCLE 3 **********)
00680
00690 (MINIMUM TIME)=1.279999E+00
00700
00710 (ELAPSED TIME)=1.000330E+04
00720 ==============*3.300781E-01
00730
00740 (CONTACTED LINKS)=(R*WORKER-VERB1 ZVERB-1 R*WORKER-OBJE1 ZOBJECT-1 R*WOR
00750 KER-SUBJ1 ZSUBJECT-1)
00760
00770 (W1-URAMU =======* URAMU)(W1-URAMU =======* SENT-1)(W1-TITIOYA =======*
00780 TITIOYA)(W1-TITIOYA =======* SENT-1)(W1-MUSUKO =======* MUSUKO)(W1-MUSUK
00790 O =======* SENT-1)
00800
00810 (CONTACTED NODES)=(SENT-1 URAMU TITIOYA MUSUKO)
00820
00830 (ACTIVE NODES LIST)=(SENT-1 URAMU TITIOYA MUSUKO W1-MUSUKO W1-TITIOYA W1
00840 -URAMU)
00850
00860 (ACTIVATION)=1.500000E+01
00870
00880
00890
00900 (********** CYCLE 4 **********)
00910
00920 (MINIMUM TIME)=1.920000E+00
00930
00940 (ELAPSED TIME)=1.000521E+04
00950 ==============*5.218749E-01
00960
00970 (CONTACTED LINKS)=(R*ZSUBJECT-1 R*ZOBJECT-1 R*ZVERB-1)
00980
00990 (SENT-1 =======* W1-MUSUKO)(SENT-1 =======* W1-TITIOYA)(SENT-1 =======*
01000 W1-URAMU)
01010
01020 (CONTACTED NODES)=(W1-MUSUKO W1-TITIOYA W1-URAMU)
01030
01040 (ACTIVE NODES LIST)=(W1-MUSUKO W1-TITIOYA W1-URAMU SENT-1 URAMU TITIOYA
01050 MUSUKO)
```

図5－2　符号化プロセスの出力例②

```
01060
01070 (ACTIVATION)=1.500000E+01
01080
01090
01100
01110 (********** CYCLE 5 **********)
01120
01130 (MINIMUM TIME)=6.399993E-01
01140
01150 (ELAPSED TIME)=1.000585E+04
01160 ===============*5.855468E-01
01170
01180 (CONTACTED LINKS)=(WORKER-SUBJ1 DEF-D1MUSUKO DEF-D2MUSUKO WORKER-OBJE1 D
01190 EF-D1TITIOYA DEF-D2TITIOYA WORKER-VERB1 DEF-D1URAMU SYNONYM-URAMU)
01200
01210 (MUSUKO ========* W1-MUSUKO)(MUSUKO ========* D1&MUSUKO)(MUSUKO ========* D
01220 2&MUSUKO)(TITIOYA ========* W1-TITIOYA)(TITIOYA ========* D1&TITIOYA)(TITI
01230 OYA ========* D2&TITIOYA)(URAMU ========* W1-URAMU)(URAMU ========* D1&URAM
01240 U)(URAMU ========* NIKUMU)
01250
01260 (CONTACTED NODES)=(W1-MUSUKO W1-TITIOYA W1-URAMU D1&MUSUKO D2&MUSUKO D1&
01270 TITIOYA D2&TITIOYA D1&URAMU NIKUMU)
01280
01290 (ACTIVE NODES LIST)=(W1-MUSUKO W1-TITIOYA W1-URAMU D1&MUSUKO D2&MUSUKO D
01300 1&TITIOYA D2&TITIOYA D1&URAMU NIKUMU SENT-1 URAMU TITIOYA MUSUKO)
01310
01320 (ACTIVATION)=1.500000E+01
01330
01340
01350 (CONSTRUCTED NODES NO2)=(MUSUKO TITIOYA URAMU W1-MUSUKO W1-TITIOYA W1-UR
01360 AMU SENT-1 NIKUMU D1&URAMU D2&TITIOYA D1&TITIOYA D2&MUSUKO D1&MUSUKO W1-
01370 D2&TITIOYA W1-D1&TITIOYA W1-D2&MUSUKO W1-D1&MUSUKO)
01380
01390 (*** NETWORK WAS RECONSTRUCTED ***)
01400
01410
01420 (CONSTRUCTED NODES NO3)=(MUSUKO TITIOYA URAMU W1-MUSUKO W1-TITIOYA W1-UR
01430 AMU SENT-1 NIKUMU D1&URAMU D2&TITIOYA D1&TITIOYA D2&MUSUKO D1&MUSUKO W1-
01440 NIKUMU W1-D1&URAMU W1-D2&TITIOYA W1-D1&TITIOYA W1-D2&MUSUKO W1-D1&MUSUKO
01450 )
01460
01470 (*** NETWORK WAS RECONSTRUCTED ***)
01480
01490
01500
01510 (********** CYCLE 6 **********)
01520
01530 (MINIMUM TIME)=1.920000E+00
01540
01550 (ELAPSED TIME)=1.000777E+04
01560 ===============*7.773437E-01
01570
01580 (CONTACTED LINKS)=(R*WORKER-VERB1 ZVERB-1 R*WORKER-OBJE1 ZOBJECT-1 R*WOR
01590 KER-SUBJ1 ZSUBJECT-1)
01600
01610 (W1-URAMU ========* URAMU)(W1-URAMU ========* SENT-1)(W1-TITIOYA ========*
01620 TITIOYA)(W1-TITIOYA ========* SENT-1)(W1-MUSUKO ========* MUSUKO)(W1-MUSUK
01630 O ========* SENT-1)
01640
01650 (CONTACTED NODES)=(SENT-1 URAMU TITIOYA MUSUKO)
01660
01670 (ACTIVE NODES LIST)=(SENT-1 URAMU TITIOYA MUSUKO W1-MUSUKO W1-TITIOYA W1
01680 -URAMU D1&MUSUKO D2&MUSUKO D1&TITIOYA D2&TITIOYA D1&URAMU NIKUMU)
01690
01700 (ACTIVATION)=1.499999E+01
01710
01720
01730
01740 (********** CYCLE 7 **********)
01750
01760 (MINIMUM TIME)=3.987692E+00
```

図5－3　符号化プロセスの出力例③

```
01770
01780 (ELAPSED TIME)=1.001175E+04
01790 ==============*1.175781E+00
01800
01810 (CONTACTED LINKS)=(DEFINE-D1WV1 SYNONYM-W1 DEFINE-D1WO1 DEFINE-D2WO1 DEF
01820 INE-D1WS1 DEFINE-D2WS1)
01830
01840 (W1-URAMU ========* W1-D1&URAMU)(W1-URAMU ========* W1-NIKUMU)(W1-TITIOYA
01850 ========* W1-D1&TITIOYA)(W1-TITIOYA ========* W1-D2&TITIOYA)(W1-MUSUKO ===
01860 =====* W1-D1&MUSUKO)(W1-MUSUKO ========* W1-D2&MUSUKO)
01870
01880 (CONTACTED NODES)=(W1-D1&URAMU W1-NIKUMU W1-D1&TITIOYA W1-D2&TITIOYA W1-
01890 D1&MUSUKO W1-D2&MUSUKO)
01900
01910 (ACTIVE NODES LIST)=(W1-D1&URAMU W1-NIKUMU W1-D1&TITIOYA W1-D2&TITIOYA W
01920 1-D1&MUSUKO W1-D2&MUSUKO SENT-1 URAMU TITIOYA MUSUKO W1-MUSUKO W1-TITIOY
01930 A W1-URAMU D1&MUSUKO D2&MUSUKO)
01940
01950 (ACTIVATION)=1.041665E+01
01960
01970 (*** NETWORK WAS RECONSTRUCTED ***)
01980
01990 (*** ALL NETWORK WAS CONSTRUCTED ***)
02000 (*** NETWORK WAS RECONSTRUCTED ***)
02010
02020
02030 (********** CYCLE 8 **********)
```

（中　略）

図5－4　符号化プロセスの出力例④

ンにおける理論的経過時間を示す。300，310行はコンタクトしたリンクを，330～350行はコンタクトしたSNとGNを表している。390，400行は，コンタクトが生起したGNを加えた新しい活性化ノードリストを示す。また420行は，その時点におけるネットワーク内の総活性化量を示す。以下，同様の活性化サイクルが続くが，冗長になるので第8サイクル以降は省略した。

3　検索プロセス

CPSモデルの検索プロセスは，Miller モデルと同様に，活性化システムによるテスト文の走査として実行される。本モデルでは検索の停止基準を，テスト文の主語，動詞，目的語に対応する作業記憶ノードから，文自体を表す符号化文脈ノードへ至る3つのリンクの発見に限定した。

図6は図5の続きであり，「息子が父親をうらむ。」という学習文を前述のように符号化した後で，「息子が父親をにくむ。」という類義ディストラクター文を検索した結果を示している。図6のサイクル4において，主語と目的語が確認され（5,210行），最後にサイクル6

```
04010 (RETRIEVAL SENTENCE)=(MUSUKO TITIOYA NIKUMU)
04020
04030 (ACTIVE NODES LIST)=(MUSUKO TITIOYA NIKUMU)
04040
04050
04060 ################  (ELAPSED TIME) = 1  ################
04070
04080 (*** DIVIDE 15 UNITS OF ACTIVATION AMONG ACTIVE NODES ***)
04090
04100
04110
04120 (********** CYCLE 1 **********)
04130
04140 (MINIMUM TIME)=1.800000E+00
04150
04160 (ELAPSED TIME)=1.000179E+04
```

図6－1　検索プロセスの出力例[7]①

7）入力形式は，RETRIEVAL ((MUSUKO TITIOYA NIKUMU))。

```
04170 ===============*1.796875E-01
04180
04190 (CONTACTED LINKS)=(SYNONYM-NIKUMU WORKER-YSYM1 INT-NIKUMU WORKER-OBJE1 D
04200 EF-D1TITIOYA DEF-D2TITIOYA WORKER-SUBJ1 DEF-D1MUSUKO DEF-D2MUSUKO)
04210
04220 (NIKUMU ========* URAMU)(NIKUMU ========* W1-NIKUMU)(NIKUMU ========* D1&UR
04230 AMU)(TITIOYA ========* W1-TITIOYA)(TITIOYA ========* D1&TITIOYA)(TITIOYA =
04240 ======* D2&TITIOYA)(MUSUKO ========* W1-MUSUKO)(MUSUKO ========* D1&MUSUKO
04250 )(MUSUKO ========* D2&MUSUKO)
04260
04270 (CONTACTED NODES)=(URAMU W1-NIKUMU D1&URAMU W1-TITIOYA D1&TITIOYA D2&TIT
04280 IOYA W1-MUSUKO D1&MUSUKO D2&MUSUKO)
04290
04300 (POSITIVE EVIDECE LINKS)=NIL
04310      (LINKSTRG SUMATION)=0.000000E-78
04320 (NEGATIVE EVIDENCE LINKS)=NIL
04330      (LINKSTRG SUMATION)=0.000000E-78
04340
04350
04360 (ACTIVE NODES LIST)=(URAMU W1-NIKUMU D1&URAMU W1-TITIOYA D1&TITIOYA D2&T
04370 ITIOYA W1-MUSUKO D1&MUSUKO D2&MUSUKO MUSUKO TITIOYA NIKUMU)
04380
04390 (ACTIVATION)=1.500000E+01
04400
04410
04420
04430 (********** CYCLE 2 **********)
04440
04450 (MINIMUM TIME)=5.400000E+00
04460
04470 (ELAPSED TIME)=1.000719E+04
04480 ===============*7.195312E-01
04490
04500 (CONTACTED LINKS)=(R*DEF-D2MUSUKO WORKER-D2SU1 INT-D2MUSUKO R*DEF-D1MUSU
04510 KO WORKER-D1SU1 INT-D1MUSUKO R*DEF-D2TITIOYA WORKER-D2OB1 INT-D2TITIOYA
04520 R*DEF-D1TITIOYA WORKER-D1OB1 INT-D1TITIOYA R*DEF-D1URAMU WORKER-D1VE1 IN
04530 T-D1URAMU WORKER-VERB1 DEF-D1URAMU SYNONYM-URAMU)
04540
04550 (D2&MUSUKO ========* MUSUKO)(D2&MUSUKO ========* W1-D2&MUSUKO)(D2&MUSUKO =
04560 ======* D1&MUSUKO)(D1&MUSUKO ========* MUSUKO)(D1&MUSUKO ========* W1-D1&M
04570 USUKO)(D1&MUSUKO ========* D2&MUSUKO)(D2&TITIOYA ========* TITIOYA)(D2&TIT
04580 IOYA ========* W1-D2&TITIOYA)(D2&TITIOYA ========* D1&TITIOYA)(D1&TITIOYA
04590 ========* TITIOYA)(D1&TITIOYA ========* W1-D1&TITIOYA)(D1&TITIOYA ========*
04600  D2&TITIOYA)(D1&URAMU ========* URAMU)(D1&URAMU ========* W1-D1&URAMU)(D1&
04610 URAMU ========* NIKUMU)(URAMU ========* W1-URAMU)(URAMU ========* D1&URAMU)
04620 (URAMU ========* NIKUMU)
04630
04640 (CONTACTED NODES)=(MUSUKO TITIOYA NIKUMU W1-D2&MUSUKO D1&MUSUKO W1-D1&MU
04650 SUKO D2&MUSUKO W1-D2&TITIOYA D1&TITIOYA W1-D1&TITIOYA D2&TITIOYA URAMU W
04660 1-D1&URAMU W1-URAMU D1&URAMU)
04670
04680 (POSITIVE EVIDECE LINKS)=NIL
04690      (LINKSTRG SUMATION)=0.000000E-78
04700 (NEGATIVE EVIDENCE LINKS)=NIL
04710      (LINKSTRG SUMATION)=0.000000E-78
04720
04730
04740 (ACTIVE NODES LIST)=(MUSUKO TITIOYA NIKUMU W1-D2&MUSUKO D1&MUSUKO W1-D1&
04750 MUSUKO D2&MUSUKO W1-D2&TITIOYA D1&TITIOYA W1-D1&TITIOYA D2&TITIOYA URAMU
04760  W1-D1&URAMU W1-URAMU D1&URAMU)
04770
04780 (ACTIVATION)=1.000000E+01
04790
04800
04810
04820 (********** CYCLE 3 **********)
04830
04840 (MINIMUM TIME)=8.099992E+00
04850
04860 (ELAPSED TIME)=1.001529E+04
04870 ===============*1.529296E+00
```

図6－2　検索プロセスの出力例②

