修 士 論 文

マルチエージェント・モデルによる共通文法の獲得

指導教官　東条 敏 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

中村 誠

1997 年 1 月 22 日

目 次

第 1 章

はじめに

1.1 目的

自然言語の文法の形式化は N.Chomsky 以来盛んに行なわれてきた. 自然言語文を処理するためには効率の良い形式化が求められるが, それをすることにより, 自然言語のもつ本質的な特性は極力損なわれてはならない. それは, 自然言語の文法は形式言語のそれとは違い, 統計現象であるということである. なぜならば, 未知語や未知の文法に遭遇した際に柔軟に対応して処理しなければならず, それは自然言語処理をするうえで, このような現象は頻発するからである.

いかに柔軟な文法表現をするかという問題について, 本研究ではどのようにして統計的な自然言語文法が発生するかという点に着目した. そのためには現在話されている自然言語の文法が, いかにして出現したかというその歴史を考察しなければならない.

現在話されている諸言語は, 有史以来変化し続けて現在の文法に至っている. その変化というのは, 会話人口をはじめとする政治的, 軍事的, 経済的圧力等の理由により, 頻繁に使われる言語へと淘汰されてきた結果である. 異なる言語を喋る民族が接触する度に混合言語が発生し, それまでに話されていた言語はそれぞれの特性を併せて, あたかも遺伝子の交配のように進化, または変化したのである.

このような頻繁に起こる変化によって, 先に述べた形式言語では処理しきれない自然言語のもつ不自然さが現れたのである.

以上のことを踏まえて, 実際に起こった自然言語文法の淘汰する様子, すなわち互いに異なる文法を持つコミュニティが, コミュニケーションをとることを目的として, 互いの文

法を獲得する過程をシミュレートすることができるならば, 自己改編するような文法表現をすることが可能ではないかとここでは提案する.

本研究の位置づけとしては, 自らメッセージを発信できる, 能動的で, 自律的なエージェント間の協調, 競合または組織化だと考えられる. そして, 本研究の目的は, ある文法を持ったエージェント群の中に, 異なった文法を持つエージェント群が加わったときに, 多数のメッセージ交換を通じて, 試行錯誤しながら, 各エージェントが群の中での共通な文法を獲得していく過程を実現することである. 具体的には, 世界中に存在する pidgin, creole といった混合言語が発生する過程を, シミュレーションモデルとして実現することである.

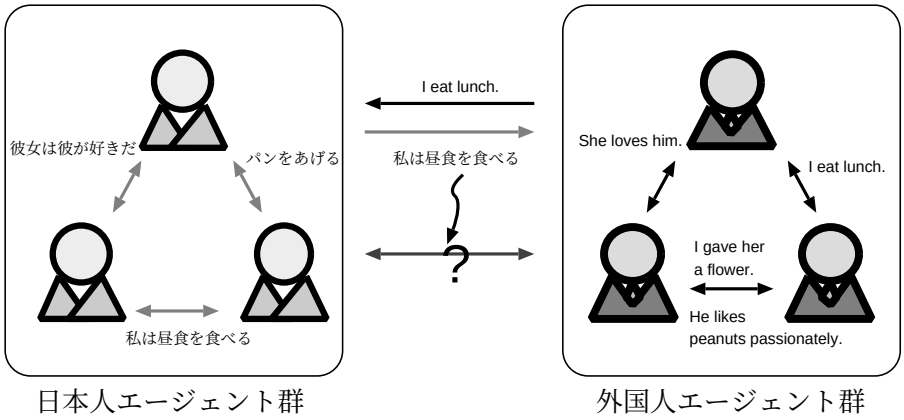

図 1.1: 本研究モデルのイメージ

1.2 本論文の構成

本稿の構成は以下の通りである. 本章ではこの後, 工学的, 言語学的な背景を述べる. 第２章ではこのモデルを扱うために必要な基礎知識, 具体的には TAG, 遺伝的アルゴリズム (GA), 遺伝的プログラミング (GP) について述べる. 第３章では第２章で述べた知識をもとに, 本研究モデルの具体的な説明を述べる. 第４章で計算機上に実装し, 実験した結果とその考察を述べる. 最後に第５章で本稿のまとめを行なう.

第 2 章

背景

本章では本研究における工学的背景と言語学的背景について述べる. 工学的背景では, 主にマルチエージェント・モデルでの文法獲得や言語獲得に関する過去の研究事例について述べる. 言語学的背景では, 現実に起こっている, 又は過去に起こった混合言語の発生と, 言語の特徴やその必要性について述べる. 本研究では工学的背景を基にして, 言語学的に発生している事象を検証すべく行なうことを認識されたい.

2.1 工学的背景

本節では, これまでに行なわれてきた自律エージェントによる文法獲得に関する研究について述べる.

2.1.1 マルチエージェント・モデルにおける文法獲得

これまでに自律的なエージェントが協調を目的として言語獲得や言語を進化させる研究としては, まず Werner と Dyer による研究があげられる [13]. これは人工生物の集団を自律的なエージェントと見立て, 雄が雌に近づくために, 雌が発するメッセージを雄が理解していくというものである. メッセージは４桁のビット列からなり, 学習機構はニューラルネットを用いている. この研究では, 一種の言語や通信規約を多数の自律的なエージェント群に自己組織化させようとした基本的なモデルを提供した.

また, 小野典彦らは [18], この Werner らの研究の延長として, この人工生物群に一部修正を施したモデルを提案している. ここでは, 学習機構として分類システムを用いて経験

的かつ遺伝的に学習させている.

Werner らや小野典彦らの研究はあくまで自己組織化を目的とし, その手段のひとつとしてメッセージプロトコルを形成させているため, このプロトコルを言語とよぶにはあまりにも単純なものであった. しかし, 橋本 [16] らの行なった研究は, 各々のエージェントが文法を持ち, その文法を学習することによって, 他のエージェントとコミュニケーションをすることを試みた. ここでは, エージェントは以下の３つの基準によって学習をしている.

1. speaking: いかに長く, あまり話されない単語を話すか.
2. recognizing: いかに長い単語を, いかに早く認識したか.
3. being recognized: どのようにエージェントが話した長い単語が認識されたか.

以上の３つが学習によって満たされることを, 文法の進化と彼らは呼んでいる.

ここではこれまでになされた, 学習によって言語獲得をする自律エージェントに関する研究について述べた. ここにあげた研究例は, エージェントが話す言語はどれも 0,1 で書かれたビット列ではあるが, コミュニケーションをするために言語を変化させていく能力をもつマルチエージェント・モデルとして興味深い.

2.1.2 小野モデル

本研究においては自然言語を話す自律エージェントが自分の意思を相手に伝達すべく学習をし, 結果として双方がそれまでに話していた言語の特徴を合わせ持った混合言語が発生するようなモデルを提案する.

したがって, 上述の研究事例と大きく異なるところは, エージェントは自然言語文で会話をするということと, 意味のある文を発話することにより, 文が認識されるということは, 単にパースできる文法を備えているというだけではなく, 伝達すべき意味が伝わったかどうかということである.

エージェントを人に見立てて, 自然言語を話すモデルを提案したのは小野哲雄 [14],[15] らである.

小野 [14],[15] らは, 子供が大人の会話を聞くことにより, 自らの文法を精緻するという問題を扱った. 文法が未熟な子供が大人との会話を聞くことにより, 素性を精緻していくというモデルである. ここでは言語は自然言語文で, 文脈自由文法で書かれた文法を, 遺伝的アルゴリズムと分類システムを用いて学習している.

ここでは, 子供が大人の文法を学ぶためには大人が子供の文法を真似て, 歩み寄ってやらなければ子供の文法は決して収束しないということが述べられている. つまり, 既に完全な文法を持っている大人のエージェントが, 学習機構を備えた子供のエージェントに対して協調することにより, 子供のエージェントの学習効果が高められるということである.

このことは言語学的にも同じようなことが述べられている. Pinker [17] は, 耳が不自由な親が子供を育てる場合, TV を見せ続けても子供は決して言語を習得しない. 子供の言語獲得に必要なのは, ゆっくりと喋り, 子供に歩み寄った文法を使う母親語だと述べている. また, Todd [1] は, 人間は文法を簡単にするレジスタをもっており, 自分を相手に歩み寄らせたり, 文法を学ばせるために, このレジスタを使うのだと述べている.

つまり, 人間の内部で行なわれている言語獲得に関する学習を, 自律エージェントによる協調という形でモデル化したものとして評価できる.

2.2 言語学的背景

ここでは, 世界中で現実に起こっている, 言語融合について述べる. また, 言語融合に似た学習を必要とする幼児期における言語獲得や, 第２言語獲得についても述べる.

2.2.1 Lingua franca

リンガ・フランカには, 以下の２つの異なった意味からなる.

1. 歴史的実体を持った言語の通称.
2. 母語を異にする言語集団が, 限られた状況で用いる, ある一定の広がりをもつ地域での通用語ないしは共通語, をさす言語学上の述語.

である. 後者の意味での リンガ・フランカ は, 混成的言語特徴の不安定な集まりで, それは言語体系を構成し得ない, とされる.

前者における リンガ・フランカ とは, 地中海域の港において, 中世から１９世紀頃まで用いられた交易上の通用語で, イタリア語をもとにしたロマンス諸語の要素と, 多少のアラビア語的要素が混じりあった地域的共通語を指す.

主な話者は, 地中海沿岸のイタリア人, スペイン人, フランス人などのロマンス語系, 地中海北岸のアラブ系の貿易商人であったとされるが, まとまった記述がないために, 言語

構造も言語人口も不明である. この言語が使われたのは, 港を中心とした限られた地域で, 外交や商業取り引きの交渉に際し, 相互理解の必要から生まれた, その場限りの不完全な言語であったようである. この言語の話者は, それとは別に, 各々の母語を持ち, 地域社会生活は, 各地方の言語で行なっていた. 現在の世界の大部分で英語が国際語として用いられることを考えると, 上の状況を理解しやすいであろう.

１３世紀から１９世紀にかけて話されていたこの言語は, 音声的には中部イタリア語に近く, 語彙的にはイタリア語, スペイン語に加えて, プロヴァンス語, カタロニア語, ギリシア語, トルコ語の単語が混じる. 形態・統語上は単純化され, 曲用, 活用は消えて不変化語を並置する. また, 単数, 複数の形態上の区別がない. 人称代名詞は, 格変化を持たず, 例えば, mi は,「私は, 私を, 私に」を表す. 動詞の時制形には, たとえば, mi andar 「私は行く」/ mi andato 「私は行った」のように, 現在形 (未来形も意味する) と, 過去形しかない.

以下の リンガ・フランカ で話された文が記録に残っている.

> Mira no trovar mi altra volta, sino afee de Dio, mi parlar patron donar bona bastonada, mucho, mucho.
>
> 「もう二度と俺の前に姿を現わすな. さもなくば, 本当に, めちゃくちゃに棒で思いきりたたいてもらうように旦那に言いつけてやるぞ」 (P.Perego,1968 による)

2.2.2 pidgin, creole

ピジンとクレオールの特徴を簡単に述べると以下のようになる.

pidgin 共通の言語をもっていないが, 通称その他の目的で互いに話をしたいと思っている人びとの間に発達した伝達のシステムである. pidgin は, そのもとになった言語に比べて, 語彙は限られており, 文法構造は単純化され, 機能する範囲ははるかに狭い.pidgin を母語とする者は誰もいない.

creole creole は, ある共同体の母語となった pidgin のことをいう. この定義は,pidgin と creole が言語発達という単一の過程における２つの段階であるという点を強調したものである.

以下に, ピジンとクレオールについての詳細を述べる.

2.2.3 pidgin

共通言語を持たない２つ以上の集団が, 一定期間以上, しばしば特定の目的を持って交渉を持つ際, 何らかの接触言語が形成される. ピジンは接触言語のひとつのタイプであり, いずれの集団の母語でもなく, 文法的・語彙的な簡略化が著しく, 使用する場面, 用途が限定されている.

しかし, ピジンについては, 子供向けの漫画や映画で長年にわたって行なわれてきたような, 固定化した見方をしないようにすることが一番重要である.「ワタシ, ターザン. オマエ, ジェーン」といったイメージは現実から程遠い. ピジンは壊れた言語ではないし, 幼児のような言葉, 怠慢, 堕落, 原始的な思考過程, 精神的欠陥などが原因で出来たものではない. これとは逆に, ピジンが自然言語を想像力によって改造したものであり, 独特の構造と規則を備えたものである.

ピジンは, 本来それを必要とする状況がなくなれば, 自然消滅することが多い. わずか数年で消滅することもあり, １００年以上使用されることは滅多にない. 例えば, ヴェトナムで使用されていたフランス語のピジンは, フランス人が引き上げるとほとんど消滅し, 同様に, ヴェトナム戦争の時に現れたピジン英語も, 戦争が終わると消滅したのである.

しかし, 場合によっては, 異なる社会状況のもとで存続し, 語彙的・文法的に拡張され, 安定的に使用される地域共通語 (stable pidgin, expanded pidgin, lingua franca) となったり, また, それを母語として話す人々の登場によりクレオールとなったりする. ピジンの中には, 異言語間の伝達の手段として非常に有用になり, 日常的な補助言語として, それまでよりも公に認められた役割を果たすようになったものもある. さらに, 共通語としての公的な地位を社会によって与えられることもある. これは,「ピジンの拡張」といわれ, それは, 使用者の要求に答えるために新しいかたちが付け加えられ, 以前よりもはるかに広い範囲の状況で用いられるようになるからである.

集団接触にともない, 特にピジンが出現し安定化する要因としては, 次の３つが考えられる.

