自然言語とプログラミング言語から見る言語の特徴

平野 正徳
東京大学 教養学部 (前期課程)
理科一類 1 年

I. はじめに

大学の展開科目自然科学ゼミナール「社会シミュレーション演習」という授業で artisoc を使用して，エージェントシミュレーションを行った．この際に使用した書籍『人口社会構築指南』(山影進著) において，artisoc を使用して，人口社会を構築し，様々なことをシミュレーションする方法を見た．その上で，僕自身，プログラミングを色々と行うだけでなく，今回，artisoc でまた異なる言語を使用したこともあり，プログラミング言語間の関係や使用者の変化に興味を持った．そのため，プログラミング言語に限らずに，言語全般も含めて，artisoc を使用して，人口社会を構築することでエージェントシミュレーションを行い，研究を行おうと考えた．

近年，グローバル化が進み，ますます他国との交流が進み，母語を異とする人同士の交流が活発となり，母語以外の言語を利用する機会も多くなった．しかしながら，1993 年の言語の使用者数順位 [1] と 2009 年言語の使用者数順位 [2] の間では，3 つ以上の順位の変化が上位 10 言語において見られないことから言語の使用者数に大きな変動は見られないと考えられる．一方で，Github の統計 [3] 及び TIOBE の統計 [4] 結果に基づけば，プログラミング言語においては使用数の順位変動が頻繁に起きている．この原因はプログラミング言語が主にインターネットを媒介として広まっていくのに対し，自然言語は人と人の直接のコミュニケーションで広まっていくという点で土地的拘束が発生することと，プログラミング言語は自然言語に比べて，各言語間での共通点が多く，似ている点にあると考えられるので，これらの条件だけを変化させることで自然言語とプログラミング言語のユーザー数の変動を再現できるのではないかと考えた．この研究を通して，共通語に適した性質や言語の使用を支配する要素を分析することで，さらなるグローバルなコミュニケーションの手助けとなるかもしれない．

II. モデルの作成

前節で述べた通り，土地拘束の有無，言語間での共通点をパラメーター化した親和度を含めて考慮に入れ，artisoc のモデルを作成した．

A. モデルの要件

登場するエージェントは「話者」及び，言語の発信源となる「terminal」である．以下で，それぞれのエージェント及びエージェントを配置する空間，設定項目，集計項目を説明する．なお，実際の artisoc ファイルにおいては，プログラミングの便宜上，エージェント等のパラメーターに以下では述べていない要件を満たすものがあるが，あくまで便宜上であり，モデルには影響がない．

1) エージェント：話者*:*

- 各ステップにおいて使用言語を設定できる．
- 言語ごとに習熟度が設定でき，言語の獲得・忘却ステータスを設定できる．
- 空間内を移動することもしないこともできる．
- 最初に空間内のランダムな位置に配置される．

2) エージェント：*terminal:*

- 固有の言語は一つである．
- 空間内を移動しない．
- 出現する際に空間内のランダムな位置に配置される．

3) 空間*:*

- サイズが 100×100(ループあり) の二次元空間である．
- この二次元空間内をエージェントはモデルに沿って移動したり，しなかったりできる．

4) 設定項目 *(*括弧内はモデルにおける変数の対応等*):*

- 話者エージェントの数 (number_of_people)
- 言語 (terminal) の個数 (number_of_lang)
- 話者エージェントに対する土地拘束の有無 (bind)
- 土地拘束を有効にした場合の話者エージェントの移動範囲 (moverange)
- 獲得済み言語の習熟度の増加ペース (usingplus)
- 獲得済み言語の習熟度の減少ペース (notusingminus)
- 獲得済み言語の忘却のための習熟度減少量 (forgotlimit)
- 各言語間の親和度 (lang 二次元配列の一部,CSV 入力, 入力のない場合は [0,0.1) のランダム)
- 各言語の出現タイミング (step 数)(lang 二次元配列の一部,CSV 入力, 入力のない場合は 0)
- 各言語の習得の難易度 (lang 二次元配列の一部,CSV 入力, 入力のない場合は [35,50))
- 各話者エージェントが周りを見て言語を決める確率 (propotion_of_looking_around)
- 各話者エージェントの視野 (range_of_looking)

5) 集計項目*:*

- 各言語の各ステップでの使用者数 (number_of_users)
- 各言語の習得者数 (number_of_those_who_has)

