確率文脈自由文法を持つエージェント間での言語共有

北陸先端科学技術大学院大学
情報科学研究科
武田稔也　小保方幸次　東条 敏
E-mail: {ttakeda,obokata,tojo}@jaist.ac.jp

1　はじめに

本研究ではマルチエージェント環境での確率文脈自由文法 (Probabilistic Context-Free Grammar, PCFG) [2] の共有化を自動化 [6] のプロセスの一部と考え, 各エージェントの持つ PCFG の文法規則の木構造に確率を与え, これをエージェント間での発話, 認識という経験的学習によりチューニングし共通言語を自己組織化する.

また, マルチエージェント環境での相互学習という設定であるため, 学習が局所的なループに陥る危険性がある. そこで, 本研究では負事例の文を設定し, 正事例と負事例の文の入力から負事例を排除するような確率を学習することとする.

この研究は言語学的な正当性に基づくものであると同時に, 異なるプロトコルを持つ人工エージェント同士が柔軟に新たなプロトコルを自己組織化する複雑系のシミュレーションの側面を持つ.

実装には, これまでの同様な研究とは異なり多数のプロセッサを用い, 各々のプロセッサに各エージェントを割り当てた分散環境でのモデル化を行う. そして, マルチエージェントモデルの研究における実験上のシステムの提案を行う.

2　文法共有化モデルの提案

2.1　文法の枠組

各エージェントは異なる文法規則を持ち, 他のエージェントとのコミニュケーションから, 帰納推論により新しい文法規則を生成する. ここでは, エージェントの持つ文法について考察する. 便宜上, 本システムでは 2 進数で表された数字列を一つの語と見なす.

仮定 1 エージェントの有する文法規則は文脈自由文法の形態をとる.

仮定 2 語彙は異なる文法規則を有するエージェント間においても共有されているものとする.

仮定 3 エージェントの発話は, エージェントの持つ文法規則から確率的に選択されるものとする.

仮定 4 エージェントの発話は他のエージェントの持つ文法規則により構文解析できたとき, 理解されたとする.

各エージェントからは他のエージェントが持っている文法セットを直接見ることはできず, それによって生成される文を交換するのみである. この過程において, エージェントは他のエージェントの発話を理解できるように新しい文法規則を推論し, それを自分の文法セットに追加する.

本モデルでは, エージェントが持つことのできる文法の最大数は, パラメータとして設定される. 仮定 1, 仮定 3 にあるようにモデルは文脈自由文法の文法規則に重みをつけた確率文脈自由文法である. 使用頻度の低い文法規則や, 他のエージェントに認識されなかった文法規則に対する重みは減少していき, 閾値以下になると不要な規則として捨てられる. このようにエージェント内部では, 生得的な機構により発現した文法が変化して, 後で出来る部分に順応していく.

2.2　エージェント間通信

エージェントの発話は基本的に他のすべてのエージェントに対して行われる. 本来, 任意のエージェントが他の任意のエージェントにランダムに話しかけられるよう設定することが理想であるが, 本モデルにおいても

十分長い時間をとれば, この本来の意図と同等の効果を期待できるため, 便宜上このように設定する. エージェントの発話に使用する文法規則は, それぞれのエージェントの持つ確率文脈自由文法により決定される.

確率文脈自由文法を次の 4 つの組 $\langle V_T, V_N, P, S\rangle$ で定式化する. ここで, V_T は終端記号の集合 $\{w_1, ..., w_n\}$, V_N は非終端記号の集合 $\{A_1, ..., A_\nu\}$ である. ここで, A_1 は開始記号であり特に S で表す. P は生成規則で, α_j を任意の終端記号と非終端記号の列とすると $A_i \to \alpha_j$ と表される. これらの生成規則にはそれぞれ確率 $P(A_i \to \alpha_j)$ が存在する. これは非終端記号 A_i を展開する際, 全ての A_i に対する文法規則の中から α_j を選択する確率である . また $A_i \to \alpha_j$ において, $\sum_j P(A_i \to \alpha_j) = 1$ である. 各エージェントは, 各状態 A_i で, この選択確率にしたがって文を生成していく.