1. 共通コード体系の欠如
2. 使用領域の限定
3. 短期間での形成

接触する複数の集団の間に, 共通の言語が事前には全く存在していない場合, 双方は全く手探りで, しかも限られた時間の内に, 新しいコード体系を構築していかなければならない. とはいえ, その素材は, それぞれが知っている言語に求めるほかない. どの言語がより多くの語彙素材を提供するか, どの言語の音声組織が土台になるか, 語順はどうするか等, これらを左右するパラメータとなるのは, 接触当事者の人口構成比, 集団間の政治的, 経済的, 軍事的, あるいは宗教的な力関係, そして, その地域のもともとの言語的多様性の度合などである.

ピジンは, 特定の限られた状況 (例えば, 労働者同士, ないし, 労働者と監督者の間の支持応答, 交易場面での意思表示や交渉等) でのみ使用される. しかも, ピジン以外に有効な意思伝達手段がない場合にのみ成立する. 言い替えれば, ピジン使用者達も, ピジンを必要とする状況以外の場面では, 自分たちの言語ないしは別の共通語を使用し続ける.

pidgin の具体的な用法

成立当初の「初期ピジン (incipient pidgin)」は, 文法の極端な単純化をともなうが, そこでは, 同時に新しい規範の形成が進行している. ピジンは, 最小限度の労力によって最大限の効果を生むための方法であり, 音声体系や文法体系は簡素化, 合理化され, 語彙形式の透明度も高い. 語彙を極端に少なくするため, 例えば, bad といわずに, no good という. また,「痩せている」を表現するのには, thin ではなく, bone got no meat という迂言形を用いることが多い.

pidgin における文法

性, 数, 格, 人称, 時制, 法, 態といった文法カテゴリは, ピジンやクレオールからはほとんど欠けている. しかし, それと同時にピジン独自の語形変化も生まれている. たとえば, Tok Pidgin には代名詞, 形容詞, 動詞に対して, 語尾変化をもつ. 具体的には以下にしめす通りである.

- 代名詞に *-pela* をつけると複数形になる.
  mi(“I,me”):*mipela*(“we,us”)
  yu(“you”,singular):*yupela*(“you”,plural)
- *-pela* は, 形容詞のマーカでもある.
  naispela “pretty”

dispela “this”
sampela “some”
wanpela “one”

- 直接目的語をもつ動詞には, *-im* がつく. この接尾時を抜いた動詞は自動詞もしくは受動になる.
  rausim “eject,remove”: *raus* “be out, come out”

また, 単語の数が少ないので, 別の文法的な情報を意味や音韻が同じ単語に結びつけている. 例えば,*askim* は名詞としても動詞としても使われる.

“mi gat wanpela askim.”
“I have a question.”

“mi laik askim em”
“I want to ask him/her.”

基本的な句や節のような組合せは, 英語とほぼ同じであるが, 細かな所で異なる. Tok Pidgin において, 所有句, 修飾名詞, 動詞, 副詞, 節は名詞の後にくる. また, ３人称が主語になると, 述部には述語マーカ *i-*が先に付く.

haus bilong mi“house of me,my house” *haus pepa*“house [for] paper,office” *haus kuk*“house [for] cooking,kitchen” *man nogud*“evil man”「悪い男」(*nogud* は「望ましくない」“undesirably” の副詞) *man mi lukim em*“the man I saw” *ol i-singaut*“they call” *balus i-no kamap yet*“the plane hasn’t arrived yet”

語彙は制限されているので, 各語には英語に比べて, 非常に広い意味を持つ.

- *sori* は “sorry” ではなく,“emotionally moved”(感情の動き) であり, それを拡張して,“sympathetic, grateful, glad” となる.
- *dai* は “cease”(終わる).“die” は *dai tru*“stop for good”
- *stap* は “be located;remain;continue”

pidgin は単語単体よりもむしろ, 句を用いる.

skru“screw;joint” から,*skru bilong arm*“elbow,” *skru bilong leg*“knee,”etc.(*bilong*=of)

英語以外からの語彙の使用は少ない.Tok Pidgin は, およそ 2000 単語から成り立っているが, そのうち, 英語以外から用いている単語は 10%以下である. そのうちの,50%が現地語であるメラネシア語, 25%がドイツ語, 残りの 25%がマレー語やロマンス諸語である.

2.2.4 creole

クレオールは, 典型的には, ピジンが母語化したものであり, 語彙および文法構造の完成度は高く, 音声面でも通常の人間言語と変わらない複雑度と体系性を備えている.

ここでいう母語化とは, 母語として使用する集団の発生, すなわち, 新しい言語共同体の成立にほかならない. この場合, 新集団, すなわち子供たちにとって, 母語とは, 母親の言語でも父親の言語でもないという点が重要である. また, 多くの場合, 子供たちが十分発達させたクレオールは, 両親たちピジン使用者にとって, 聞いても理解できない別個の言語となる.

厳密にいえば, ピジン使用者を親とする子供たちが, ピジンを第 1 言語として習得し, 子供社会での様々な場面での使用を通じて独自に発達させたもので, しかも, 周囲の大人たちが使用する他の様々な言語の影響をあまり受けていないものが, 理論的な意味で, 純粋なクレオールである. たとえば, ハワイ・クレオール語などがそうである.

また, 子供に限らず, 成人のピジン使用者たちが, ピジンの使用範囲を広げ, 本来は別の言語を使用していた家族間, 同一民族間での会話等にまで, ピジンを汎用するようになり, それを通じて, ピジン言語を語彙的, 文法的に発達させ, 機能的にほぼ完成された言語を生み出している場合も少なくない. パプアニューギニアで話されている Tok pidgin がそれである.

ピジンが特定の目的や場面でのコミュニケーションを成功させれば一応その機能を果たしたことになるのに対し, クレオールは, 人間生活のあらゆる領域での意思疎通, 表現の要請に対応したものとして発達しなければならない.

2.2.5 pidgin, creole の違い

以上において述べたことは, ピジン・クレオールの特徴であると同時に違いであるのだが, 厳密にいうと, ピジンやクレオールを厳密にわける明確な定義はない. しかし言語学者はそのようなグループの存在を認識している. その特徴は, 地域的・起源的なものではなく社会・歴史的発展等の目的による環境の共有にある. 現在のところ, ある言語がピジンであるとかクレオールであるというのは, 3つの基準, すなわち言語学的, 社会的, 歴史的なものを勘案して決定するしかない.

ピジンについての多くの定義があげる, ピジン化過程の構成要素として, 以下のようなものがあげられる.

- 簡略化 (simplification)
- 縮小 (reduction)
- 再構成 (restructuring)
- 混合 (mixture)

これらはクレオールの特徴とされることもある.

緻密にする (elaboration), 複雑性 (complication), 拡張 (expansion) といった概念で説明しようという試みはあるものの, ピジンと同様クレオールにも合意の得られるような定義はない. 一般的な見方からすれば, ピジン化とクレオール化は, お互いにミラーイメージであるプロセスである. これは, 二つのステージによる発展を意味している. 最初に, 基になる言語を簡略化して急速に言語の多様性を再構築する. 次のステップでは, その多様性を機能の拡張にともなって緻密化し, 母国語化する. 社会学的な観点からは, クレオールは, それを話すほとんどの人に取って母国語であるのに対し, ピジンは誰にとっても母国語ではない. ピジンは母国語ではないので, 最低限の文法的な道具で何とかすることになるが, クレオールの場合は普通の言語としてコミュニケートするだけの最低限の言語的資源がなければならない.

結論を述べると, ピジンとクレオールがあまりにも似た性質を持っているので, はっきりした境界を定めるのは問題であることがわかる. クレオールは単に前身となるピジンの性質を引き継ぐのであって, 新たに作り出すのではない. ピジンとクレオールは, 言語学的には相対的な概念である. ピジンやクレオールといわれるものの実体は, それぞれピジン

化, クレオール化の過程を経たものの実体ではあるのだが, それら固有の, あるいは完全な生成物ではなく, あくまで生成過程の最中にあるものなのである.

2.2.6 世界中の混合言語

上で述べたように, 世界中で混合言語が作り出されている. それを全てここで列挙することは不可能であるが, 代表的なピジン・クレオールを以下にしめす ([20] より抜粋).

ハワイ・ピジン/クレオール 英語基盤で, 中国語, 日本語, ハワイ語, ポルトガル語, フィリピン諸語の影響を受けている. 話者約５０万で, 多くは第１言語として用いている.

チヌーク・ジャーゴン カナダ, アメリカ北部で話されていた言語. チヌーク語基盤で, 英語, フランス語, ヌートゥカ語の方言の影響がある. １９世紀には１０万人の話者がいたが, 現在ではほとんど消滅している.

ハイチ・フランス語クレオール ハイチに主として３つの変種があり, ４００万以上の話者によって使用されている.

クリオ語 アフリカ, シエラレオネのフリータウンを中心とする地域の, 英語基盤のクレオール. 約５万人の第１言語であり, また第２言語としても広く使用されている. このクレオールの古い変種が, リベリアでも話されている.

カメルーン・ピジン英語 約２００万人の話者が第２言語としてカメルーンで使用している, 英語基盤のピジン. 都市部ではクレオール化している場合もある. これと関係のある変種が, ナイジェリア西部およびフェルナドボー島で使用されている.

トク・ピジン語 (ネオメラネシア語) 英語基盤のピジンで, 土着のパプア諸語の影響を受けている. 約１００万人によってパプアニューギニアで広く使用されている. クレオール化している地域もある.

日本ピジン 英語基盤のピジンで, １９世紀の終わり頃に日本の港で広く使用され, １９４０年代にアメリカに占領された地域でも話されていた. 現在ではもう使用されていない.

横浜ピジン 日本語, 英語, フランス語, マライ語他から成るピジン. 横浜祖開周辺で発生した. 現在ではもう使用されていない.

2.2.7　幼児期における言語獲得

2.1.2 節で述べたように, 幼児期における第１言語獲得に必要なプロセス, つまり, 子供の文法を精緻するためには親も子供に歩み寄った寛大な文法を使わなければならないことは, 共通言語獲得に関しても同じことがいえる. 大人にとって, 既知の言語から新たな言語を理解するということは, 子供にとっては, 心的言語から母語を学び取るということと同じことなのである.

ここで述べるのは, 人間には生得的に言語を身につける普遍的な能力が備わっており, それによって幼児が第１言語を獲得することと同じような現象が, ピジンを生成する過程にも現れるのではないかということである. もし, 全ての人間が, 言語を習得し, 操作するという生物的な素質を持って生まれてくるならば, 表面上は異なって現れていても, あらゆる言語に共通する言語普遍性が存在すると思われる. さらに, いろいろ異なる接触状況を調べてみると, 話し手が, 時には個のような言語普遍性, すなわち, いろいろの言語状況に対処しようとするこのような傾向を, 利用または再活性化できるのではないかと思われる.

ブラウンとベルージは, 子供の言語獲得の様子を調査し, 母親が子供に話す時, ほとんど必ず, しかも無意識に, 自分の言葉を単純化していることに気づいた. とても幼い子供たちに話すとき, 言葉を自動的に簡単にする親が多いのである. 従属文を少なくし, 単純時制を使い, 新しい単語を繰り返すのである. しかも大人たちは, 自分が何をしているのか, どのようにしているのか, そのやり方についても, あまり自覚しないで話している.

ファーガソンは, さらに一歩進んで,「多くの, おそらくどの言語社会においても, 何らかの理由のために, その社会の正常な話し方をすぐに理解できないと考えられている人たち (例えば, 赤ん坊, 外国人, 耳の聞こえない人たち) に対して使われる, 特殊なレジスタがいくつか存在する」と述べている. ファーガソンは, 「正規の」言語に屈折があるの,「単純な」レジスタでは, それが捨て去られがちだということを発見した.

さらに, 誰でも, 時には「単純レジスタ」を利用できるのではないかという考えを支持して, ローマン・ヤーコブソンは, 次のように指摘する.

> 自分の言語を完全に掌握している子供が, 突然, 赤ん坊に戻ったように振る舞って楽しんでみたり, 弟や妹の真似をしたり, または, ある程度自分自身のことを思い出したりして, もう一度赤ん坊のように話そうとする様子は, 繰り返し立証されてきた. 程度は異なるが, この幼児本能は, 特に精神分析学者が強調するように, 大人になっても現れることがある. ガーベレンツは, 恋人同志が愛を語

るとき, 幼児語で話す場合が非常に多いと指摘した.

大人も, ある条件下におかれると, 幼児期における言語に逆戻りするということは, 興味深い. 大人であっても, 必要に迫られると, 自分の正常の言語を単純化して使うことが可能なのである.

第 3 章

実験に必要な知識

本章では, 実験システムを作成するにあたって, 必要な文法やアルゴリズムについて述べる. 具体的には, Fillmore の深層格に基づいた概念の表現方法,TAG (Tree Adjoining Grammar), 遺伝的アルゴリズム (GA, Genetic Algorithm), 遺伝的プログラミング (GP, Genetic Programming) についてである. また, 並列プロセッサ AP1000 上に本実験システムを実装するために用いた, 並列オブジェクト志向言語,ABCL/f についても述べる.

3.1 概念の表現

本節では, 文を生成する前に発生する概念について述べる. ここでは, 人間が相手に自分の要求を伝えることを前提に, 文を生成する際, 文のもとになるものを概念とよぶことにする. この概念は, たとえ文を生成するための文法が異なっていようとも, 全ての人間に対して共通であると仮定する.

以下に, 概念を表現するにあたって, そのもととなった Fillmore の提唱する格文法と, それを用いた概念の表現方法を提案する.

3.1.1 Fillmore の深層格

文法概念としての格 (case) とは, 文の各構成要素が文の述語に対してどういう役割であるかを表したものである. このような格システムは, おもに文法との関わりにおいて述べられるものであり, 表層格と呼ばれる.

それに対して, 深層格というものは, 文の意味を担う動作主, 行動の起点等について求めるものであり, 文の意味解釈として深層格を文中から得るために, 格文法 (case grammar) というものを Fillmore 等が提唱した.