B. その他表示項目

- エージェントの配置状況
  - 赤：言語使用中の話者エージェント
  - 黒：言語を使用してない話者エージェント
  - 黄：terminal

C. モデルのフローチャート

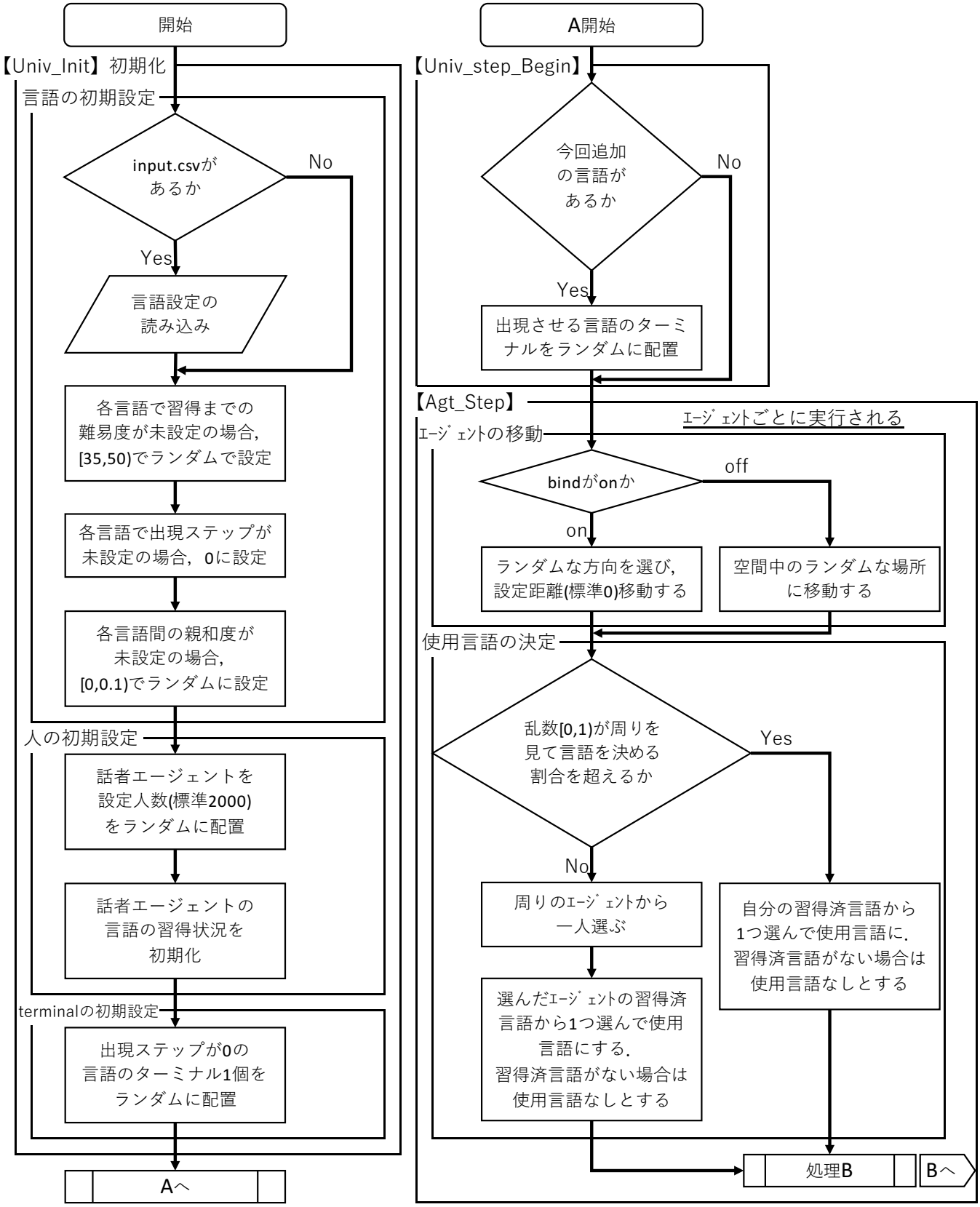

Fig. 1. 作成したモデルのフローチャート．2 ページにわたる．一部，便宜上，実際のフローとは異なる部分があるが，基本的に同じである．

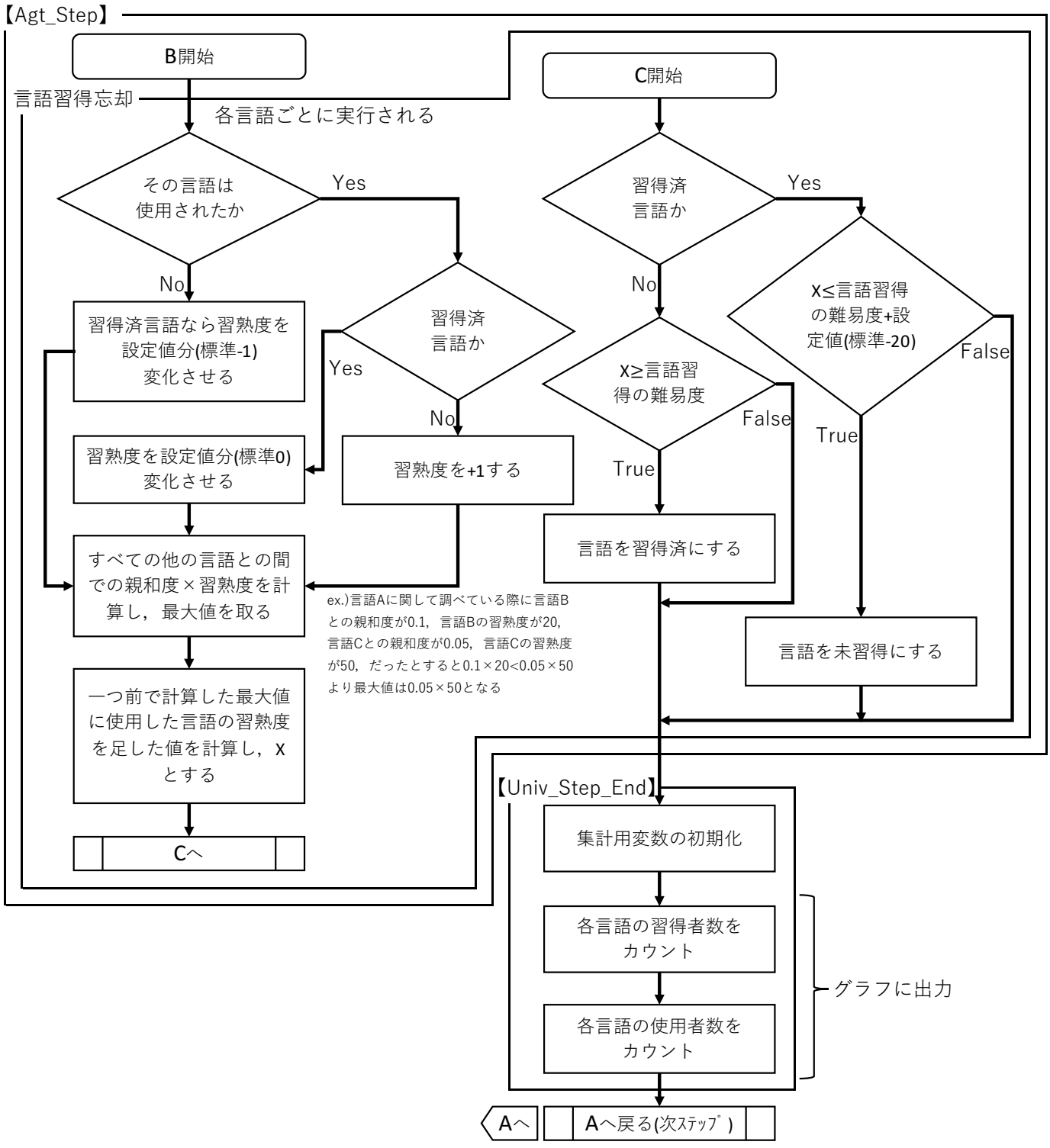

Fig. 1. *Continued.*

D. コントロールパネル