エージェントは, 発話された文字列に対し形態素解析を行う. この段階では, 各エージェントの持つ文法規則の確率をもとにした形態素解析は行わず, 試行錯誤的に繰り返される. そして, 各エージェントの持つ文法規則に従って構文解析を行う. これに成功する, つまりコミニュケーションに成功した場合, これを入力文として繰り返し最適化を行い, エージェントの持つ文法規則の選択確率 $P(A_i \to \alpha_j)$ を自己組織化してゆく.

また, コミニュケーションに失敗した場合, 発話したエージェントの使用した文法規則の確率を減らし, 聞き手側のエージェントは, それを満たす文法規則を帰納推論の枠組みで新たに生成する.

通常の確率文脈自由文法のアルゴリズムでは, 文 W に対する導出 T 中の生成規則 $A \to \alpha$ が使われた回数 $C(A \to \alpha; T, W)$ に, 導出 T の相対確率 $P(W,T)/P(W)$ を考慮して, 次のように定義している.

$$
\begin{aligned}
C(A \to \alpha) &= \sum_T \frac{P(W,T)}{P(W)} C(A \to \alpha; T, W) \\
P(A \to \alpha) &= \frac{C(A \to \alpha)}{\sum_\beta C(A \to \beta)}
\end{aligned}
$$

これに対し, 本モデルでは文 W に対する導出 T 中において, それぞれの確率を独立に計算している. つまり,

$$
P(A \to \alpha; T, W) = \frac{C(A \to \alpha; T, W)}{\sum_\beta C(A \to \beta; T, W)}
$$

となる. このような定義は, 本研究が文の解析を目的としたものではなく, エージェント間でのコミュニケーションを目的としているためである. 本研究では, 語に対する意味や目的達成のための表現などは付加していない. しかし, このような定義をすることによって, 同じ文法規則でも, その使われ方に応じた確率が形成されることになる. つまり, 例えば日常言語における主語として使用される語 (名詞) と目的語として使用される語 (動詞) の対応関係のような構造的意味を持たせることができる.

エージェントの保有できる文法規則の最大数は確率文脈自由文法に依存する. つまり, コミニュケーションに失敗し選択される確率の低くなった文法規則は淘汰され, 使用頻度の高い文法規則のみが保持される. また, 自らが保持していない全ての文法規則を帰納推論によって生成すると過度に一般化されてしまい, 文法の共有化とは言えない. そこで, 負例によって過度の一般化を防ぐ必要がある. しかし, 言語を取り扱うというモデルの性質上, 正例と負例の判断基準を定めるのが困難である. そこで本稿では, いくつかのエージェントを教師エージェントとした, 一般の教師つき学習モデルとして取り扱う.

2.3 並列分散プログラミング

本モデルにおける実験環境について検討を行う. 分散処理を行うことにより, 得られる利益として処理速度の向上を考えるのは, ごく当然のことである. しかし, 本モデルにおける分散処理の目的は, 計算速度の向上よりも, 各プロセッサを一つのエージェントに見立てた大規模なシミュレーションにある. 純粋に計算速度の向上を目指す立場から見れば, 資源の無駄遣いに思えるだろう. しかし, マルチエージェントモデルを用いた言語共有において大規模なシミュレーションを行うことは, 実際の言語活動により近づくという意味では, 非常に意義のあるものである.

本モデルのシミュレーションにおける分散プログラミングへの要求は以下の通りである (図 1).

1. 各プロセッサでそれぞれのエージェントの動作を模倣する
2. エージェント間のコミニュティとしてプロセッサ間の通信を行う
3. 各プロセッサの発話数はプロセッサの処理速度に依存する

図 2 に分散プ ログラミングのタイムテーブルを示

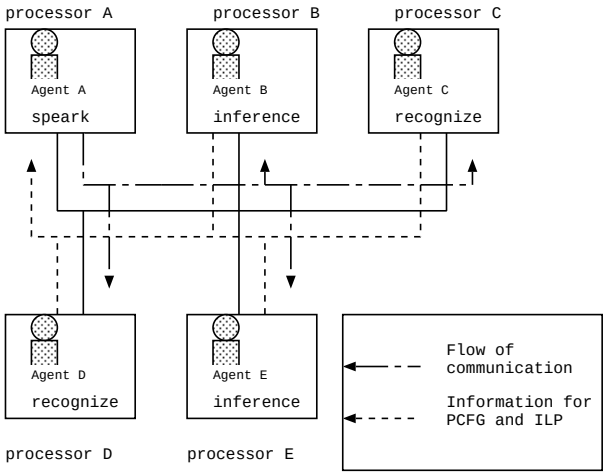

図 1: 並列化の概念

す. ここで, 図中の記号 A, B, C はエージェントを表している. ただし, S, W, R は, プロセッサ A における処理状態である.

