構成論的方法論を用いた自動詞構文における文法の進化モデルと規則の般化

小林昌博

1 はじめに

ヒトが用いる言語は進化の過程でどのように発生し、進化して現在のような形式になったのかという疑問は、関連する派生的ないくつかの問いを生み出した。それらの問いに関する答えは、昨今様々な知見とともに次第に明らかになってきている (Christiansen and Kirby 2003)。本稿では、コンピュータシミュレーションを用いて、主語と動詞のみからなる単純な自動詞構文においてランダムな発話の中から文法規則がどのように発生するのかを考察し、その文法の進化モデルとシミュレーション実験の結果を報告する。構成論的アプローチの手段の一つであるコンピュータシミュレーションのモデルは先行研究のなかでいくつか提案されているが、本稿は、Kirby (2002a) の繰り返し学習 (Iterated Learning) モデルを採用し、修正したものを用いている。本モデルは、比喩的にいうと親が生成した発話を子供が学習し、ランダムな発話の中から規則を見つけ出して、次の世代では自分が親として自分の子供に発話（のセット）を与えるという文化進化 (cultural evolution) プロセスを模している。本研究で行った自動詞構文を対象にしたシミュレーションにおいて、(1) ランダムな言語が飛び交うのは最初の数世代で、いわゆる抽象的な「文法」は進化の比較的初期段階で発生し、語順が一意に収束すると、文レベル・語彙レベルの規則ともにその後安定状態に至ること、また、(2) 語順（この場合「SV」の語順か「VS」の語順か）が一意に決まる場合と、同一文法の中に「SV」と「VS」の二つの規則が生じる場合で、文規則と語彙規則の総数と規則の抽象化プロセスの間に特定のパターンが存在することがわかった。

本稿の構成は以下のようになる。次節では、文法の進化に関する研究の背景や先行研究、Kirby (2002a) で提案されている繰り返し学習モデルを概観する。3 節では、本稿で用いたシミュレーションのアルゴリズムの解説や自動詞構文における進化の詳細をみる。4 節では、実際のシミュレーションの結果とその含意、また今後の研究の展開や方向性を議論する。5 節は結論として議論のまとめを行う。なお、付録として本研究のシミュレーションのために用いたコンピュータプログラムを掲載している（システムの詳細は付録を参照）。

2　背景および先行研究

2.1　文法進化に関する研究の背景

いつヒトの言語が発生したのか、という疑問は前適応からの転用など生物学的な基盤を考慮に入れる必要があるが、文法がある程度の抽象度を持ったのは9万年から14万年という説(Bickerton 2007)もある。このように長い間のヒトの進化過程において、自然言語がいつどのように発生し進化・発展していったのかという疑問は絶えず問われ続けてきた。しかし、当初その研究の多くは確固たる証拠に基づくものではなく、非常に思索的なものであったため、19世紀後半には言語の起源や進化の問題を少なくとも科学的な方法で問うことは困難であるという見方が広がった。しかし、20世紀の生物学、文化人類学、心理学、計算機科学や言語学などの発展を通して言語の進化に関する理解は深まっている。例えば言語学においては、ノーム・チョムスキーがヒトは生まれながらに言語を使用する言語能力(faculty of language)を持っているというアイデアに基づき、その内在的な言語能力を自身が打ち立てた「生成文法」という形式文法理論研究によって明らかにしようとしてきた。生成文法理論は、1960年代に提唱された標準理論(Chomsky 1965)から80年代の原理とパラメータのアプローチ(Chomsky 1981, 1982)、さらに90年代のミニマリスト・プログラム(Chomsky 1995)へと理論の変遷をたどりながらヒトが有する言語能力の研究を発展させてきた。チョムスキーの言語観は、他の形式言語・意味研究の引き金となったと言えよう。これら形式言語の研究を踏まえ、DNA解析への言語理論の応用の研究は盛んにおこなわれた(Searls 2002, Chiang, Joshi and Searls 2006)。さらに人工生命研究も言語進化モデルと深い関係があることがわかった(Kirby 2002b)。また、霊長類とヒトの言語能力の差異に関する数多い研究の中で最近のものとして、Fitch and Hauser (2004)はワタボウシタマリンが正規言語を理解できるが文脈自由言語を理解できないことを明らかにした[1]。霊長類だけでなく、鳥類の歌声の構造原理・文法の研究も多くなされている(Catchpole and Slater 2008, Okanoya 2004)。さらに、多くの言語学者の関心を言語の進化の問題へと向けた研究としてHauser, Chomsky and Fitch (2002)が挙げられる。Hauser et al. (2002)は、狭義の言語能力として言語の再帰性を取り上げている。

その他、言語の進化に関する研究で重要な研究として、構成論的なアプローチが挙げられる。言語進化研究における構成論的な方法論とは、実際の進化のプロセス（のエッセンス部分）をコンピュータ上に再現し、実際にその動作を観察してその仕組みを理解しようという試みである。本研究の枠組みも構成論的アプローチと呼ばれる方法論を採用している。構成論的な枠組みは、1990年代後半から(Hashimoto and Ikegami 1996)など計算機科学の発展とともに研究が行われきた。Hashimoto and Ikegami (1996)は、ネットワークモデルを用いて言語と文法の進化モデルの構築を行っている。また、レプリケータ・ミューテータ方程式を使った進化・言語習得モデルはNowak (2006)に詳しい。その他、コンピュータシミュレーションモデルの構築とアルゴリズムに関して、Cangelosi and Parisi (2002)に詳細がある。本研究が採用している繰り返し学習モデルもこの構成論的な方法論の一つである。次に繰り返し学習モデルの概要を述べる。

[1]正規言語は正規文法により受理・生成ができる言語の集合である。正規文法とは有限状態オートマトンと同じ受理・生成能力を持つ。一方で、自然言語に見られる中央埋め込みタイプの再帰(A^nB^n)などは正規文法では受理・生成不可能であり、メモリ機能を付けたプッシュダウンオートマトン（文脈自由文法）で受理・生成ができる。

2.2　繰り返し学習モデルの概観

本稿で採用した繰り返し学習 (Kirby 2002a) の枠組みは、Kirby (2000) の枠組みの簡略版である。本小節では、本モデルの大枠を解説する。Kirby (2002a) の文化進化モデルでは、各々の世代において親（発話者）と子（聞き手 / 学習者）が存在し、親は自分の持っている文法に基づいてある特定数の意味概念[2]とその意味をあらわす記号列（文）のペアを発話することになっている。子どもは、発話としての意味と文のペアを受け取り、そこからある語順の規則を抽出してより一般化された規則を学習し、次の世代では自分が親となり同じく意味とその意味に該当する文のペアを自分の持つ文法を使って発話・出力する。シミュレーションとしては、各世代において発話者が持つ規則としての文法を観察することになる。子供が意味概念と文のペアの集合から規則を推論して般化するコンポートネントの構築が実装の際には中心となる。本小節では、Kirby (2002a) 規則の般化において重要な概念を紹介する。

• 規則

まず、規則であるが Kirby (2002a) における規則は (1) のように書き換え規則の形式をとる。

(1) a. S/walk(taro) → `abcdef`

b. A/walk → `def`

(1a) は「文規則」で (1b) は「語彙規則」の例である。(1a) の左辺にある「taro」や「walk」は意味概念を表し「S/」の S はラベルを表す[3]。一階述語論理表記の「walk(taro)」は日本語で「太郎が歩く」、英語では "Taro walks." という文が「指し示す意味概念」を表している。規則 (1a) は、「太郎が歩く」という意味概念は「`abcdef`」という記号列で表されるということを表している。一方、(1b) はいわゆる語彙規則に相当する規則で「walk」という意味は「`def`」によって表現されるということを意味している。

• 規則の般化 (rule subsumption)

次に重要な概念として 2 つの規則を推論をまとめあげて一つの規則にする「般化」という操作を挙げる。例として (2a,b) のような二つの規則を考えてみる。

(2) a. S/love(john, mary) → `abcdexyz`

b. S/love(mike, mary) → `vwdexyz`

(2a, b) の左辺はそれぞれ「John loves Mary.」と「Mike loves Mary.」という意味概念を表す表記である。(2a, b) の左辺を比べてみると「john」と「mike」のみが異なり、「love」と「mary」は共通している。右辺を見ると「`dexyz`」が共通している文字列である。したがって、聞き手は文字列「`dexyz`」は「love」と「mary」という文字列を表す記号列と推論をする[4]。さらに (2a) において、「love」と「mary」が「`dexyz`」であるならば「john」は「`abc`」であると推論でき、(2b) において、「mike」は「`vw`」であると推論できる。このような学習プロセスによって聞き手は (2) の 2 つの規則から (3) のように規則を般化、あるいは抽象化することができる。

[2] Kirby (2002a) では「concept」と呼んでいる。

[3] ラベル S は特殊なラベルで、このラベルが付いた規則は文レベルの規則である。

[4] この場合、`dexyz` のうち、どの部分文字列が「love」でどの部分文字列が「mary」を表しているのかは、学習者はこの時点では知るすべがない。

(3) a. S/love(x, mary) → W/x dexyz

b. W/john → abc

c. W/mike → vw

(2) の 2 つの文規則は、(3) に見られるように 1 つの文規則 (3a) と 2 つの語彙規則 (3b, c) へと一般化（抽象化）される。この操作を「般化」と呼ぶ。(3a) は主語の位置が変数化されており、右辺には該当する変数と任意のラベルからなる「W/x」という表記がある。「W/x」は、W というラベルを語彙規則で持つ単語であればいかなる単語でも文字列の当該の箇所に生起することができるという意味を表す。左辺に出てくる要素は、ラベル以外は般化の操作によりすべて変数化することが可能である。(3b) と (3c) の語彙規則は、文規則においてラベルが W が現れている場所に右辺の文字列を挿入することができることを表している。