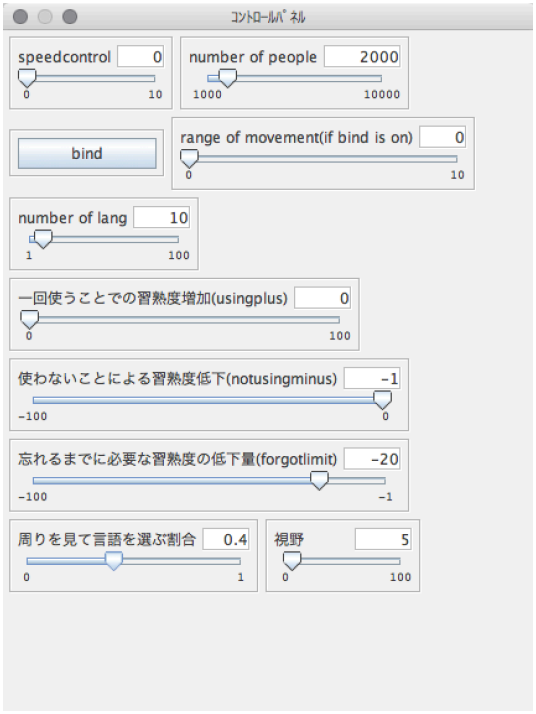

Fig. 2. コントロールパネル．これでパラメーターの一部を操作できる．

上図のようなコントロールパネルでパラメーター値を操作する．ここまでで説明が十分でないパラメーターを説明する．視野だが，これは話者エージェントの視野である．主に周りのエージェントを選ぶ際の範囲となる．標準値を 5 としてある．また，speedcontrol だが，これはモデルには影響なく，単にステップ間の待ち時間の有無及び，待ち時間の設定に使用する．これを使用するのは，あまりにも変化が早すぎて見えない場合などであるため，ほとんど使うことはない．それ以外のパラメーターは既出なので，説明はここでは割愛する．