分散処理の実行管理はフラグのチェックによって行われる. 各エージェントは文の入力を行う Input 処理を完了したら, 実行完了のフラグ (Input Flag) を立てる. 各エージェントはこのフラグをチェックし, 全てのエージェントのフラグが立てられるまで次の処理を行わない. 同様に Speak 処理, Output 処理の各セクションでフラグによる実行制御を行う. ここで, 図中の “Lock Flag” は, 文の出力処理を行う際に, ファイルの多重アクセス回避のためのフラグである.

プログラムの主な動作は Input, Parse, Speak, Output である. このうちファイルアクセスを行っているのは, Input と Output である. 制約として各段階において, 各エージェントは全てのエージェントがその処理を終えるまで, つまり各段階におけるフラグが全てのエージェントによって立てられるまで, そのままの状態で待機することとする. これにより, ファイルの多重ア クセスによるデッドロックを防いでいる. ただし, Speak においてはファイルアクセスを行っていないので, 待機状態中に繰り返し Speak 処理を行うことが可能である.

図 2 では, 全てのエージェントが入力を行った後, つまり, 全てのエージェントが Input Flag を立てた後に, 次の構文解析 (Parse), 文生成 (Speak) を行っている. この段階で, エージェント B は Speak 処理に時間がかかっている. このため, エージェント A はここで 2 度繰り返し処理 (R) を行っている. エージェント B の Speak 処理終了後, エージェント A は出力 (Output) の開始状態 (S) に入るが, これよりも先にエージェント B が出力処理を開始したため, 待機状態 (W) に入っている. この図では, エージェント B が 1 度 Speak 処理を行う間に, エージェント C は 5 度の Speak 処理を行っている.

このように, 処理速度の早いプロセッサが次の処理に移るまでの待機状態を利用して, 繰り返し Speak 処理を行うことにより, エージェント間の発話頻度にばらつきを持たせている.

3 実験

使用した文法規則と語彙セットを図 3, 4 に示す. 文の生成には表の文法規則に基づいて行われる.

また, これらの文法規則に割り振られた選択確率には閾値を設定し, 使用頻度の低い文法規則については淘汰を行う. 各エージェントは 1 ステップに一定の文を生成し, 他の全てのエージェントに発話を行い, 聞き手側のエージェントはそれについて構文解析を行う.

$S \to A_1 \ \ A_2$	$A_3 \to A_9 \ \ A_1$
$A_1 \to A_4 \ \ A_8$	$A_2 \to A_{10} \ \ A_7 \ \ A_1$
$A_1 \to A_5 \ \ A_8$	$A_2 \to A_7 \ \ A_1$
$A_1 \to A_6 \ \ A_8$	$A_2 \to A_7 \ \ A_3$
$A_1 \to A_8$	

図 3: 文法規則の集合

000	A_8	0001	A_{10}		1001	A_4	
001	A_8	0010	A_{10}		1010	A_8	A_4
011	A_5	0011	A_8	A_7	1011	A_8	A_4
100	A_9	0100	A_7	A_{10}	1100	A_8	
101	A_8	0101	A_7		1101	A_4	
110	A_4	0110	A_7		1110	A_8	A_7
111	A_8	0111	A_4		1111	A_9	
0000	A_8	1000	A_9				