規則（般化されているかいないかに関わらず）を用いることで、シミュレーションにおいて文を「解析」したり、意味概念を表す文を「生成」したりすることができる。実際に規則を用いた文の解析と生成の例を見てみる。(3) の規則を用いると (4) にみられるような意味概念を表す文を解析することができる。ここでいう「解析」とは、学習者が発話者から受け取った意味概念とそれを表す文に対して、学習者自身の持つ文法を使ってその文が当該の意味を表す文として正しいかをチェックすることである。

(4) a. love(john, mary) : abcdexyz

b. love(mike, mary) : vwdexyz

(4) は意味概念と文のペアの例である。文法を用いた解析の仕組みを見てみる。(3a) によると、「x loves Mary」という意味を表す文字列（文）は、「$\cdots$ dexyz」という文字列である。ここで「$\cdots$」は、W のラベルを持つ単語規則の右辺の文字列を表す。(4a) を解析対象の文とすると、変数 x に該当する単語は「John」であり、(3b) によると「John」はラベルが W のラベルがついている文規則の右辺記号列の部分文字列「abc」として生起することが可能であることがわかる。(4a) によると、「love(john, mary)」を表す文字列は「abcdexyz」であり、これは上記の「$\cdots$」を「abc」と置き換えた文字列とマッチするため、(4a) の意味と文は (3) の規則によって解析が可能であることがわかる。同じ判断手順により (4b) の意味と文も (3a,c) の規則によって解析できる。

次は生成の例を見てみる。発話者が (5) の文法規則を持っており、「love(john, mary)」と「love(mike, mary)」の 2 つの意味概念を表す文を生成・発話しようとしているとする。

(5) a. S/love(x, mary) → W/x dexyz

b. W/john → abc

c. Q/mike → vw

「love(john, mary)」と (5a) の左辺を比較してみる。変数 x を「John」に置き換えると意味概念と一致する。(5a) の右辺をみると、変数 x はラベル W を持っていることがわかる。語彙規則を見ると「John」はラベル W を持っているため、(5a) の左辺である「W/x dexyz」の「W/x」の部分を (5b) の右辺「abc」と置き換えることで「love(john, mary)」を表す文として「abcdexyz」

を生成することができる。一方で、「love(mike, mary)」を表す文を生成する際には少し注意が必要である。「love(john, mary)」の場合の生成過程と同様に、「love(mike, mary)」でも (5a) の規則を使用する。しかし、変数 x のラベルが W である一方、「mike」の語彙規則のラベルは Q であるため、(5c) の語彙規則を適用することができない。このような場合ランダムな文字列を生成し、(5a) の右辺の W/x と置き換えることで文を生成する[5]。結果として例えば「love(mike, mary)」の意味概念を表す文字列として「mudexyz」などが生成される。このランダムな文字列生成の操作に関して、何文字までの文字列をランダムに生成するかという点は実験のパラメータとして変更可能である。

• 語彙規則のラベルの単一化操作

語彙規則において (6) のように書き換え規則の右辺が同じで単語も同じであるのにラベルが異なる 2 つの規則が存在する場合、ラベルを一つにまとめる操作を適用する。

(6) a. W/john → abc

b. Y/john → abc

(6) の 2 つの規則は、「W/john → abc」という 1 つの規則になる。注意が必要な点は、語彙規則でラベルの変更が行われた場合、文規則においてもラベルの変更が適用される。具体的には、この場合、すべての文規則の中のラベル Y は W に変更される。本研究でも基本的に本節で解説した学習アルゴリズムを利用している。

次節では、本研究で用いたシミュレーションアリゴリズムと自動詞構文の文法進化の概要について述べる。

3　コンピュータのシミュレーション実験の概要

本研究のシミレーションは前述のとおり、Kirby (2002a) を修正したものを用いている。本節では、まずシミュレーション全体のアルゴリズムを概観し、流れを確認する。そして本研究においてどのように自動詞構文を定義し取り扱ったのかを概観し、簡単な進化の例を見てみる。

3.1　シミュレーションのアルゴリズム

2 節ですでに述べているように、本研究は Kirby (2002a) の繰り返し学習モデルを採用している。モデルのイメージは図 1 のように図式化される。本モデルは、親に相当する個体がランダムに生成された意味概念（run(david) のような概念）の集合に対して[6]、自分が持っている文法規則を使ってその意味概念に相当する文字列を生成する。ここで注意しなければならないこととして次の 2 点が挙げられる。まず、シミュレーションの最初の世代の個体は文法の知識を何も持ち合わせていないため、意味概念に対してランダムに文字列を生成して意味概念を表す文とする可能性が極めて高いということが挙げられる。次に 2 点目として、発話は意味概念とそ

[5] Kirby (2002a) ではこの操作を invention と呼んでいる

[6] 各個体がいくつの意味概念を扱うかはシミュレーションのパラメータとして設定可能である。イメージとしては各世代の話者に対して話す意味内容は自動的にどこからか（実際にはシミュレーションの実施者だが）降ってきて、話者は保持している文法を用いてその意味を表す文（文字列）を生成する。

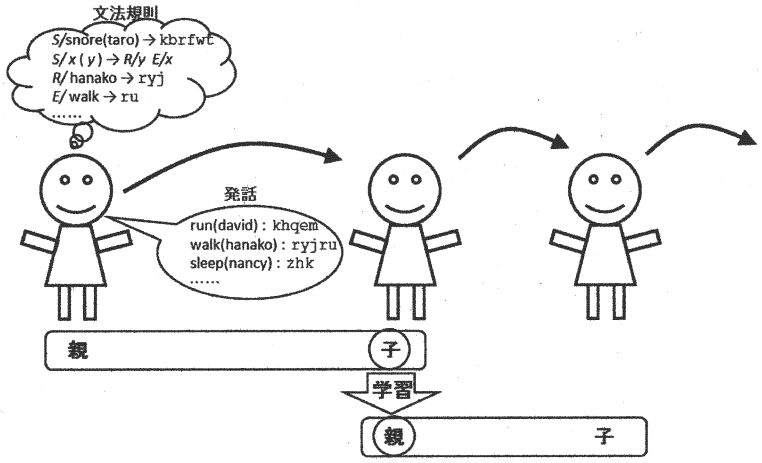

図 1: 世代間のコミュニケーションモデル

れを表す文字列のペアの集合という形式をとるのであるが、自分の発話内容に矛盾がないようにそれぞれの文字列を生成しながら規則の般化も同時に行うこととしている。これは、ある一定量の意味概念の集合を処理させる場合、その中に同じ意味概念が含まれる可能性があることは想像に難くなく、その場合、規則を般化によって学習していない場合は、ランダムな文字列生成によって同じ意味概念を異なる文字列で表現してしまう可能性があるからである。同じ意味概念を表す文字列は同じでなければ子ども世代の学習にノイズがかなり含まれてしまうことになりかねない。このように親は子どもに対して意味と文（文字列）のペアの集合を発話する。

子どもの個体は、入力として意味と文（文字列）のペアの集合を発話として受け取る。子どもは般化の操作を使って意味と文字列のペアの集合から規則性を抽出し学習する。学習のプロセスが終了すると、親の個体はシミュレーションからはずれ、子どもが新しい次世代の親となり、次世代の子どもとして文法知識のない個体を新たにシミュレーションに加えて同じ作業を続ける。親が発話をし終わったときに親の文法をその世代の文法として記録することになる。世代間のループ数もシミュレーションのパラメータとして設定する。シミュレーションの全体の流れを図 2 に示す。本実験では、60 世代までのシミュレーション実験を行っている。

次に意味概念の生成についてであるが、実験では、8 つの固有名詞（Taro や Hanako など）と 8 つの自動詞（run や walk など）からランダムに組み合わせて 60 の意味をそれぞれの世代で生成して話者に与えた。8 つの名詞と 8 つの自動詞の組み合わせなので確率的に 64 個以下の 60 個の意味概念には、すべての組み合わせが含まれていないこともある。つまり各世代の発話や学習において、同じ意味がないような可能性を残すようにデザインした。

3.2 自動詞構文の文法の進化

次に、実際のシミュレーションのプロセスにおいてどのような文法の発生・進化が観察されたかについて詳細を見てみる。本研究で取り上げた構文は単純な自動詞構文である。したがって、規則は例えば (7) のような形式になる。