E. CSV 入力

今回作成したプログラムでは，CSV 入力により，言語の細かい設定ができるようになっている．ただ，CSV 入力がなくても自動でランダム値や規定値など適切な値が設定される．(詳細はフローチャート) また，空白も入力がないものとして自動でランダム値や規定値など適切な値が設定される．

	A	B	C	D	E	F	G
1		習得までの値	出現ステップ	lang0との親和度	lang1との親和度	lang2との親和度	lang3との親和度
2	lang0			1	0.5	0.5	0.5
3	lang1			0.5	1	0.5	0.5
4	lang2			0.5	0.5	1	0.5
5	lang3			0.5	0.5	0.5	1
6	lang4			0.5	0.5	0.5	0.5
7	lang5			0.5	0.5	0.5	0.5
8	lang6			0.5	0.5	0.5	0.5
9	lang7			0.5	0.5	0.5	0.5
10	lang8			0.5	0.5	0.5	0.5
11	lang9			0.5	0.5	0.5	0.5

Fig. 3. 入力用 CSV のデータ例．Microsoft Office Excel 2016 for Mac で表示した．この場合，習得までの値は [35,50) でランダムで設定され，出現ステップは 0 に設定される．なお，CSV はカンマ区切り．

III. パラメーターの検討

ここまでのモデルで使用しているパラメーターの設定には検討が必要である．そこで，実際のユーザー数の変化に似たようなものになるようなパラメーター値を探すことにした．この際の基準としたことと条件を挙げる．

- 自然言語においては，bind を on とし，移動範囲は 0 とする．
- プログラミング言語においては bind を off とする．
- 特定の言語におけるユーザー数がブレイクアウト (9 割以上が一つの言語に集中してしまう状態と定義する．) しないこと．
- 自然言語は全言語の累計使用者数の変化がほぼなくなってから (以下これを安定期と呼ぶ) 首位言語が 3 位以下に落ちること (以下これを陥落と呼ぶ) が少ない．
- プログラミング言語においては安定期に入ってから首位言語の陥落が頻繁に見られる．
- 計測期間は安定期の 3 倍の step 数か 500steps のうち大きい方に至るまでとした．

また，変化させたパラメーターは以下の通りである

- 自然言語の場合
  - 周りを見て言語を選ぶ割合:0.1〜1.0(0.1 間隔)
  - 親和度:明示的に指定しない ([0,0.1) のランダム)
- プログラミング言語の場合
  - 周りを見て言語を選ぶ割合:0.1〜1.0(0.1 間隔)
  - 親和度:全言語間において 0.1,0.5(ただし，周りを見て言語を選ぶ割合が 1.0 の場合，フローからわかるように親和度は関係ない．また，親和度は大きすぎると何もせずに自動的に言語習得してしまい，おかしな動きをするので 0.1 と 0.5 のみとしている．)