図 4: 語彙セット

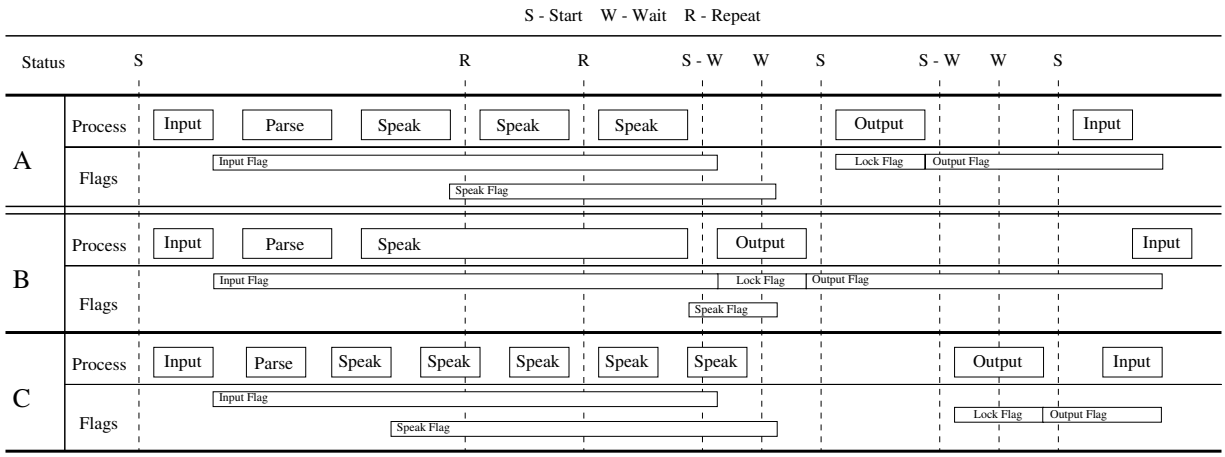

図 2: 分散プログラミングのタイムテーブル

3.1 実験 1

実験 1 ではエージェント数を 10 にしてシミュレーションを行った. また初期の段階で各エージェントが保有する文法規則数に制限を加えた.

この実験ではエージェントコミュニケーションに正例, 負例を設定した. 具体的には常に一定の選択確率に基づいた発話を行う教師エージェントを作り, このエージェントにより生成される文を正例とし, また, 予め各エージェントに負例文を与え, これと同じ入りだしで生成された文を負例としてペナルティを加えた. また, 初期の文法セットに組み込まれなかった文法規則は, 帰納推論の枠組において背景知識として存在する. 実験条件を以下に示す. ここで, 発話数は, 淘汰パラメータとの兼ね合いにおいて適当な 10 文に設定した. 教師エージェント数と負例数については, その数値を変えて実験を行った結果より設定した. その結果については省略する.

エージェント数	10
教師エージェント数	2
1 ステップの発話数	10 文
負例数	2
ステップ数	1000 step

負例数を 2 として行った実験結果として $A_2 \rightarrow A_7$ A_1 における A_1 の展開についての各エージェントの選択確率の推移を図 5 に示す.

正例のみからの学習, 負例のみからの学習, そして正例と負例両方からの学習のそれぞれの選択可能な文法規則数の推移を図 6 に示す. 初期の段階では, 各エージェントとも有用な文法規則を見出していないため, 各自が勝手に発話を行い, 多くの文法規則を学習していることが確認される. しかし, ステップが進むごとに確率文脈自由文法の共有化が進むにつれ, 発話は特定の文法規則を使用されていった.

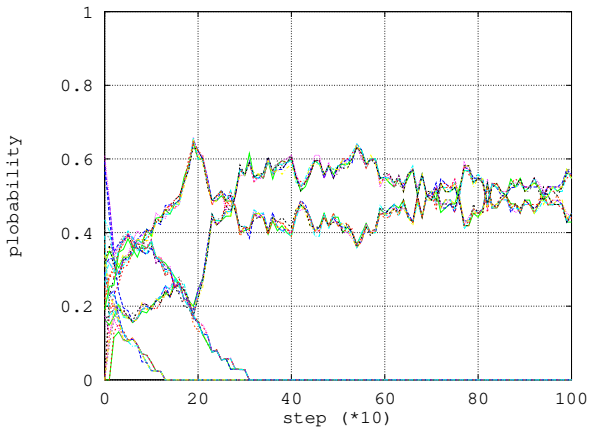

図 5: $A_2 \rightarrow A_7$ A_1 における A_1 の展開

負例のみからの学習モデルと, 正例と負例からの学習モデルを比較してみると, はじめのうちは, 正例と負例からの学習モデルが, 負例のみからの学習モデルと同様に, 使用可能な文法規則数が減少している. しかし, 430 ステップ付近から, 正例のみからの学習モデルの文法規則数に近づいているのが確認される. また, 負例のみからの学習モデルについては, 逆に 450 ステップからの値に変化がなく局所的な文法構造に陥ってしまった. また, 正例のみからの学習モデルは, 他の学習

モデルと比較してエージェントの持つ文法規則数の変動が安定するまでに, 多少時間がかかっている.