具体的には, 単文には動詞を中心要素として, 1個以上の名詞句から構成される命題が埋め込まれているというものである. 例えば,

John opened the door with the key.

という文においては, 動詞 open を中心に, John は行為者格 (Agent),the door は対象格 (Object), the key は道具格 (Instrument) であると考える.

格の定義には様々あるが, Fillmore の提唱した深層格システムを表3.1.1に挙げる ([4] より抜粋).

表 3.1: Fillmore による深層格システム

行為者格 (Agent)	出来事の行為者・動作主
対行為者格 (Counter-Agent)	行為が及ぶ相手
対象格 (Object)	行為によって何らかの変化をこうむる実体
結果格 (Result)	行為の結果として存在するに至った事物
道具格 (Instrument)	出来事を起こした刺激・物理的原因
始点格 (Source)	何かが移動を開始した場所
終点格 (Goal)	何かが移動の結果至った場所
経験者格 (Experiencer)	何らかの出来事の経験者

中心となる動詞は, 自分のまわりにどういう格を集めるかを特定しておく必要がある. この格を集める順序集合を格フレーム (case frame) と呼ぶ. 格フレームは, その動詞にとって不可欠 (obligatory) なものか, あってもなくても良いものか (optional) を指定される. 従って, 格文法の考え方は, 動詞の表層構造支配を, 動詞の深層意味格支配へと発展させたものであると考えられる.

3.1.2 深層格をもとにした概念の表現

上述の深層格を用いて, 概念を表現する方法を提案する. ここでは, 文を生成する際, 文のもとになるものを概念とよぶことにする.

例えば「行く」という動詞に対しての概念は, 以下のようになる.

[*go: (Obligatory: Agent) (Optional: Source, Goal, Instrument, etc.)]

ここで Obligatory は必ずなくてはならない必須格 (Obligatory Case), Optional はあってもなくても良い任意格 (Optional Case) を意味する. また, *go のように, "*" がついたものは, 動作や物事を指し示すもので, 言語を越えて共通である. これを意味と呼ぶ.

概念はどのような文法を持つものにも共通であり, 「あげる」という動詞に対し,「あげる人」は自分自身,「もらう人」は対発話者で,「あげるもの」が花であるということを意味する概念は,

[*give: (Agent: *I)(Counter-Agent: *you)(Object: *flower)]

となり, この概念をもとに日本人ならば「私はあなたに花をあげる」と文生成をし, 英語を話す人ならば, "I give you a flower." という文を作りあげる.

重要なのは, 文生成をするために必要な概念は, 任意の言語を話す人間に関して共通であり, それを表層の文に変換する文法や変換すべき単語が, 言語によって異なるということなのである. ここで共通である概念とは, 各動詞に対して指定される必須格, 任意格に関しても同じことがいえる.

日本語文で,「君にあげる」という文は状況によって成立するのだが, この文を生成するにあたって「あげる」という動詞の概念は,

[*give: (Obligatory: Counter-Agent)]

ではないことに注意しなければならない. 日本語の場合, 表層では格を省略することが可能であるが, 表層で表現しなくてもそれが暗に深層に伝えることができるということであり, 日本語に関しても, また他の言語に関しても「あげる」を意味する概念は,

[*give: (Obligatory: Agent, Counter-Agent, Object)(Optional: etc.)]

と共通である.

Fillmore は, 表層構造に現れる主語・目的語といった文法関係が, 実際の意味解釈に役に立たないと指摘し, 深層格のシステムを導入するに至ったのだが, 本研究においては, 非常に簡単な文に対しての (深層格をもとにした表層の) 文法獲得を提案するため, 敢えてこれを否定する立場にとる. つまり, 表層から得られる格は, 動詞に依存するが, 深層格につながるということである.

例えば, 動詞 give を用いた文には 主語と目的語が２つ使われるが, 主語はそのまま深層格における動作主 (Agent) となり, ２つの目的語には, 順に対行為者格 (Counter-Agent), 対象格 (Object) が割り当てられる. この動詞 give には, 深層の必須格 (obligatory case) として, 上の３つが指定されているのだが, それと同時に, 表層にも主格と目的格を２つ必要とするので, あくまで簡単な文には適応できる.

3.2 TAG

文脈自由文法の拡張として, 記号列を書き換える文法規則ではなく, 木構造を書き換える文法規則を持った文法を考えることができる. このような文法形式を **TAG(Tree-Adjoining Grammar)** とよぶ. TAG では,initial tree と auxiliary tree という形で与えられる木構造規則を, 代入と接合という操作で組み合わせていくことによって文の解析, 生成を行なう. 以下に initial tree と auxiliary tree の説明をする.

initial tree ある文法の根ノードと同一のラベルをもつ導出木の葉ノードと置き換えられる木構造規則. この置き換え操作を代入 (substitution) とよぶ.

auxiliary tree ある根ノードと同一ラベルをもつ導出木中のノードにその構造全体が挿入される木構造規則. この挿入操作を接合 (adjunction, adjoining) とよぶ. auxiliary tree には根ノードと同一のラベルをもつ葉ノードが必ずある. この葉ノードをフットノードと呼び,“*” をつけて区別する.

図 3.1 に TAG で記述された文法の例を示す. 図中の α で示されるルールは initial tree, β で示されるものは auxiliary tree である.

TAG は弱文脈依存文法 (mildly context sensitive grammar) というクラスに属し, 文脈自由文法と文脈依存文法の間の生成能力を持つ.

TAG は強力な文法生成能力を特徴とする, 木を生成する形式からなる. それは文を連想する構造の記述を特徴とするものである.

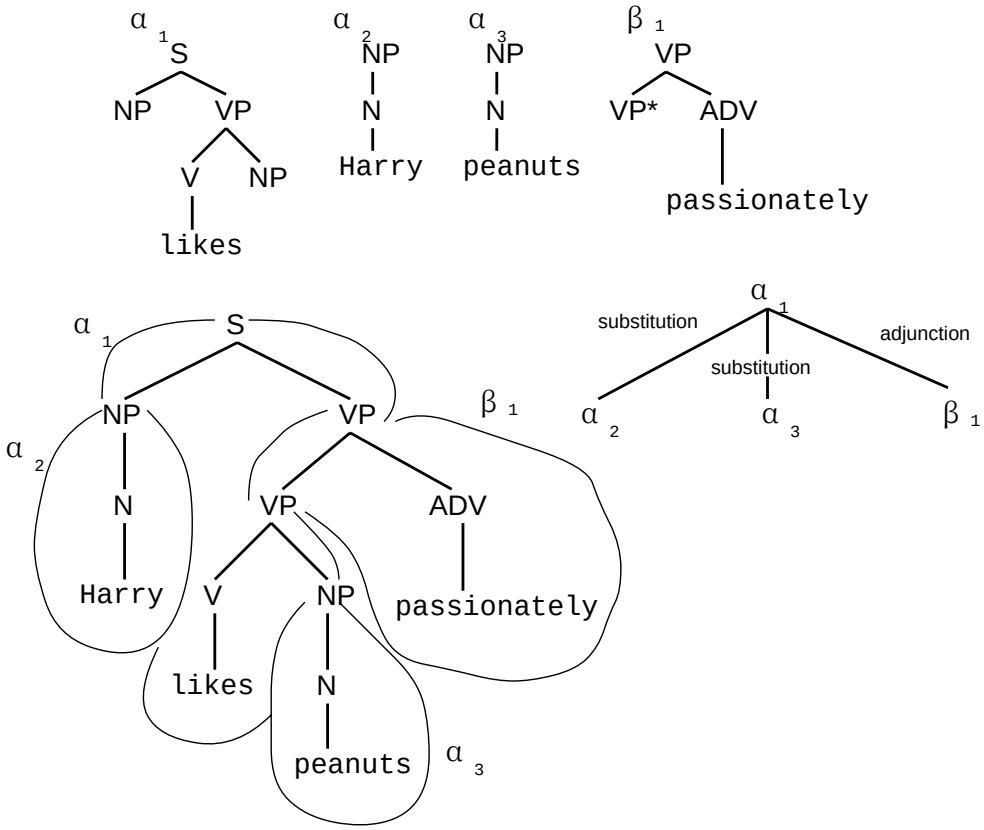

図 3.1: LTAG ルールと代入, 接合操作の例

TAG が木を構成するために持つ操作は接合だけである. 接合は代入を模擬することができるのだが, 明示的に代入のオペレーションを加えることにより,lexicalized TAG(LTAG) を得る.LTAG は, 形式的に TAG と同等である. これゆえ, 今後 LTAG のみで記述する. 更に,LTAG は直接言語学上の関連を持つ. 最近の linguistic and computational の研究では,LTAG のフレームワークで研究がなされている.

3.2.1 LTAG

TAG において, 各木構造が必ずアンカー (anchor) とよばれる一つの単語を持つようにしたものを LTAG とよぶ.LTAG を導入することで, 文法の語彙化をはかることが可能となる. 文脈自由文法の形式のままでは文法の語彙化をおこなうことはできない.TAG は語彙化を可能にするために文脈自由文法を最低限拡張した文法形式であるとみなすことも

できる. 図 3.1 は,LTAG の形式で描かれている.

3.2.2 深層格フィーチャ付 TAG の提案

深層格フィーチャ付 TAG の提案

ここでは LTAG に深層格フィーチャを持たすことを提案する. LTAG では, 動詞の引数を指定することが可能である. そもそもなぜ動詞が引数を持つかというと, 指定した引数に意味がある, すなわち格を指定しているということであり, 文生成に関してこのようなフィーチャを提案することは自然な拡張といえる.

深層格を素性として持たせた LTAG の例を図 3.2 に示す.

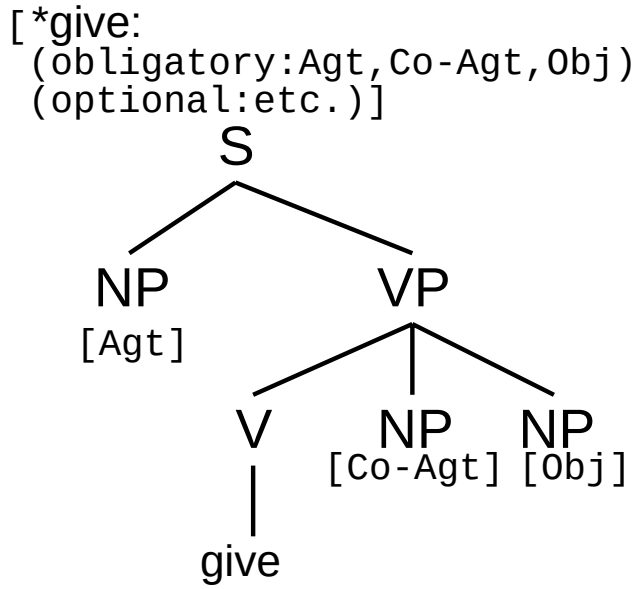

図 3.2: 深層格フィーチャ付 TAG の例

代入, 接合操作に対するユニフィケーション

3.1.2 項で述べたとおり,「行く」という動詞に対しての概念は以下のようになる.

[**go*: (Obligatory: Agent) (Optional: Source, Goal, Instrument, etc.)]

ここで指定された格について, 表層の位置情報を TAG によって指定されたと考えれば良い.

また, 動詞以外のルールに関しても, その根ノードに, 品詞に加えて, 素性としてどの格と代入できるかということを指定する. そして, 代入または接合の操作をする際, 格情報をユニフィケーションするのである.

ここで問題が生じる. それは, 各言語が持つ文法ルールに対して格を割り当てるために, 3.1.2 項で述べたような任意の言語に対する格の共通性がなくなってしまうことである. 例えば, 英語文法における「自分自身」を示す単語は, “I”, “me” 等があげられるが, 各単語にはそれぞれ行為者格と対行為者格であるという情報が既に含まれているのに対し, 日本語文法における「自分自身」を示す単語は「私」だけであり, ここには格に関する情報は含まれていない. 日本語では行為者格であることを示すために,「は」という格情報のみを持つ助詞が必要なのである. ここであげられるルールを図 3.3 に示す.

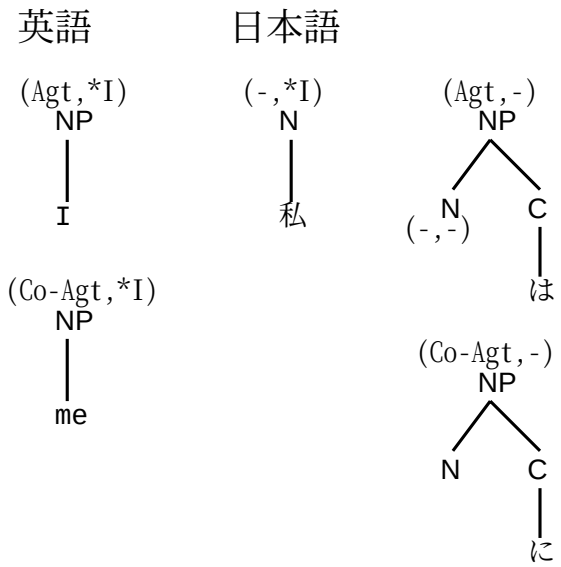

図 3.3: 「私」を表す英語と日本語の TAG

しかし, これはあまり問題にならない. なぜなら, 動詞に関する格情報は変わることがないために, 共通な概念というものは変わらないからである. 表層である TAG に深層格を素性として持たせた時点で, その文法に特化した深層の表現が必要である.

ユニフィケーションによる文生成とパース

上述のユニフィケーションを用いることにより, 文生成やパースが可能である. 以下にその手順を示す.

文生成に関しては例えば

[*give: (Agent: *I)(Counter-Agent: *you)(Object: *flower)]

をもとにして図 3.2 のルールで文を生成する際,

1. *give を意味として持つルールを探索する.
2. 葉ノードの品詞と, 概念で指定された格と意味をもとに (NP, Agent, *I) をユニフィケーションできるルールを探索する. 他も同じ.
3. 代入または接合操作が必要でなくなるまで (品詞, 格, 意味) をもとに適応する文法を探索し, ユニフィケーションを繰り返す.