A. 自然言語の結果

- 周りを見て言語を選ぶ割合:0.1

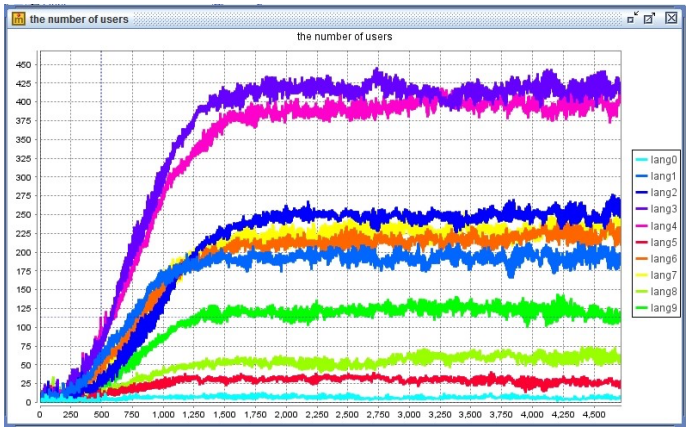

Fig. 4. 周りを見て言語を選ぶ割合:0.1 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.2

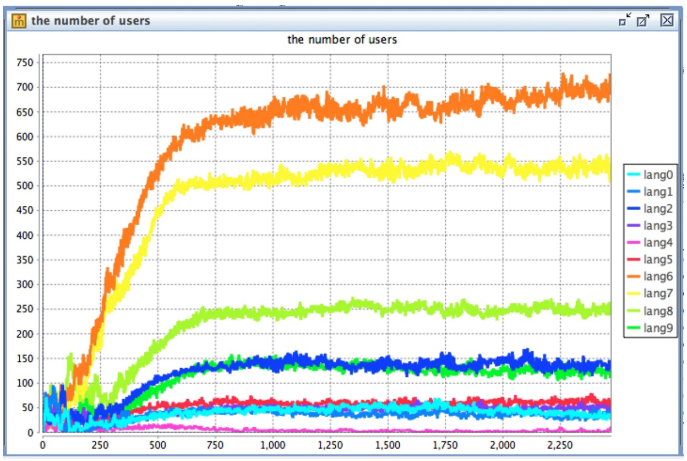

Fig. 5. 周りを見て言語を選ぶ割合:0.2 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.3

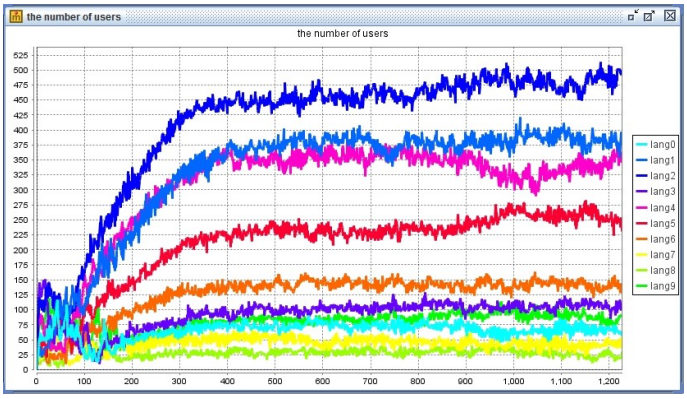

Fig. 6. 周りを見て言語を選ぶ割合:0.3 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.4

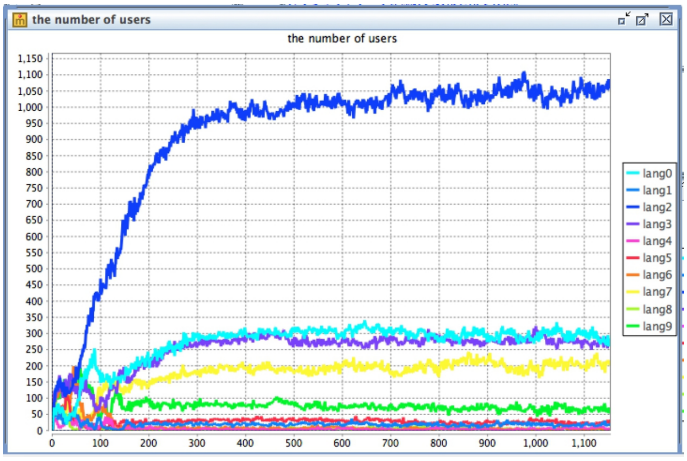

Fig. 7. 周りを見て言語を選ぶ割合:0.4 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.5

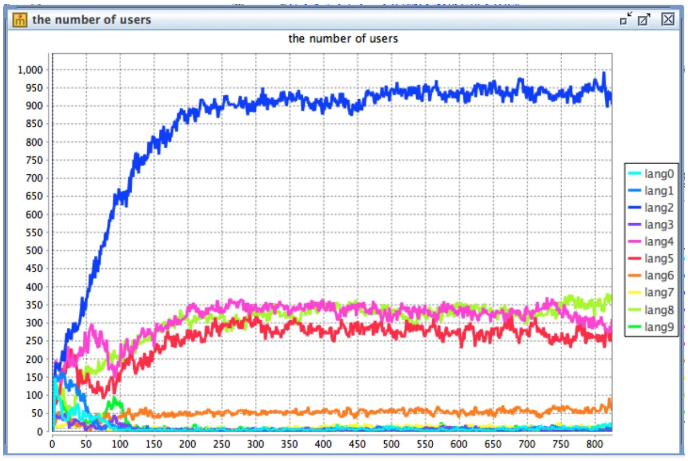