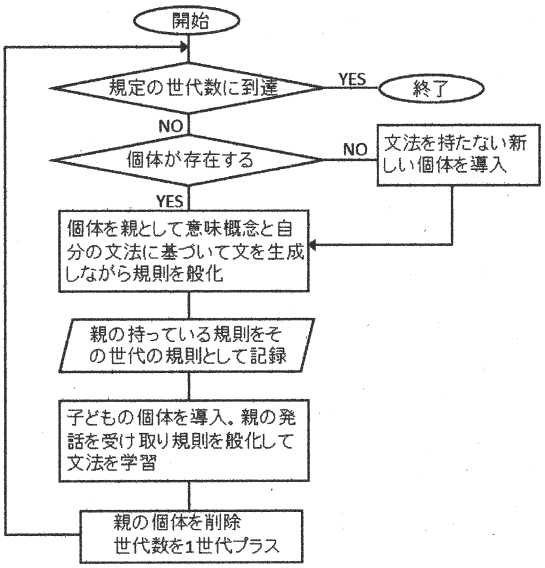

図 2: シミュレーションのアルゴリズム

(7) S/walk(david) → duyalzh

自動詞構文は単語が主語と動詞の 2 つしかないため、仮に主語が規則の般化操作により般化された場合、自動的に動詞が右辺のどの文字列と対応するかがわかるため、動詞も変数化されて般化される。またその逆も成り立つ。このように他動詞構文と比較して進化プロセス（世代交代）数を少なく設定することが可能となった。今回は、各世代において発話に用いる意味概念の数を 60 に、文生成時にランダムに発生させる文字数の最大数を 4 文字[7] に設定してシミュレーション実験を行った。

シミュレーションの一般的な傾向として、最初の数世代は特に般化されていない規則、つまり意味概念そのものと文字列の関係がそのまま規則化されているパターンが多く見てとれる。例えば図 3 はとあるシミレーション結果の最初の世代の文法規則である。ほとんどの文規則において般化が行われておらず、意味概念と任意の文字列が関連付けられているのみだが、唯一般化された規則があり（$S/\ x\,(\,y\,) \rightarrow A/y\ \ B/x$）、語順が「VS」の言語になっていることがわかる。また、語彙規則のラベルをみると、8 つの固有名詞と 8 つの自動詞に関してそれぞれの品詞の分類（ここでは動詞が「A」で、名詞が「B」）がすでに学習されていることも見てとれる。ここで注意が必要なこととして、すでに般化された文規則と語彙規則が存在するのにも関わらず他の文規則が存在する必要があるのかという疑問が挙げられるかもしれないが、生成の過程で般化されている規則とされていない規則では、般化されている規則が優先的に適用されるようにプログラムされているため、使用頻度の低い規則は進化の過程で消滅してしまうことが予想される。図 4 は図 3 と同じ試行における 60 世代目の文法規則である。図 4 をみると、60 世代経過時にはすでに文規則は般化された規則（$S/y(x) \rightarrow AP/y\ \ AO/x$）のみとなり、8 つの固有名詞と 8 つの動詞も品詞ごとにラベリングされていることがわかる。60 世代目の話者は図

[7]今回は、最大 4 文字として実際に何文字になるかもランダムに決まるようにした。

世代1:

S/ laugh(john) → swvjjq
S/ drink(mary) → nuutqsa
S/ sleep(hanako) → fiwr
S/ snore(mike) → thkgypk
S/ y (*x*) → *A/y B/x*
S/ smoke(john) → hd
S/ smoke(david) → dfpdo
S/ smoke(hanako) → kxdteyw
S/ laugh(ichiro) → vyj
S/ snore(taro) → zwj
S/ cry(hanako) → ykhyth
S/ cry(mary) → ouk
S/ cry(taro) → gchxy
S/ run(taro) → rrykjxqc
S/ cry(ichiro) → ncuhzu
S/ run(john) → kgthfulw

A/ laugh → p
A/ snore → dr
A/ walk → asx
A/ smoke → ht
A/ drink → qy
A/ cry → ne
A/ sleep → tm
A/ run → nmd
B/ david → bvcjr
B/ mike → ktnv
B/ hanako → qsm
B/ mary → ook
B/ taro → p
B/ john → s
B/ ichiro → y
B/ nancy → z

図 3: あるシミュレーションの試行における第 1 世代目の文法規則

世代60:

S/ y (*x*) → *AP/y AO/x*

AP/ snore → upe
AP/ laugh → p
AP/ cry → ou
AP/ smoke → h
AP/ drink → umoj
AP/ sleep → ijixen
AP/ walk → asx
AP/ run → nmd

AO/ mary → ook
AO/ hanako → qsm
AO/ ichiro → y
AO/ taro → p
AO/ david → bvcjr
AO/ mike → ktnv
AO/ nancy → z
AO/ john → d

図 4: 図 3 の試行と同じシミュレーションの第 60 世代目の文法規則

世代4:

S/ y (x) → AR/x AQ/y
S/ y (x) → AT/y AS/x

AQ/ smoke → hk
AQ/ laugh → gyvj
AQ/ cry → qk
AQ/ snore → lurg
AQ/ drink → v
AQ/ run → jke
AQ/ sleep → ufh
AQ/ walk → qxr

AR/ mary → dnrb
AR/ hanako → w
AR/ ichiro → fbz
AR/ taro → aiyp
AR/ david → sfj
AR/ mike → cdkf
AR/ nancy → wy

AT/ walk → a
AT/ smoke → smgo
AT/ cry → ny
AT/ snore → rlh
AT/ run → yi
AT/ laugh → vzsn

AS/ john → sorw

図 5: 語順が 2 パターン観察されたシミュレーションの第 4 世代目の文法規則

4 の文法規則を持っているので、例えば「ijixenqsm」が「花子は寝ている（sleep(hanako)）」という意味を表す文として正しい文字列であると理解・認識（解析）することができ、「マイクが歩いている（walk(mike)）」という意味概念を表す文として「asxktnv」を出力（生成）することができる能力を有しているということができる。

今回のシミュレーションでは、自動詞構文を対象としているため、語順に関しては理論的には「SV」パターンに帰着する場合と「VS」パターンの言語が進化する確率は半々である。興味深いのだが、シミュレーションの試行において「SV」のパターンも「VS」のパターンも両方文法内に存在し、それが長期間世代をこえて受け継がれることがあることがわかった。図 5 は 2 通りの語順が発生したあるシミュレーション試行における 4 世代目の文法の状態である。シミュレーション上の文法進化が始まって間もない世代だが、文規則としてすでに「SV」の語順と「VS」の語順の 2 パターンの般化規則が観察される。また、両方の語順とも動詞と名詞が品詞ごとに識別されている。「SV」パターンの主語は *AR* のラベルを持ち、動詞は *AQ* のラベルを持っていることがわかる。「VS」パターンの主語は *AS* のラベルで、動詞は *AT* のラベルを持つということになる。図 5 を見ると、例えば同じ動詞でも「SV」の時と「VS」の時では違う文字列で表現する（例えば「SV」パターンでは「walk」は「qxr」だが、「VS」パターンでは「a」であるなど）ことがわかる。仮に図 5 を文法として持つような言語が話されている世界があったとすれば、「太郎が寝ている」という意味は「aiypufh」と表現され、「ジョンが笑っている」という意味を表す場合は「vzsnsorw」と言わなければばらない。その世界では「ジョン」のことを言及するときのみ「VS」の語順を使用しなければならず、かつ「VS」の語順は「SV」と比較して表現能力（様々な意味を発話したり、文字列を解析したりする能力）が低いということが言えるだろう。図 6 は図 5 と同じシミュレーション試行の 60 世代目の文法規

世代60:

S/ y (x) → *AQ/x AR/y*
S/ y (x) → *AO/y AP/x*

AR/ smoke → dmei
AR/ snore → lurg
AR/ cry → rg
AR/ drink → v
AR/ run → z
AR/ laugh → gyvj
AR/ walk → qkwf
AR/ sleep → ufh

AQ/ nancy → ycdbajeu
AQ/ mike → cajdkf
AQ/ hanako → w
AQ/ taro → aiyp
AQ/ ichiro → fbz
AQ/ david → sfjq

AO/ smoke → caj
AO/ run → u
AO/ sleep → iwmp
AO/ cry → i
AO/ snore → o
AO/ walk → c
AO/ laugh → zr
AO/ drink → b

AP/ john → kxhysorw
AP/ mary → nrbjke

図 6: 図 5 と同じ試行の第 60 世代目の文法規則

則である。図 6 と図 5 を比較してみると次の 2 つのことがわかった。まず、それぞれの語順のパターンにおいていくつかの単語が進化の過程で生き残っていることが指摘できる。例えば、「SV」パターンの語順において 4 世代目も 60 世代目も「snore」は「lurg」として表記されるし、「hanako」は「w」として受け継がれている。2 つ目として非常に興味深いのだが、1 つの言語体系において語順が 2 通り存在することは無駄であるように思われるかもしれないが、図 6 を見てみると全ての表現が片方どちらの語順で実現されているわけではないことがわかる。具体的には動詞に関しては、「SV」「VS」のどちらも 8 つの動詞が語彙規則として規則化されているが、名詞に関しては「SV」パターンで 6 つが規則化され、残りの 2 つは「VS」パターンにおいて規則化されている。つまり、「ジョン」と「メアリー」について言及するときは「VS」の語順を使い、それ以外は「SV」の語順で表現するというような、いわば相補的な関係を進化させていることがわかる。語順に関して冗長な文法体系においても意味のある相補関係が発生する点が興味深い。