```
04880
04890 (CONTACTED LINKS)=(SYNONYM-NIKUMU WORKER-YSYM1 INT-NIKUMU WORKER-OBJE1 D
04900 EF-D1TITIOYA DEF-D2TITIOYA WORKER-SUBJ1 DEF-D1MUSUKO DEF-D2MUSUKO)
04910
04920 (NIKUMU =======* URAMU)(NIKUMU =======* W1-NIKUMU)(NIKUMU =======* D1&UR
04930 AMU)(TITIOYA =======* W1-TITIOYA)(TITIOYA =======* D1&TITIOYA)(TITIOYA =
04940 ======* D2&TITIOYA)(MUSUKO =======* W1-MUSUKO)(MUSUKO =======* D1&MUSUKO
04950 )(MUSUKO =======* D2&MUSUKO)
04960
04970 (CONTACTED NODES)=(URAMU D1&URAMU D1&TITIOYA D2&TITIOYA D1&MUSUKO D2&MUS
04980 UKO W1-NIKUMU W1-TITIOYA W1-MUSUKO)
04990
05000 (POSITIVE EVIDECE LINKS)=NIL
05010      (LINKSTRG SUMATION)=0.000000E-78
05020 (NEGATIVE EVIDENCE LINKS)=NIL
05030      (LINKSTRG SUMATION)=0.000000E-78
05040
05050
05060 (ACTIVE NODES LIST)=(URAMU D1&URAMU D1&TITIOYA D2&TITIOYA D1&MUSUKO D2&M
05070 USUKO W1-NIKUMU W1-TITIOYA W1-MUSUKO MUSUKO TITIOYA NIKUMU W1-D2&MUSUKO
05080 W1-D1&MUSUKO W1-D2&TITIOYA)
05090
05100 (ACTIVATION)=8.333335E+00
05110
05120
05130
05140 (********** CYCLE 4 **********)
05150
05160 (MINIMUM TIME)=2.338092E+00
05170
05180 (ELAPSED TIME)=1.001762E+04
05190 ==============*1.762889E+00
05200
05210 (CONTACTED LINKS)=(R*WORKER-SUBJ1 ZSUBJECT-1 R*WORKER-OBJE1 ZOBJECT-1)
05220
05230 (W1-MUSUKO =======* MUSUKO)(W1-MUSUKO =======* SENT-1)(W1-TITIOYA ======
05240 =* TITIOYA)(W1-TITIOYA =======* SENT-1)
05250
05260 (CONTACTED NODES)=(SENT-1 MUSUKO TITIOYA)
05270
05280 (ZOBJECT-1 LINKSTRG 3.101540E-01 )
05290 (ZSUBJECT-1 LINKSTRG 3.101540E-01 )
05300 (POSITIVE EVIDECE LINKS)=(ZSUBJECT-1 ZOBJECT-1)
05310      (LINKSTRG SUMATION)=6.203081E-01
05320 (NEGATIVE EVIDENCE LINKS)=NIL
05330      (LINKSTRG SUMATION)=0.000000E-78
05340
05350
05360 (ACTIVE NODES LIST)=(SENT-1 MUSUKO TITIOYA URAMU D1&URAMU D1&TITIOYA D2&
05370 TITIOYA D1&MUSUKO D2&MUSUKO W1-NIKUMU W1-TITIOYA W1-MUSUKO NIKUMU W1-D2&
05380 MUSUKO W1-D1&MUSUKO)
05390
05400 (ACTIVATION)=7.777780E+00
05410
05420
05430
05440 (********** CYCLE 5 **********)
05450
05460 (MINIMUM TIME)=1.261906E+00
05470
05480 (ELAPSED TIME)=1.001889E+04
05490 ==============*1.889061E+00
05500
05510 (CONTACTED LINKS)=(R*WORKER-YSYM1 R*SYNONYM-W1 INTER-YSYM1)
05520
05530 (W1-NIKUMU =======* NIKUMU)(W1-NIKUMU =======* W1-URAMU)(W1-NIKUMU =====
05540 ==* W1-D1&URAMU)
05550
05560 (CONTACTED NODES)=(NIKUMU W1-URAMU W1-D1&URAMU)
05570
05580 (POSITIVE EVIDECE LINKS)=(ZSUBJECT-1 ZOBJECT-1)
05590      (LINKSTRG SUMATION)=6.633193E-01
```

図6—3　検索プロセスの出力例③