Fig. 8. 周りを見て言語を選ぶ割合:0.5 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.6

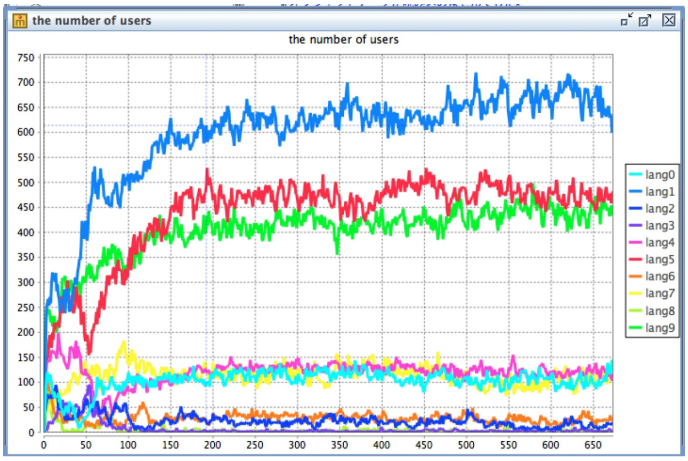

Fig. 9. 周りを見て言語を選ぶ割合:0.6 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.7

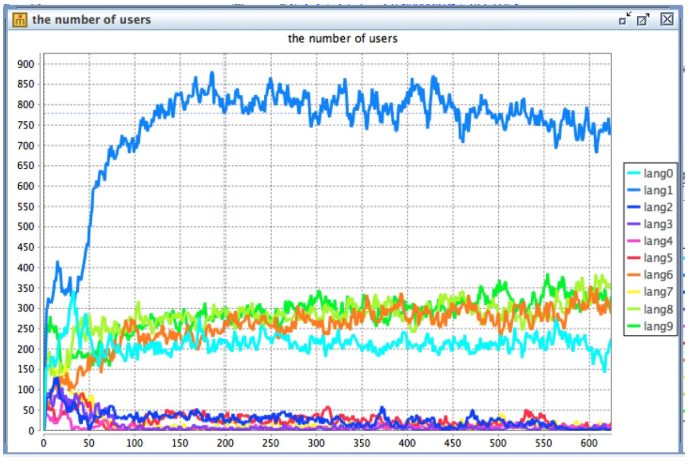

Fig. 10. 周りを見て言語を選ぶ割合:0.7 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.8

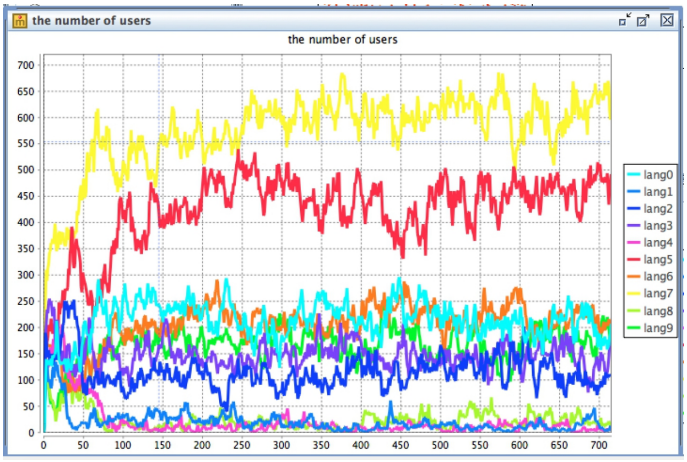

Fig. 11. 周りを見て言語を選ぶ割合:0.8 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.9

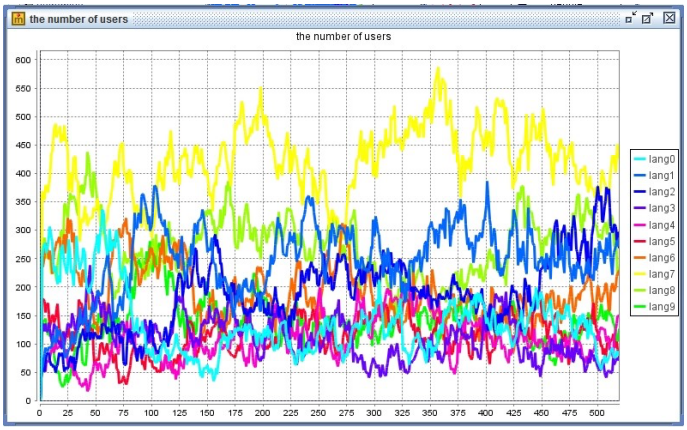

Fig. 12. 周りを見て言語を選ぶ割合:0.9 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:1.0