といった処理をする. 文をパースする場合は, ユニフィケーションすることによって得られる, 格と意味を記憶しておき, 得られた動詞にこれらの情報を付け加えてやれば良い.

3.3 遺伝的アルゴリズムについて

本節では遺伝的アルゴリズムについて述べる. 本研究ではこの遺伝的アルゴリズムは用いないのだが, 次節で述べる遺伝的プログラミングのもととなっているので, 簡単に述べることにする.

3.3.1 遺伝的アルゴリズムとは

遺伝的アルゴリズム (GA) は, 生物進化の原理に基づいたアルゴリズムである. 求めるべき解の候補を遺伝子に見立て, 生物における選択 (selection), 交叉 (crossover), 突然変異 (mutation) といった遺伝的操作をすることにより, 遺伝子を解に近づけていくといった方法である.

GA での遺伝子は, 遺伝子型 (genotype) と表現型 (phenotype) からなる. 遺伝子型は遺伝子の組合せのパターンを指し,GA のオペレータはこの遺伝子型に適用される. 表現型は

遺伝子型に基づいて形成されるここの性質を指し, 環境により表現型から適応度が決まり, 選択はこの適応度に依存する.

3.3.2 遺伝的アルゴリズムでの処理の流れ

GA での基本的な処理は以下の通りである.

1. **初期集団の生成** ここで生成される遺伝子の個数は予め決められており, 遺伝子はランダムに生成される.

2. **終了条件のチェック** 処理手順 3 から 6 のループの終了判定. 終了条件には主にある一定の世代数まで達したかどうか, または解が求まったかどうかが採用される.

3. **遺伝的操作** 選択, 交叉, 突然変異といった遺伝的操作を行なう.

6. **適応度の評価** 終了条件, 遺伝的操作に用いるために, 評価関数によって遺伝子の適応度を評価する. 処理手順 2. に戻る.

3.3.3 選択

ここでは遺伝的操作のひとつである選択 (selection) の方法について述べる. これは, 基本的に適応度の高い個体を次世代に残す戦略である.

選択の方法は数多く提唱されているが, ここで述べるのは, 実験システムで用いている方法を述べる. 選択の方法は, 後に述べる遺伝的プログラミング (GP) についても同じである. 実験システムで用いられている選択の方法はランク戦略とエリート保存戦略である. 以下にランク戦略とエリート保存戦略について述べる.

ランク戦略とは, 適応度によって各個体をランク付けし, 予め各ランクに対して決められた確率で子孫を残せるようにする. 各個体は, その適応度毎にランキングされており, 選択確率は適応度にはよらず, ランクに依存する. ランク戦略の問題点は, 適応度とランクによって与えられる選択確率の違いである. つまり, 適応度があまりにも解とかけ離れている場合でも, 子孫を残すことができるということである.

エリート保存戦略とは, 集団中で最も適応度の高い個体をそのまま次世代に残す方法である. この方法は, その時点で最も良い解が後に述べる交叉や突然変異で破壊されないと

いう利点がある. ただし, エリート個体の遺伝子が集団中に急速に広がる可能性が高いため, 局所解に陥る危険もある.

3.3.4 交叉

交叉 (crossover) は, ２つの親の遺伝子を組み替えて子の遺伝子をつくる操作である. 遺伝子が交叉する位置を決めて, その前と後で, どちらの親の遺伝子を受け継ぐかを変える. したの例では, ２つの遺伝子の３番目と４番目を堺にして交叉させることにより, 次世代に残す遺伝子が得られる.

$$\begin{array}{cc} 001|0111 & \\ 101|1100 & \end{array} \quad \Rightarrow \quad \begin{array}{c} 0011100 \\ 1010111 \end{array}$$

3.3.5 突然変異

突然変異は, 遺伝子を一定の確率で変化させる操作である. 下の例では遺伝子の２番目が０から１に変異している.

$$1011001 \quad \Rightarrow \quad 1111001$$

3.4 遺伝的プログラミングについて

遺伝的プログラミングとは, 遺伝子に構造的表現が扱えるようにしたもので, John Koza によって提唱された.

以下に遺伝的プログラミングについて述べる.

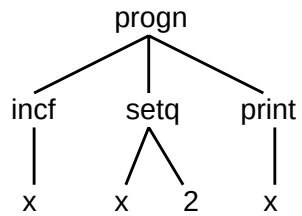

図 3.4: GP における遺伝子の例

3.4.1 遺伝的プログラミングでの処理の流れ

GP では GA で行なわれてきた交叉や突然変異は, 全て木構造に対して行なわれる. GP の処理手順も GA と同様である. 遺伝子の具体例として図 3.4 に示す. この遺伝子構造は, 下の S 式

(progn (incf x)(setq x 2)(print x))

と同義である.

図に示すように, 遺伝子は S 式で書かれるプログラムとなり, その遺伝子を環境上で評価することにより, 適応度が得られる.

遺伝的操作は, 基本的には GA と同じであるが, 交差のオペレーションが GP にとって非常に重要なものとなるため, 以下に述べる. 遺伝的操作を図 3.5 , 図 3.6 , 図 3.7 に示す.

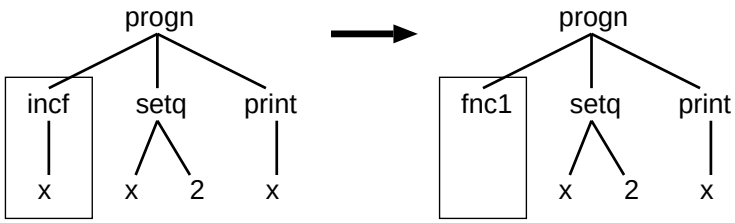

図 3.5: GP における突然変異の例

3.4.2 交叉

GP では遺伝子構造が木構造であるため, 交叉は２つの遺伝子のランダムに選択された部分木を入れ換える操作となる. 図 3.6 がそれである.

GA の交叉との大きな違いは, 同じ遺伝子を持った個体同士を交叉させても, 同じ遺伝子を持った子供が必ずしも生まれないという点である. GA の場合,

$$\begin{array}{ll} 101|1100 & \\ 101|1100 & \end{array} \Rightarrow \begin{array}{l} 1011100 \\ 1011100 \end{array}$$

となり, 遺伝子構造が変わらないのに対し,GP における交叉では, 図 3.7 に示す通り, 交差点によって生成される個体の遺伝子が変化してしまうのである.

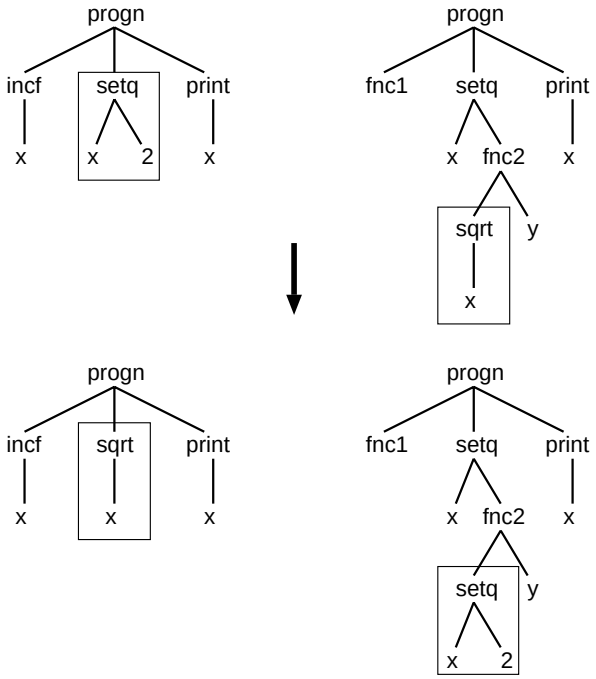

図 3.6: GP における交叉の例 1

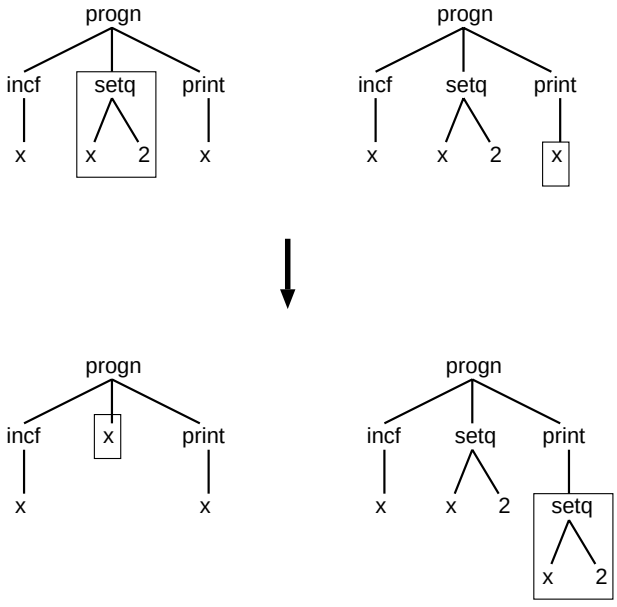

図 3.7: GP における交叉の例 2

3.5 ABCL/f

ABCL/f は並列オブジェクト志向プログラミング言語である. 以下の特徴によって, プログラマは簡単に並列プログラムを記述することができる.

- future の導入
  高効率な並列処理を実現するため,ABCL/f には Halstead により提案された future の変形を導入している. ただし, 暗黙的に同期をとるもともとの future とは異なり, 明示的な同期処理を行なうことができる. 通常のファンクション呼出しもすべてこの future から派生している. 起動の際, 通常の引数に加えて返答届け先と呼ばれる future オブジェクトを生成し, 結果は起動の返答届け先を経て伝達される. この future オブジェクトは, 一階の値として扱うことができる.

- デフォルト規則の提供
  ABCL/f はプログラマがデータの配置と計算処理の並列性を可視とすることを基本としている. しかしこれは, すべての処理をプログラマに委ねるということではない. 特に指定を行なわなければ, コンパイラはデフォルト規則に従い処理する. 加えて, プログラマはこのデフォルト規則から明示的に逃れる方法もあわせて利用できる. 例えば, プロシージャ呼び出しは指定がなければローカルノードで呼び出されるが, プログラマは付加的な支持を用いて明示的に指定したノードで起動することも可能である.

上記 future の導入とデフォルト規則の提供によって, 非同期起動を基本とする多くの並列プログラムを逐次版に大きな変更を加えることなく記述することができる. プログラム開発時に, プログラマは基本アルゴリズムを確かめられる逐次のものから着手でき, アプリケーションを段階的に改良しながら開発を進めることができる.

第 4 章

共通文法獲得モデルの提案

本章では, これまでに述べてきたことをもとに, 共通文法獲得モデルをマルチエージェントの枠組で提案する. 本モデルの目的は,2.2 節で述べたような pidgin 化や creole 化の過程をシミュレートすることである. 本モデルの評価として, 実際に存在する pidgin や creole と比較することが最良なのであるが,

1. 研究者自身の言語習得の困難性.
2. 実際の pidgin における多種言語からの融合による比較の複雑性.

を理由に, 本実験では融合する対象言語として日本語と英語を選んだ.

本研究で取り上げる日本語と英語の差異は

- 日本語は助詞をマーカとして格を指定している.
- 英語は名詞句の表層位置により, 格を指定している.

ことである. 本モデルによって, これらの特徴を学習によって合わせ持つような文法へと変化し, 結果として最初のうちは所有する文法の違いにより, コミュニケーションできなかったエージェント同士がコミュニケーションできるようになることが望ましい.

4.1 マルチエージェント環境と文法獲得のプロセス

本モデルは, 以下のエージェントから構成される.

- 日本人エージェント群: 日本語の文法を既に獲得している.
- 外国人エージェント群: 英文法を獲得している.

これらの構成要素が相互作用することにより, 各エージェントが共通の文法を獲得する過程をシミュレートする (図 1.1参照). そのプロセスを以下に示す.

1. エージェントが発話したいという欲求から, 概念が発生し, その概念をもとに, 自分の持つ文法を用いて文を生成する.
2. エージェントが誰かに対して発話をおこなう.
3. 文を受け取ったエージェントは自分の持つ文法を用いてパースし, そこから得られた概念を返答する.
4. 最初に発話したエージェントは概念を受け取ると, 最初の概念と, 受け取った概念をもとに学習をし, 自分の文法を変えていく.
5. 共通の文法を獲得する.

本モデルで共通文法を獲得したエージェントは,bilingual ではない. つまり, スイッチを切替えたように, 突然以前に獲得していた日本語または英語を発話できるわけではない.

また, 学習によって得られた文法は, 英語でも日本語でもない. 双方が入り混じった混合言語である.

4.2 同国人エージェント同士の会話モデル

同国人エージェントしか存在しない場合の会話モデルは以下のようになる (図 4.1参照). ここでは文法の学習は存在しない. なぜなら, 自分が持っている文法で, 既にコミュニケーションが成立しているからである.

本節では, 学習機構を省いたモデル, すなわち, エージェント間通信, 文生成器, パーサについて述べる.