```
05600 (NEGATIVE EVIDENCE LINKS)=NIL
05610       (LINKSTRG SUMATION)=0.000000E-78
05620
05630
05640 (ACTIVE NODES LIST)=(NIKUMU W1-URAMU W1-D1&URAMU SENT-1 MUSUKO TITIOYA U
05650 RAMU D1&URAMU D1&TITIOYA D2&TITIOYA D1&MUSUKO D2&MUSUKO W1-NIKUMU W1-TIT
05660 IOYA W1-MUSUKO)
05670
05680 (ACTIVATION)=6.666668E+00
05690
05700
05710
05720 (********** CYCLE 6 **********)
05730
05740 (MINIMUM TIME)=3.000002E+00
05750
05760 (ELAPSED TIME)=1.002189E+04
05770 ==============*2.189062E+00
05780
05790 (CONTACTED LINKS)=(ZYSYMVERB-1)
05800
05810 (W1-NIKUMU =======* SENT-1)
05820
05830 (CONTACTED NODES)=(SENT-1)
05840
05850 (ZYSYMVERB-1 LINKSTRG 1.661538E-01 )
05860 (POSITIVE EVIDECE LINKS)=(ZYSYMVERB-1 ZSUBJECT-1 ZOBJECT-1)
05870       (LINKSTRG SUMATION)=8.294731E-01
05880 (NEGATIVE EVIDENCE LINKS)=NIL
05890       (LINKSTRG SUMATION)=0.000000E-78
05900
05910
05920 (ACTIVE NODES LIST)=(SENT-1 NIKUMU W1-URAMU W1-D1&URAMU MUSUKO TITIOYA U
05930 RAMU D1&URAMU D1&TITIOYA D2&TITIOYA D1&MUSUKO D2&MUSUKO W1-NIKUMU W1-TIT
05940 IOYA W1-MUSUKO)
05950
05960 (ACTIVATION)=6.666668E+00
05970
05980
05990 (##### RETRIEVAL WAS COMPLETED #####)
06000
06010 (RETRIEVAL SENTENCE)=(MUSUKO TITIOYA NIKUMU)
06020 (LAST CYCLE)=6
06030 (POSITIVE EVIDENCE LINKS LIST)=(ZYSYMVERB-1 ZSUBJECT-1 ZOBJECT-1)
06040   (POSITIVE LINKSTRG SUMATION)=8.452632E+00
06050 (NEGATIVE EVIDENCE LINKS LIST)=NIL
06060   (NEGATIVE LINKSTRG SUMATION)=0.000000E-78
06070 (REACTION TIME)=2.189062E+00
06080
06090 (ZYSYMVERB-1 LINKSTRG 1.819440E-01 )
06100 (ZSUBJECT-1 LINKSTRG 3.316596E-01 )
06110 (ZOBJECT-1 LINKSTRG 3.316596E-01 )
```

図6－4　検索プロセスの出力例④

で類義動詞が確認されたので（5,790行），反応は誤警報である。つまり，CPSモデルでは符号化プロセスにおいて，学習文の動詞の類義動詞に対応する作業記憶ノードから符号化文脈ノードへ至るリンクは，何度も活性化されてリンク強度が増加しているため，特定の条件下では人間の被験者と同様に，類義ディストラクター文に対してイエス反応を出力するという誤り（類義誤警報）も生ずる。図6の出力結果の最後には，確認されたリンクのリスト（6,030行），確信度に相当するこれらのリンク強度の合計（6,040行），反応時間（6,070行）が示されている。

Ⅳ　実験のシミュレーション

1　文記憶に関する実験的研究

文記憶に関する研究では，文の表現形式や個々の単語といった逐語的な（verbatim）情報と，文全体から構成される意味的な情報の保持成績を比較することによって，文の記憶表象や保持特性に関する検討がなされてきた。

従来の研究では，Brewer（1975）やSachs（1967, 1974）に代表されるように，再認テストにおいて逐語的情報は約10秒で検出できなくなるが，意味的情報はその後も検出されることから，文の意味的情報は長期的に保

持されるが，逐語的情報は急速に減衰すると主張された。これに対して，Anderson & Paulson (1977) は反応時間を測度として実験を行った結果，約8分後の遅延再認時にも，逐語的情報がある程度保持されていることを見い出した。さらに，Hayes-Roth & Hayes-Roth (1977) や都築 (1983) は，被験者に単文を偶発学習させた後で，表1に示したように，文中の動詞を入れかえたテスト文を用いて遅延再認テストを行い，再認率，反応時間，確信度を測定した。その結果，文の意味的情報だけでなく逐語的情報も，少なくとも5分間は保持されていることが見い出された。こうした知見は，反応時間や確信度といった比較的敏感な測度を用いることによって，文の逐語的情報が従来主張されているほど急速に減衰するわけではなく，ある程度の期間は保持されていることを示している。

次節以降では，CPSモデルによるHayes-Roth & Hayes-Roth (1977) や都築 (1983) の実験のシミュレーションについて述べるが，次節ではまず，パラメータ推定の手続きについて説明する。

表1 再認テスト文の構成

文の種類	文の構成
学習文	「S_1がO_1をV_1†。」
	「S_2がO_2をV_2。」
類義ディストラクター文	「S_1がO_1をSV_1††。」
	「S_2がO_2をSV_2。」
非類義ディストラクター文	「S_1がO_1をV_2。」
	「S_2がO_2をV_1。」
	「S_1がO_1をSV_2。」
	「S_2がO_2をSV_1。」

†英語のS－V－O型文は，日本語のS－O－V型文に相当する。
††SV_1 はV_1 の類義動詞を示す。

2 パラメータ推定

前述のようにCPSモデルは，活性化ノードリストの最大値と符号化プロセスの処理時間という2つの主要なパラメータをもつ。前記の文記憶実験に対応したパラメータの推定をシステマティックに行うため，次のような手続きを用いた。

Ⅲ－3節で述べたように，CPSモデルでは，検索時における作業記憶ノードから符号化文脈ノードへ至るリンクが重要であり，また，今回のシミュレーションでは動詞と類義動詞の弁別が大きな意味をもつ。従って，パラメータ推定の手がかりとして，学習文の動詞の作業記憶ノードから符号化文脈ノードへのリンク（ラベル名は，ZVERB—x ; xは文番号）と，類義動詞の作業記憶

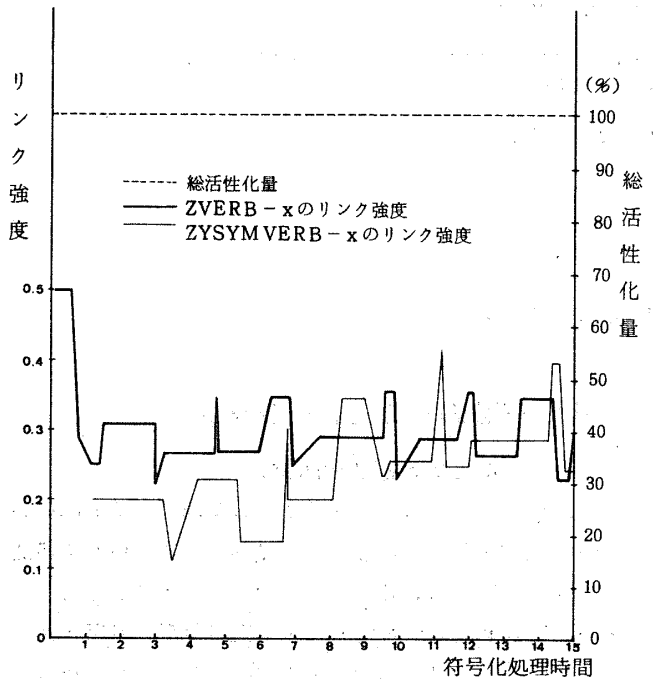


図7 符号化処理時間の経過に伴うリンク強度と総活性化量の変化
——活性化ノード数19(100%)の場合——

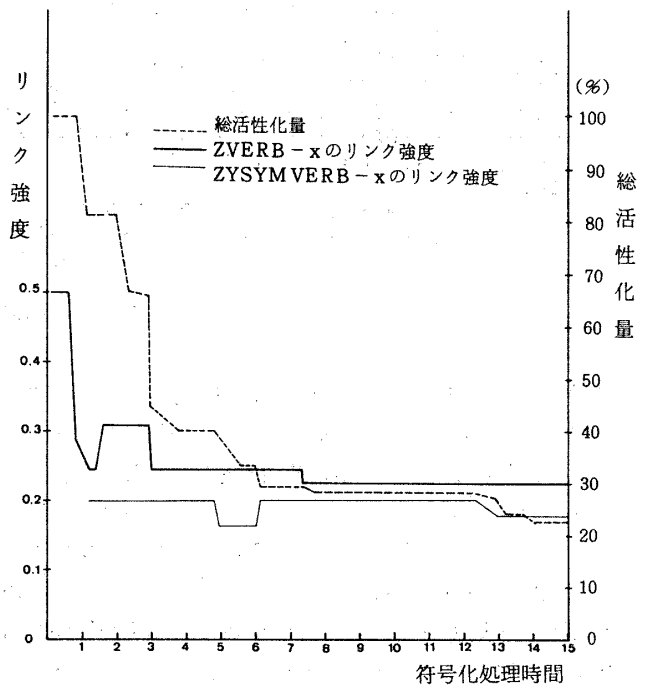


図8 符号化処理時間の経過に伴うリンク強度と総活性化量の変化
——活性化ノード数17(89%)の場合——

ノードから符号化文脈ノードへのリンク（ラベル名は，ZYSYMVERB－x）の強度に着目した。具体的には，活性化ノードリストの最大値を上限値の19（全ノード数に対して100%）から順に，17（89%），15（79%），13（68%）と4水準に変化させ，リンクZVERB－xとZYSYMVERB－xのリンク強度の変化と，ネットワーク内の総活性化量の減衰を調べることにより，最適と考えられる活性化ノード数と符号化処理時間の決定を試みた。その際のデータを，図7～10に示す。

まず図7は，活性化ノード数が全ノード数19と等しい場合の，ZVERB－xとZYSYMVERB－xのリンク強度と総活性化量を，符号化プロセスにおけるコンタクト時点ごとにプロットしたものである。この条件では活性化量が減衰しないので，2つのリンク強度はかなり時間が経過しても激しく変動しており，検索結果の予測は困難である。また，人間の情報処理過程において，活性化量は時間とともに減衰すると一般には主張されており（Anderson, 1983；Hayes-Roth & Hayes-Roth, 1977；McClelland, 1979；McClelland & Rumelhart, 1981），この点からもこの条件は不適切である。

図8は，活性化ノード数が17（89%）の場合である。図に示されたように，活性化量の減衰は比較的ゆるやかであり，処理時間が経過しても投入された活性化量の約20%が保持されているため，2つのリンク強度は時間が経過しても収束せずに変動している。

図9は，活性化ノード数が15（79%）の場合である。図に示されたように活性化量の減衰は急だが，処理時間が経過しても約10%の活性化量が保持され，リンク強度は比較的早い時点で収束している。

最後に図10は，活性化ノード数が13（68%）の場合である。図から明らかなように活性化量の減衰は非常に急激であり，そのためリンク強度はほとんど変動していない。

以上のデータから，ある程度の活性化量を残した状態で2つのリンク強度が収束する，図9に示した活性化ノード数15（79%）の条件が，4条件の中で最も適切であると考えられる。また，Anderson（1983）も，活性化の減衰率を20%，維持率を80%としてシミュレーションを行っており，この値は図9の活性化ノード数15の場合（維持率79%）と極めて近い。

図9はリンク強度の変化によって，処理時間を①0.00～1.21，②1.22～4.92，③4.93～6.09，④6.10以降の4段階に分けることができる。その各段階において，1文を符号化して1文を検索する単純なシミュレーションを行った結果，第2段階が最も適切であったため，この段階の一時点（処理時間＝2.95，活性化サイクル＝15）に符号化処理時間を決定した。

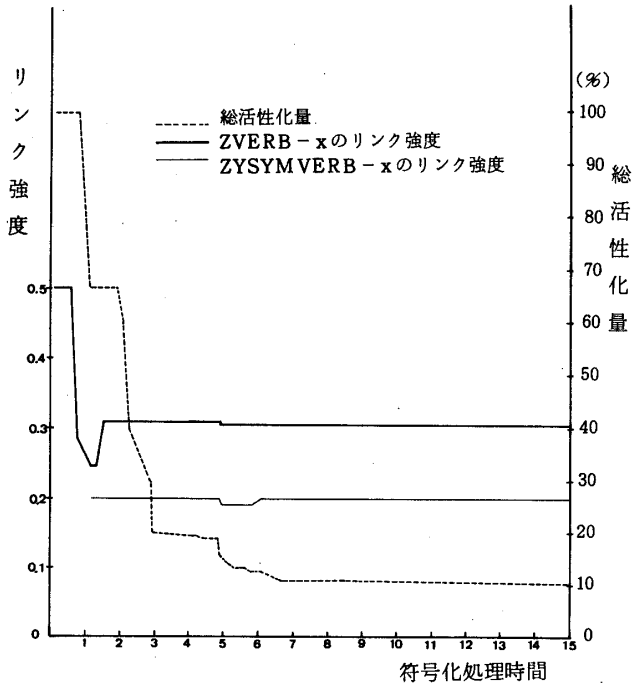


図9　符号化処理時間の経過に伴うリンク強度と総活性化量の変化
——活性化ノード数15(79%)の場合——

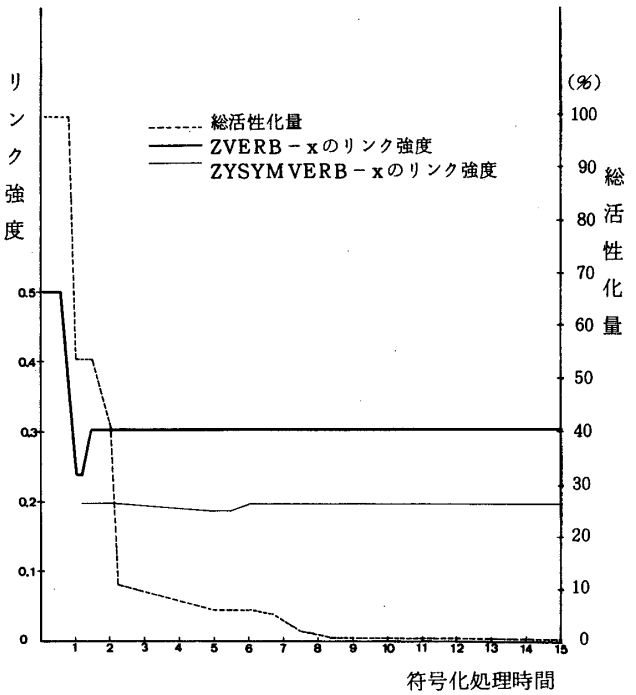


図10　符号化処理時間の経過に伴うリンク強度と総活性化量の変化
——活性化ノード数13(68%)の場合——

3　シミュレーション

表1に示したように，前述の文記憶実験のテスト文は，学習文と3種類のディストラクター文から構成されている。今回のシミュレーションでは，推定したパラメータを用いて，4種類の再認テスト文の全順列である24通りについてプログラムを実行させた。つまり，対になった2文を符号化処理した後，4回の検索を連続して行う過程を，24通りの順列全てについて実行した。

遅延再認の実験事態に対応して，今回のシミュレーションでは符号化処理の後に，リンク強度以外の活性化に関する属性値を全て消去した。また，実験事態では，主語と目的語が同じである4種類のテスト文の検索は，連続して行われず相互に隔たっており，その間に活性化量は減衰してしまうと仮定できる。従って，シミュレーションにおいても，毎回検索が終了するたびに活性化量に関する属性値を全て消去した。しかし，記憶痕跡を代表するリンク強度は毎回保持され，その結果として前回の検索が次の検索に影響を及ぼした。

表2に，24通りのシミュレーションによる反応率，反応時間，確信度の平均と標準偏差を示す。イエス反応について，まず反応率を比較すると，正再認率と類義誤警報率は等しく（87.5%），非類義誤警報率はそれよりもかなり低い（18.8%）。この結果は，再認率を測度とすると，遅延再認テスト時には意味的情報だけが優位であるとする知見（例えば，Sachs, 1967, 1974）と一致している。しかしながら，反応時間と確信度に関しては，表1から明らかなように，正再認，類義誤警報，非類義誤警報の順で反応時間は速く，確信度は大きい値が示されている。この結果は，反応率よりも敏感な測度である反応時間と確信度においては，非類義誤警報と類義誤警報・正再認の差に反映される意味的情報の保持だけでなく，類義誤警報と正再認の差に反映される逐語的情報の保持も確認できたことを示しており，Hayes-Roth & Hayes-Roth（1977）や都築（1983）の結果と一致している。ネットワーク活性化説の観点からすれば，文記憶において意味的情報が逐語的情報よりも優位であるという Sachs らの知見は，意味的情報が長期記憶に転送されやすいことによるのではなく，逆に，単語を構成単位としたネットワーク表象が活性化され，意味的に関連した単語ノード間で活性化が拡大した結果であると解釈できる（都築，1983）。

従って結論として，CPSモデルは，文記憶における意味的情報と逐語的情報の保持に関する従来の知見を，うまくシミュレートしているとみなすことができる。

V　考　察

本研究は，文記憶における記憶表象の構造と活性化による処理プロセスを扱うため，Miller（1981）の研究を参考にシミュレーション・モデルを作製し，シミュレーション結果と実験データとを比較することによって，文記憶のネットワーク活性化説の有効性を確認する目的で行った。前節で述べたように，この所期の目的はほぼ達せられたと考えられる。

しかし，問題点は少なくない。今回作製したCPSモデルは，約1,500行にわたるプログラム・リストからもわかるように，かなり複雑で大がかりなシステムである。このモデルの一般的な特徴として，①シミュレーション・クロックをもつ動的で構成的なモデルである，②離散的な変量（活性化ノード数）と連続的な変量（他の活性化に関する属性値）をもつ混合モデルである，③確率論的（stochastic）モデルではなく決定論的（deterministic）モデルである，といった3点をあげることができる。これに対して，最近提出されたAnderson（1983）のACT*モデルは，CPSモデルと同様に文記憶のネットワーク活性化モデルの1つであるが，①試行錯誤なしに行列計算によって漸近的な最適解が得られる，解析的かつ静

表2　シミュレーション結果

反　応	反応数（%）	反応時間 平均（SD）	確信度 平均（SD）
正再認	21（87.5）	2.02（0.59）	8.42（0.99）
類義誤警報	21（87.5）	2.40（0.57）	6.23（1.94）
非類義誤警報	9（18.8）	2.76（0.21）	4.68（1.05）
ミス	3（12.5）	2.28（0.85）	5.20（1.20）
類義正棄却	3（12.5）	3.16（0.17）	7.91（0.14）
非類義正棄却	39（81.3）	2.19（0.58）	7.91（1.14）

的なモデルである，②変量の特性において連続的モデルである，③確率論的モデルである，といった特徴をもつ。つまり，ACT*モデルの活性化システムは，ネットワーク構造を仮定した確率論的な数理モデルの形をとっている。シミュレーションにおけるモデルの安定性という観点からすれば，解析的なACT*モデルの方が，プロセスを重視した今回のCPSモデルよりも優れていると考えられる。しかしながら，CPSモデルがあえて上記の方略を用いた背景には，人間の情報処理システムに関する現段階の知見とシミュレーション・モデルとの間に，単なる機能的な等価性だけでなく，機能を生み出す統合的な構造的等価性（安西，1984）を実現しようとする試みがあったと言うことができる。こうした方向性が，検証不可能なリアリティーをモデルに付与するのみにすぎないと考えるか，構造的等価性の追究こそが心理学的モデルの目標であると考えるかは，議論の分かれるところであろう。

これまでの考察から示唆されたように，今回作製したCPSモデルは1つの可能性にすぎず，同程度の説明力をもつ複数のモデルが存在しうる。その際に，どのモデルが心理学的妥当性をもつかを決定する基準は，最終的にはより精密な実験データにあることは言うまでもない。コンピュータ・シミュレーションという技法は，例えば，文記憶のネットワーク活性化説といった複雑な理論を厳密化し，様々なパラメータや処理プロセスを，同時にシステマティックに操作しうる優れた研究方法ではあるが，多くの選択肢の中から厳密化の方向を決定する実験データが，まだまだ不足していると考えられる。

文　　献

Anderson, J.R. 1976 *Language, memory, and thought.* Hillsdale, N.J. : Lawrence Erlbaum Associates.

Anderson, J.R. 1983 *The architecture of cognition.* Cambridge, MA : Harvard University Press.

Anderson, J.R., Kline, P.J., & Beasley, Jr. C.M. 1979 A general learning theory and its application to schema abstraction. In G.H. Bower (Ed.). *The Psychology of Learning and Motivation,* vol. **13**, Academic Press.

Anderson, J.R., & Paulson, R. 1977 Representation and retention of verbatim information. *Journal of Verbal Learning and Verbal Behavior,* **16**, 439-452.

安西祐一郎　1984　認識研究におけるコンピュータシミュレーションの意義と限界　理想　№617，144—169.

Brewer, W.F. 1975 Memory for ideas : Synonym substitution. *Memory & Cognition*, **3**, 458-464.

Collins, A.M., & Loftus, E.L. 1975 A spreading activation theory of semantic processing. *Psychological Review,* **82**, 407-428.

Collins, A.M., & Quillian, M.R. 1969 Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior,* **8**, 240-247.

FACOM 1976 FACOM OS IV LISP手引書　富士通

Fillmore, C.J. 1968 The case for case. In E. Bach & R.T. Harms (Eds.), *Universals in linguistic theory.* Chicago : Holt, Rinehart & Winston.

Hayes-Roth, B. 1977 Evolution of cognitive structures and processes. *Psychological Review*, **84**, 260-278.

Hayes-Roth, B., & Hayes-Roth, F. 1977 The prominence of lexical information in memory : Representations of meaning, *Journal of Verbal Learning and Verbal Behavior*, **16**, 119-136.

Kieras, D.E. 1981 Component processes in the comprehension of simple prose. *Journal of Verbal Learning and Verbal Behavior,* **20**, 1-23.

McClelland, J.L. 1979 On the time relations of mental processes : An examination of systems of processes in cascade. *Psychological Review,* **86**, 287-330.

McClelland, J.L., & Rumelhart, D. 1981 An interactive activation model of context effects in letter perception : Part 1. An account of basic findings. *Psychological Review*, **88**, 375-407.

Miller, J.R. 1981 Constructive processing of sentences : A simulation model of encoding and retrieval. *Journal of Verbal Learning and Verbal Behavior,* **20**, 24-45.

Sachs, J.S. 1967 Recognition memory for syntactic and semantic aspects of connected discourse. *Perception and Psychophysics,* **2**, 437-442.

Sachs, J.S. 1974 Memory in reading and listening to discourse. *Memory & Cognition,* **2**, 95-100.

Siklóssy, L. 1976 *Let's talk LISP.* Prentice-Hall.（後藤英一・戸島　熙訳　1981　LISP入門　日本コンピュータ協会）

都築誉史　1983　単文偶発学習課題による word-based network モデルの検討　日本心理学会第47回大会発

表論文集，282.
都築誉史　1984　単文記憶におけるネットワーク活性化モデルのコンピュータ・シミュレーション　日本認知科学会第1回大会発表論文集，40—41.
Winston, P.H., & Horn, B.K.P. 1981 *LISP*. Addison-Wesley.

（1985年7月31日　受稿）

付　表