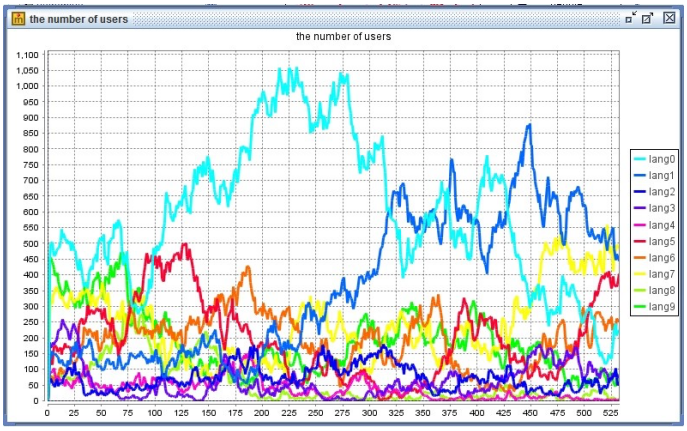

Fig. 13. 周りを見て言語を選ぶ割合:1.0 の時の自然言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

各条件で 5 回ずつ試行したところ，おおよそ周りを見て言語を選ぶ割合は 0.6 以下とわかった．ただ，これでは周りを見て言語を選ぶ割合の値が分からないので，首位言語のシェアでフィッティングを行うことにしたが，以下に結果を示す通り，明確な相関を見出せなかったため，この方法ではできなかった．

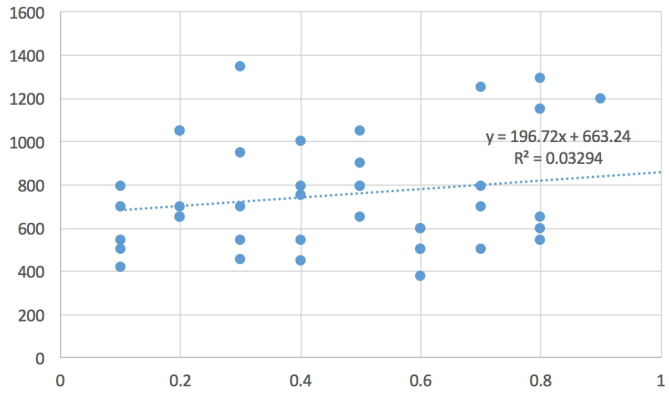

Fig. 14. 周りを見て言語を選ぶ割合と首位言語の使用者数．周りを見て言語を選ぶ割合が各値ごとに各 5 回試行し，そのうち，首位言語の陥落が見られなかったものについて，首位言語のおおよその使用者数を 50 人単位で概算した．

B. プログラム言語の結果

周りを見て言語を選ぶ割合が 1.0 でない場合はどれもブレイクアウトを起こした．そのため，ここでは周りを見て言語を選ぶ割合が 1.0 の時と周りを見て言語を選ぶ割合が 0.9 の時の結果のみを掲載する．

- 周りを見て言語を選ぶ割合:1.0

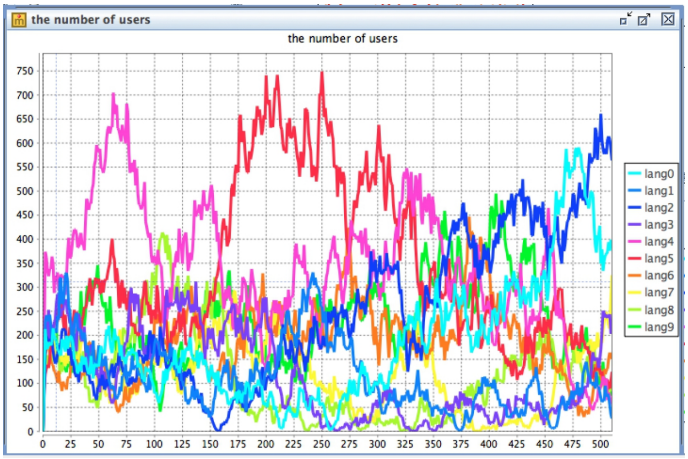

Fig. 15. 周りを見て言語を選ぶ割合:1.0 の時のプログラミング言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.9 及び親和度:0.1

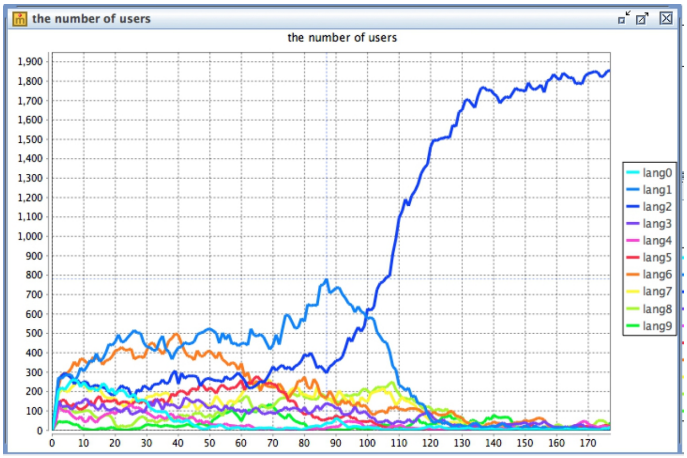

Fig. 16. 周りを見て言語を選ぶ割合:0.9 及び親和度:0.1 の時のプログラミング言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