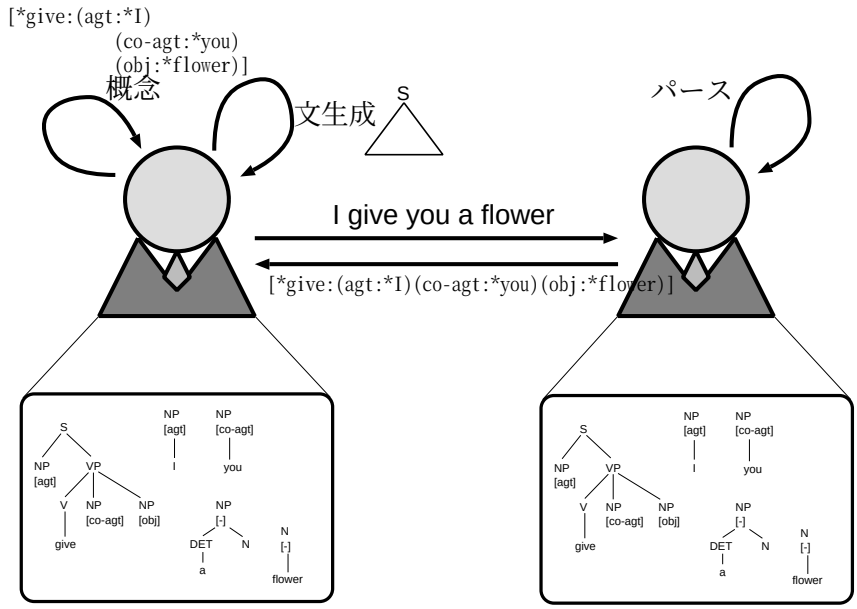

図 4.1: 同国人エージェントによる会話

4.2.1 エージェント間通信

エージェント間での文のやりとりは以下のようになる.

- あるエージェントが以下の操作にしたがって, 他のエージェントに向かって自然言語文を送信する.

  1. エージェントがランダムに概念を発生させる.
     具体的には「私はあなたに花をあげる」という概念は,
     [*give: (agt:*I) (co-agt:*you) (obj:*flower)] となる.
  2. 概念と, 自分の文法を基に文を生成する.
  3. ランダムに選んだエージェントに対し, 生成した文を発話する.

- 文を受け取ったエージェントは以下の操作にしたがって, 文を理解する.

  1. エージェントが文を受け取る.
  2. 文を自分の文法にしたがってパースする.
  3. パースした結果から, 概念を得る.

4. 発話したエージェントへ概念を返す.

4.2.2 文法の定義

本モデルで扱う文法は, CFG (Context Free Grammar, 文脈自由文法) ではなく 3.2 節で述べた TAG である. 文法獲得モデルのほとんどが CFG を用いているのに対し, なぜ TAG を採用したかというと, TAG による構造的表現が, 3.4 節で述べた GP によって, 効率良く学習ができないかと考えたからである.

以下に 3.2.2 項で提案した深層格フィーチャ付 TAG を, 本モデルにおいてどのように定義するかについて述べる.

TAG のオペレーションと格の対応

本研究においては, モデルを簡単にするために, 各文法における格の定義に次のような制約を用いた.

- 必須格は代入ノードによって定義される.
- 任意格は接合ノードによって定義される.

動詞のルールに関していえば, 表層構造において必ず存在しなければならない格は, 代入されるべき非終端記号としてルール内に存在する. 3.1 図においては,α_1のルール中の “eat” 以外の葉ノードである非終端記号がそれである. このルールが構文木として成立するためには, これら非終端記号に対して必ず代入操作をしなければならない. つまりこれは必須格であると定義しても差し支えがないのである.

同様に,β_1のルール “passionately” は,α_1の木に接合されなくてもα_1のルールは構文木として成立することができる. つまりこれは任意格として定義することができる. ここでは前置詞を auxiliary tree, つまり接合によって得ることができる任意格としている. 必ず接合しなければならない, つまり必須格であるのにそれが前置詞句からなる構文も存在するかも知れないが, ここでは簡単な文を対象としているので考えないでおく.

日本語の文法に関して述べる. 日本語の特徴として, 格マーカとして助詞を付加することによって, 表層上格の語順が自由で, 最後に動詞がくるようになっている. これは 4.2 図に示すように, 格マーカを auxiliary tree として定義すると, 語順が自由であることを表現できる.

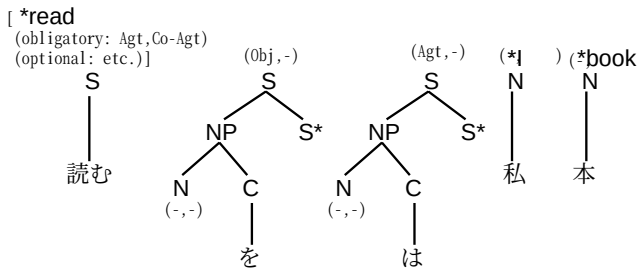

図 4.2: TAG による日本語の表現

しかし, ここで述べた定義と, 日本語の表現に問題が生じる. 3.2.2 項で述べたように, 任意の言語に対して, 動詞の必須格と任意格は共通であると定義し, それはここまで保たれてきた. しかし, 格マーカを auxiliary tree, つまり接合によって定義するならば, 本項で定義したとおり, それはたとえ必須格であっても, 任意格とみなされてしまうのである. これはモデルを簡単に定義しようとして生じた問題であり, 結果として図 4.3 のように日本語の動詞をここでの制約に見合うように再定義し, 日本語の特徴である語順の自由を奪う形になる.

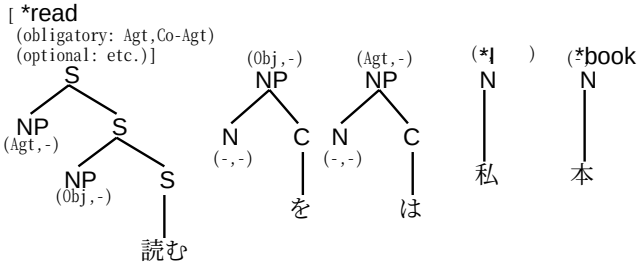

図 4.3: 制約通りの日本語の表現

ここで提案するモデルは上述の制約通り, 必須格であるものは代入ノードで定義することにする.

4.2.3 文生成器

深層格フィーチャ付 TAG を用いることにより, 概念からの文生成が容易に行なえる. 文生成のおおまかな手順は 3.2.2 項で示した通りである. ユニフィケーションをしながら文生成を行なう様子を図 4.4 に示す. これは, 図 4.3 の文法を,

[*read: (Agt: *I)(Obj: *book)]

という概念をもとに文生成を行なっている様子である.

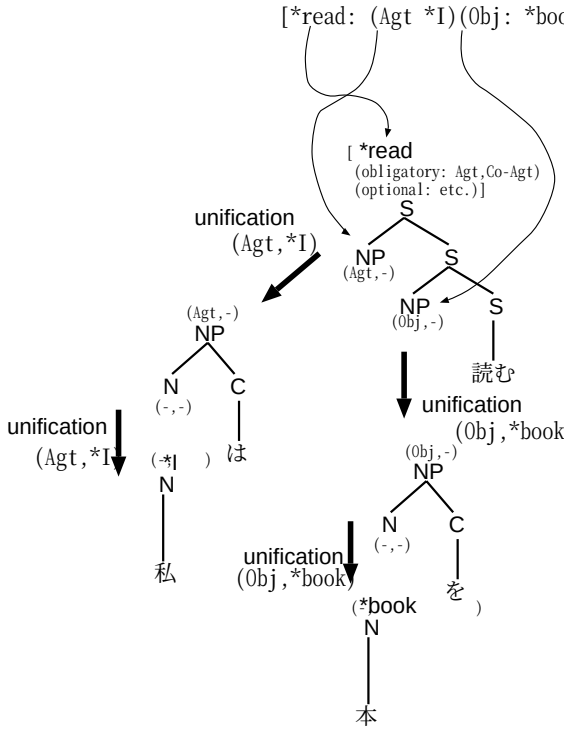

図 4.4: TAG による文生成の様子

4.2.4 パーサ

パーサに関しても LTAG を用いることにより, 比較的容易になる. 文法は全て語彙化されているので, 入力文に対して, 文中の各単語をアンカーに持つ文法を対応させてやれば良い. 代入操作, 接合操作は隣り合った文法にしか対応できないという制約をもうけるこ

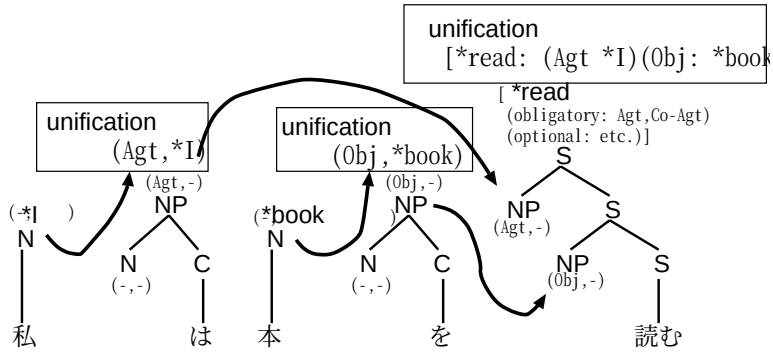

図 4.5: TAG によるパースの様子

とにより,CFG のチャートパーサとほぼ同じアルゴリズムでパースができ, また, ユニフィケーションによって概念が求められる.

図 4.5 に,「私は本を読む」という文を入力とし,

[*read:(Agt: *I) (Obj: *book)]

という概念を求める様子を示す.

4.3 学習機構

本節では学習機構について述べる. このモデルでは, 学習をすることにより, 自分の文法を変化させるといった処理をする. 学習には遺伝的プログラミング (GP) を用いる. 以下に図 4.6を追いながら説明する. この図は概念 “*give” に対応する文法の学習機構である. このように, 学習はそれぞれのエージェントが持つ各文法ごとに行なわれる.

4.3.1 遺伝子の生成

もともとエージェントが持っていた各文法ごと, つまり各概念に対する文法ごとに, 遺伝子を持たせる. この遺伝子は,S 式でかかれており, この式を評価すると, 文法が変化する仕組みになっている. 図 4.6では original がもともと持っていた文法で, この文法が遺伝子群を生成する.

遺伝子の初期値は乱数で決められるが, 最低 1 つの遺伝子は, 何もしない関数 “(none)” にしなければならない. 関数 “(none)” を評価すると, もともとの遺伝子を返す. 適応度の初期値は一定である. 便宜的に, 関数 “(none)” の初期値は他の遺伝子の適応度よりも大きな値にしておいても良い.

4.3.2 文生成用辞書

文生成には original の文法から, １番適応度の高い遺伝子と, ２番目に適応度の高い遺伝子が用いられる. ある概念から文を生成する際, このどちらかの遺伝子を評価した文法が文生成用辞書にエントリーされる. この選択は１文生成するごとにされる.

文生成に使われた文法は, もととなった遺伝子の適応度に反映される.

もし文生成用の候補が１種類しかないならば, つまり１番適応度の高い遺伝子 “(none)” を評価した文法ということで original の文法そのままであるが, これを使って文生成すると, ネイティブである言語をそのまま話すことになる. しかし, このモデルにおいてはもともと自分の持っている文法を使った文では理解できないエージェントに対して発話するのであるから, すこし変化させた文法で冒険的に会話をする必要があるのである. original の文法を使えば, 同国人であるエージェントには話が通じるという安心感と, 少し文法を変化させて同国人を含めた他のエージェントに話して通じるかどうかという挑戦的な試みを合わせ持った辞書であるといえよう.

4.3.3 パース用辞書

パース用の辞書には, 全ての遺伝子を評価した文法セットを用いる. 文生成は適応度の高い遺伝子を評価した文法しか使わないのに対し, パースは適応度の低い遺伝子に対しても文法として用意し, 様々は文に対し, 広く受け入れてやらなければならない. 勿論, パースに使われた文法は, もととなった遺伝子の適応度に反映されなければならない.

エージェント群の中で会話をしていると, 自分が発話する量よりも, 自分の耳に入ってくる文の方が圧倒的に多い. つまり, まわりが話していることを聞きながら学習をし, 自分もそれにかなった文を発話できるようになっていくわけである.

4.3.4 単語の翻訳

本モデルにおいては, 各エージェントが持っている文法の, 単語の変換を許す. このモデルの背景はあくまで pidgin であり, コミュニティが共通文法を獲得する以前にすでに bilingual であるエージェントが存在しても不思議ではない. 辞書を片手にお互いにカタコトと話をしつつ文法を学ぶという状況を考えると, それほど強い制約ではない.

単語の翻訳は単語を翻訳する遺伝子を評価することによって可能となる. つまり, 学習によって単語を翻訳することを学ばなければならないわけである.

4.3.5 学習による文法の入れ換え

original の文法をもとに変化するために遺伝子を生成したのであるが, もし遺伝子を評価した文法のほうが, 頻繁に使われるようになると, すなわち 遺伝子 “(none)” よりも適応度が高い遺伝子が出現するようになると, original の文法はその遺伝子を評価した文法に交換しなければならない. その際, 遺伝子も新たに作り直される. つまり, 適応度が一番高い遺伝子を評価した文法が original の文法となる訳であり, それは遺伝子 “(none)” が一番高い適応度でなければならないことを意味している.

4.3.6 エージェントの文法

エージェントの持つ文法は, 基本的には外国人エージェントが図 4.7 , 日本人エージェントが図 4.8 であるが, 本実験で期待している具体的な学習とは主に

1. 語順の変化

2. 格マーカの進化

であり, 学習によって得られる遺伝子を評価することによってこれを実現するような文法を生成するわけである.

したがって, 文法を見て頂ければ分かるのだが,(1) については, 語順を変化させる文法, つまり複雑な木構造を持つ文法はほとんどが動詞であるので, それ以外のルールには, 外国語の文法を既に持たせておく. そのかわり, そのような文法にたいしては遺伝子による学習は行なわない.

4.4 ABCL/f への実装

本モデルを東京大学で開発中の並列オブジェクト指向言語 ABCL/f へ実装する. これで書かれたプログラムは, 並列プロセッサ AP1000 への実装が可能である.

本モデルを ABCL/f へ実装することの利点としては, エージェントごとのプロセッサ割り当てによる, マルチエージェント環境の実現が可能であることなどがあげられる.