次節では、般化規則の数と文法の進化の関係を世代経過とともに考察する。

4　データの解析結果と今後の展開

4.1　解析結果

本節では、語順が一意に収束した場合と 2 通りの語順が生き残った場合、そしてそれらの規則の般化の世代経過を観察する。本研究では、文を生成する際に用いる意味概念の数を 60 に、文生成時にランダムに発生させる文字数の最大値を 4 に設定してシミュレーション実験を行った。20 回のシミュレーションを行い、「SV」か「VS」のどちらか 1 つの語順パターンに収束した試行が 11 回で 2 通りの語順を持つ文法規則に収束した例が 9 回であった。ほぼ半分の確率で 1 つしか語順を持たない場合と 2 通りの語順パターンを持つ場合が観察された。また最終的に般化されない規則が残るということもなかった。毎回のシミュレーションの試行において般化の推移に関してわずかな違いはあるものの、1 通りの語順か 2 通りの語順か、それぞれのケースごとにある一定の般化推移パターンに落ち着くことが明らかになった。まず語順が 1 つのパターンに収束した例から見てみる。図 7 は語順が一意に収束したシミュレーション試行の典型的な規則の推移と世代の関係のパターンを表している。なお、シミュレーション自体は 60 世代まで実行しているが、40 世代目以降は基本的に変化がないため、それ以降は図の中では省略をしている。般化された規則数は、最初の世代は 1 つであるが次に 2 通りの規則に進化し、17 世代目から再び一意の語順に収束している。語順が 1 つのパターンに収束する場合、それぞれのシミュレーション試行において次のことが共通して観察される。まず、(1) 進化の過程の最初の数世代において、語彙に関する規則は増加する傾向にある。(2) その一方で、般化されていない文レベルの規則（図 7 では点線で表示）は最初の数世代で減少する傾向にある [8]。興味深いのは (3) 基本的に語彙レベルの規則数は徐々に減少していくのだが、般化された規則数が 2 から 1 に収束するのと同時に語彙規則数は 16 へと収束する。これは 8 つの固有名詞と 8 つの自動詞、合計 16 の単語が全て規則化されることに起因する。

次に同一の個体内に「SV」「VS」の 2 つの語順パターンが定着する例（図 8）を見てみる。図

[8] 実験のデザインをもう少し複雑なものにすれば（例えば、一つの世代で複数の個体を導入して会話をさせ、その後次の世代へと発話をするなど）、規則が般化されるまでもう少し長い世代を観察する必要がでてくるだろう。

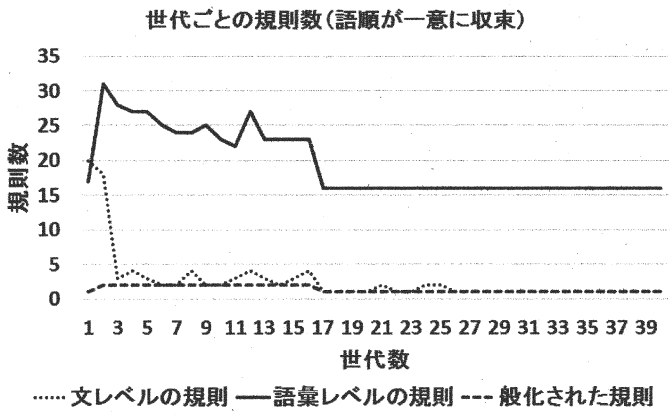

図 7: 語順が 1 パターンに収束した世代と規則の典型的な推移関係

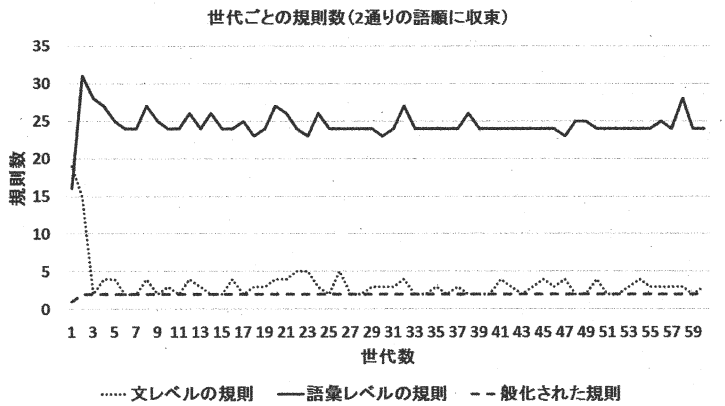

図 8: 「SV」と「VS」のパターンを持つ文法の世代と規則の典型的な推移関係

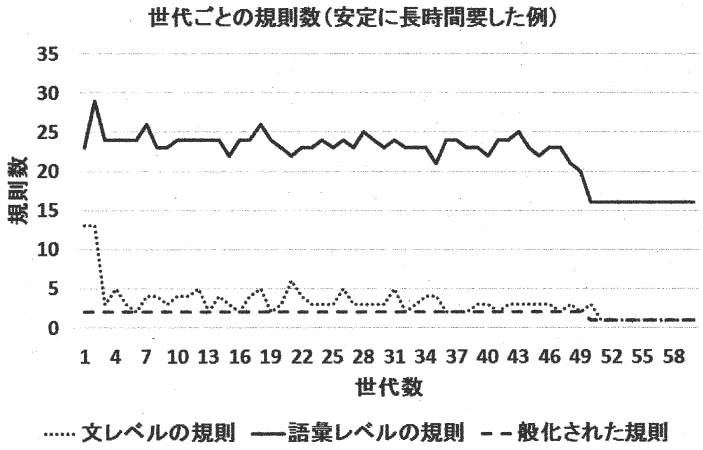

図 9: 収束に時間を要した珍しい規則数の推移

8 に見られるように、2 世代目から 60 世代目まで般化規則は 2 つの語順を持っている。図 7 では語順が 1 つに収束すると語彙規則数は 16 で安定状態となったが、図 8 では文レベル・語彙レベル両方の規則において完全に安定はせずに推移することがわかる。

最後に、最終的に語順が 1 つのパターンに安定したのだが、図 7 のパターンから外れたシミュレーション試行が観察されたので報告する。語順が 1 つのパターンに収束する場合、微妙な差があるにせよほとんど 20 世代目までにどちらか 1 つの語順に安定したが、20 回のシミュレーション試行のうち一度だけ図のように約 50 世代目になって語順が一つに定まった試行があった（図 9）。この場合も、語順が 1 つに定まった世代で語彙規則の数も 16 になって、以後安定状態になった。このように、語順が 2 通りある場合も進化シミュレーションを長期間世代に渡って観察すれば何らかのきっかけで語順が 1 つに定まる可能性も否定できない。どのような理由によってこれが生じうるかは現在のところはっきりとはしていない。

次に、今回の研究を踏まえて今後の研究の方向性と発展を考察する。

4.2　今後の展開

今回は、比較的シンプルなデザインに基づいたコンピュータシミュレーションであったため、進化計算に要する世代数も少なめであった。今後は、本研究のシミュレーション結果をさらに詳細に見ていくことはもちろん、主に以下のような項目に関しての研究を発展・展開させていく予定である。

- シミュレーションに参加する個体の増加
- 他動詞構文への展開
- 複文構造への展開
- 再帰構造の発生のシミュレーション

今回は各々の世代において親の個体と子どもの個体の2個体のインタラクションが進化計算の中心であったが、ノイズが少ないためか比較的順調に規則の般化が進んでいった。今後は、各世代で複数の個体からなるコミュニティなどを想定し、それらのグループ内での発話や文理解なども導入して進化モデルに深みを持たせることを考えている。他動詞構文を対象にすると名詞がもう一つ加わることとなり、複雑さも大幅に増すであろう。今後は自動詞構文と他動詞構文をまぜた意味概念からの文生成や理解を採用する。複文構造は再帰構造を許す文法へと進化する際に必要な構造と考えているので、今後の重要な研究トピックである。また正規文法からどのように中央埋め込み構文を生成する文脈自由文法が発生するのかを本研究の方法論である文化進化の観点から考察した研究はまだないので、今後の発展の重要な方向の一つである。

5 結論

本稿では、Kirby (2002a) のモデルに基づき自動詞構文を使った意味概念からの文生成と解析を通して、文法がどのように発生・進化していくかをコンピュータシミュレーションを利用して実証した。主にシミュレーション実験から以下の2つが明らかになった。まず、ランダムな言語が飛び交うのは最初の数世代で、いわゆる抽象的な「文法」は進化の比較的初期段階で発生し、般化操作によって語順が一意に収束すると、文レベル・語彙レベルの規則ともその後安定状態に至ったことがわかった。今回の実験はモデル自体と学習アルゴリズムが比較的シンプルだったため進化のプロセス自体も単純化されがちであり、12万年前に発生したともいわれる文法メカニズムが明らかになったわけではないが、文法の般化が予想より初期で安定することから生物学的な前適応からの転用の準備ができていれば比較的早く文法の進化と抽象化は進んだと考えている。

次に明らかになったことは、語順が一意に収束した文法と2通りの般化された文規則を許す文法とでは、規則の数において顕著なパターンの違いがあることである。1通りの語順で安定した文法系では、語彙規則もある一定の数（今回は名詞と動詞の数の合計である16）で安定したが、2通りの語順を持つ文法では語彙レベルでも文レベルでも規則数は世代が経過しても不安定なままであった。これは、現在私たちが使用している自然言語を見てみればわかることで、基本的にそれぞれの言語では1つの語順を持つ言語が多数であることとも関連しているだろう。2通りではなく1通りの語順に収束するように今回の実験にバイアスをかけることができればさらに明らかになることがあるであろう。本研究で明らかになったことを踏まえて、今後は他動詞構文や複文構造などへの展開、そして正規文法から文脈自由文法への文法の進化のメカニズムを調べることを考えている。

参考文献

Bickerton, D. 2007. Language Evolution: A brief guide for linguists. *Linguia*. 117. pp. 510-526.