```
00010 SQC(')
00020 DEFINE((
00030 (COMMENT (LAMBDA NIL
00040   (PRINT (LIST '***CPS12*** 'ON1984/10/3 'BY
00050   'TAKASHI 'TSUZUKI)) )) ))
00060 CSET(VMXAL 15)
00070 CSET(QTIME 2.953515)
00080 CSET(INTVL 10000)
00090 CSET(P P)
00100 CSET(XXX XXX)
00110 CSET(NOWBG NOWBG)
00120 CSET(NOWP1 NOWP1)
00130 CSET(NOW15 NOW15)
00140 CSET(NOWP2 NOWP2)
00150 CSET(NOWP3 NOWP3)
00160 CSET(NOWP4 NOWP4)
00170 DEFINE((
00180 (ENCODINGCS (LAMBDA NIL
00190   (PROG NIL
00200     (CSETQ VCYCL 15) (CSETQ VSTIM 10.0)
00210     (CSETQ VACTI 15.0) (CSETQ VLKTH 1.0)
00220     (CSETQ ZTIME 0.0) (CSETQ ZTIM2 0.0)
00230     (CSETQ MTIME 0.0)
00240     (CSETQ DISBG NOWBG) (CSETQ DISP1 NOWP1)
00250     (CSETQ DIS15 NOW15) (CSETQ DISP2 NOWP2)
00260     (CSETQ DISP3 NOWP3) (CSETQ DISP4 NOWP4)
00270     (CSETQ CUEP2 NIL) (CSETQ CUEP3 NIL)
00280     (CSETQ CUEP4 NIL) (CSETQ CUE45 NIL)
00290     (CSETQ PUTAC NIL) (CSETQ DTIME NIL)
00300     (CSETQ VTIME 0.0) (CSETQ LKACT NIL)
00310     (CSETQ TLIST NIL) (CSETQ ANSWR VACTI)
00320     (PRINT (LIST '*** 'ALL 'VARIABLES 'WERE
00330     'CLEARED '***))
00340     (RETURN NIL) ))) ))
00350 DEFINE((
00360 (RETRIEVALCS (LAMBDA NIL
00370   (PROG NIL
00380     (CSETQ VCYCL 50)
00390     (CSETQ NESTG 0.6) (CSETQ VSTIM 10.0)
00400     (CSETQ VACTI 15.0) (CSETQ VLKTH 1.0)
00410     (CSETQ ZTIME 0.0) (CSETQ ZTIM2 0.0)
00420     (CSETQ MTIME 0.0) (CSETQ PESUM 0.0)
00430     (CSETQ NESUM 0.0)
00440     (CSETQ TLIST NIL) (CSETQ ZLIST NIL)
00450     (CSETQ PELIS NIL) (CSETQ NELIS NIL)
00460     (CSETQ DTIME NIL) (CSETQ VTIME 0.0)
00470     (CSETQ RVLST NIL) (CSETQ ANSWR VACTI)
00480     (CSETQ VLINK XXX) (CSETQ SVLNK XXX)
00490     (PUT VLINK 'LINKSTRG 10)
00500     (PUT SVLNK 'LINKSTRG 10)
00510     (PRINT (LIST '*** 'ALL 'VARIABLES 'WERE
00520     'CLEARED '***))
00530     (RETURN NIL) ))) ))
00540 DEFINE((
00550 (FONUMNAME (LAMBDA NIL
00560   (PROG NIL
00570     (PUT 1 'NAME '¥¥¥1¥) (PUT 2 'NAME '¥¥¥2¥)
00580     (PUT 3 'NAME '¥¥¥3¥) (PUT 4 'NAME '¥¥¥4¥)
00590     (PUT 5 'NAME '¥¥¥5¥) (PUT 6 'NAME '¥¥¥6¥)
00600     (PUT 7 'NAME '¥¥¥7¥) (PUT 8 'NAME '¥¥¥8¥)
00610     (PUT 9 'NAME '¥¥¥9¥) (PUT 0 'NAME '¥¥¥0¥))))))
00620 DEFINE((
00630 (PUT (LAMBDA (ATM INDIC PROP)
00640   (DEFLIST (LIST (LIST ATM PROP)) INDIC) )) ))
00650 DEFINE((
00660 (SPACES (LAMBDA (N)
00670   (COND
00680     ((ZEROP N) NIL)
00690     (T (PROG2 (PRIN1 BLANK) (SPACES (SUB1 N)) )))
00700 )) ))
00710 DEFINE((
00720 (UNPACK (LAMBDA (ATM)
00730   (PROG (ANS LIS)
00740     (SETQ ANS NIL)
00750     (SETQ LIS (EXPLODE ATM))
00760       LOOP
00770         (COND
00780           ((NULL LIS) (RETURN ANS))
00790           (T (SETQ ANS (CONS (RLIT (CAR LIS))
00800              ANS) )))
00810     (SETQ LIS (CDR LIS))
00820     (GO LOOP) ) ) )) ))
00830 DEFINE((
00840 (PACK (LAMBDA (LIS)
00850   (PROG (ANS)
00860     (SETQ ANS NIL)
00870       LOOP
00880         (COND
00890           ((NULL LIS) (RETURN (MKATOM)) ))
00900         (SETQ ANS (APPEND ANS (UNPACK (CAR
00910         LIS))))
00920         (SETQ LIS (CDR LIS))
00930         (GO LOOP) ))) ))
00940 DEFINE((
00950 (F2READTEXT (LAMBDA (TXT NUM)
00960   (PROG (PTSPL)
00970     (SETQ PTSPL (LIST 'SUBJECT 'OBJECT 'VERB))
00980     (FONUMNAME)
00990     (FOSYNONYMS)
01000     (PUT (GET (CADDR TXT) 'SYNONYMV)
01010       'PTSPEECH 'YSYMVERB)
01020       LOOP
01030         (COND
01040           ((NULL TXT) (RETURN (PRINT (LIST
01050            'ENCODING 'SENTENCE NUM)) )))
01060         (PUT (CAR TXT) 'PTSPEECH (CAR PTSPL))
01070         (SETQ PTSPL (CDR PTSPL))
01080         (SETQ TXT (CDR TXT))
01090         (GO LOOP) ) )) ))
01100 DEFINE((
01110 (FOSYNONYMS (LAMBDA NIL
01120   (PROG (LIS)
01130     (SETQ LIS (FOSYNONYMDT))
01140       LOOP
01150         (COND
01160           ((NULL LIS) (RETURN NIL)))
01170         (PUT (CAAR LIS) 'SYNONYMV (CADAR LIS))
01180         (PUT (CADAR LIS) 'SYNONYMV (CAAR LIS))
01190         (SETQ LIS (CDR LIS))
01200         (GO LOOP) ) )) ))
01210 DEFINE((
01220 (FOSYNONYMDT (LAMBDA NIL
01230   '((NIKUMU URAMU) (NAGURU TATAKU)
01240     (ORIRU KUDARU) (KAKERU HASIRU)) )) ))
01250 DEFINE((
01260 (F2WKMLINK (LAMBDA (OND NUM)
01270   (PROG (PNAME NODTG WLINK RWLNK )
01280     (SETQ PNAME (EXPLODE (GET OND 'PTSPEECH)))
01290     (SETQ NODTG (PACK (LIST (CAR PNAME) (CADR
01300     PNAME) (CADDR PNAME) (CAR (CDDDR PNAME)) )))
01310     (SETQ WLINK (PACK (LIST 'WORKER- NODTG (GET
01320     NUM 'NAME))))
01330     (SETQ RWLNK (PACK (LIST 'R* WLINK)))
01340     (PUT WLINK 'STRTNODE OND)
01350     (F2LNKLIST OND WLINK)
01360     (F1LSLENGTH OND)
01370     (PUT WLINK 'GOALNODE (F2MKWORKM OND NUM))
01380     (RETURN (F2RVSLINK WLINK RWLNK)) ))) ))
01390 DEFINE((
01400 (F2MKWORKM (LAMBDA (OND NUM)
01410   (PROG (WORKM)
01420     (SETQ WORKM (PACK (LIST 'W (GET NUM 'NAME)
01430     DASH OND)))
01440     (PUT WORKM 'PTSPEECH (PACK (LIST 'Z (GET
01450     OND 'PTSPEECH))))
01460     (RETURN WORKM) ))) ))
01470 DEFINE((
01480 (F2ECNLINK (LAMBDA (WND NUM)
01490   (PROG (PLINK RPLNK)
01500     (SETQ PLINK (PACK (LIST (GET WND 'PTSPEECH)
01510     DASH (GET NUM 'NAME))))
01520     (SETQ RPLNK (PACK (LIST 'R* PLINK)))
01530     (PUT PLINK 'STRTNODE WND)
01540     (F2LNKLIST WND PLINK)
01550     (F1LSLENGTH WND)
01560     (CSETQ ECNOD (PACK (LIST 'SENT- (GET
01570     NUM 'NAME))))
01580     (PUT PLINK 'GOALNODE ECNOD)
01590     (RETURN (F2RVSLINK PLINK RPLNK) ))) ) ))
01600 DEFINE((
01610 (F1SYMLINK (LAMBDA (VND)
01620   (PROG (SLINK RSLNK)
01630     (CSETQ SVERB (GET VND 'SYNONYMV))
01640     (SETQ SLINK (PACK (LIST 'SYNONYM- VND)))
01650     (SETQ RSLNK (PACK (LIST 'SYNONYM- SVERB)))
01660     (PUT SLINK 'STRTNODE VND)
01670     (F2LNKLIST VND SLINK)
01680     (F1LSLENGTH VND)
01690     (PUT SLINK 'GOALNODE SVERB)
01700     (RETURN (F2RVSLINK SLINK RSLNK) ))) ) ))
01710 DEFINE((
01720 (F2WSYMLNK (LAMBDA (VND NUM)
01730   (PROG (SLINK RSLNK WVERB)
01740     (SETQ SLINK (PACK (LIST 'SYNONYM-W (GET NUM
01750     'NAME))))
01760     (SETQ RSLNK (PACK (LIST 'R*SYNONYM-W (GET
01770     NUM 'NAME))))
01780     (SETQ WVERB (PACK (LIST 'W (GET NUM 'NAME)
01790     DASH VND)))
01800     (PUT SLINK 'STRTNODE WVERB)
01810     (F2LNKLIST WVERB SLINK)
01820     (F1LSLENGTH WVERB)
01830     (PUT SLINK 'GOALNODE (PACK (LIST 'W (GET
01840     NUM 'NAME) DASH (GET VND 'SYNONYMV))))
01850     (RETURN (F2RVSLINK SLINK RSLNK) ))) ) ))
01860 DEFINE((
01870 (F2SEMLINK (LAMBDA (OND  DNB)
01880   (PROG (PNAME NODTG DLINK RDLNK)
```

```
01890      (SETQ PNAME (EXPLODE (GET OND 'PTSPEECH)))
01900      (SETQ NODTG OND)
01910      (SETQ DLINK (PACK (LIST 'DEF-D (GET DNB
01920      'NAME) NODTG)))
01930      (SETQ RDLNK (PACK (LIST 'R* DLINK)))
01940      (PUT DLINK 'STRTNODE OND)
01950      (F2LNKLIST OND DLINK)
01960      (F1LSLENGTH OND)
01970      (F2MKSEMAM DLINK DNB)
01980      (PUT DLINK 'GOALNODE (F2MKSEMAM OND DNB))
01990      (RETURN (F2RVSLINK DLINK RDLNK) ))) ) ))
02000 DEFINE((
02010 (F2MKSEMAM (LAMBDA (OND DNB)
02020   (PROG (SEMAM)
02030      (SETQ SEMAM (PACK (LIST 'D (GET DNB 'NAME)
02040      '& OND)))
02050      (PUT SEMAM 'PTSPEECH (PACK (LIST 'D (GET
02060      DNB 'NAME) (GET OND 'PTSPEECH))))
02070      (RETURN SEMAM) ))) ))
02080 DEFINE((
02090 (F3WSEMLNK (LAMBDA (OND NUM DNB)
02100   (PROG (PNAME NODTG WORKM DLINK RDLNK)
02110      (SETQ PNAME (EXPLODE (GET OND 'PTSPEECH)))
02120      (SETQ NODTG (PACK (LIST 'W (CAR PNAME))))
02130      (SETQ DLINK (PACK (LIST 'DEFINE- 'D (GET
02140      DNB 'NAME) NODTG (GET NUM 'NAME))))
02150      (SETQ RDLNK (PACK (LIST 'R* DLINK)))
02160      (SETQ WORKM (F2MKWORKM OND NUM))
02170      (PUT DLINK 'STRTNODE WORKM)
02180      (F2LNKLIST WORKM DLINK)
02190      (F1LSLENGTH WORKM)
02200      (F3MKWSEMA OND NUM DNB)
02210      (PUT DLINK 'GOALNODE (F3MKWSEMA OND NUM DNB))
02220      (RETURN (F2RVSLINK DLINK RDLNK) ))) ) ))
02230 DEFINE((
02240 (F3MKWSEMA (LAMBDA (OND NUM DNB)
02250   (PROG (SEMAM)
02260      (SETQ SEMAM (PACK (LIST 'W (GET NUM 'NAME)
02270      DASH 'D (GET DNB 'NAME) '& OND)))
02280      (PUT SEMAM 'PTSPEECH (PACK (LIST 'D (GET
02290      DNB 'NAME) (GET OND 'PTSPEECH))))
02300      (RETURN SEMAM))) ) ))
02310 DEFINE((
02320 (F2RVSLINK (LAMBDA (LNK RLK)
02330   (PROG (GNODE)
02340      (PUT RLK 'GOALNODE (GET LNK 'STRTNODE))
02350      (SETQ GNODE (GET LNK 'GOALNODE))
02360      (F2LNKLIST GNODE RLK)
02370      (F1LSLENGTH GNODE)
02380      (RETURN (PUT RLK 'STRTNODE GNODE) ))) ) ))
02390 DEFINE((
02400 (F2INTERLINK (LAMBDA (OND DNB)
02410   (PROG (GODNB ILINK SEMAS SEMAG)
02420      (COND
02430        ((EQUAL DNB 1) (SETQ GODNB 2))
02440        ((EQUAL DNB 2) (SETQ GODNB 1)) )
02450      (SETQ ILINK (PACK (LIST 'INT-D (GET DNB
02460      'NAME) OND)))
02470      (SETQ SEMAS (F2MKSEMAM OND DNB))
02480      (SETQ SEMAG (F2MKSEMAM OND GODNB))
02490      (PUT ILINK 'STRTNODE SEMAS)
02500      (PUT ILINK 'GOALNODE SEMAG)
02510      (F2LNKLIST SEMAS ILINK)
02520      (F1LSLENGTH SEMAS)
02530      (RETURN NIL) ))) ))
02540 DEFINE((
02550 (F3WINTERLINK (LAMBDA (OND NUM DNB)
02560   (PROG (PNAME NODTG GODNB ILINK WORKS WORKG)
02570      (COND
02580        ((EQUAL DNB 1) (SETQ GODNB 2))
02590        ((EQUAL DNB 2) (SETQ GODNB 1)) )
02600      (SETQ PNAME (EXPLODE (GET OND 'PTSPEECH)))
02610      (SETQ NODTG (PACK (LIST 'W (CAR PNAME))))
02620      (SETQ ILINK (PACK (LIST 'INTER-D (GET DNB
02630      'NAME) NODTG (GET NUM 'NAME))))
02640      (SETQ WORKS (F3MKWSEMA OND NUM DNB))
02650      (SETQ WORKG (F3MKWSEMA OND NUM GODNB))
02660      (PUT ILINK 'STRTNODE WORKS)
02670      (PUT ILINK 'GOALNODE WORKG)
02680      (F2LNKLIST WORKS ILINK)
02690      (F1LSLENGTH WORKS)
02700      (RETURN NIL) ))) ))
02710 DEFINE((
02720 (F1INTSYLINK (LAMBDA (OND)
02730   (PROG (ILINK SLINK SEMAM)
02740      (SETQ ILINK (PACK (LIST 'INT-D (GET 1
02750      'NAME) OND)))
02760      (SETQ SEMAM (F2MKSEMAM OND 1))
02770      (PUT ILINK 'STRTNODE SEMAM)
02780      (PUT ILINK 'GOALNODE SVERB)
02790      (F2LNKLIST SEMAM ILINK)
02800      (F1LSLENGTH SEMAM)
02810      (SETQ SLINK (PACK (LIST 'INT- SVERB)))
02820      (PUT SLINK 'STRTNODE SVERB)
02830      (PUT SLINK 'GOALNODE SEMAM)
02840      (F2LNKLIST SVERB SLINK)
02850      (F1LSLENGTH SVERB)
02860      (RETURN NIL) ))) ))
02870 DEFINE((
02880 (F2WINTSYLINK (LAMBDA (OND NUM)
02890   (PROG (ILINK SLINK WORKM WSVRB)
02900      (SETQ ILINK (PACK (LIST 'INTER-D1WV (GET
02910      NUM 'NAME))))
02920      (SETQ WORKM (F3MKWSEMA OND NUM 1))
02930      (SETQ WSVRB (F2MKWORKM SVERB NUM))
02940      (PUT ILINK 'STRTNODE WORKM)
02950      (PUT ILINK 'GOALNODE WSVRB)
02960      (F2LNKLIST WORKM ILINK)
02970      (F1LSLENGTH WORKM)
02980      (SETQ SLINK (PACK (LIST 'INTER-YSYM (GET
02990      NUM 'NAME))))
03000      (PUT SLINK 'STRTNODE WSVRB)
03010      (PUT SLINK 'GOALNODE WORKM)
03020      (F2LNKLIST WSVRB SLINK)
03030      (F1LSLENGTH WSVRB)
03040      (RETURN NIL) ))) ))
03050 DEFINE((
03060 (F2LNKLIST (LAMBDA (NOD LNK)
03070   (PROG (SMLIS)
03080      (SETQ SMLIS (CONS LNK (DELETE LNK (GET NOD
03090      'LINKLIST))))
03100      (RETURN (PUT NOD 'LINKLIST SMLIS) ))) ) ))
03110 DEFINE((
03120 (F1LSLENGTH (LAMBDA (NOD)
03130   (PROG (LKLIS)
03140      (SETQ LKLIS (GET NOD 'LINKLIST))
03150      (RETURN (PUT NOD 'LINKNUMS (FLOAT (LENGTH
03160      LKLIS)))) )) ) ))
03170 DEFINE((
03180 (GETPRINT4 (LAMBDA (NOD ATR PNM STN)
03190   (PROG NIL
03200      (COND
03210        ((EQUAL PNM 1)
03220         (PROG2
03230           (PRIN1 (LIST NOD ATR (GET NOD ATR)))
03240           (RETURN (SPACES STN)) )))
03250      (PRINT (LIST NOD ATR (GET NOD ATR)))
03260      (COND
03270        ((EQUAL STN P)
03280         (PROG NIL
03290           (PRINT
03300 '=============================================
03310             ))
03320           (RETURN (TERPRI)) ))) ))) ))
03330 DEFINE((
03340 (F1BNODLNK (LAMBDA (NLS)
03350   (PROG (FSTND LKNUM LKLST BLKSG FSTLK)
03360     LOOP
03370       (COND
03380         ((NULL NLS)
03390           (RETURN (PRINT (LIST '*** 'NETWORK
03400           'WAS 'RECONSTRUCTED '***)) )))
03410       (SETQ FSTND (CAR NLS))
03420       (PUT FSTND 'NODEACTV (GREAT 0.0 (GET FSTND
03430       'NODEACTV)))
03440       (SETQ LKNUM (GET FSTND 'LINKNUMS))
03450       (SETQ LKLST (GET FSTND 'LINKLIST))
03460       (SETQ BLKSG (QUOTIENT 1.0 LKNUM))
03470         LOOP2
03480           (COND
03490             ((NULL LKLST)
03500              (PROG2
03510                (SETQ NLS (CDR NLS)) (GO LOOP))))
03520           (SETQ FSTLK (CAR LKLST))
03530           (PUT FSTLK 'LINKACTV (GREAT 0.0 (GET
03540           FSTLK 'LINKACTV)))
03550           (PUT FSTLK 'LINKSTRG (GREAT BLKSG (GET
03560           FSTLK 'LINKSTRG)))
03570           (COND
03580             ((EQUAL DISBG NOWBG)
03590              (PROG2
03600                (PUT FSTLK 'ELPSTIME NIL)
03610                (CSETQ DISBG NIL)) ))
03620           (PUT FSTLK 'ELPSTIME (GREAT NIL (GET
03630           FSTLK 'ELPSTIME)))
03640           (SETQ LKLST (CDR LKLST))
03650           (GO LOOP2) ))) ))
03660 DEFINE((
03670 (F1BNODLNK# (LAMBDA (NLS)
03680   (PROG (FSTND LKNUM LKLST FSTLK)
03690     LOOP
03700       (COND
03710         ((NULL NLS)
03720           (RETURN (PRINT (LIST '*** 'ALL
03730           'ACTIVATION 'WAS 'CLEARED '***)) )))
03740       (SETQ FSTND (CAR NLS))
03750       (PUT FSTND 'NODEACTV 0.0)
03760       (SETQ LKNUM (GET FSTND 'LINKNUMS))
03770       (SETQ LKLST (GET FSTND 'LINKLIST))
03780         LOOP2
03790           (COND
03800             ((NULL LKLST)
03810              (PROG2
03820                (SETQ NLS (CDR NLS)) (GO LOOP))))
03830           (SETQ FSTLK (CAR LKLST))
03840           (PUT FSTLK 'LINKACTV 0.0)
03850           (SETQ LKLST (CDR LKLST))
```

原

