プログラミング学習環境における能動的推論の活用とその評価

北島茂樹[*1]・山中脩也[*2]・長慎也[*2]・今野貴之[*1]・中川 智之[*3]
Email: shigeki.kitajima@meisei-u.ac.jp

*1：明星大学教育学部教育学科
*2：明星大学情報学部情報学科
*3：明星大学データサイエンス学環

◎Key Words　　プログラミング学習環境，自由エネルギー原理，能動的推論

1. はじめに

本研究グループでは，自由エネルギー原理（free-energy principle，FEP）に基づくプログラミング学習環境を提案し，多様な学習環境で実践を行ってきた(1)。明星大学教育学部も，2020 年度から「コンピュータ概論」の受講者に対し，同様のプログラミング学習環境をオンラインで実施し，その評価を行ってきた(2)。

2021 年度の実践では，形成的アセスメントにおける評価ツールであるパフォーマンス・クライテリアを導入するなど，学生も評価活動に参画できるようにした。その結果，学生の取り組みは質・量ともに向上し，パフォーマンス・クライテリアには自律的にプログラミングを学ぶ際に，学びに向かわせる役割や学びを支える役割，他者との学びをつなぐ役割があることが明らかになった(3)。

また，本研究グループは，プログラミング等の演習過程で理解したことなどを言語化し他者と共有する活動を支援するための Bot を開発し，2022 年度から明星大学情報学部の「プログラミング演習 3」に導入してきた(4)。その結果，学習者が能動的推論（active inference）をより自律的に行えるようになった。

明星大学教育学部でも，パフォーマンス・クライテリアをもとに，成功したパフォーマンスを示すチェックリストであるプロダクト・クライテリアを開発し，2022 年度の「コンピュータ概論」から導入してきた。そこで，本研究では，同環境の実践において収集された多様なデータログを分析し，その結果について考察を行う。

2. 能動的推論の活用したプログラミング学習環境

2.1 自由エネルギー原理と能動的推論

自由エネルギー原理では，脳はあらゆる感覚の予測器であると捉え，入力された感覚信号に基づき，外環境や内環境の状態について，学習を通じてベイズ最適な（Bayes optimal）推論を行っていると考える(5)。また，自由エネルギーとは，Helmholtz が考え出した概念であり，あるシステムに対して人間が自由に使えるエネルギーのことである(6)。それは，自由エネルギー最小化原理ともいうべきものであり，自己組織化されたシステムが環境内で平行状態であるためには，システムの自由エネルギーを最小化しなくてはならず，それは，システムが無秩序へ向かう自然な傾向に抗って持続的に存在するための必要条件であるという(7)。

そのため，主体（Agent）は外部世界に関する生成モデルと現在の推測から計算される自由エネルギーを最小化するために，現在の推測を変える無意識的推論と，行動によって感覚入力を変える能動的推論とを組み合わせている(8)。そして，自由エネルギーを下げるために，現在の推測を変えて，生成モデルを一致させていき，正しい認識として一致したときに 0 となり，それが最小となるのである。つまり，自由エネルギー原理の本体は，能動的推論を含む，能動的なプロセスにあるのである(8)。

ここで，能動的推論を，期待する感覚入力に合うように周囲の環境を予測しやすい状態に変化させる行動制御・意識決定とし，「行動」と「知覚」の循環的因果性のもとで外部世界を認知していくことを「学習」と捉え，本研究におけるプログラミング学習環境を図式したものが図 1 となる(1)。また，主体（Agent）は学習者であり，左の外部世界との循環的因果性を「言語処理系」，右の外部世界との循環的因果性を「主体以外の Agent」とする。