Cangelosi, A. and D. Parisi. (eds.) 2002. *Simulating the Evolution of Language*. Springer.

Catchpole, C. K. and P. J. B. Slater. 2008. *Bird Song: Biological Themes and Variations*. Cambridge University Press.

Chiang, D., A. Joshi and D. Searls. 2006. Grammatical Representations of Macromolecular Structure. *Journal of Computational Biology.* vol. 13. pp. 1077-1100.

Chomsky, N. 1965. *Aspects of the Theory of Syntax.* The MIT Press.

Chomsky, N. 1981. *Lectures on Government and Binding: The Pisa Lectures.* Mouton de Gruyter.

Chomsky, N. 1982. *Some Concepts and Consequences of the Theory of Government and Binding.* The MIT Press.

Chomsky, N. 1995. *The Minimalist Program.* The MIT Press.

Christiansen, M. H. and S. Kirby. 2003. Language Evolution: Consensus and Controversies. *Trends in Cognitive Sciences.* vol. 7. pp. 300-307.

Fitch, W. T. and M. D. Hauser. 2004. Computational Constraints on Syntactic Processing in a Non-human Primate. *Science.* vol. 303. pp. 377-380.

Hashimoto, T. and T. Ikegami. 1996. Emergence of net-grammar in communicating agents. *BioSystems.* vol. 38. pp. 1-14.

Hauser, M., D., N. Chomsky and W. T. Fitch. 2002. The Faculty of Language: What Is It, Who Has It, and How Did It Evolve? *Science.* vol. 298. pp. 1569-1579.

Kirby, S. 2000. Syntax without Natural Selection: How Compositionality Emerges from Vocabulary in a Population of Learners. In Knight, C., J. Hurford and M. Studdert-Kennedy (eds.) *The Emergence of Language.* pp. 303-323. Cambridge University Press.

Kirby, S. 2002a. Learning, Bottlenecks and the Evolution of Recursive Syntax. In Briscoe, T. (ed.) *Linguistic Evolution through Language Acquisition: Formal and Computational Models.* pp. 173-294. Cambridge University Press.

Kirby, S. 2002b. Natural Language from Artificial Life. *Artifical Life.* vol. 8. pp. 185-215.

Nowak, M. 2006. *Evolutionary Dynamics: Exploring the Equations of Life.* Belknap Press.

Okanoya, K. 2004. Song Syntax in Bengalese Finches: Proximate and Ultimate Analyses. *Advances in the Study of Behavior.* vol. 34. pp. 297-346.

Searls, D. 2002. The Language of Genes. *Nature.* vol. 420. pp. 211-217.