```
03860           (GO LOOP2) ))) ))
03870 DEFINE((
03880 (F1BSEMLNK (LAMBDA (TXT)
03890   (PROG (SULIS OBLIS VELIS SVLIS)
03900     (SETQ SULIS (GET (CAR TXT) 'LINKLIST))
03910     (SETQ OBLIS (GET (CADR TXT) 'LINKLIST))
03920     (SETQ VELIS (GET (CADDR TXT) 'LINKLIST))
03930     (SETQ SVLIS (GET SVERB 'LINKLIST))
03940     (CSETQ DFLIS (LINEARIZE (LIST (F2DISCRLNK
03950     SULIS 'W) (F2DISCRLNK OBLIS 'W) (F2DISCRLNK
03960     VELIS 'W) (F2DISCRLNK SVLIS 'W) )))
03970     (F1SMSTRGTH DFLIS)
03980     (CSETQ RDFLS (F1GETRLNK (APPEND (F2DISCRLNK
03990     SULIS 'W) (F2DISCRLNK OBLIS 'W) )))
04000     (COND
04010       ((EQUAL 1 (LENGTH (GET CUEP2 'LINKLIST)))
04020        (F1RSMSTRG RDFLS) ))
04030     (COND
04040       ((EQUAL 1 (LENGTH (GET SVERB 'LINKLIST)))
04050        (PROG2
04060          (CSETQ RDEFV (F1GETRLNK (F2DISCRLNK
04070          VELIS 'W))) (F1RSMSTRG RDEFV) ))
04080       (T (CSETQ RDEFV (F1GETRLNK (APPEND
04090          (F2DISCRLNK VELIS 'W) (F2DISCRLNK
04100          SVLIS 'W) ))) ))
04110     (RETURN NIL) ))) ) ))
04120 DEFINE((
04130 (F2DISCRLNK (LAMBDA (LLS CHR)
04140   (PROG (ANSLS)
04150     (SETQ ANSLS NIL)
04160       LOOP
04170         (COND
04180           ((NULL LLS) (RETURN ANSLS))
04190           ((NOT (EQUAL CHR (CAR (EXPLODE (CAR
04200            LLS))) ))
04210            (SETQ ANSLS (CONS (CAR LLS) ANSLS)) ))
04220         (SETQ LLS (CDR LLS))
04230         (GO LOOP) ))) ))
04240 DEFINE((
04250 (F1GETRLNK (LAMBDA (LLS)
04260   (PROG (SNODE GNODE LKLIS RLIST)
04270     (SETQ RLIST NIL)
04280       LOOP
04290         (COND
04300           ((NULL LLS) (RETURN RLIST)) )
04310         (SETQ SNODE (GET (CAR LLS) 'STRTNODE))
04320         (SETQ GNODE (GET (CAR LLS) 'GOALNODE))
04330         (SETQ LKLIS (GET GNODE 'LINKLIST))
04340           LOOP2
04350             (COND
04360               ((NULL LKLIS)
04370                (PROG2
04380                  (SETQ LLS (CDR LLS)) (GO LOOP)))
04390               ((EQUAL (GET (CAR LKLIS) 'GOALNODE)
04400                SNODE)
04410                (SETQ RLIST (CONS (CAR LKLIS)
04420                RLIST)) ))
04430             (SETQ LKLIS (CDR LKLIS))
04440             (GO LOOP2) ))) ))
04450 DEFINE((
04460 (F1SMSTRGTH (LAMBDA (LKL)
04470   (PROG NIL
04480     LOOP
04490       (COND
04500         ((NULL LKL) (RETURN (TERPRI)) ))
04510       (PUT (CAR LKL) 'LINKSTRG 3.333333E-01)
04520       (SETQ LKL (CDR LKL))
04530       (GO LOOP) ))) ))
04540 DEFINE((
04550 (F1RSMSTRG (LAMBDA (RDL)
04560   (PROG NIL
04570     LOOP
04580       (COND
04590         ((NULL RDL) (RETURN (TERPRI)) ))
04600       (PUT (CAR RDL) 'LINKSTRG 3.333333E-01)
04610       (SETQ RDL (CDR RDL))
04620       (GO LOOP) ))) ))
04630 DEFINE((
04640 (F1RSMSTRG2 (LAMBDA (VLS)
04650   (PROG NIL
04660     LOOP
04670       (COND
04680         ((NULL VLS) (RETURN (TERPRI)) ))
04690       (PUT (CAR VLS) 'LINKSTRG 3.333333E-01)
04700       (SETQ VLS (CDR VLS))
04710       (GO LOOP) ))) ))
04720 DEFINE((
04730 (F2PROCESS1 (LAMBDA (TXT NUM)
04740   (PROG NIL
04750     (CSETQ WMLST NIL)
04760     (CSETQ SMLST NIL)
04770     (CSETQ TEXT2 TXT)
04780     (CSETQ VRBND (CADDR TXT))
04790     (TERPRI)
04800     (F2READTEXT TXT NUM)
04810     (PRIN1 (LIST 'ENCODING 'SENTENCE))
04820     (PRIN1 '=) (PRINT TXT)
04830     (TERPRI)
```

著

```
04840       LOOP
04850         (F2WKMLINK (CAR TXT) NUM)
04860         (F2ECNLINK (F2MKWORKM (CAR TXT) NUM) NUM)
04870         (CSETQ WMLST (CONS (F2MKWORKM (CAR TXT)
04880         NUM) WMLST))
04890         (F2SEMLINK (CAR TXT) 1)
04900         (CSETQ SMLST (CONS (F2MKSEMAM (CAR TXT)
04910         1) SMLST))
04920         (COND
04930           ((NULL (CDR TXT))
04940            (RETURN (F2REPORT25 TEXT2 NUM)) ))
04950         (F2SEMLINK (CAR TXT) 2)
04960         (CSETQ SMLST (CONS (F2MKSEMAM (CAR TXT)
04970         2) SMLST))
04980         (SETQ TXT (CDR TXT))
04990         (GO LOOP) ))) ))
05000 DEFINE((
05010 (F2REPORT25 (LAMBDA (TXT NUM)
05020   (PROG NIL
05030     (CSETQ SVERB (GET VRBND 'SYNONYMV))
05040     (CSETQ CUEP2 (F2MKSEMAM (CAR TEXT2) 1))
05050     (CSETQ CUEP3 SVERB)
05060     (CSETQ CUEP4 (F3MKWSEMA (CAR TEXT2) NUM 1))
05070     (CSETQ CUE45 (F2MKWORKM SVERB NUM))
05080     (CSETQ NLIST NIL)
05090     (CSETQ WMLS2 NIL)
05100     (CSETQ WMLST (REVERSE WMLST))
05110     (F1SYMLINK VRBND)
05120     (CSETQ NLIST (LINEARIZE (LIST TXT WMLST
05130     ECNOD SVERB SMLST)))
05140     (CSETQ NLIS1 NLIST)
05150     (TERPRI)
05160     (PRIN1 (LIST 'CONSTRUCTED 'NODES 'NO1))
05170     (PRIN1 '=) (PRINT NLIST)
05180     (TERPRI)
05190     (F1BNODLNK NLIST)
05200     (CSETQ AVLST (LINEARIZE (LIST WMLST ECNOD)))
05210     (TERPRI)
05220     (PRIN1 (LIST 'ACTIVE 'NODES 'LIST))
05230     (PRIN1 '=) (PRINT AVLST)
05240     (RETURN (TERPRI)) ))) ))
05250 DEFINE((
05260 (F2PROCESS2 (LAMBDA (TXT NUM)
05270   (PROG NIL
05280     LOOP
05290       (COND
05300         ((NULL (CDR TXT))
05310          (RETURN (FOREPORT35) )) )
05320       (F3WSEMLNK (CAR TXT) NUM 1)
05330       (F3WSEMLNK (CAR TXT) NUM 2)
05340       (F2WKMLINK (F2MKSEMAM (CAR TXT) 1) NUM)
05350       (CSETQ WMLS2 (CONS (F2MKWORKM (F2MKSEMAM
05360       (CAR TXT) 1) NUM) WMLS2))
05370       (F2WKMLINK (F2MKSEMAM (CAR TXT) 2) NUM)
05380       (CSETQ WMLS2 (CONS (F2MKWORKM (F2MKSEMAM
05390       (CAR TXT) 2) NUM) WMLS2))
05400       (F2INTERLINK (CAR TXT) 1)
05410       (F2INTERLINK (CAR TXT) 2)
05420       (F3WINTERLINK (CAR TXT) NUM 1)
05430       (F3WINTERLINK (CAR TXT) NUM 2)
05440       (SETQ TXT (CDR TXT))
05450       (GO LOOP) ))) ))
05460 DEFINE((
05470 (FOREPORT35 (LAMBDA NIL
05480   (PROG NIL
05490     (CSETQ NLIST (APPEND NLIS1 WMLS2))
05500     (TERPRI)
05510     (PRIN1 (LIST 'CONSTRUCTED 'NODES 'NO2))
05520     (PRIN1 '=) (PRINT NLIST)
05530     (TERPRI)
05540     (F1BNODLNK NLIST)
05550     (F1RSMSTRG2 RDFLS)
05560     (RETURN NIL) ))) ))
05570 DEFINE((
05580 (DELETE (LAMBDA (SEX LIS)
05590   (COND
05600     ((NULL LIS) NIL)
05610     ((EQUAL SEX (CAR LIS)) (CDR LIS) )
05620     (T (CONS (CAR LIS) (DELETE SEX (CDR LIS))))
05630        ) )) ))
05640 DEFINE((
05650 (LINEARIZE (LAMBDA (LIS)
05660   (COND
05670     ((NULL LIS) NIL)
05680     ((ATOM (CAR LIS))
05690      (CONS (CAR LIS) (LINEARIZE (CDR LIS))) )
05700     (T (APPEND (LINEARIZE (CAR LIS)) (LINEARIZE
05710        (CDR LIS)))) ))) ))
05720 DEFINE((
05730 (XDIFFERENCE (LAMBDA (NM1 NM2)
05740   (COND
05750     ((EQUAL NM1 XXX) NM1)
05760     ((EQUAL NM2 NIL) NM1)
05770     ((EQUAL NM1 NIL) (DIFFERENCE 0.0 NM2))
05780     (T (DIFFERENCE NM1 NM2)) )) )) ))
05790 DEFINE((
05800 (SMALL (LAMBDA (NM1 NM2)
05810   (COND
```