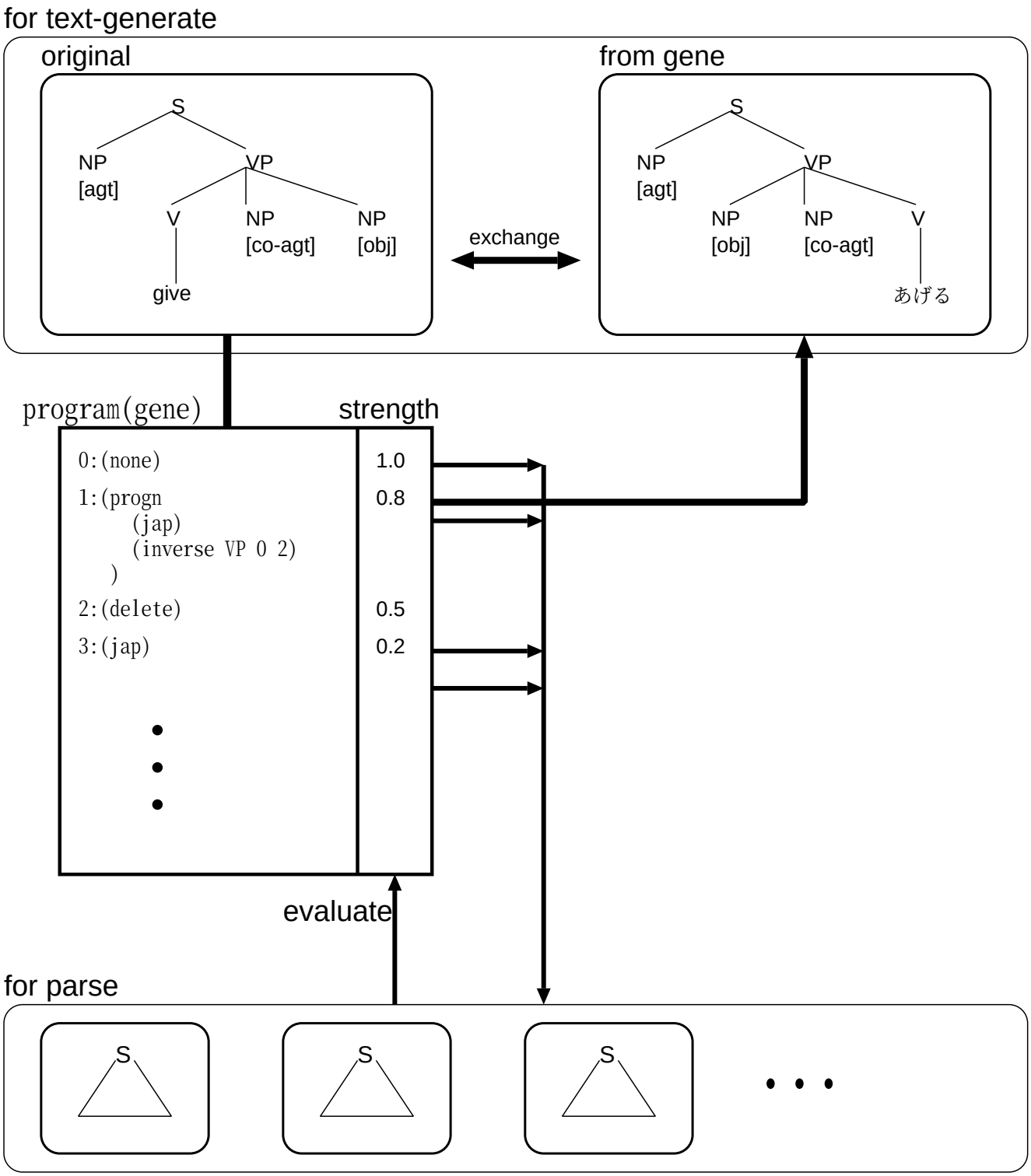

図 4.6: *give に関する辞書

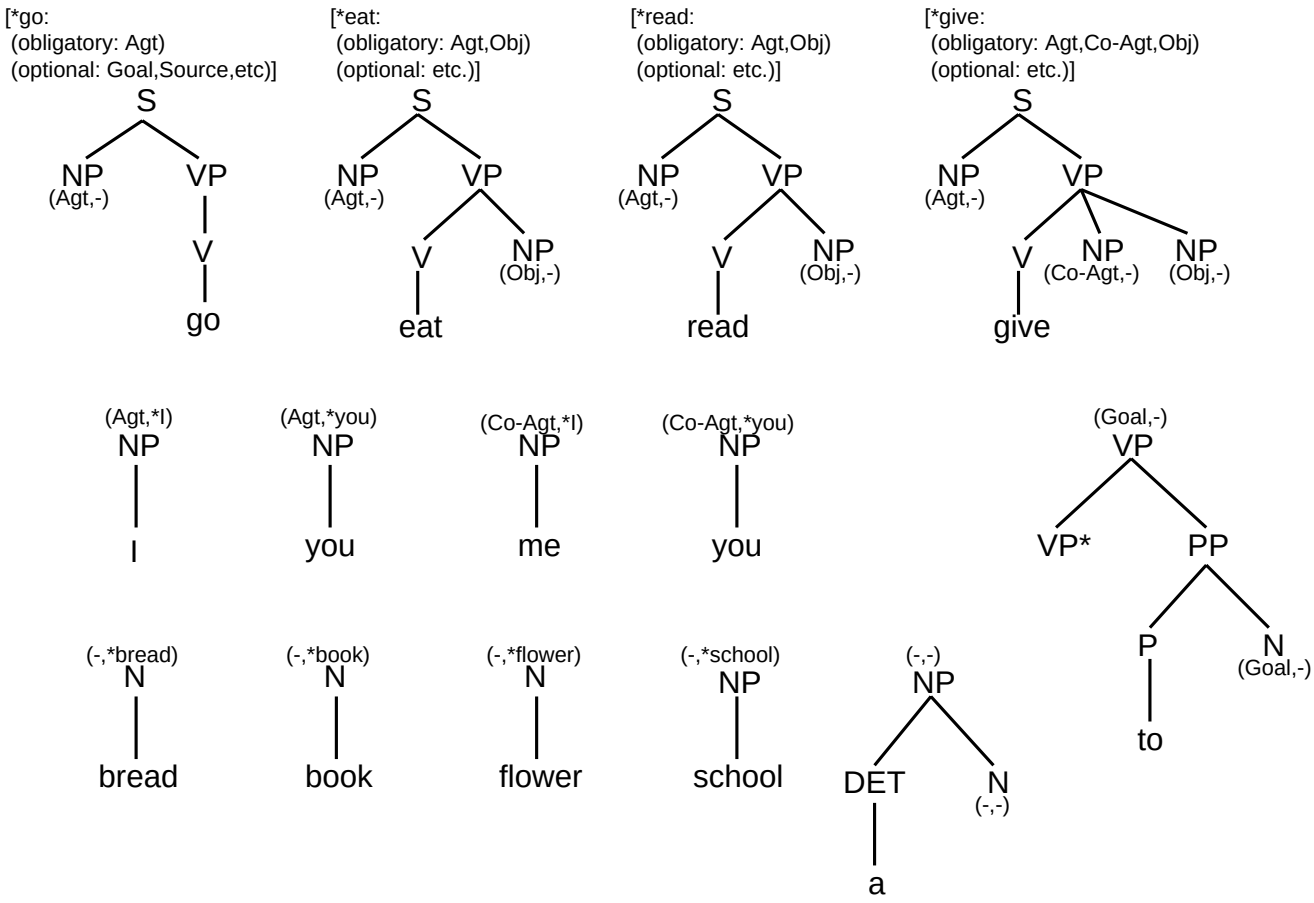

図 4.7: エージェントが持つ英語辞書

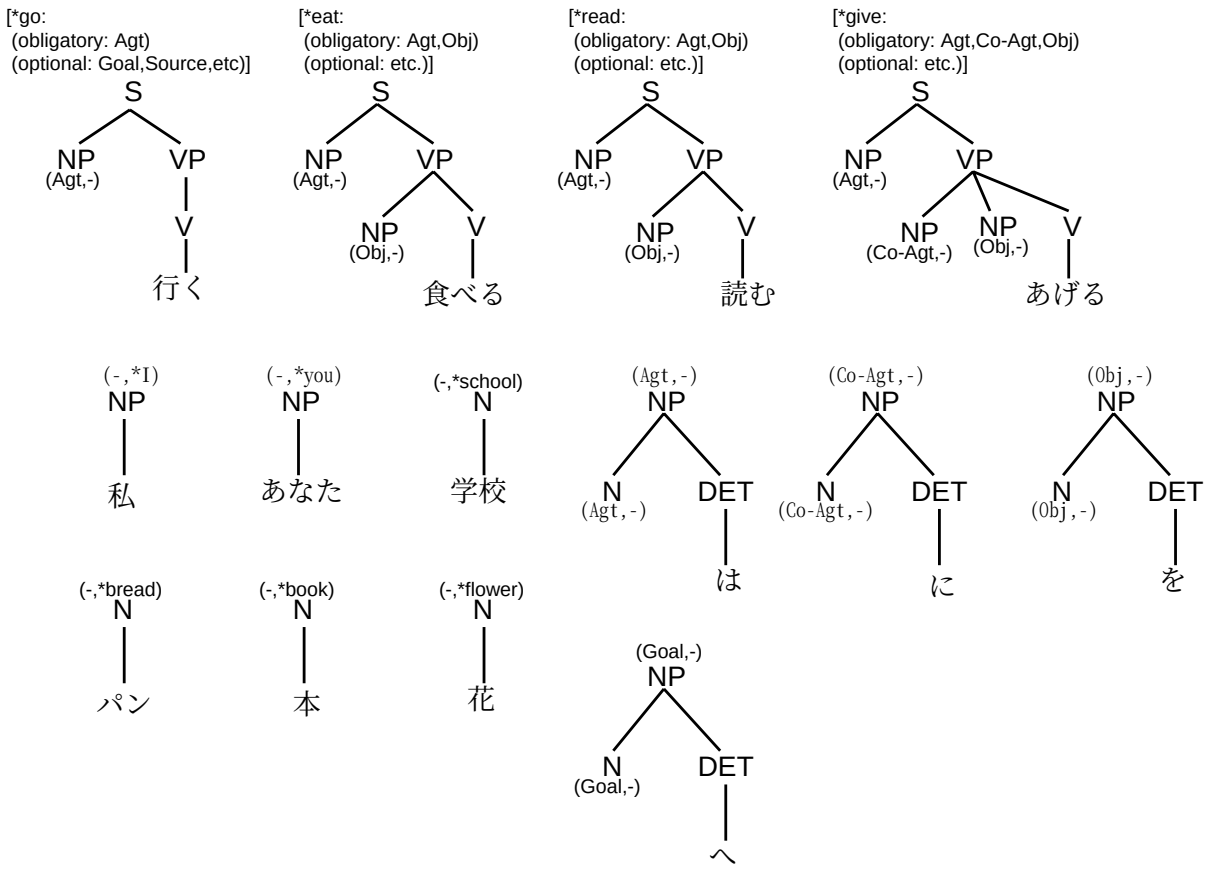

図 4.8: エージェントが持つ日本語辞書

第 5 章

実験

本章ではこれまでに提案してきたモデルに対しての実験を行なう.

5.1 並列プロセッサ上の処理速度の比較実験

本節では並列プロセッサ上の処理速度の比較実験を行なう. 本モデルは並列オブジェクト指向言語 ABCL/f で記述されており, その並列性についての検証が可能である. この実験は, 本研究の目的とは特に関係はないのだが, ABCL/f 自体も東京大学で開発中の言語であるため, 並列プロセッサで実験をする以上, 処理速度の検証が求められる.

この実験は学習をしない同国人エージェント同士の会話モデルのプログラムを用いて実験をする.

並列オブジェクト指向言語 ABCL/f による, 処理速度に大きく影響を与える特徴や, 本モデルにおける処理速度に大きく影響を与えるであろうプログラムレベルでの特徴を以下に述べる.

ABCL/f の特徴

- オブジェクトはプロセッサ依存であり, オブジェクトを定義する際, オブジェクトを置くプロセッサを指定する必要がある.
- オブジェクトのメソッドは, オブジェクトが置いてあるプロセッサで処理される.

モデルの特徴

- TAG ルールにおける木構造の各ノードはオブジェクトとして定義した. これにより, 代入や接合といった操作は関数の副作用として表現が可能である.

- 通信コストを考慮し, エージェントが所有するオブジェクトは, そのエージェントが定義されているのと同じプロセッサに全て定義した. これにより,TAG ルールの木の探索によってかかる通信コストは, 指定したプロセッサがバラバラであるときと比べて格段に減る.

5.1.1 実験環境

実験環境は表 5.1.1 に示す通りである. また, 表 5.1.1 に示すようなパラメータ指定のもとで行なった. パラメータはエージェント数ごとに会話数をそろえるべきであるのだが, 計算機の能力等の関係で, 多少不揃いとなった. ちなみに, プロセッサ数 64 とは並列プロセッサ AP1000 の最大プロセッサ数である.

表 5.1: 並列実験の設定

目的	同国人エージェントの会話プログラムを, 使用したプロセッサ数に対する処理速度をグラフ化し, 各種パラメータを切替え, 処理の高速化を確認する.
パラメータ	エージェント数, 発話数
文法	深層格フィーチャ付 TAG
概念	[*go: (Agt: *I)(Goal: *school)], [*give: (Agt: *I)(Co-Agt: *you)(Obj: *flower)] の２種類をランダムに選択.
発話形態	ランダムに選択したエージェントに向かって概念から生成した自然言語文を発話する.
返事	発話したエージェントに向かってパースして得られた概念を返す.
学習	無し
出力	プロセッサ数ごとの処理時間

表 5.2: 並列実験におけるパラメータの指定

エージェント数 10	会話数 30,60
エージェント数 40	会話数 20,40,60
エージェント数 64	会話数 20,40

5.1.2 エージェントの文法

エージェントの持つ文法は図 4.7 に示すような英語文法である. 各エージェントはこの文法セットを共通に持っている.