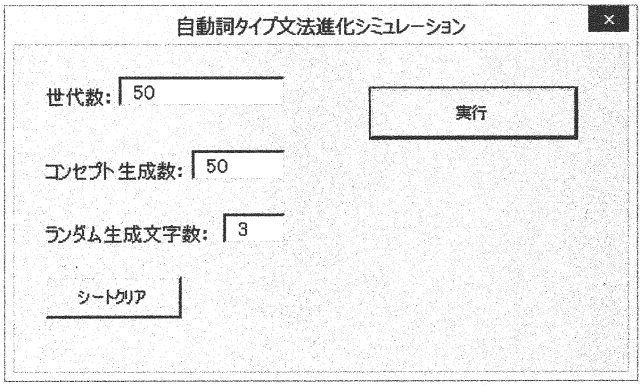

図 10: シミュレーション実験のコンソール

6 付録

ここでは、実際にシミュレーションに用いたシステム及び内部のコードを示す。図 10 はシミュレーションのコンソール画面である。今回の実験ではパラメータとして使った「世代数」と「コンセプト生成数」、「ランダム生成文字数」を設定できるようにしてある。今回は Microsoft Excel に組み込みの VBA を用いてシステムを構築した。世代数を表すテキストボックスはシステム上で「TextBox1」として、コンセプトの生成数は「TextBox2」として、ランダム生成文字数をしているテキストボックスは「TextBox3」として記述されている。本システムを走らせるためには Excel 上で、次の 7 つのシートを設定しなければならない。シート名は、文規則が表示される「srule」、語彙規則が表示される「wrule」、規則の般化に使うラベルを表示している「label」、使用する名詞と動詞が書いてある「iv」、それぞれの世代で使用する意味概念が生成・表示される「concepts」、発話が表示される「utterances」、シミュレーションの結果が表示される「result」である。シミレーション開始時には「srule」と「wrule」、「concepts」、「utterances」、「result」のシートは空シートで構わない。「label」シートには、A の列に 78 個、異なる文字等を使用するラベルで記入しておく。B 列の 1 行目には「1」を記入する。「iv」シートの A 列には 8 つの名詞、B 列には 8 つの自動詞を記入しておく。コンソール上の「実行」ボタンを押すと、以下の「exec_evolution()」が実行される。以下がコードである。

```
Private Sub exec_evolution()
    Dim i, generation_num As Long
    generation_num = TextBox1.Value

    For i = 1 To generation_num
        Call generate_concepts

        Call produce_utterances
        Call print_result((i))

        Sheets("srule").Cells.Clear
        Sheets("wrule").Cells.Clear
```

```
        Call copy_utterances_to_srule
        Call abstract_rules

        Sheets("utterances").Cells.Clear
    Next
End Sub

Sub print_result(i As Long)
    Dim x, result_maxrow, srule_maxrow As Long
    Dim wrule_maxrow, srule_i, wrule_i As Long

    Call delete_dup_wrule

    If Sheets("result").Cells(Rows.Count, 1).End(xlUp).Row < _
       Sheets("result").Cells(Rows.Count, 7).End(xlUp).Row Then
        result_maxrow = _
        Sheets("result").Cells(Rows.Count, 7).End(xlUp).Row
    Else
        result_maxrow = _
        Sheets("result").Cells(Rows.Count, 1).End(xlUp).Row
    End If
    srule_maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row
    wrule_maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row

    If Sheets("result").Cells(1, 1) = "" And _
       Sheets("result").Cells(1, 7) = "" Then
        result_maxrow = 0
    End If

    If result_maxrow = 0 Then
        Sheets("result").Cells(1, 3) = "to"
        Sheets("result").Cells(1, 4) = TextBox1.Value
    End If
    Sheets("result").Cells(result_maxrow + 1, 1) = "Generation"
    Sheets("result").Cells(result_maxrow + 1, 2) = i
    srule_i = 1
    For x = result_maxrow + 2 To result_maxrow + 2 + srule_maxrow
        Sheets("result").Cells(x, 1) = Sheets("srule").Cells(srule_i, 1)
        Sheets("result").Cells(x, 2) = Sheets("srule").Cells(srule_i, 2)
        Sheets("result").Cells(x, 3) = Sheets("srule").Cells(srule_i, 3)
        Sheets("result").Cells(x, 4) = Sheets("srule").Cells(srule_i, 4)
        Sheets("result").Cells(x, 5) = Sheets("srule").Cells(srule_i, 5)
        srule_i = srule_i + 1
    Next
    wrule_i = 1
    For x = result_maxrow + 2 To result_maxrow + 2 + wrule_maxrow
        Sheets("result").Cells(x, 7) = Sheets("wrule").Cells(wrule_i, 1)
        Sheets("result").Cells(x, 8) = Sheets("wrule").Cells(wrule_i, 2)
        Sheets("result").Cells(x, 9) = Sheets("wrule").Cells(wrule_i, 3)
        wrule_i = wrule_i + 1
    Next

End Sub

Sub delete_dup_wrule()
    Dim i, j, maxrow, flag As Long
```

```
    maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row

    flag = 0
    For i = 1 To maxrow
        For j = i + 1 To maxrow
            If Sheets("wrule").Cells(i, 1) = Sheets("wrule").Cells(j, 1) And _
               Sheets("wrule").Cells(i, 2) = Sheets("wrule").Cells(j, 2) And _
               Sheets("wrule").Cells(i, 3) = Sheets("wrule").Cells(j, 3) Then
                Sheets("wrule").Rows(j).Delete
                flag = 1
                Exit For
            End If
        Next
        If flag = 1 Then
            Exit For
        End If
    Next

    If flag = 1 Then
        Call delete_dup_wrule
    End If

End Sub

Sub copy_utterances_to_srule()
    Dim maxrow, i As Long

    Call delete_dup_utterances

    maxrow = Sheets("utterances").Cells(Rows.Count, 1).End(xlUp).Row

    For i = 1 To maxrow
        Sheets("srule").Cells(i, 1) = Sheets("utterances").Cells(i, 1)
        Sheets("srule").Cells(i, 2) = Sheets("utterances").Cells(i, 2)
        Sheets("srule").Cells(i, 3) = Sheets("utterances").Cells(i, 3)

    Next
End Sub

Sub delete_dup_utterances()
    Dim maxrow, i, j As Long
    maxrow = Sheets("utterances").Cells(Rows.Count, 1).End(xlUp).Row

    For i = 1 To maxrow
        For j = i To maxrow
            If i <> j Then
                If Sheets("utterances").Cells(i, 1) = _
                   Sheets("utterances").Cells(j, 1) And _
                   Sheets("utterances").Cells(i, 2) = _
                   Sheets("utterances").Cells(j, 2) And _
                   Sheets("utterances").Cells(i, 3) = _
                   Sheets("utterances").Cells(j, 3) Then
                    Sheets("utterances").Rows(j).Delete
                    Exit For
                End If
            End If
        Next
```

```
    Next

    If maxrow = Sheets("utterances").Cells(Rows.Count, 1).End(xlUp).Row Then
    Else
        Call delete_dup_utterances
    End If

End Sub

Sub abstract_rules()
    Dim flag, before, after, maxrow, i, j As Long
    Dim shared_str As String
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row
    before = maxrow
    shared_str = ""
    flag = 0
    For i = 1 To maxrow
        For j = i To maxrow
            If i <> j Then
                flag = top_level_str_compare((i), (j))
                If flag = 1 Then
                    Call delete_dup_wrule
                    Exit For
                End If
            End If
        Next
        If flag = 1 Then
            Exit For
        End If
    Next
    after = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row
    If before <> after Then
        Call abstract_rules
    End If
End Sub

'Procedure for comparing strings
Function top_level_str_compare(i As Long, j As Long) As Long
    Dim flag As Integer
    flag = 0

    If Sheets("srule").Cells(i, 1) <> 1 And _
       Sheets("srule").Cells(i, 2) <> 2 And _
       Sheets("srule").Cells(j, 1) <> 1 And _
       Sheets("srule").Cells(j, 2) <> 2 Then
        If Right(Sheets("srule").Cells(i, 3), 1) = _
           Right(Sheets("srule").Cells(j, 3), 1) And _
           Left(Sheets("srule").Cells(i, 3), 1) = _
           Left(Sheets("srule").Cells(j, 3), 1) Then
        Else
            If InStr(Sheets("srule").Cells(i, 3), _
                     Sheets("srule").Cells(j, 3)) = 0 And _
               InStr(Sheets("srule").Cells(j, 3), _
                     Sheets("srule").Cells(i, 3)) = 0 Then
                 flag = i_not_analyzed_j_not_analyzed(i, j)
            End If
        End If
```

```
    Else
        flag = i_not_analyzed_j_analyzed(i, j)
        If flag = 0 Then
            flag = i_analyzed_j_not_analyzed(i, j)
            If flag = 0 Then
                flag = i_analyzed_j_analyzed(i, j)
            End If
        End If
    End If
    top_level_str_compare = flag
End Function

Function i_analyzed_j_analyzed(i As Long, j As Long) As Long
    Dim flag, maxrow, x As Long, i_word_order, j_word_order As String
    Dim i_fst_lab, i_snd_lab, j_fst_lab, j_snd_lab As Variant
    flag = 0
    If Sheets("srule").Cells(i, 1) = 1 And Sheets("srule").Cells(i, 2) = 2 And _
       Sheets("srule").Cells(j, 1) = 1 And Sheets("srule").Cells(j, 2) = 2 Then
        If Sheets("srule").Cells(i, 3) Like "*:1,*" Then
            i_word_order = "sv"
        Else
            i_word_order = "vs"
        End If
        If Sheets("srule").Cells(j, 3) Like "*:1,*" Then
            j_word_order = "sv"
        Else
            j_word_order = "vs"
        End If
        If i_word_order = j_word_order Then
            i_fst_lab = Split(Sheets("srule").Cells(i, 4), ":")
            j_fst_lab = Split(Sheets("srule").Cells(j, 4), ":")
            i_snd_lab = Split(Sheets("srule").Cells(i, 5), ":")
            j_snd_lab = Split(Sheets("srule").Cells(j, 5), ":")
            Sheets("srule").Rows(j).Delete

            maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 2) = j_fst_lab(0) Then
                    Sheets("wrule").Cells(x, 2) = i_fst_lab(0)
                ElseIf Sheets("wrule").Cells(x, 2) = j_snd_lab(0) Then
                    Sheets("wrule").Cells(x, 2) = i_snd_lab(0)
                End If
            Next
            flag = 1
        End If
    End If
    i_analyzed_j_analyzed = flag
End Function

Function check_dup_label(word As String, w_label As String, _
                                 obj_word As String) As Long
    Dim maxrow, i, flag As Long
    flag = 0
    maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
    For i = 1 To maxrow
        If Sheets("wrule").Cells(i, 1) = word And _
           Sheets("wrule").Cells(i, 2) = w_label And _
```

```
            obj_word = Sheets("wrule").Cells(i, 3) Then
             flag = 1
             Exit For
        End If
    Next
    check_dup_label = flag
End Function

Function i_analyzed_j_not_analyzed(i As Long, j As Long) As Long
    Dim flag, maxrow, x As Long
    Dim subj_lab, verb_lab, word_order, fst_str, snd_str As String
    Dim array_item As Variant
    flag = 0

    If Sheets("srule").Cells(i, 1) = 1 And Sheets("srule").Cells(i, 2) = 2 And _
       Sheets("srule").Cells(j, 1) <> 1 And _
       Sheets("srule").Cells(j, 2) <> 2 Then
        maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
        If Sheets("srule").Cells(i, 3) Like "*:1,*" Then
            word_order = "sv"
        Else
            word_order = "vs"
        End If
        If word_order = "sv" Then
            array_item = Split(Sheets("srule").Cells(i, 4), ":")
            subj_lab = array_item(0)
            array_item = Split(Sheets("srule").Cells(i, 5), ":")
            verb_lab = array_item(0)
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(j, 1) And _
                   Sheets("wrule").Cells(x, 2) = subj_lab Then
                    fst_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(j, 2) And _
                   Sheets("wrule").Cells(x, 2) = verb_lab Then
                    snd_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
        Else
            array_item = Split(Sheets("srule").Cells(i, 4), ":")
            verb_lab = array_item(0)
            array_item = Split(Sheets("srule").Cells(i, 5), ":")
            subj_lab = array_item(0)
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(j, 1) And _
                   Sheets("wrule").Cells(x, 2) = subj_lab Then
                    snd_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(j, 2) And _
                   Sheets("wrule").Cells(x, 2) = verb_lab Then
```

```
                    fst_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
        End If
        If fst_str & snd_str = Sheets("srule").Cells(j, 3) Then
            Sheets("srule").Rows(j).Delete
            flag = 1
        End If

    End If
    i_analyzed_j_not_analyzed = flag
End Function

Function i_not_analyzed_j_analyzed(i As Long, j As Long) As Long