これから言えることは, モデルに負例を与えることにより, 早い段階での文法規則の淘汰 (特殊化) が実現し, モデルに正例を与えることにより, 会話に多様性を持たせること (一般化) が可能である. そして, 正例と負例からの学習モデルでは, この二つの性質がうまくマッチングしていることが確認される.

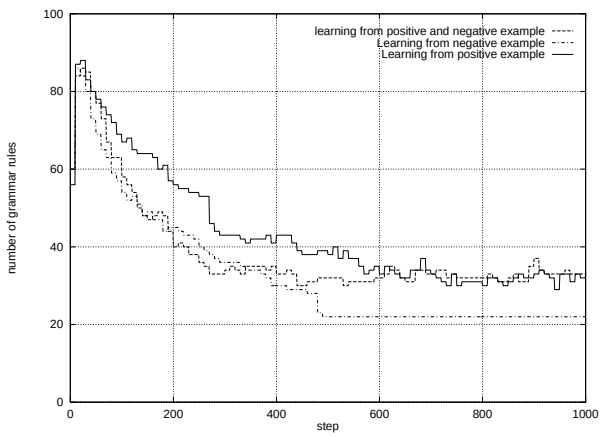

図 6: 正例と負例の影響

3.2 実験 2

実験 1 のモデルにおいて複数のプロセッサを使用した分散処理実験を行った. これまでの実験では 1 ステップにおけるエージェントの発話数が限られており, 全てのエージェントが同数の文を生成していた. 本実験では分散処理の性質を利用してエージェントの発話数にばらつきを持たせた. つまり, 処理速度の早いプロセッサに割り振られたエージェントの発話数は, 他のプロセッサに割り振られたエージェントよりも多くなる. 本実験は, 前の実験よりも, より社会的なモデルのシミュレーションといえるだろう. 実験条件を以下に示す.

エージェント数	4
1 ステップの最大発話数	18 文 (3×6 *loops*)
ステップ数	1000 step
淘汰パラメータ	0.008
使用計算機	SS5, ultra5

実験 2 では, 分散環境がモデルに与える影響を評価するため, 4 台の計算機を使用して実験を行った. まず, 発話回数の多いエージェント群の中に一つだけ発話回数の少ないエージェントを入れた場合について考察する. 図 7 に, 発話回数の多い 3 つのエージェントと発話回数の少ない 1 つのエージェントでのコミュニケーションの結果を示す. やはり, 発話数の多いエージェン

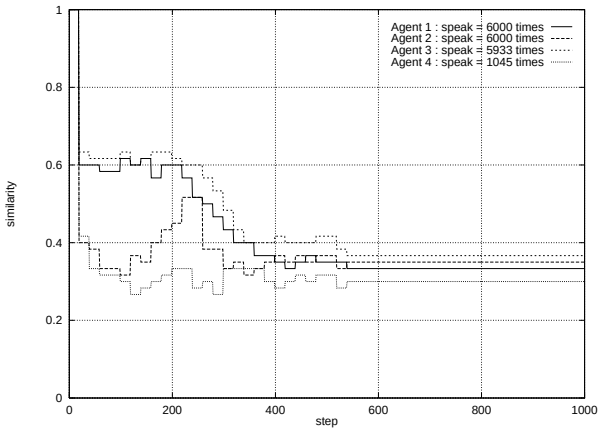

図 7: 3 対 1 のモデル

トのほうが高い類似度を保ったまま世代を重ねているのが確認される. また, この 2 つの実験より各エージェントの発話数の違いによる影響は, 初期の段階で大きく作用することが分かった. 初期の段階での各エージェントの文法規則の選択確率はまだ試行錯誤的な段階のため, より多く発話を行ったほうが他のエージェントの文法規則の選択確率に与える影響が大きいものと思われる. ただし, 図 7 では発話数の多いエージェントが 3 つ存在するため, イニシアチブの獲得で衝突が起こっている. エージェント 3 は, 他の 2 つのエージェントと発話数がそれほど変わらないにも関わらず, その類似度を落している. エージェント 1 とエージェント 2 のもつ文法セットが, 互いにエージェント 3 のもつ文法セットよりも比較的似ていたためにこの現象が生じたものである. 集団社会において, その部分集団の規模が大きい程, 全体に与える影響力が強いことを示している.