- 周りを見て言語を選ぶ割合:0.9 及び親和度:0.5

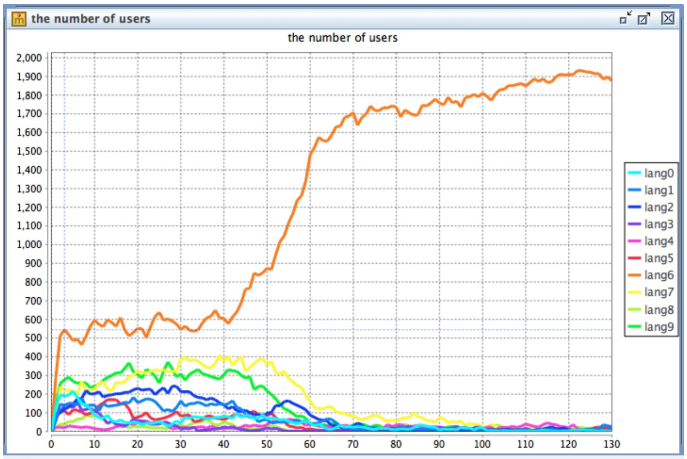

Fig. 17. 周りを見て言語を選ぶ割合:0.9 及び親和度:0.5 の時のプログラミング言語のユーザー数のシミュレーション結果 (試行回数 1 回目)

IV. 考察

自然言語における周りを見て言語を選ぶ割合は具体的に決定できなかったものの，およそ 0.6 以下である一方，プログラミング言語においてはおよそ 1 だとわかった．また，プログラミング言語において，親和度は直接的に影響はなかった．つまり，自然言語とプログラミング言語における違いは土地拘束の存在以外には周りを見て言語を選ぶ割合しかないということになる．では，周りを見て言語を選ぶ割合の違いはどこから来るだろうか．自然言語に比べ，プログラミング言語が言語間の親和度が大きいことは推察できるが，それが直接影響しないという結果が出たものの，周りを見て言語を選ぶ割合を通して間接的に影響しているのではないだろうか．それぞれの言語の親和度が高くなると，人々が使用言語を変更することがたやすくなると考えればつじつまが合う．

また，プログラミング言語において，周りを見て言語を決める確率が低い場合，ブレイクアウトが起きやすい傾向が見受けられたが，そうであっても自然言語においてはブレイクアウトしていないのは土地拘束の影響が大きいと考えられる．これらに基づくと，自然言語は人々の移動距離が大きくなるとブレイクアウトしていき，言語統一が起こるかもしれないが，居住地という概念があるうちは極めて厳しいと考えられる．オンラインなのどの仮想空間が広がれば統一言語ができるかもしれない．

今後の研究としては，自然言語における周りを見て言語を選ぶ割合の検討が考えられる．今回は首位言語のシェアで検討しようとしたが，周りを見て言語を選ぶ割合を変化させて，明らかに現れる変化は言語が空間全体に浸透するまでの step 数の増加である．これをうまく使えば周りを見て言語を選ぶ割合を決めることができるかもしれない．

また，プログラミング言語は今回，完全に土地拘束のないものとしたが，実際にはインターネット上にもコミュニティーが存在するわけで，完全にないとみなすことはできない．このような要素が抜けてしまっているためにプログラミング言語における周りを見て言語を選ぶ割合が 1 になってしまっている可能性が高い．ただ，これは土地拘束が完全にあるとみなした自然言語において，首位言語の陥落が頻繁に観測されたのは周りを見て言語を選ぶ割合が 0.9 以上の場合であったことを鑑みれば，プログラミング言語の周りを見て言語を選ぶ割合は極めて高いと考えることは間違いないと思われる．

謝辞

このシミュレーションは先生，TA さんの指導・協力により行うことができた．ここで改めて感謝の意を表したい．

REFERENCES

[1] Cambridge University, The Cambridge factfinder, Cambridge [England] ; New York : Cambridge University Press, 1994.
[2] Time, INC., TIME Almanac 2013, ENCYCLOPAEDIA BRITANNICA, INC., 2013, p.502.
[3] ”Language Trends on GitHub” [online]. Available:https://github.com/blog/2047-language-trends-on-github. [2015/10/24 Access]
[4] ”TIOBE Software: The Coding Standards Company” [online]. Available:http://www.tiobe.com/index.php/content/paperinfo/tpci/. [2015/10/24 Access]