    Dim flag, maxrow, x As Long
    Dim subj_lab, verb_lab, word_order, fst_str, snd_str As String
    Dim array_item As Variant
    flag = 0

    If Sheets("srule").Cells(i, 1) <> 1 And Sheets("srule").Cells(i, 2) <> 2 And _
       Sheets("srule").Cells(j, 1) = 1 And _
       Sheets("srule").Cells(j, 2) = 2 Then
        maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
        If Sheets("srule").Cells(j, 3) Like "*:1,*" Then
            word_order = "sv"
        Else
            word_order = "vs"
        End If
        If word_order = "sv" Then
            array_item = Split(Sheets("srule").Cells(j, 4), ":")
            subj_lab = array_item(0)
            array_item = Split(Sheets("srule").Cells(j, 5), ":")
            verb_lab = array_item(0)
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(i, 1) And _
                   Sheets("wrule").Cells(x, 2) = subj_lab Then
                    fst_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(i, 2) And _
                   Sheets("wrule").Cells(x, 2) = verb_lab Then
                    snd_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
        Else
            array_item = Split(Sheets("srule").Cells(j, 4), ":")
            verb_lab = array_item(0)
            array_item = Split(Sheets("srule").Cells(j, 5), ":")
            subj_lab = array_item(0)
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(i, 1) And _
                   Sheets("wrule").Cells(x, 2) = subj_lab Then
                    snd_str = Sheets("wrule").Cells(x, 3)
```

```
                    Exit For
                End If
            Next
            For x = 1 To maxrow
                If Sheets("wrule").Cells(x, 1) = Sheets("srule").Cells(i, 2) And _
                   Sheets("wrule").Cells(x, 2) = verb_lab Then
                    fst_str = Sheets("wrule").Cells(x, 3)
                    Exit For
                End If
            Next
        End If
        If fst_str & snd_str = Sheets("srule").Cells(i, 3) Then
            Sheets("srule").Rows(i).Delete
            flag = 1
        End If

    End If
    i_not_analyzed_j_analyzed = flag
End Function

Function i_not_analyzed_j_not_analyzed(i As Long, j As Long) As Long
    Dim flag, maxrow, shared_len, w_len As Long
    Dim shared_str, subj_lab, verb_lab, other_str As String
    flag = 0

    If Sheets("srule").Cells(i, 1) = Sheets("srule").Cells(j, 1) Then
        If Left(Sheets("srule").Cells(i, 3), 1) = _
           Left(Sheets("srule").Cells(j, 3), 1) Then
            shared_str = return_left(Sheets("srule").Cells(i, 3), _
                                     Sheets("srule").Cells(j, 3))
            If shared_str <> "" Then
                Call pick_up_new_lab
                maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
                subj_lab = Sheets("wrule").Cells(maxrow, 2)
                Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 1)
                Sheets("wrule").Cells(maxrow, 3) = shared_str

                Call pick_up_new_lab
                maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
                verb_lab = Sheets("wrule").Cells(maxrow, 2)
                shared_len = Len(shared_str)
                other_str = Mid(Sheets("srule").Cells(i, 3), shared_len + 1)

                Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 2)
                Sheets("wrule").Cells(maxrow, 3) = other_str

                Sheets("wrule").Cells(maxrow + 1, 1) = Sheets("srule").Cells(j, 2)
                Sheets("wrule").Cells(maxrow + 1, 2) = verb_lab
                other_str = Mid(Sheets("srule").Cells(j, 3), shared_len + 1)
                Sheets("wrule").Cells(maxrow + 1, 3) = other_str

                Sheets("srule").Cells(i, 1) = 1
                Sheets("srule").Cells(i, 2) = 2
                Sheets("srule").Cells(i, 3) = subj_lab & ":1," & verb_lab & ":2"
                Sheets("srule").Cells(i, 4) = subj_lab & ":1"
                Sheets("srule").Cells(i, 5) = verb_lab & ":2"
```

```
            Sheets("srule").Rows(j).Delete
            flag = 1
        End If
    ElseIf Right(Sheets("srule").Cells(i, 3), 1) = _
            Right(Sheets("srule").Cells(j, 3), 1) Then
        shared_str = return_right(Sheets("srule").Cells(i, 3), _
                                  Sheets("srule").Cells(j, 3))
        If shared_str <> "" Then
            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            subj_lab = Sheets("wrule").Cells(maxrow, 2)
            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 1)
            Sheets("wrule").Cells(maxrow, 3) = shared_str

            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            verb_lab = Sheets("wrule").Cells(maxrow, 2)
            w_len = Len(Sheets("srule").Cells(i, 3)) - Len(shared_str)
            other_str = Left(Sheets("srule").Cells(i, 3), w_len)

            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 2)
            Sheets("wrule").Cells(maxrow, 3) = other_str

            Sheets("wrule").Cells(maxrow + 1, 1) = Sheets("srule").Cells(j, 2)
            Sheets("wrule").Cells(maxrow + 1, 2) = verb_lab
            w_len = Len(Sheets("srule").Cells(j, 3)) - Len(shared_str)
            other_str = Left(Sheets("srule").Cells(j, 3), w_len)
            Sheets("wrule").Cells(maxrow + 1, 3) = other_str

            Sheets("srule").Cells(i, 1) = 1
            Sheets("srule").Cells(i, 2) = 2
            Sheets("srule").Cells(i, 3) = verb_lab & ":2," & subj_lab & ":1"
            Sheets("srule").Cells(i, 4) = verb_lab & ":2"
            Sheets("srule").Cells(i, 5) = subj_lab & ":1"

            Sheets("srule").Rows(j).Delete
            flag = 1
        End If
    End If
ElseIf Sheets("srule").Cells(i, 2) = Sheets("srule").Cells(j, 2) Then
    If Left(Sheets("srule").Cells(i, 3), 1) = _
       Left(Sheets("srule").Cells(j, 3), 1) Then
        shared_str = return_left(Sheets("srule").Cells(i, 3), _
                                 Sheets("srule").Cells(j, 3))
        If shared_str <> "" Then
            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            verb_lab = Sheets("wrule").Cells(maxrow, 2)
            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 2)
            Sheets("wrule").Cells(maxrow, 3) = shared_str

            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            subj_lab = Sheets("wrule").Cells(maxrow, 2)
            shared_len = Len(shared_str)
            other_str = Mid(Sheets("srule").Cells(i, 3), shared_len + 1)
```

```
            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 1)
            Sheets("wrule").Cells(maxrow, 3) = other_str

            Sheets("wrule").Cells(maxrow + 1, 1) = Sheets("srule").Cells(j, 1)
            Sheets("wrule").Cells(maxrow + 1, 2) = subj_lab

            other_str = Mid(Sheets("srule").Cells(j, 3), shared_len + 1)
            Sheets("wrule").Cells(maxrow + 1, 3) = other_str

            Sheets("srule").Cells(i, 1) = 1
            Sheets("srule").Cells(i, 2) = 2
            Sheets("srule").Cells(i, 3) = verb_lab & ":2," & subj_lab & ":1"
            Sheets("srule").Cells(i, 4) = verb_lab & ":2"
            Sheets("srule").Cells(i, 5) = subj_lab & ":1"

            Sheets("srule").Rows(j).Delete
            flag = 1
        End If
    ElseIf Right(Sheets("srule").Cells(i, 3), 1) = _
            Right(Sheets("srule").Cells(j, 3), 1) Then
        shared_str = return_right(Sheets("srule").Cells(i, 3), _
                                  Sheets("srule").Cells(j, 3))
       If shared_str <> "" Then
            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            verb_lab = Sheets("wrule").Cells(maxrow, 2)
            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 2)
            Sheets("wrule").Cells(maxrow, 3) = shared_str

            Call pick_up_new_lab
            maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
            subj_lab = Sheets("wrule").Cells(maxrow, 2)
            w_len = Len(Sheets("srule").Cells(i, 3)) - Len(shared_str)
            other_str = Left(Sheets("srule").Cells(i, 3), w_len)

            Sheets("wrule").Cells(maxrow, 1) = Sheets("srule").Cells(i, 1)
            Sheets("wrule").Cells(maxrow, 3) = other_str

            Sheets("wrule").Cells(maxrow + 1, 1) = Sheets("srule").Cells(j, 1)
            Sheets("wrule").Cells(maxrow + 1, 2) = subj_lab
            w_len = Len(Sheets("srule").Cells(j, 3)) - Len(shared_str)
            other_str = Left(Sheets("srule").Cells(j, 3), w_len)
            Sheets("wrule").Cells(maxrow + 1, 3) = other_str

            Sheets("srule").Cells(i, 1) = 1
            Sheets("srule").Cells(i, 2) = 2
            Sheets("srule").Cells(i, 3) = subj_lab & ":1," & verb_lab & ":2"
            Sheets("srule").Cells(i, 4) = subj_lab & ":1"
            Sheets("srule").Cells(i, 5) = verb_lab & ":2"

            Sheets("srule").Rows(j).Delete
            flag = 1
        End If
    End If
End If

i_not_analyzed_j_not_analyzed = flag
```

```
End Function

Sub pick_up_new_lab()
    Dim lab, new_lab As String, i, num, maxrow, flag As Long
    flag = 0
    num = Sheets("label").Cells(Rows.Count, 2).End(xlUp).Row
    maxrow = Sheets("wrule").Cells(Rows.Count, 2).End(xlUp).Row
    If Sheets("wrule").Cells(1, 1) = "" Then
        maxrow = 0
    End If
    For i = 1 To maxrow
        If Sheets("label").Cells(num, 1) = Sheets("wrule").Cells(i, 2) Then
            flag = 1
            Exit For
        End If
    Next
    If flag = 0 Then
        Sheets("wrule").Cells(maxrow + 1, 2) = Sheets("label").Cells(num, 1)
        Call move_label_pos
    Else
        Call move_label_pos
        Call pick_up_new_lab
    End If
End Sub

Function return_left(i_str As String, j_str As String) As String
    Dim dubplication As String, i, i_num, j_num, num As Integer
    duplication = ""
    i_num = Len(i_str)
    j_num = Len(j_str)
    If i_num < j_num Then
        num = i_num
    Else
        num = j_num
    End If

    For i = 1 To num
        If Left(i_str, i) = Left(j_str, i) Then
            duplication = Left(i_str, i)
        Else
            Exit For
        End If
    Next
    return_left = duplication
End Function

Function return_right(i_str As String, j_str As String) As String
    Dim dubplication As String, i, i_num, j_num, num As Integer
    duplication = ""
    i_num = Len(i_str)
    j_num = Len(j_str)
    If i_num < j_num Then
        num = i_num
    Else
        num = j_num
    End If
```

```
    For i = 1 To num
        If Right(i_str, i) = Right(j_str, i) Then
            duplication = Right(i_str, i)
        Else
            Exit For
        End If
    Next
    return_right = duplication
End Function


Private Sub produce_utterances()
    Dim maxrow, i As Long
    maxrow = Sheets("concepts").Cells(Rows.Count, 1).End(xlUp).Row

    For i = 1 To maxrow
        Call subj_verb_complete_match_pattern(i)
        If Sheets("utterances").Cells(i, 1) = "" Then
            Call only_subj_match_pattern(i)