```
05820      ((OR
05830         (AND (EQUAL NM1 NIL) (EQUAL NM2 XXX))
05840         (AND (EQUAL NM1 XXX) (EQUAL NM2 NIL)) )
05850      XXX)
05860      ((NOT (NUMBERP NM1)) NM2)
05870      ((NOT (NUMBERP NM2)) NM1)
05880      ((EQUAL NM1 NM2) NM1)
05890      ((LESSP NM1 NM2) NM1)
05900      ((LESSP NM2 NM1) NM2)
05910      (T NIL) )))
05920 (MINIMUM (LAMBDA (NLS )
05930   (PROG (ANS)
05940      (SETQ ANS (CAR NLS))
05950      (SETQ NLS (CDR NLS))
05960        LOOP
05970          (COND
05980            ((NULL NLS) (RETURN ANS)) )
05990          (SETQ ANS (SMALL (CAR NLS) ANS))
06000          (SETQ NLS (CDR NLS))
06010          (GO LOOP) ))) ))
06020 DEFINE((
06030 (GREAT (LAMBDA (NM1 NM2)
06040   (COND
06050      ((EQUAL NM1 NIL) NM2)
06060      ((EQUAL NM2 NIL) NM1)
06070      ((EQUAL NM1 NM2) NM1)
06080      ((LESSP NM1 NM2) NM2)
06090      ((LESSP NM2 NM1) NM1)
06100      (T NIL) )))
06110 (MAXIMAM (LAMBDA (NLS)
06120   (PROG (ANS)
06130      (SETQ ANS (CAR NLS))
06140      (SETQ NLS (CDR NLS))
06150        LOOP
06160          (COND
06170            ((NULL NLS) (RETURN ANS)) )
06180          (SETQ ANS (GREAT (CAR NLS) ANS))
06190          (SETQ NLS (CDR NLS))
06200          (GO LOOP) ))) ))
06210 DEFINE((
06220 (SORTING (LAMBDA (LIS)
06230   (COND
06240      ((NULL LIS) NIL)
06250      (T (CONS (MINIMUM LIS) (SORTING (REPLACE
06260         (MINIMUM LIS) LIS)))) )))
06270 (MSORTING (LAMBDA (LIS)
06280   (COND
06290      ((NULL LIS) NIL)
06300      (T (CONS (MAXIMAM LIS) (MSORTING (REPLACE
06310         (MAXIMAM LIS) LIS)))) ))
06320 DEFINE((
06330 (REPLACE (LAMBDA (ATM LIS)
06340   (COND
06350      ((NULL LIS) NIL)
06360      ((EQUAL ATM (CAR LIS)) (CDR LIS) )
06370      (T (CONS (CAR LIS) (REPLACE ATM (CDR LIS))))
06380      ))) ))
06390 DEFINE((
06400 (F2PROCESS3 (LAMBDA (TXT NUM)
06410   (PROG NIL
06420      (F3WSEMLNK VRBND NUM 1)
06430      (F2WKMLINK (F2MKSEMAM VRBND 1) NUM)
06440      (CSETQ WMLS2 (CONS (F2MKWORKM (F2MKSEMAM
06450      VRBND 1) NUM) WMLS2))
06460      (F2WKMLINK SVERB NUM)
06470      (F2WSYMLNK VRBND NUM)
06480      (CSETQ WMLS2 (CONS (F2MKWORKM SVERB NUM)
06490      WMLS2))
06500      (CSETQ NLIST (APPEND NLIS1 WMLS2))
06510      (F1INTSYLINK VRBND)
06520      (F2WINTSYLINK VRBND NUM)
06530      (TERPRI)
06540      (PRIN1 (LIST 'CONSTRUCTED 'NODES 'NO3))
06550      (PRIN1 '=) (PRINT NLIST)
06560      (TERPRI)
06570      (F1BNODLNK NLIST)
06580      (F1RSMSTRG2 RDEFV)
06590      (RETURN NIL) ))) ))
06600 DEFINE((
06610 (F2PROCESS4 (LAMBDA (TXT NUM)
06620   (PROG NIL
06630     LOOP
06640       (COND
06650         ((NULL (CDR TXT))
06660          (PROG NIL
06670            (F2ECNLINK (F2MKWORKM (F2MKSEMAM (CAR
06680            TXT) 1) NUM) NUM)
06690            (F2ECNLINK (F2MKWORKM SVERB NUM) NUM)
06700            (F1BNODLNK NLIST)
06710            (SETQ TXT (CDR TXT))
06720            (TERPRI)
06730            (PRINT (LIST '*** 'ALL 'NETWORK 'WAS
06740            'CONSTRUCTED '***))
06750            (RETURN NIL) )))
06760         (COND
06770           ((NULL TXT) (RETURN NIL) ))
06780         (F2ECNLINK (F2MKWORKM (F2MKSEMAM (CAR TXT)
06790         1) NUM) NUM)
06800         (F2ECNLINK (F2MKWORKM (F2MKSEMAM (CAR TXT)
06810         2) NUM) NUM)
06820         (SETQ TXT (CDR TXT))
06830         (GO LOOP) ))) ))
06840 DEFINE((
06850 (F1BGNACTV (LAMBDA (ALS)
06860   (PROG NIL
06870     (CSETQ DACTV (QUOTIENT VACTI (TIMES 1.0
06880     (FLOAT (LENGTH ALS)))))
06890       LOOP
06900         (COND
06910           ((NULL ALS)
06920             (PROG2
06930               (PRINT (LIST '*** 'DIVIDE '15
06940               'UNITS 'OF 'ACTIVATION 'AMONG
06950               'ACTIVE 'NODES '***))
06960               (TERPRI) )))
06970         (COND
06980           ((NULL ALS) (RETURN NIL) ))
06990         (PUT (CAR ALS) 'NODEACTV (PLUS DACTV
07000         (GET (CAR ALS) 'NODEACTV)))
07010         (SETQ ALS (CDR ALS))
07020         (GO LOOP) ))) ))
07030 DEFINE((
07040 (F1PUTACTV (LAMBDA (ALS)
07050   (PROG (ANODE)
07060     LOOP
07070       (COND
07080         ((NULL ALS)
07090           (RETURN NIL) ))
07100       (SETQ ANODE (CAR ALS))
07110       (CSETQ LKACT (QUOTIENT (GET ANODE 'NODEACTV)
07120       (GET ANODE 'LINKNUMS) ))
07130       (F1RMDTIME (GET ANODE 'LINKLIST))
07140       (SETQ ALS (CDR ALS))
07150       (GO LOOP) ))) ))
07160 DEFINE((
07170 (F1RMDTIME (LAMBDA (ALK)
07180   (PROG (LKSTG ETIME)
07190     (COND
07200       ((NULL ALK)
07210         (RETURN NIL) ))
07220     (PUT (CAR ALK) 'LINKACTV LKACT)
07230     (SETQ LKSTG (GET (CAR ALK) 'LINKSTRG))
07240     (COND
07250       ((AND (NUMBERP (GET (CAR ALK) 'ELPSTIME))
07260         (MINUSP (GET (CAR ALK) 'ELPSTIME)) )
07270         (PROG NIL
07280           (PUT (CAR ALK) 'ELPSTIME XXX)
07290           (F1RMDTIME2 (CAR ALK))
07300           (RETURN NIL) ))
07310       ((EQUAL LKACT 0.0)
07320         (PROG NIL
07330           (PUT (CAR ALK) 'ELPSTIME (SMALL (GET
07340           (CAR ALK) 'ELPSTIME) XXX))
07350           (F1RMDTIME2 (CAR ALK))
07360           (RETURN (F1RMDTIME (CDR ALK))) ))
07370       ((NULL ALK) (RETURN NIL))
07380       (T (PROG NIL
07390             (SETQ ETIME (QUOTIENT 1.0 (TIMES
07400             LKSTG LKACT)))
07410             (PUT (CAR ALK) 'ELPSTIME (SMALL ETIME
07420             (GET (CAR ALK) 'ELPSTIME)))
07430             (F1RMDTIME2 (CAR ALK))
07440             (RETURN (F1RMDTIME (CDR ALK))) ))) )))
07450
07460 (F1RMDTIME2 (LAMBDA (ALI)
07470   (CSETQ TLIST (DELETE XXX (DELETE NIL (CONS (GET
07480   ALI 'ELPSTIME) TLIST))) )))) ))
07490 DEFINE((
07500 (F2MNMTIME (LAMBDA (ALS TLS)
07510   (PROG NIL
07520     (CSETQ RTLIS NIL)
07530     (CSETQ MTIME (MINIMUM (LINEARIZE TLS)))
07540       LOOP
07550         (COND
07560           ((NULL ALS)
07570             (RETURN NIL) ))
07580         (F1MNMTIME2 (GET (CAR ALS) 'LINKLIST))
07590         (SETQ ALS (CDR ALS))
07600         (GO LOOP) )))
07610
07620 (F1MNMTIME2 (LAMBDA (ALK)
07630   (PROG (ALINK)
07640     LOOP
07650       (COND
07660         ((NULL ALK)
07670           (RETURN NIL)) )
07680       (SETQ ALINK (CAR ALK))
07690       (F1MNMTIME3 (GET ALINK 'ELPSTIME))
07700       (PUT ALINK 'ELPSTIME RTIM1)
07710       (CSETQ RTLIS (DELETE XXX (DELETE NIL (CONS
07720       RTIM1 RTLIS))))
07730       (SETQ ALK (CDR ALK))
07740       (GO LOOP) )))
07750
07760 (F1MNMTIME3 (LAMBDA (EPT)
07770   (PROG (DIFRT RTIM1)
07780     (SETQ DIFRT (XDIFFERENCE EPT MTIME))
```

```
07790      (COND
07800        ((EQUAL DIFRT 0.0)
07810         (RETURN (CSETQ RTIM1 NIL)) )
07820        (T (RETURN (CSETQ RTIM1 DIFRT))) ) ))) ))
07830 DEFINE((
07840 (F1CONTACT (LAMBDA (ALS)
07850   (PROG NIL
07860     (CSETQ ALIS2 ALS)
07870     (CSETQ CLIST NIL)
07880     (CSETQ CLIS2 NIL)
07890       LOOP
07900         (COND
07910           ((NULL ALS)
07920            (PROG NIL
07930              (TERPRI)
07940              (PRIN1 '(CONTACTED LINKS))
07950              (PRIN1 '=) (PRINT CLIST)
07960              (TERPRI)
07970              (F1CONTACT2 CLIST)
07980              (TERPRI)
07990              (PRIN1 '(CONTACTED NODES))
08000              (PRIN1 '=)
08010              (F1SORTING CLIS2)
08020              (PRINT SLIST)
08030              (CSETQ CLIS2 SLIST)
08040              (CSETQ ALIS2 (F2UNIONDEL CLIS2
08050              ALIS2))
08060              (RETURN NIL) )))
08070      (COND
08080        ((NULL ALS) (RETURN NIL)) )
08090      (F1MNMLIST (GET (CAR ALS) 'LINKLIST))
08100      (SETQ ALS (CDR ALS))
08110      (GO LOOP) )))
08120
08130 (F1CONTACT2 (LAMBDA (CLS)
08140   (PROG NIL
08150     LOOP
08160       (COND
08170         ((NULL CLS)
08180          (RETURN (TERPRI))) )
08190       (PRIN1 (LIST (GET (CAR CLS) 'STRTNODE)
08200       '======== (GET (CAR CLS) 'GOALNODE)))
08210       (SETQ CLS (CDR CLS))
08220       (GO LOOP) ))) ))
08230 DEFINE((
08240 (F1MNMLIST (LAMBDA (KLS)
08250   (PROG (ALNK1 GNODE)
08260     LOOP
08270       (COND
08280         ((NULL KLS) (RETURN NIL))
08290         ((EQUAL NIL (GET (CAR KLS) 'ELPSTIME))
08300          (PROG NIL
08310            (SETQ ALNK1 (CAR KLS))
08320            (PUT ALNK1 'ELPSTIME XXX)
08330            (PUT (GET ALNK1 'STRTNODE) 'NODEACTV
08340            (DIFFERENCE (GET (GET ALNK1 'STRTNODE)
08350            'NODEACTV)
08360            (GET ALNK1 'LINKACTV) ))
08370            (SETQ GNODE (GET ALNK1 'GOALNODE))
08380            (CSETQ CLIST (CONS ALNK1 CLIST))
08390            (CSETQ CLIS2 (CONS GNODE (DELETE GNODE
08400            CLIS2)))
08410            (PUT GNODE 'NODEACTV (PLUS (GET GNODE
08420            'NODEACTV) (GET ALNK1 'LINKACTV))) )))
08430       (SETQ KLS (CDR KLS))
08440       (GO LOOP) ))) ))
08450 DEFINE((
08460 (F1SORTING (LAMBDA (LIS)
08470   (PROG (LIST2 ANDLS ANDL2)
08480     (SETQ LIST2 LIS)
08490     (SETQ ANDLS NIL)
08500     (CSETQ SLIST NIL)
08510       LOOP
08520         (COND
08530           ((NULL LIS)
08540            (PROG NIL
08550              (SETQ ANDL2 (MSORTING ANDLS))
08560                LOOP2
08570                  (COND
08580                    ((NULL ANDL2)
08590                     (RETURN NIL) ))
08600                  (F2SORTING2 LIST2 ANDL2)
08610                  (SETQ ANDL2 (CDR ANDL2))
08620                  (GO LOOP2) )))
08630         (COND
08640           ((NULL LIS) (RETURN SLIST) ))
08650         (SETQ ANDLS (CONS (GET (CAR LIS)
08660         'NODEACTV) ANDLS))
08670         (SETQ LIS (CDR LIS))
08680         (GO LOOP) )))
08690
08700 (F2SORTING2 (LAMBDA (LS2 AN2)
08710   (PROG NIL
08720     LOOP
08730       (COND
08740         ((NULL LS2) (RETURN NIL))
08750         ((EQUAL (CAR AN2) (GET (CAR LS2)
08760          'NODEACTV))
08770          (CSETQ SLIST (APPEND (DELETE (CAR LS2)
08780            SLIST) (LIST (CAR LS2)))) ))
08790       (SETQ LS2 (CDR LS2))
08800       (GO LOOP) ))) )))
08810 DEFINE((
08820 (F1STRENGTH (LAMBDA (CLS)
08830   (PROG (LACTV LSTRG)
08840     (COND
08850       ((NULL CLS) (RETURN NIL) ))
08860       (SETQ LACTV (GET (CAR CLS) 'LINKACTV))
08870       (SETQ LSTRG (GET (CAR CLS) 'LINKSTRG))
08880     (COND
08890       ((EQUAL LACTV 0.0)
08900        (F1STRENGTH (CDR CLS)) ))
08910     (SETQ LSTRG (PLUS LSTRG (TIMES (DIFFERENCE
08920     1.0 LSTRG) (TIMES LACTV VLKTH))))
08930     (PUT (CAR CLS) 'LINKSTRG LSTRG)
08940     (F1STRENGTH (CDR CLS)) ))) ))
08950 DEFINE((
08960 (F1RASNSTRG (LAMBDA (NLS)
08970   (PROG NIL
08980     LOOP
08990       (COND
09000         ((NULL NLS) (RETURN (TERPRI)) ))
09010       (F1REASSIGN (GET (CAR NLS) 'LINKLIST))
09020       (SETQ NLS (CDR NLS))
09030       (GO LOOP) ))) ))
09040 DEFINE((
09050 (F1REASSIGN (LAMBDA (CLK)
09060   (PROG (CLKL2 SGLST CSTLK)
09070    (SETQ CLKL2 CLK)
09080    (SETQ SGLST NIL)
09090      LOOP
09100        (COND
09110          ((NULL CLK)
09120           (PROG (STLK2 RESUM SUMAT)
09130             LOOP2
09140               (COND
09150                 ((NULL CLKL2) (RETURN NIL)) )
09160               (SETQ STLK2 (CAR CLKL2))
09170               (SETQ SUMAT (SUMATION SGLST))
09180                 (COND
09190                   ((EQUAL SUMAT 0.0)
09200                    (SETQ RESUM 0.0))
09210                   (T (SETQ RESUM (QUOTIENT 1.0
09220                      SUMAT))) )
09230                 (PUT STLK2 'LINKSTRG (TIMES (GET
09240                 STLK2 'LINKSTRG) RESUM))
09250                 (SETQ CLKL2 (CDR CLKL2))
09260                 (GO LOOP2) )))
09270        (COND
09280          ((NULL CLKL2) (RETURN NIL) ))
09290        (SETQ CSTLK (CAR CLK))
09300        (SETQ SGLST (CONS (GET CSTLK 'LINKSTRG)
09310        SGLST))
09320        (SETQ CLK (CDR CLK))
09330        (GO LOOP) ))) ))
09340 DEFINE((
09350 (SUMATION (LAMBDA (NLS)
09360   (COND
09370     ((NULL NLS) 0.0)
09380     (T (PLUS (CAR NLS) (SUMATION (CDR NLS)) ))
09390        ))) ))
09400 DEFINE((
09410 (F1MAXALIST (LAMBDA (ALS)
09420   (PROG (LNGTH RLIST AVLST)
09430     (SETQ LNGTH (LENGTH ALS))
09440     (SETQ AVLST ALS)
09450     (COND
09460       ((LESSP LNGTH VMXAL)
09470        (RETURN AVLST) )
09480       ((EQUAL LNGTH VMXAL)
09490        (RETURN AVLST) ))
09500       (SETQ RLIST (REVERSE ALS))
09510         LOOP
09520           (COND
09530             ((EQUAL LNGTH VMXAL)
09540              (PROG NIL
09550                (CSETQ AVLST (REVERSE AVLST))
09560                (RETURN AVLST) )))
09570           (COND
09580             ((EQUAL LNGTH VMXAL) (RETURN NIL)))
09590           (CSETQ AVLST (CDR RLIST))
09600           (PUT (CAR RLIST) 'NODEACTV 0.0)
09610           (SETQ LNGTH (DIFFERENCE LNGTH 1))
09620           (SETQ RLIST (CDR RLIST))
09630           (GO LOOP) ))) ))
09640 DEFINE((
09650 (F1TDISPLAY (LAMBDA (NUM)
09660   (PROG NIL
09670     (TERPRI)
09680     (PRIN1 '################)
09690     (SPACES 2)
09700     (PRIN1 (LIST 'ELAPSED 'TIME))
09710     (SPACES 1)
09720     (PRIN1 '=)
09730     (SPACES 1)
09740     (PRIN1 NUM)
09750     (SPACES 2)
09760     (PRIN1 '################)
```