5.2 共通文法獲得の実験

本節においては学習機構を備えたエージェントの共通文法の獲得実験を行なう. 学習機構については前章に述べた通りであり, 学習成果の評価としてヒット率を世代ごとに計算したグラフを出力とした. ヒット率とは

$$\text{ヒット率} = (\text{発話先エージェントが認識した数})/(\text{発話数})$$

であり, 各世代ごとに計算され, エージェントごとの学習成果を導き出すことができる. 本実験では, 以下のパラメータ, 文法のもとに日本人エージェントと外国人エージェントを配置し, 実験を行なった.

5.2.1 パラメータ

各種定義は表 5.2.1 の通りである. また, パラメータは表 5.2.1 に示す通りに変化させて実験している.

表 5.3: 言語獲得実験の設定

目的	外国人エージェントと日本人エージェントを複数配置することにより, 学習によって共通言語を導き出すことを目的とする. そのサブゴールとしてヒット率を高めることを目的とする.
パラメータ	エージェント数,1 世代における発話数, 世代数
文法	深層格フィーチャ付 TAG
概念	[*go: (Agt: *I)(Goal: *school)], [*go: (Agt: *I)] [*give: (Agt: *I)(Co-Agt: *you)(Obj: *flower)] [*give: (Agt: *you)(Co-Agt: *I)(Obj: *book)] [*read: (Agt: *you)(Obj: *book)] [*eat: (Agt: *I)(Obj: *bread)] の６種類をランダムに選択.
発話形態	ランダムに選択したエージェントに向かって概念から生成した 自然言語文を発話する.
返事	発話したエージェントに向かってパースして得られた概念を返す.
学習	GP によって生成したプログラムにより, 文法を書き換える. 文のパース, 概念の返答から適応度の評価をする. GP の遺伝子となる関数は表 5.2.1 参照.
出力	各エージェントから各世代のヒット率.

表 5.4: 言語獲得実験におけるパラメータの指定

エージェント数 4	１世代での会話数 10	世代数 50
エージェント数 4	１世代での会話数 20	世代数 50
エージェント数 4	１世代での会話数 50	世代数 50
エージェント数 4	１世代での会話数 100	世代数 50

表 5.5: 遺伝子となる関数の定義

(progn g1 g2 g3)	遺伝子g1,g2,g3 の順に評価をする関数. 0 を返す.
(none)	何もしない関数. 1 を返す.
(translate)	単語を翻訳する関数. そのルールのアンカーが 英語ならば日本語に, 日本語ならば英語に翻訳する. 2 を返す.
(inverse g1)	木構造をひっくり返す関数.g1 が 0 ならば ノード S,1 ならばノード VP につながっている 子ノードの順番を入れ換える. 3 を返す.
(1)-(9)	何もしない関数だが,(1)-(9) に対応した値を返す.

第 6 章

実験結果と考察

6.1 並列プロセッサ上の処理速度の比較実験

6.1.1 実験結果

前節でのパラメータをもとに実験を行なった. 実験結果をグラフで出力したものを図 6.1 , 図 6.2 , 図 6.3 に示す. これはエージェント数ごとに会話数をまとめたグラフであり, 表 5.1.1 におけるエージェント数 10 が図 6.1 ,40 が図 6.2 ,64 が図 6.3 に対応している.

6.1.2 考察

グラフ全体を見ると, プロセッサ数が増えるごとに演算時間が減少していることが分かる. また, 途中まで単調減少であったのに, あるプロセッサ数まで進むと, 減少具合が鈍くなり, ついには単調減少でなくなってしまうという点も 3 つのグラフに共通している.

これはオブジェクト指向言語でいうところのオブジェクトがプロセッサ依存ということが大きく影響している. つまり, エージェントもプロセッサ依存なのである.

プロセッサ数と処理時間の関係を以下に示す. ここで処理するプロセッサ台数を $num(PE)$, エージェント数を $num(agt)$ と定義する.

$num(PE) \ll num(agt)$ 使用したプロセッサ台数がエージェント数よりも少ない場合, エージェントを複数抱えるプロセッサが存在することになる. この場合は当然ひとつのプロセッサにエージェントが少ない方が良いわけで, 理想は 1 プロセッサにつき 1

エージェントが最良のケースと考えることができ, そこへ向かってグラフは単調減少していくのである.

$num(PE) < num(agt)$ プロセッサ台数がエージェント数に近づいていくにしたがって, 処理速度の伸び悩みが発生する. これは１プロセッサが抱える平均エージェント数がプロセッサ数が１つ増えてもたいして変わらなくなってきたからである.

$num(PE) \geq num(agt)$ プロセッサ台数がエージェント数を上回ると, もはや処理速度が向上する要素は何もなくなる. したがって, 処理速度が早くならないどころか, 使用しないプロセッサまで管理しなければならなくなり, 結果としてプロセッサが増えるごとに遅くなっていくのである. しかし, この増加具合は処理時間に対する誤差を大きく下回っているため, このグラフには出て来ない.

図 6.1 , 図 6.2 , 図 6.3 はエージェント数が固定で, エージェントの会話数によるグラフである. これを見ると会話数が増えるごとに処理時間は増える. では次に会話数が固定でエージェント数による処理時間の違いを図 6.4 に示す. これは会話数を４０に固定して, エージェント数が４０と６４に関して比較をしたものである.

当然の結果ながら, エージェント数を増やしたほうが処理時間がかかることが確認された.

図 6.5 はプロセッサ数の増加にともない, エージェント数も増やしたグラフである. つまり $num(PE) = num(agt)$ である. 会話数は１０としている. このグラフを見ると, 処理時間はほぼ比例している.

本節の実験として, 並列プログラムによる並列プロセッサの処理速度の比較実験を行なった. これは 並列オブジェクト指向言語 ABCL/f に対する実験である. ここで得られた結果は処理効率について述べることはできるが, 言語獲得の見地からは何も評価はできない. 何故ならば, 各エージェントがエージェント内, つまり割り当てられているプロセッサ内で行なっているプロセスはただ単に文生成とパースを繰り返すだけだからである.

ここでの実験によって,ABCL/f によって実装したプログラムが, 並列性による処理の高速化が確認できた. これによって次節で述べる学習機構を備えたエージェントの高速処理に十分期待ができる.

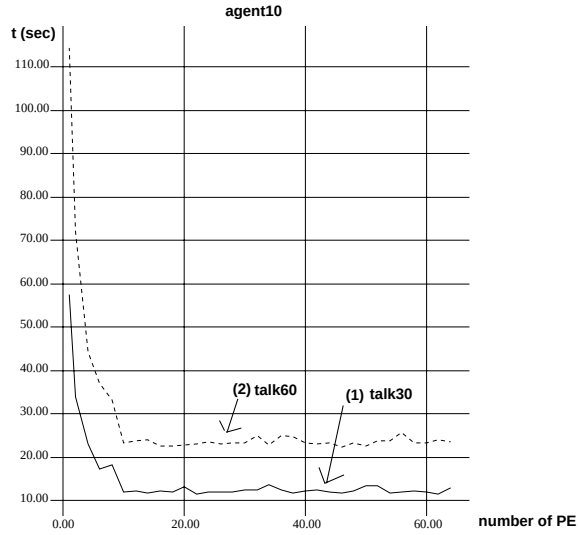

図 6.1: プロセッサ台数に対する処理速度 (1)

6.2 共通文法獲得の実験

6.2.1 実験結果

実験結果を示す. 上で指定したパラメータに対応する図を表 6.2.1 に示す.

表 6.1: 言語獲得実験におけるパラメータの指定

エージェント数 4	会話数 10	世代数 50	図 6.6
エージェント数 4	会話数 20	世代数 50	図 6.7
エージェント数 4	会話数 50	世代数 50	図 6.8
エージェント数 4	会話数 100	世代数 50	図 6.9

6.2.2 考察

グラフを見比べてわかることは, 図 6.6 よりも図 6.9 の方が右上がりのグラフであり, 1 世代に会話をする量が多い方が着実に学習していることが伺える. 逆に, 1 世代での会話数が少ないケースではヒット率が非常に不安定であり, 会話が高い確率で成立する世代も

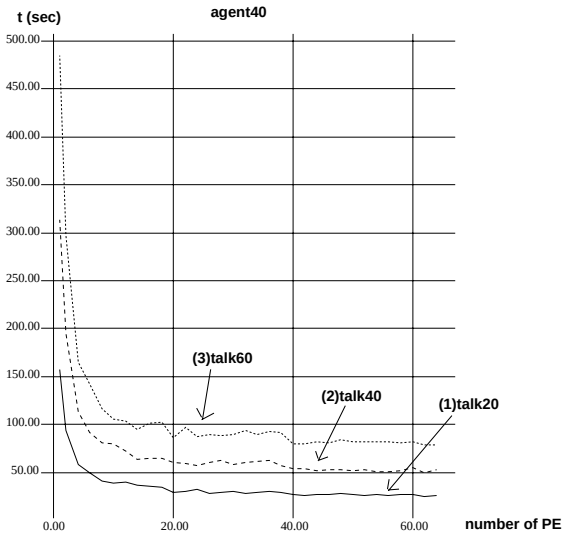

図 6.2: プロセッサ台数に対する処理速度 (2)

あるのだが, 次の世代もそうとは限らない. このケースは, とても学習効率が良いとはいえない.

この理由は, 適応度の点数づけが遺伝的操作に大きく影響しているということである. 適応度は発話をする側は,

- 発話した文から正しい概念が返ってきたか.

また文を受け取る側は

- 入力文をパースできたか.

によって加点される. つまり, 適応度の値は, 文生成をするごと, パースするごとに変化するのである. 遺伝的操作をすると, 遺伝子の適応度はまた初期値に戻ってしまうので, 本当に良い遺伝子かどうかを知るためには, 長い時間をかけてじっくりと適応度を決めるべきなのである. ただやみくもに遺伝的操作をして, せっかく育った適応度を振り出しにするのはあまり効率が良くはないということである.

学習による言語獲得の実験において, 以下の 2 つを学習によって実現することを期待すると述べた.

- 語順の変化

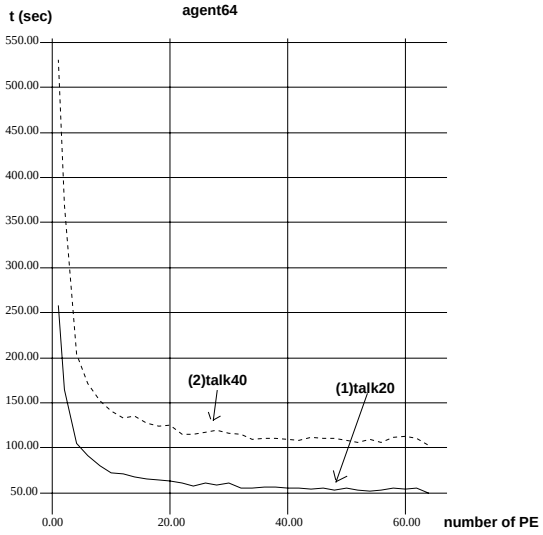

図 6.3: プロセッサ台数に対する処理速度 (3)

- 格マーカの進化

これを得るためには 3 段階の学習が必要である.

1. 自分のもつ文法とは異なる文法から生成された文をパースするために, ランダムに生成された遺伝子を評価して得られたでたらめな文法をパースに用いることにより, それに対応した文法の適応度が上がる. ここではまず語彙が翻訳されているかが鍵となる.

2. 適応度の高い遺伝子をもとに遺伝的操作がなされる. ここで語順が変化する.

3. 淘汰された遺伝子を評価することによって得られた文法を用いて発話をしてみる.

(3) に行きつくまでには (1) と (2) を何世代か繰り返す必要がある. これを行なって生き残った遺伝子, つまり最も適応度の高くなった遺伝子を評価した文法が original である文法と入れ替わるわけである.

具体的に, 異国人エージェントに対して会話を行ない, 実際に認識された文を含む表示例を以下に示す.

```
[*read:([obj]:*book)([agent]:*you)]
anata c-a hon c-o yomu
```

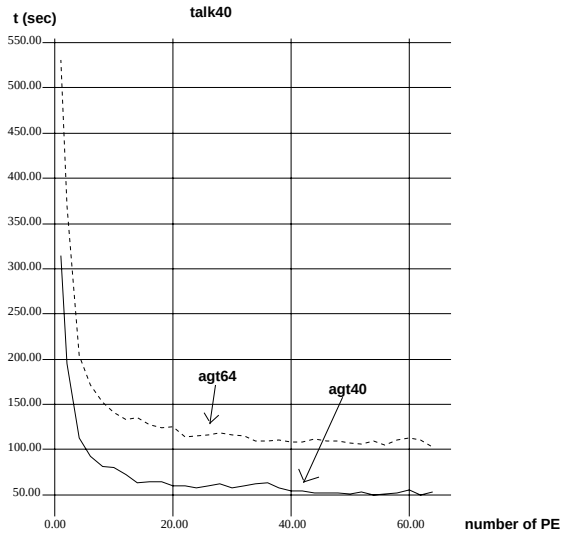

図 6.4: プロセッサ台数に対する処理速度 (4)

```
[subst:NP [agent] -]
(NP[agent] (N [agent])(C (c-a )))
[subst:N [*] *you]
(N[*] (anata ))
[subst:NP [obj] -]
(NP[obj] (N [obj])(C (c-o )))
[subst:N [*] *book]
(N[*] (hon ))
[verb:*read(obligate:[obj][agent])(optional:)]
(S (NP [agent])(VP (NP [obj])(V (yomu ))))
```

これは日本人エージェントが外国人エージェントに対して会話をした例であり,

[*read:([obj]:*book)([agent]:*you)]

をもとに文生成をし,

anata c-a hon c-o yomu

という文を発話し, 外国人エージェントに認識されているということを意味している. 文中に出現する “c-a”,“c-o” は進化した格マーカであり, 日本語の「は」及び「を」を「翻

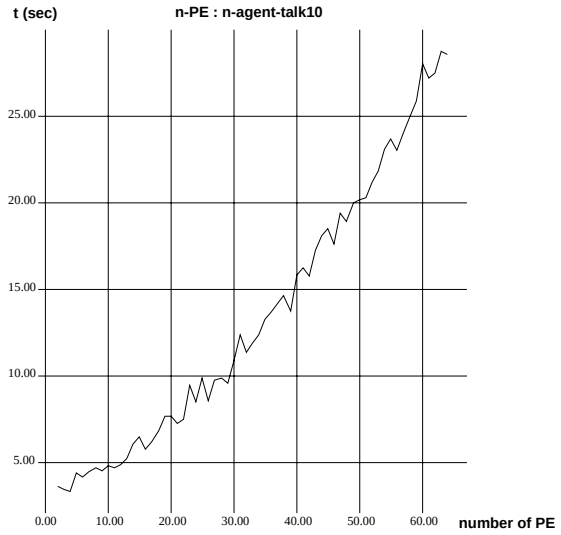

図 6.5: n プロセッサ:n エージェントの処理速度の変化

訳」という形で変換し, 外国人にも通用する共通の格マーカへと進化した. ちなみにその他の S 式等は文生成に用いた文法である.