            If Sheets("utterances").Cells(i, 1) = "" Then
                Call only_verb_match_pattern(i)
                If Sheets("utterances").Cells(i, 1) = "" Then
                    Call generate_with_no_variables(i)
                    If Sheets("utterances").Cells(i, 1) = "" Then
                        Call generate_randomly(i)
                    End If
                End If
            End If
        End If
        Call abstract_rules
    Next
End Sub

Sub generate_randomly(input_num As Long)
    Dim fst_str, snd_str As String, maxrow As Long
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row
    fst_str = generate_ran_str()
    snd_str = generate_ran_str()

    If Sheets("srule").Cells(1, 1) = "" Then
        maxrow = 0
    End If

    Sheets("utterances").Cells(input_num, 1) = _
    Sheets("concepts").Cells(input_num, 1)
    Sheets("utterances").Cells(input_num, 2) = _
    Sheets("concepts").Cells(input_num, 2)
    Sheets("utterances").Cells(input_num, 3) = fst_str & snd_str
    Sheets("srule").Cells(maxrow + 1, 1) = _
    Sheets("concepts").Cells(input_num, 1)
    Sheets("srule").Cells(maxrow + 1, 2) = _
    Sheets("concepts").Cells(input_num, 2)
    Sheets("srule").Cells(maxrow + 1, 3) = fst_str & snd_str

End Sub

Sub generate_with_no_variables(input_num As Long)
```

```
    Dim i, maxrow As Long
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row
    If Sheets("srule").Cells(1, 1) <> "" Then
        For i = 1 To maxrow
            If Sheets("srule").Cells(i, 1) = _
              Sheets("concepts").Cells(input_num, 1) And _
              Sheets("srule").Cells(i, 2) = _
              Sheets("concepts").Cells(input_num, 2) Then
                Sheets("utterances").Cells(input_num, 1) = _
                Sheets("concepts").Cells(input_num, 1)
                Sheets("utterances").Cells(input_num, 2) = _
                Sheets("concepts").Cells(input_num, 2)
                Sheets("utterances").Cells(input_num, 3) = _
                Sheets("srule").Cells(i, 3)
            End If
        Next
    End If
End Sub

Sub only_verb_match_pattern(input_num As Long)
    Dim fst_lab, snd_lab, fst_id, snd_id, fst_str, snd_str As String
    Dim maxrow, i As Long, my_array As Variant
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row

    If Sheets("srule").Cells(1, 1) <> "" Then
        For i = 1 To maxrow
            If Sheets("srule").Cells(i, 1) = 1 And _
               Sheets("srule").Cells(i, 2) = 2 Then
                my_array = Split(Sheets("srule").Cells(i, 4), ":")
                fst_lab = my_array(0)
                fst_id = my_array(1)
                my_array = Split(Sheets("srule").Cells(i, 5), ":")
                snd_lab = my_array(0)
                snd_id = my_array(1)

                If fst_id = "1" Then
                    fst_str = retrieve_str(Sheets("concepts"). _
                                           Cells(input_num, 1), (fst_lab))
                    snd_str = retrieve_str(Sheets("concepts"). _
                                           Cells(input_num, 2), (snd_lab))
                    If fst_str = "" And snd_str <> "" Then
                        fst_str = generate_ran_str()
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Call insert_wrule_with_label( _
                             Sheets("concepts").Cells(input_num, 1), _
                             (fst_str), (fst_lab))
                        Exit For
                    End If
                Else
                    fst_str = retrieve_str(Sheets("concepts"). _
                                           Cells(input_num, 2), (fst_lab))
                    snd_str = retrieve_str(Sheets("concepts"). _
                                           Cells(input_num, 1), (snd_lab))
                    If fst_str <> "" And snd_str = "" Then
                        snd_str = generate_ran_str()
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Call insert_wrule_with_label( _
```

```
                        Sheets("concepts").Cells(input_num, 1), _
                        (snd_str), (snd_lab))
                    Exit For
                End If
            End If
        End If
    Next
End If
End Sub

Sub only_subj_match_pattern(input_num As Long)
    Dim fst_lab, snd_lab, fst_id, snd_id, fst_str, snd_str As String
    Dim maxrow, i As Long, my_array As Variant
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row

    If Sheets("srule").Cells(1, 1) <> "" Then
        For i = 1 To maxrow
            If Sheets("srule").Cells(i, 1) = 1 And _
               Sheets("srule").Cells(i, 2) = 2 Then
                my_array = Split(Sheets("srule").Cells(i, 4), ":")
                fst_lab = my_array(0)
                fst_id = my_array(1)
                my_array = Split(Sheets("srule").Cells(i, 5), ":")
                snd_lab = my_array(0)
                snd_id = my_array(1)

                If fst_id = "1" Then
                    fst_str = retrieve_str( _
                              Sheets("concepts").Cells(input_num, 1), (fst_lab))
                    snd_str = retrieve_str( _
                              Sheets("concepts").Cells(input_num, 2), (snd_lab))
                    If fst_str <> "" And snd_str = "" Then
                        snd_str = generate_ran_str()
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Call insert_wrule_with_label( _
                             Sheets("concepts").Cells(input_num, 2), _
                             (snd_str), (snd_lab))
                        Exit For
                    End If
                Else
                    fst_str = retrieve_str( _
                              Sheets("concepts").Cells(input_num, 2), (fst_lab))
                    snd_str = retrieve_str(_
                              Sheets("concepts").Cells(input_num, 1), (snd_lab))
                    If fst_str = "" And snd_str <> "" Then
                        fst_str = generate_ran_str()
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Call insert_wrule_with_label( _
                             Sheets("concepts").Cells(input_num, 2), _
                             (fst_str), (fst_lab))
                        Exit For
                    End If
                End If
            End If
        Next
    End If
End Sub
```

```
Sub subj_verb_complete_match_pattern(input_num As Long)
    Dim fst_lab, snd_lab, fst_id, snd_id, fst_str, snd_str As String
    Dim maxrow, i As Long, my_array As Variant
    maxrow = Sheets("srule").Cells(Rows.Count, 1).End(xlUp).Row

    If Sheets("srule").Cells(1, 1) <> "" Then
        For i = 1 To maxrow
            If Sheets("srule").Cells(i, 1) = 1 And _
               Sheets("srule").Cells(i, 2) = 2 Then
                my_array = Split(Sheets("srule").Cells(i, 4), ":")
                fst_lab = my_array(0)
                fst_id = my_array(1)
                my_array = Split(Sheets("srule").Cells(i, 5), ":")
                snd_lab = my_array(0)
                snd_id = my_array(1)

                If fst_id = "1" Then
                    fst_str = retrieve_str( _
                      Sheets("concepts").Cells(input_num, 1), (fst_lab))
                    snd_str = retrieve_str( _
                      Sheets("concepts").Cells(input_num, 2), (snd_lab))
                    If fst_str <> "" And snd_str <> "" Then
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Exit For
                    End If
                Else
                    fst_str = retrieve_str( _
                      Sheets("concepts").Cells(input_num, 2), (fst_lab))
                    snd_str = retrieve_str( _
                      Sheets("concepts").Cells(input_num, 1), (snd_lab))
                    If fst_str <> "" And snd_str <> "" Then
                        Call insert_utterance(input_num, fst_str & snd_str)
                        Exit For
                    End If
                End If
            End If
        Next
    End If
End Sub

Function retrieve_str(word As String, label As String) As String
    Dim maxrow, i As Long, w_str As String
    maxrow = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
    w_str = ""

    For i = 1 To maxrow
        If Sheets("wrule").Cells(i, 1) = word And _
           Sheets("wrule").Cells(i, 2) = label Then
            w_str = Sheets("wrule").Cells(i, 3)
            Exit For
        End If
    Next
    retrieve_str = w_str
End Function

Sub insert_utterance(line_num As Long, str As String)
```

```
    Sheets("utterances").Cells(line_num, 1) = _
    Sheets("concepts").Cells(line_num, 1)
    Sheets("utterances").Cells(line_num, 2) = _
    Sheets("concepts").Cells(line_num, 2)
    Sheets("utterances").Cells(line_num, 3) = str
End Sub

Private Sub generate_concepts()
    Dim i, random_num, loop_num As Long, subj, verb As String

    loop_num = TextBox2.Value
    For i = 1 To loop_num
        random_num = Int(8 * Rnd + 1)
        subj = Sheets("iv").Cells(random_num, 1)
        random_num = Int(8 * Rnd + 1)
        verb = Sheets("iv").Cells(random_num, 2)
        Sheets("concepts").Cells(i, 1) = subj
        Sheets("concepts").Cells(i, 2) = verb
        loop_num = loop_num - 1
    Next
End Sub

Function generate_ran_str() As String
    Dim let_num, rand_num, i, j, alpha_num As Integer
    Dim alphabet(1 To 26) As Variant, str As String

    rand_num = TextBox3.Value

    alphabet(1) = "a"
    alphabet(2) = "b"
    alphabet(3) = "c"
    alphabet(4) = "d"
    alphabet(5) = "e"
    alphabet(6) = "f"
    alphabet(7) = "g"
    alphabet(8) = "h"
    alphabet(9) = "i"
    alphabet(10) = "j"
    alphabet(11) = "k"
    alphabet(12) = "l"
    alphabet(13) = "m"
    alphabet(14) = "n"
    alphabet(15) = "o"
    alphabet(16) = "p"
    alphabet(17) = "q"
    alphabet(18) = "r"
    alphabet(19) = "s"
    alphabet(20) = "t"
    alphabet(21) = "u"
    alphabet(22) = "v"
    alphabet(23) = "w"
    alphabet(24) = "x"
    alphabet(25) = "y"
    alphabet(26) = "z"

    str = ""
    let_num = Int(rand_num * Rnd + 1)
```

```
    For j = 1 To let_num
        alpha_num = Int(26 * Rnd +
        str = str & alphabet(alpha_num)
    Next
    generate_ran_str = str
End Function

Sub insert_wrule_with_label(word As String, w_str As String, _
                            label As String)
    Dim i, last_l, flag As Long

    last_l = Sheets("wrule").Cells(Rows.Count, 1).End(xlUp).Row
    Sheets("wrule").Cells(last_l + 1, 1) = word
    Sheets("wrule").Cells(last_l + 1, 2) = label
    Sheets("wrule").Cells(last_l + 1, 3) = w_str
End Sub

Sub move_label_pos()
    Dim last_l As Long
    last_l = Sheets("label").Cells(Rows.Count, 2).End(xlUp).Row
        If last_l < 78 Then
            Sheets("label").Cells(last_l, 2) = ""
            Sheets("label").Cells(last_l + 1, 2) = 1
        Else
            Sheets("label").Cells(last_l, 2) = ""
            Sheets("label").Cells(1, 2) = 1
        End If
End Sub

Sub delete_last_line(sheet_name As String, line_num As Long)
    Sheets(sheet_name).Rows(line_num).Delete
End Sub
```