```
09770      (TERPRI)
09780      (RETURN (TERPRI)) ))) ))
09790 DEFINE((
09800 (ENCODING (LAMBDA (TX2 NU2)
09810    (PROG NIL
09820       (TERPRI) (TERPRI)
09830       (ENCODINGCS)
09840       (CSETQ SGNUM 0)
09850       (CSETQ ZTIME INTVL)
09860       (CSETQ NUMBR 0)
09870       (CSETQ MTIME 10)
09880       (F2PROCESS1 TX2 NU2)
09890       (RETURN (F2TIMECONDE TX2 NU2)) ))) ))
09900 DEFINE((
09910 (FONODESMK15 (LAMBDA NIL
09920    (COND
09930       ((AND
09940          (MEMBER (CAR TEXT2) AVLST)
09950          (EQUAL DIS15 NOW15))
09960        (PROG2
09970          (F1BSEMLNK TEXT2)
09980          (CSETQ DIS15 NIL) ))
09990       (T NIL) ))) ))
10000 DEFINE((
10010 (F2NODESMK2 (LAMBDA (TXT NUM)
10020    (COND
10030       ((AND
10040          (MEMBER CUEP2 AVLST)
10050          (EQUAL DISP2 NOWP2))
10060        (PROG2
10070          (CSETQ DISP2 NIL)
10080          (F2PROCESS2 TXT NUM) ))
10090       (T NIL) ))) ))
10100 DEFINE((
10110 (F2NODESMK3 (LAMBDA (TXT NUM)
10120    (COND
10130       ((AND
10140          (MEMBER CUEP3 AVLST)
10150          (EQUAL DISP3 NOWP3))
10160        (PROG (WSLNK)
10170          (F2PROCESS3 TEXT2 NUM)
10180          (RETURN (CSETQ DISP3 NIL)) ))
10190       (T NIL) ))) ))
10200 DEFINE((
10210 (F2NODESMK4 (LAMBDA (TXT NUM)
10220    (COND
10230       ((AND (OR
10240                (MEMBER CUEP4 AVLST)
10250                (MEMBER CUE45 AVLST))
10260              (EQUAL DISP4 NOWP4))
10270        (PROG NIL
10280          (F2PROCESS4 TEXT2 NUM)
10290          (F1BNODLNK NLIST)
10300          (RETURN (CSETQ DISP4 NIL)) ))
10310       (T NIL) ))) ))
10320 DEFINE((
10330 (F2TIMECONDE (LAMBDA (TX2 NU2)
10340    (COND
10350       ((GREATERP VTIME QTIME) (FOENDING))
10360       ((LESSP ANSWR (QUOTIENT VACTI 10))
10370        (FOENDING))
10380       ((EQUAL NUMBR VCYCL) (FOENDING))
10390       ((GREATERP  (DIFFERENCE (PLUS ZTIM2 INTVL)
10400        ZTIME) MTIME)
10410        (PROG NIL
10420          (F2ACTIVATION2 TX2 NU2)
10430          (FONODESMK15)
10440          (F2NODESMK2 TX2 NU2)
10450          (F2NODESMK3 TX2 NU2)
10460          (F2NODESMK4 TX2 NU2)
10470          (F1CLKMTIME AVLST)
10480          (RETURN (F2TIMECONDE TX2 NU2)) ))
10490       (T (PROG NIL
10500             (F2NODESMK2 TX2 NU2)
10510             (F2NODESMK3 TX2 NU2)
10520             (F2NODESMK4 TX2 NU2)
10530             (CSETQ DTIME (DIFFERENCE (PLUS ZTIM2
10540             INTVL) ZTIME))
10550             (RETURN (ENCODING155 TX2 NU2)))) ))) ))
10560 DEFINE((
10570 (ENCODING155 (LAMBDA (TX2 NU2)
10580    (PROG NIL
10590       (F1TIMECORRECT NLIST)
10600       (CSETQ ZTIME (PLUS ZTIM2 INTVL))
10610       (CSETQ ZTIM2 (PLUS ZTIM2 INTVL))
10620       (CSETQ SGNUM (PLUS SGNUM 1))
10630       (F1BNODLNK# NLIST)
10640       (F1TDISPLAY SGNUM)
10650       (F1BGNACTV AVLST)
10660       (F1CLKMTIME AVLST)
10670       (RETURN (F2TIMECONDE TX2 NU2)) ))) ))
10680 DEFINE((
10690 (F1CLKMTIME (LAMBDA (ALS)
10700    (PROG NIL
10710       (CSETQ TLIST NIL)
10720       (F1PUTACTV ALS)
10730       (CSETQ MTIME (MINIMUM (LINEARIZE TLIST)))
10740       (RETURN NIL) ))) ))
10750 DEFINE((
10760 (F2ACTIVATION2 (LAMBDA (TX2 NU2)
10770    (PROG NIL
10780       (TERPRI) (TERPRI)
10790       (CSETQ NUMBR (PLUS NUMBR 1))
10800       (PRINT (LIST '********** 'CYCLE NUMBR
10810       '**********))
10820       (TERPRI)
10830       (PRIN1 (LIST 'MINIMUM 'TIME)) (PRIN1 '=)
10840       (PRINT MTIME)
10850       (TERPRI)
10860       (CSETQ ZTIME (PLUS ZTIME MTIME))
10870       (PRIN1 (LIST 'ELAPSED 'TIME)) (PRIN1 '=)
10880       (PRINT ZTIME)
10890       (PRIN1 '==============*)
10900       (CSETQ VTIME (DIFFERENCE ZTIME INTVL))
10910       (CSETQ VTIME (QUOTIENT VTIME 10.0))
10920       (PRINT VTIME)
10930       (F2MNMTIME AVLST TLIST)
10940       (F1CONTACT AVLST)
10950       (RETURN (FOACTIVATION3)) ))) ))
10960 DEFINE((
10970 (FOACTIVATION3 (LAMBDA NIL
10980    (PROG NIL
10990       (F1STRENGTH CLIST)
11000       (TERPRI)
11010       (CSETQ AVLST ALIS2)
11020       (F1MAXALIST AVLST)
11030       (PRIN1 (LIST 'ACTIVE 'NODES 'LIST))
11040       (PRIN1 '=)
11050       (PRINT AVLST)
11060       (F1RASNSTRG AVLST)
11070       (F1GETACTV AVLST)
11080       (RETURN NIL) ))) ))
11090 DEFINE((
11100 (F1GETSTRG (LAMBDA (NLS)
11110    (PROG (LKLST FSTLK)
11120       (COND
11130          ((NULL NLS)
11140           (RETURN (TERPRI)) ))
11150       (SETQ LKLST (GET (CAR NLS) 'LINKLIST))
11160          LOOP2
11170          (COND
11180             ((NULL LKLST)
11190              (RETURN (F1GETSTRG (CDR NLS)) )))
11200          (GETPRINT4 (CAR LKLST) 'LINKSTRG 1 0)
11210          (TERPRI)
11220          (SETQ LKLST (CDR LKLST))
11230          (GO LOOP2) ))) ))
11240 DEFINE((
11250 (F1TIMECORRECT (LAMBDA (NLS)
11260    (PROG (LKLST FSTLK)
11270       (COND
11280          ((NULL NLS)
11290           (RETURN (PRINT (LIST 'END 'OF 'CORRECTING
11300           'ELPSTIME)))))
11310       (SETQ LKLST (GET (CAR NLS) 'LINKLIST))
11320          LOOP2
11330          (COND
11340             ((NULL LKLST)
11350              (RETURN (F1TIMECORRECT (CDR NLS))) )
11360             ((NOT (NUMBERP (GET (CAR LKLST)
11370              'ELPSTIME) ))
11380              (PROG2
11390                 (SETQ LKLST (CDR LKLST))
11400                 (GO LOOP2) )) )
11410          (PUT (CAR LKLST) 'ELPSTIME (DIFFERENCE
11420          (GET (CAR LKLST) 'ELPSTIME) DTIME))
11430          (SETQ LKLST (CDR LKLST))
11440          (GO LOOP2) ))) ))
11450 DEFINE((
11460 (FOENDING (LAMBDA NIL
11470    (PROG NIL
11480       (TERPRI)
11490       (PRINT (LIST '###### 'ENCODING 'WAS
11500       'COMPLETED '#####))
11510       (F1BNODLNK# NLIST)
11520       (F1DELETIME NLIST)
11530       (RETURN NIL) ))) ))
11540 DEFINE((
11550 (F2TIMECOND3 (LAMBDA (TXT NUM)
11560    (COND
11570       ((NULL CLIST)
11580        (PROG NIL
11590          (CSETQ NUMBR (DIFFERENCE NUMBR 1))
11600          (CSETQ ZTIME (DIFFERENCE ZTIME MTIME))
11610          (F2NODESMK2 TXT NUM)
11620          (F2NODESMK3 TXT NUM)
11630          (F2NODESMK4 TXT NUM)
11640          (CSETQ ZTIME (PLUS ZTIM2 INTVL))
11650          (CSETQ ZTIM2 (PLUS ZTIM2 INTVL))
11660          (RETURN (ENCODING155 TXT NUM)) ))
11670       (T NIL) ))) ))
11680 DEFINE((
11690 (RETRIEVAL (LAMBDA (TX2)
11700    (PROG NIL
11710       (TERPRI)
11720       (RETRIEVALCS)
```