逆に外国人エージェントが日本人エージェントに対して会話をした例を以下に示す.

```
[*give:([obj]:*book)([co-agt]:*I)([agent]:*you)]
anata c-a watashi c-c hon c-o ageru
[subst:NP [agent] -]
(NP[agent] (N [agent])(C (c-a )))
[subst:N [agent] *you]
(N[agent] (anata ))
[subst:NP [co-agt] -]
(NP[co-agt] (N [co-agt])(C (c-c )))
[subst:N [co-agt] *I]
(N[co-agt] (watashi ))
[subst:NP [obj] -]
(NP[obj] (N [obj])(C (c-o )))
[subst:N [*] *book]
(N[*] (hon ))
```

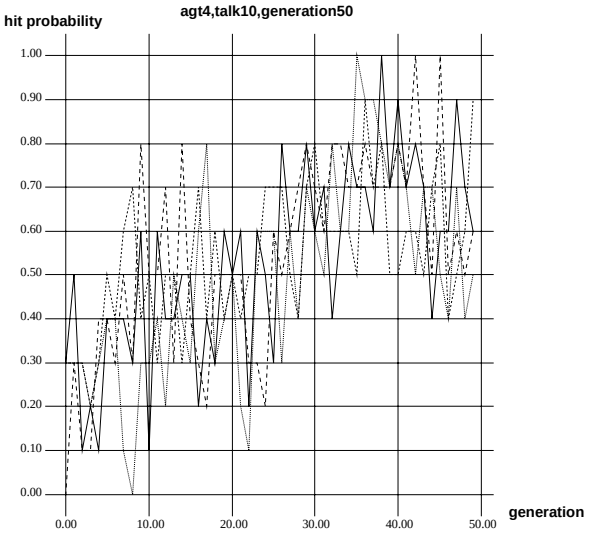

図 6.6: 4 エージェントの世代ごとのヒット率の変化 (1)

```
[verb:*give(obligate:[obj][co-agt][agent])(optional:)]
(S (NP[agent])(VP (NP [co-agt])(NP [obj])(V (ageru ))))
```

これは

[*give:([obj]:*book)([co-agt]:*I)([agent]:*you)]

をもとに文生成をし,

anata c-a watashi c-c hon c-o ageru

という文を発話し, 外国人エージェントに認識されているということを意味している. 文中に出現する “c-a” は日本語の「に」に対応する格マーカである. これは外国人が日本人の語順及び格マーカを学んだという証拠である.

上のような例を見ると, まるで外国人エージェントが日本語を学んだだけのようにも見えるのだが,

```
[*go:([goal]:*school)([agent]:*I)]
gakkou c-g go watashi c-a
[adjoin:S [goal] -]
(S (PP (N [goal])(P (c-g )))(S *))
```

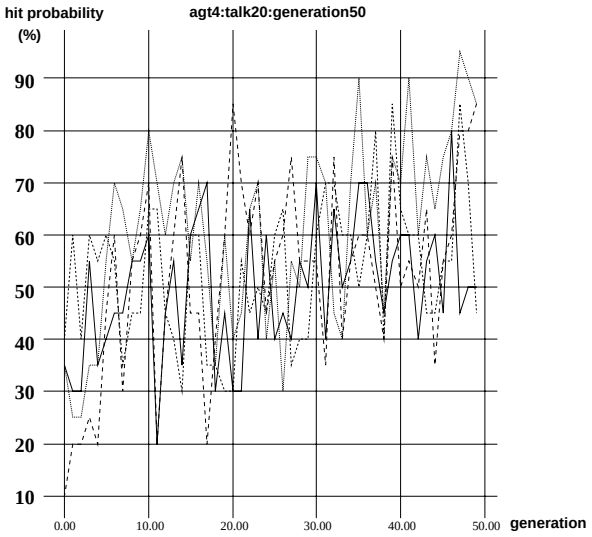

図 6.7: 4 エージェントの世代ごとのヒット率の変化 (2)

```
[subst:N [*] *school]
(N[*] (gakkou ))
[subst:NP [agent] -]
(NP[agent] (N [agent])(C (c-a )))
[subst:N [*] *I]
(N[*] (watashi ))
[verb:*go(obligate:[agent])(optional:[goal])]
(S (VP (V (go )))(NP [agent]))
```

という日本人エージェントの会話例のように,

gakkou c-g go watashi c-a

という文を生成しているものもある. これは “*go” に関しては主格の位置が日本語, 英語のどちらでもない文法に落ち着いてしまったのである.

このように, ヒット率の上昇とは, 共通言語の獲得を意味しているのである.

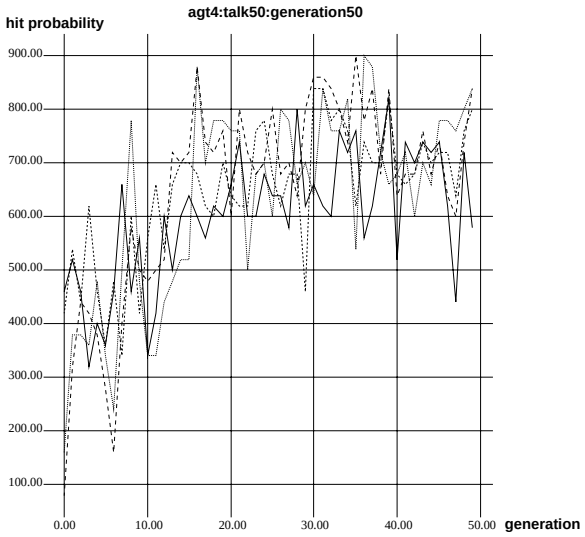

図 6.8: 4 エージェントの世代ごとのヒット率の変化 (3)

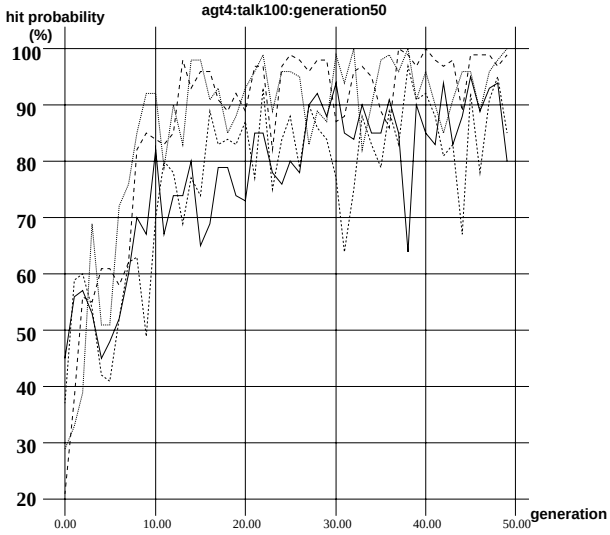

図 6.9: 4 エージェントの世代ごとのヒット率の変化 (4)

第 7 章

結論

7.1 おわりに

最後に本研究全般のまとめをする.

本研究においては言語学上の実際に存在する混合言語 **pidgin** や **creole** のシミュレーションをするという目的であった. そのために言語学上の背景としてピジン, クレオール, リンガ・フランカといった世界中の混合言語における, 言語融合へのプロセスを調査した.

それをもとに実験システムを定義し, マルチエージェントモデルの枠組で, コミュニティ内での共通言語獲得モデルの提案を行なった. 本モデルの制約としては,

- お互いの語彙の共有をゆるすが, 文法の構造までは共有しない.
- 概念を持たすことにより, 有意味な会話をする. ただパースして木構造を得るだけでは評価にはならない.

こととした. また, モデルの機能的特徴としては

- 文法表現は深層格フィーチャ付 TAG を使用した.
- 学習は GP を持たせた.
- 並列環境での処理により, 実際に起こっている言語融合の環境に近づけた.

である.

提案したモデルを実装し, 並列環境での処理速度の検証及び共通文法獲得の実験を行なった.

- 並列環境の検証については, 処理速度の向上が, 認められ, さらに並列環境という, 現実世界に近い環境を表現できた.
- 共通文法獲得に関しては, 会話の認識度の向上が認められ, さらに日本語と英語の 2 つの言語が混ざることにより, 日本語でも英語でもない言語であるが, 両者の特徴を持った混合言語がコミュニティ内に広まったことが確認された.

7.2 問題点と今後の課題

今回定義した文法は, コミュニケーションをとろうとしているエージェントはすでに文法を寛大にして接しているという課程のもとで文法を定義したので, 非常に簡単な構造でとなっている. 例として, 英語文法は人, 称, 数等は既に省いて表現しており, また日本語も,TAG により語順の自由さを表現しておきながら, ここでは定義しなかった.

本研究においては主に語順の変化の学習に焦点を当てた. 今後の課題として, もう少し複雑な文から始めて, 自ら自分のもつ文法を簡単化することから学習を始めるようなモデルを定義することをここに記しておく.

謝辞

東条 敏助教授には研究方針, 取り組み方に対して適切な助言を頂きました. 奥村 学助教授には自然言語処理に関する助言を頂きました. 東京大学米澤研究室の田浦健次郎助手には ABCL/f, プログラム作成に関する助言を頂きました. 佐藤研究室博士後期課程の小野哲雄氏には研究上有益な助言, 及び研究に対する取り組み方, 言語学上の助言を頂きました. 最後に, いつも励ましてくれた両親に心から感謝いたします.

参考文献

[1] Loreto Todd, ピジン・クレオール入門田中幸子訳, 大修館書店,1986.

[2] 北野 宏明 編遺伝的アルゴリズム産業図書,1993.

[3] John R. Koza GENETIC PROGRAMMING The MIT Press,1993.

[4] 東条 敏自然言語処理入門近代科学社,1988.

[5] 伊藤 拓也遺伝的プログラミングによるロボットプログラムの頑健性に関する研究北陸先端科学技術大学院大学 情報科学研究科 木村研究室 修士論文, 1996.

[6] Aravind K.Joshi Tree-Adjoining Grammars The Encyclopedia of Language and Linguistics (ed. R.E. Asher, University of Edinburgh), Pergamon Press, Oxford,1993.

[7] The XTAG Research Group A lexicalized Tree Adjoining Grammar for English Institute for Research in Cognitive Science University of Pennsylvania, 1995.

[8] Yves Schabes, Aravind K. Joshi AN EARLEY-TYPE PARSING ALGORITHM FOR TREE ADJOINING GRAMMARS 26th ACL Proceedings pp258-269, 1988.

[9] Yves Schabes, Anne Abeillé, Aravind K. Joshi Parsing Strategies with 'Lexicalized' Grammars: Application to Tree Adjoning Grammars COLING Proceedings vol.2 pp578-583, 1990.

[10] Anne Abeillé, Yves Schabes, Aravind K. Joshi Using Lexicalized Tags for Machine Translation COLING Proceedings vol.3 pp1-6

[11] 長尾 真 編自然言語処理岩波講座 ソフトウエア科学 15 岩波書店, 1996

[12] Kristian Lindgren Evolutionary Phenomena in Simple Dynamics ARTIFICIAL LIFE II, pp295-312, Addison Wesley,1992.

[13] Werner, G.M. and M.G.Dyer Evolution of Communication in Artificial Organisms, in C.G.Langton (Ed.), Artificial Life II, pp659-687, Addison-Wesley, 1991

[14] 小野 哲雄・東条 敏: マルチエージェント・モデルによる文法の獲得, マルチエージェントと協調計算研究会 (日本ソフトウェア科学会) 発表資料, 1995.

[15] Ono, T., Tojo, S. and Sato, S.: Common Language Acquisition by Multi-Agents, *International Computer Symposium (ICS'96), Proceedings on Artificial Intelligence*, pp.218–223, 1996.

[16] Hashimoto, T. and Ikegami, T.: Evolution of Symbolic Grammar Systems, *Third European Conference on Artificial Life*, pp.812-823, Springer, 1995.

[17] Steven Pinker THE LANGUAGE INSTINCT William Morrow & Company,Inc., 1994.

[18] 小野典彦, Torkaman Rahmani Adel 人工生物群による通信の自己組織化マルチエージェントと協調計算研究会 (日本ソフトウェア科学会) 発表資料, 1992.

[19] 亀井孝, 河野六郎, 千野栄一編 言語学大辞典三省堂,1996.

[20] ディヴィッド・クリスタル著, 風間喜代三, 長谷川欣佑監訳 言語学百科辞典大修館書店,1992.

[21] ENCYCLOPEDIA AMERICANA, GROLIE INCORPORATED, 1995.

[22] Encyclopedia Britanica, Encyclopedia Britanica Inc, 1991.