次に, このモデルとは逆に発話数の少ない 3 つのエージェントの中に発話数の多いエージェントを 1 つ加えて実験を行った (図 8).

エージェント 1 が他のエージェントよりも高い類似度を保持しているものの, 上の 2 つのモデルに比べ, そ

の差は狭まっている. つまり, 発話数の多いエージェントも集団の規模が大きくなれば, 与える影響力が少なくなってしまうことを示している.

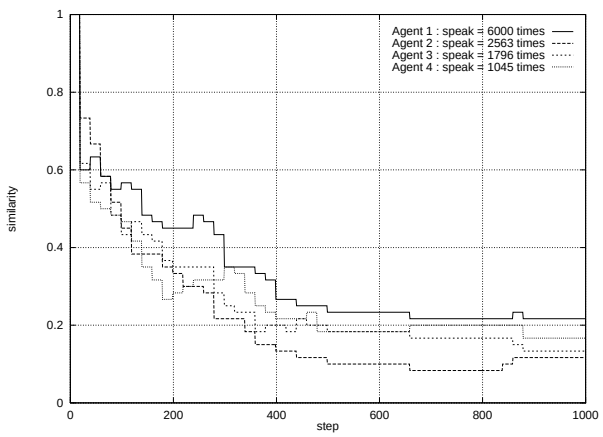

図 8: 1 対 3 のモデル

4 まとめ

本研究では, マルチエージェント環境でのコミュニケーションによる文法の共有化についてのモデル化を行った. 実験環境としては, 分散処理により各プロセッサを一つのエージェントに見立てた大規模なシミュレーションを行った. 各エージェントは, 割り当てられたプロセッサの処理速度に依存した発話を行うため, 発話数に個体差が出現した. この現象に対しては, 発話に個体差を生じさせたほうが, より現実社会の現象を反映したモデルになった.

本稿で提案したモデルは幾つかの点で人工的な設定を行っているが, 今後はこれらの設定に対しても, より根拠のある (認知科学的に説明可能な) モデルに改善していくこと, そしてそれが良いモデルであることを主張するために実際の言語データから統計的検証を常に考えていく必要がある. また, 確率文脈自由文法を用いた構文解析 (probabilistic parsing) の導入や, 帰納論理プログラミングの文脈自由文法への応用についての考察を行う必要がある.

参考文献

[1] Derek Bickerton 著, 筧 寿雄 監訳: ことばの進化論, 勁草書房, 1998.

[2] Eugene Charniak: STATISTICAL LANGUAGE LEARNING, *TechBooks*,1993.

[3] Hashimoto, T. and Ikegami, T.: Evolutin of Symbolic Grammar Systems, *Advance in Artificial Life*, pp812-823, 1995.

[4] Hashimoto, T. and Ikegami, T.: Emergence of net-grammar incommunicating agents, *BioSystems 38*, pp1-14, 1996.

[5] 北研二: 確率的言語モデル, 東京大学出版会, 1999.

[6] 大津由紀雄 他: 言語科学と関連領域, 岩波書店, 1998.

[7] 小野哲雄, 東条 敏: 推論機能を有するエージェント群による共通文法の組織化, 人工知能学会誌 *vol.13, no.4*, pp546-559, 1998.

[8] Norihiko Ono and Adel T. Rahmani: Self-Organizatin of Communicatin in a Multi-agent System, *MACC*, 1992

[9] Christopher D. Manning and Hinrich Schütze: Foundatins of Statistical natural Language Processing, *The MIT press*, 1999.

[10] Luc Steels: The Origins of Syntax in Visually Grounded Robotic Agents, *IJCAI*, pp1632-1641, 1997.

[11] Werner,G. M. and M.G.Dyer: Evolutin of Communication in Artificial Organisms, *Artificial Life II*, pp659-687, 1991