```
11730     (CSETQ AVLST TX2)
11740     (CSETQ RVLST TX2)
11750     (TERPRI) (TERPRI)
11760     (PRIN1 (LIST 'RETRIEVAL 'SENTENCE)) (PRIN1 '=)
11770     (PRINT TX2)
11780     (TERPRI)
11790     (PRIN1 (LIST 'ACTIVE 'NODES 'LIST)) (PRIN1 '=)
11800     (PRINT AVLST)
11810     (TERPRI)
11820     (CSETQ SGNUM 0)
11830     (CSETQ ZTIME INTVL)
11840     (CSETQ NUMBR 0)
11850     (CSETQ MTIME 10)
11860     (RETURN (F1TIMECONDR TX2)) ))) ))
11870 DEFINE((
11880 (F1TIMECONDR (LAMBDA (TX2)
11890   (COND
11900     ((EQUAL NUMBR VCYCL) (FOENDINGR))
11910     ((AND
11920        (MEMBER VLINK PELIS) (MEMBER SVLNK NELIS)
11930        (EQUAL 3 (LENGTH NELIS)) )
11940      (FOENDINGR))
11950     ((AND
11960        (MEMBER SVLNK PELIS) (MEMBER VLINK NELIS)
11970        (EQUAL 3 (LENGTH PELIS)) )
11980        (FOENDINGR))
11990     ((AND
12000        (MEMBER VLINK PELIS) (MEMBER SVLNK NELIS)
12010        (EQUAL 3 (LENGTH PELIS)) )
12020      (FOENDINGR))
12030     ((AND
12040        (MEMBER SVLNK PELIS) (MEMBER VLINK NELIS)
12050        (EQUAL 3 (LENGTH NELIS)) )
12060      (FOENDINGR))
12070     ((AND
12080        (NOT (OR (AND (MEMBER VLINK PELIS)
12090                      (MEMBER SVLNK NELIS))
12100                 (AND (MEMBER SVLNK PELIS)
12110                      (MEMBER VLINK NELIS))))
12120        (EQUAL 3 (LENGTH (APPEND PELIS NELIS))) )
12130      (FOENDINGR))
12140     ((LESSP NESTG (F1SUMSTRG NELIS)) (FOENDINGR) )
12150     ((GREATERP (DIFFERENCE (PLUS ZTIM2 INTVL)
12160      ZTIME) MTIME)
12170        (PROG NIL
12180          (F2ACTIVATION2R AVLST TX2)
12190          (F1CLKMTIME AVLST)
12200          (RETURN (F1TIMECONDR TX2)) ))
12210     (T (PROG NIL
12220          (CSETQ DTIME (DIFFERENCE (PLUS ZTIM2
12230          INTVL) ZTIME))
12240          (RETURN (RETRVAL155  TX2)))) ))) ))
12250 DEFINE((
12260 (RETRVAL155 (LAMBDA (TX2)
12270  (PROG NIL
12280    (CSETQ ZTIME (PLUS ZTIM2 INTVL))
12290    (CSETQ ZTIM2 (PLUS ZTIM2 INTVL))
12300    (CSETQ SGNUM (PLUS SGNUM 1))
12310    (F1TDISPLAY SGNUM)
12320    (F1BGNACTV AVLST)
12330    (F1CLKMTIME AVLST)
12340    (RETURN (F1TIMECONDR TX2)) ))) ))
12350 DEFINE((
12360 (F2ACTIVATION2R (LAMBDA (AL2 TX2)
12370   (PROG NIL
12380     (TERPRI) (TERPRI)
12390     (CSETQ NUMBR (PLUS NUMBR 1))
12400     (PRINT (LIST '********** 'CYCLE NUMBR
12410     '**********))
12420     (TERPRI)
12430     (PRIN1 (LIST 'MINIMUM 'TIME)) (PRIN1 '=)
12440     (PRINT MTIME)
12450     (TERPRI)
12460     (CSETQ ZTIME (PLUS ZTIME MTIME))
12470     (PRIN1 (LIST 'ELAPSED 'TIME)) (PRIN1 '=)
12480     (PRINT ZTIME)
12490     (PRIN1 '===============*)
12500     (CSETQ VTIME (DIFFERENCE ZTIME INTVL))
12510     (CSETQ VTIME (QUOTIENT VTIME 10.0))
12520     (PRINT VTIME)
12530     (F2MNMTIME AL2 TLIST)
12540     (F1CONTACT AL2)
12550     (TERPRI)
12560     (F1EVIDENCE CLIST)
12570     (CSETQ NESUM (F1SUMSTRG NELIS))
12580     (CSETQ PESUM (DIFFERENCE (F1SUMSTRG PELIS)
12590     NESUM))
12600     (PRIN1 (LIST 'POSITIVE 'EVIDECE 'LINKS))
12610     (PRIN1 '=) (PRINT PELIS)
12620     (SPACES 5)
12630     (PRIN1 (LIST 'LINKSTRG 'SUMATION))
12640     (PRIN1 '=) (PRINT PESUM)
12650     (PRIN1 (LIST 'NEGATIVE 'EVIDENCE 'LINKS))
12660     (PRIN1 '=) (PRINT NELIS)
12670     (SPACES 5)
12680     (PRIN1 (LIST 'LINKSTRG 'SUMATION))
12690     (PRIN1 '=) (PRINT NESUM)
12700     (TERPRI)
12710     (F1TIMECOND3R TX2)
12720     (RETURN (FOACTIVATION3)) ))) ))
12730 DEFINE((
12740 (F1TIMECOND3R (LAMBDA (TXT)
12750   (COND
12760     ((NULL CLIST)
12770      (PROG NIL
12780        (CSETQ NUMBR (DIFFERENCE NUMBR 1))
12790        (CSETQ ZTIME (DIFFERENCE ZTIME MTIME))
12800        (CSETQ ZTIME (PLUS ZTIM2 INTVL))
12810        (CSETQ ZTIM2 (PLUS ZTIM2 INTVL))
12820        (RETURN (RETRVAL155 TXT)) ))
12830     (T NIL) ))) ))
12840 DEFINE((
12850 (FOENDINGR (LAMBDA NIL
12860   (PROG NIL
12870     (CSETQ NESUM (F1SUMSTRG NELIS))
12880     (CSETQ PESUM (F1SUMSTRG PELIS))
12890     (CSETQ NESUM (TIMES NESUM 10))
12900     (CSETQ PESUM (TIMES PESUM 10))
12910     (TERPRI)
12920     (PRINT (LIST '##### 'RETRIEVAL 'WAS
12930     'COMPLETED '#####))
12940     (TERPRI)
12950     (PRIN1 (LIST 'RETRIEVAL 'SENTENCE))
12960     (PRIN1 '=) (PRINT RVLST)
12970     (PRIN1 (LIST 'LAST 'CYCLE))
12980     (PRIN1 '=) (PRINT NUMBR)
12990     (PRIN1 (LIST 'POSITIVE 'EVIDENCE 'LINKS
13000     'LIST))
13010     (PRIN1 '=) (PRINT PELIS) (SPACES 2)
13020     (PRIN1 (LIST 'POSITIVE 'LINKSTRG 'SUMATION))
13030     (PRIN1 '=) (PRINT PESUM)
13040     (PRIN1 (LIST 'NEGATIVE 'EVIDENCE 'LINKS
13050     'LIST))
13060     (PRIN1 '=) (PRINT NELIS) (SPACES 2)
13070     (PRIN1 (LIST 'NEGATIVE 'LINKSTRG 'SUMATION))
13080     (PRIN1 '=) (PRINT NESUM)
13090     (PRIN1 (LIST 'REACTION 'TIME)) (PRIN1 '=)
13100     (PRINT VTIME)
13110     (TERPRI)
13120     (F1GETSTRG2 PELIS)
13130     (F1GETSTRG2 NELIS)
13140     (RETURN NIL) ))) ))
13150 DEFINE((
13160 (F1FINDLINK (LAMBDA (CLS)
13170   (PROG (ZLIS2)
13180     (CSETQ ZLIST (F1PICKUPLNK CLS))
13190     (SETQ ZLIS2 ZLIST)
13200       LOOP
13210         (COND
13220           ((NULL ZLIS2) (RETURN NIL)))
13230         (GETPRINT4 (CAR ZLIS2) 'LINKSTRG 0 0)
13240         (SETQ ZLIS2 (CDR ZLIS2))
13250         (GO LOOP) ))) ))
13260 DEFINE((
13270 (F1PICKUPLNK (LAMBDA (LLS)
13280   (PROG (ANSLS EXLIS)
13290     (SETQ ANSLS NIL)
13300       LOOP
13310         (COND
13320           ((NULL LLS) (RETURN ANSLS)) )
13330         (SETQ EXLIS (EXPLODE (CAR LLS)))
13340         (COND
13350           ((AND
13360              (EQUAL 'Z (CAR EXLIS))
13370              (OR
13380                (EQUAL 'S (CADR EXLIS))
13390                (EQUAL 'O (CADR EXLIS))
13400                (EQUAL 'V (CADR EXLIS))
13410                (EQUAL 'Y (CADR EXLIS)) ) )
13420            (SETQ ANSLS (CONS (CAR LLS) ANSLS)) ))
13430         (SETQ LLS (CDR LLS))
13440         (GO LOOP) ))) ))
13450 DEFINE((
13460 (F1EVIDENCE (LAMBDA (CLS)
13470   (PROG (EVLIS EVNUM)
13480     (F1FINDLINK CLS)
13490     (SETQ EVLIS ZLIST)
13500       LOOP
13510         (COND
13520           ((NULL EVLIS) (RETURN NIL))
13530           ((NULL PELIS)
13540            (CSETQ PELIS (LIST (CAR EVLIS))) ))
13550         (SETQ EVNUM (F1GETNUM (CAR EVLIS)))
13560         (SETQ RSNUM (F1GETNUM (CAR PELIS)))
13570         (COND
13580           ((EQUAL EVNUM RSNUM)
13590            (CSETQ PELIS (CONS (CAR EVLIS) (DELETE
13600            (CAR EVLIS) PELIS))) )
13610           ((NOT (EQUAL EVNUM RSNUM))
13620            (CSETQ NELIS (CONS (CAR EVLIS) (DELETE
13630            (CAR EVLIS) NELIS)))))
13640         (CSETQ VLINK (PACK (LIST 'ZVERB- RSNUM)))
13650         (CSETQ SVLNK (PACK (LIST 'ZYSYMVERB-
13660         RSNUM)))
13670         (F1EVIDENCE2 (CAR EVLIS))
13680         (SETQ EVLIS (CDR EVLIS))
```

```
13690          (GO LOOP) ))) ))
13700 DEFINE((
13710 (F1GETNUM (LAMBDA (ATM)
13720   (PROG (STNUM)
13730     (SETQ STNUM (LAST (EXPLODE ATM)))
13740     (RETURN STNUM) ))) ))
13750 DEFINE((
13760 (LAST (LAMBDA (LIS)
13770   (COND
13780     ((NULL (CDR LIS)) (CAR LIS))
13790     (T (LAST (CDR LIS))) ))) ))
13800 DEFINE((
13810 (F1SUMSTRG (LAMBDA (LIS)
13820   (PROG (ANSER)
13830     (SETQ ANSER 0.0)
13840       LOOP
13850         (COND
13860           ((NULL LIS) (RETURN ANSER)) )
13870         (SETQ ANSER (PLUS ANSER (GET (CAR LIS)
13880         'LINKSTRG)))
13890         (SETQ LIS (CDR LIS))
13900         (GO LOOP) ))) ))
13910 DEFINE((
13920 (F1DELETIME (LAMBDA (NLS)
13930   (PROG (LKLST FSTLK)
13940     (COND
13950       ((NULL NLS)
13960        (RETURN (PRINT (LIST '*** 'END 'OF
13970        'DELETING 'ELPSTIME '***)))))
13980     (SETQ LKLST (GET (CAR NLS) 'LINKLIST))
13990       LOOP2
14000         (COND
14010           ((NULL LKLST)
14020            (RETURN (F1DELETIME (CDR NLS)) )))
14030         (PUT (CAR LKLST) 'ELPSTIME NIL)
14040         (SETQ LKLST (CDR LKLST))
14050         (GO LOOP2) ))) ))
14060 DEFINE((
14070 (F2UNIONDEL (LAMBDA (LS1 LS2)
14080   (PROG (LIST1)
14090     (SETQ LIST1 LS1)
14100       LOOP
14110         (COND
14120           ((NULL LS1)
14130            (RETURN (APPEND LIST1 LS2)) )
14140           ((MEMBER (CAR LS1) LS2)
14150            (SETQ LS2 (DELETE (CAR LS1) LS2)) ))
14160         (SETQ LS1 (CDR LS1))
14170         (GO LOOP) ))) ))
14180 DEFINE((
14190 (F1EVIDENCE2 (LAMBDA (EVL)
14200   (COND
14210     ((AND
14220        (EQUAL EVL VLINK)
14230        (MEMBER SVLNK PELIS)
14240        (GREATERP (GET EVL 'LINKSTRG) (GET SVLNK
14250        'LINKSTRG)) )
14260      (PROG2
14270        (CSETQ PELIS (DELETE SVLNK PELIS))
14280        (CSETQ NELIS (CONS SVLNK (DELETE SVLNK
14290        NELIS))) ))
14300     ((AND
14310        (EQUAL EVL VLINK)
14320        (MEMBER SVLNK PELIS)
14330        (LESSP (GET EVL 'LINKSTRG) (GET SVLNK
14340        'LINKSTRG)) )
14350      (PROG2
14360        (CSETQ PELIS (CDR PELIS))
14370        (CSETQ NELIS (CONS EVL (DELETE EVL
14380        NELIS))) ))
14390     ((AND
14400        (EQUAL EVL SVLNK)
14410        (MEMBER VLINK PELIS)
14420        (GREATERP (GET EVL 'LINKSTRG) (GET VLINK
14430        'LINKSTRG)) )
14440      (PROG2
14450        (CSETQ PELIS (DELETE VLINK PELIS))
14460        (CSETQ NELIS (CONS VLINK (DELETE VLINK
14470        NELIS))) ))
14480     ((AND
14490        (EQUAL EVL SVLNK)
14500        (MEMBER VLINK PELIS)
14510        (LESSP (GET EVL 'LINKSTRG) (GET VLINK
14520        'LINKSTRG)) )
14530      (PROG2
14540        (CSETQ PELIS (CDR PELIS))
14550        (CSETQ NELIS (CONS EVL (DELETE EVL
14560        NELIS)))) )
14570     (T NIL) ) ))) ))
14580 DEFINE((
14590 (F1GETSTRG2 (LAMBDA (LLS)
14600   (PROG NIL
14610     LOOP
14620       (COND
14630         ((NULL LLS) (RETURN NIL)) )
14640       (GETPRINT4 (CAR LLS) 'LINKSTRG 0 0)
14650       (SETQ LLS (CDR LLS))
14660       (GO LOOP) ))) ))
14670 DEFINE((
14680 (F1GETACTV (LAMBDA (NLS)
14690   (PROG (ANSWR)
14700     (CSETQ ANSWR 0.0)
14710       LOOP
14720         (COND
14730           ((NULL NLS)
14740            (PROG NIL
14750              (PRIN1 (LIST 'ACTIVATION))
14760              (PRIN1 '=) (PRINT ANSWR)
14770              (RETURN NIL) )))
14780         (COND
14790           ((NULL NLS) (RETURN (TERPRI))) )
14800         (CSETQ ANSWR (PLUS ANSWR (GET (CAR NLS)
14810         'NODEACTV)))
14820         (SETQ NLS (CDR NLS))
14830         (GO LOOP) ))) ))
14840 DEFINE((
14850 (F1BNODLNK#2 (LAMBDA (NLS)
14860   (PROG (FSTND LKNUM LKLST FSTLK)
14870     LOOP
14880       (COND
14890         ((NULL NLS)
14900          (RETURN (PRINT (LIST '##### 'ALL
14910          'ATTRIBUTES 'WERE 'CLEARED '#####)))))
14920       (SETQ FSTND (CAR NLS))
14930       (PUT FSTND 'NODEACTV 0.0)
14940       (SETQ LKNUM (GET FSTND 'LINKNUMS))
14950       (SETQ LKLST (GET FSTND 'LINKLIST))
14960       (PUT FSTND 'LINKNUMS 0.0)
14970       (PUT FSTND 'LINKLIST NIL)
14980         LOOP2
14990           (COND
15000             ((NULL LKLST)
15010              (PROG2
15020                (SETQ NLS (CDR NLS))
15030                (GO LOOP) )))
15040           (SETQ FSTLK (CAR LKLST))
15050           (PUT FSTLK 'LINKACTV 0.0)
15060           (PUT FSTLK 'ELPSTIME NIL)
15070           (PUT FSTLK 'LINKSTRG 0.0)
15080           (SETQ LKLST (CDR LKLST))
15090           (GO LOOP2) ))) ))
```

ABSTRACT

A COMPUTER SIMULATION FOR THE NETWORK ACTIVATION THEORY OF SENTENCE MEMORY

— A case for S-O-V sentences —

Takashi TSUZUKI

The present study is designed to examine a computer simulation programed for testing the validity of the network activation theory of sentence memory. According to Hayes-Roth & Hayes-Roth (1977), basic assumptions for the network activation theory are summarized as follows.

(1) Each word is represented as an unitary node in the network, and relationships between nodes are represented by appropriately labeled links that connect them.

(2) The meaning of a word is derived from various relationships it has with other nodes. Semantically related words are more directly interconnected in the network than semantically unrelated words.

(3) A sentence is presented as a subnetwork that includes nodes and links, and a higher-order node can represent the subnetwork connected to it.

(4) A linguistic input causes activation of the network representation, and activation spreads in parallel along all the links connected to an active node.

(5) The activation spreads from each activated node is limited at any given point in time.

(6) The strength of each link in memory is an increasing function of the total amount and the frequency of activation it has received, and a decreasing function of the time since its last activation.

For this study, the above network activation theory was translated following Miller (1981) into a LISP 1.5 computer program that consists of an activation system, encoding processes, and retrieval processes. The last two processes are carried out by the common activation system. In this model, activation originated from the active nodes spreads across links. The process of the parallel spreads of activation are simulated by computing the shortest time necessary for activation to cross a link and to active a new node. Generally, the time is computed by: Time = 1/(link strength * link activation). For the model's encoding process, a S-O-V sentence is encoded by gradually constructing a working memory structure called the Encoding Context Network based on the words of the sentence and from the nodes of the semantic network that are related to the words. For the retrieval process, the sentence is searched for along one of these Encoding Context Network structures. The model simultaneously produces estimates of the percent correct, the reaction time, and the confidence rating. These dependent variables are compared to experimental data of sentance memory.

The presented simulation is to explore the extent to which the representational format in memory preserves a sentence's meaning and wording. Many researchers have shown that the memory for a sentence's meaning is far supeior to the retention of that sentence's wording. Although recently, several experiments using the reaction time and the confidence rating showed that distractor sentences synonymous in meaning with previously studied sentences could successfully be discriminated even in the delayed recognition test. For example, in an experiment of Tsuzuki (1983), following an incidental study session and a five-minute intervention task, the subject was given a verification task that required responding true or false to studied sentences, synonymous distractor sentences (constructed by substituting verbs with their synonyms), and nonsynonymous distractor sentences (constructed by substituting verbs with nonsynonymous ones). In this experiments, subjects verified studied sentences more accurately, faster, and more confidently than synonymous and nonsynonymous sentences.

The simulation of these experiments took place in two separate stages, the encoding process and the retrieval process. The results indicated that the corresponding estimates of the percent correct, the reaction time, and the confidence rating from the simulation agreed with those derived from the experiments. This finding provids an evidence to support the network activation theory of sentence memory.