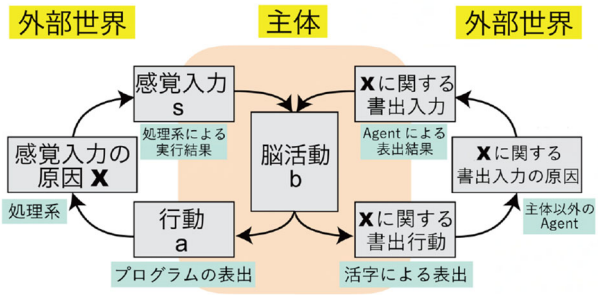

図 1 本研究におけるプログラミング学習環境

こうしたプログラミング学習環境が成り立つ前提条件として，「言語処理系」に対して主体が「行動」を起こすことができ，そこから得られた結果を感覚入力（知覚）できるとともに，「主体以外の Agent」に対して主体が「書出行動」を起こすことができ，また，主体以外の Agent の書出を「書出入力」（知覚）できることが必要となる。そして，主体は，それぞれの行動と知覚をいつでも起こすことができ，行動を起こすと知覚できる，という順序が重要となる。つまり、本研究におけるプログラミング学習環境は，成立のために必要となる前提条件を満たし，その順序をシステムと資料が担保するように構成されているのである(1)。

2.2 コンピュータ概論における学習環境とその実践

明星大学教育学部の「コンピュータ概論」のプログラミング学習環境は，Bit Arrow，作問システム，Slack，LMS（manaba）によって構成される。

Bit Arrow とは，Web ブラウザ上で動作可能なプログラミング学習環境のためのシステムであり，Python や C 言語など複数の言語に対応している。そのため，図 1 左の「言語処理系」を担っている。また，サーバには学生の実行ログが保存されており，授業者は図 2 のように，管理画面から学生ひとりひとりの実行状況や実行回数を確認することができる。さらに，ログから次の兆候を読み取ることができる[9]。

- エラーが多発している
- 手が止まっている
- 課題の進捗が遅れている
- 1 つの課題に時間をかけすぎている
- やみくもにプログラムを書き換えている

図 2 Bit Arrow の実行ログ

学生は，LMS 上で配布された資料（図 3 及び図 4）について，Bit Arrow を用いて探究を行うのであるが，その際，C.S.Peirce の三つの推論形式を学生が何度も繰り返しながらプログラミングを学んでいくのである[10]。

- 仮説形成：現象の観察と既知の規則から拡張的推論を行い，仮説を形成する。
- 演繹：既知の規則と仮説から必然的推論を行い，必然的帰結として実行結果を導く。
- 帰納：仮説と実行結果から検証的推論を行い，新たな規則を導く。

図 3 資料における課題（体験して試そう）の例

演繹において，学生は Bit Arrow にソースコードを入力し実行結果を得る。そして，学生はその結果について現象の観察を行い，これまで獲得してきた既知の規則をもとに，仮説としてソースコードを作成する。こうした仮説形成と演繹を繰り返すことでソースコードは検証され，正当化されていく。それが帰納となり新たな規則が導かれるのである。そのため，仮に思い付きでソースコードを作成したとしても，コンパイルエラーの表示を含めた，現象の観察を繰り返すことで，学生は妥当な仮説を形成していくのである。このように，このプロセスが図 1 左の「言語処理系」における循環的因果性における活動に相等する。

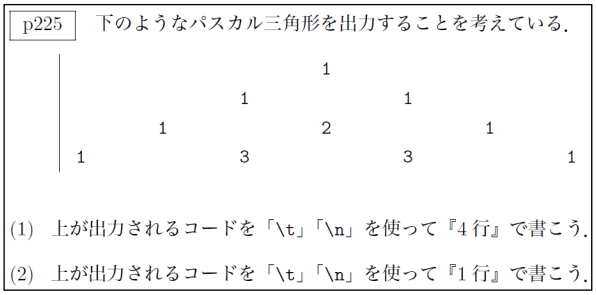

図 4 資料における課題（組み合わせて試そう）の例

コンピュータ概論では，情報学部のプログラミング演習の授業のように，bot を導入していないため，LMS において学習の進度に合わせて学生に資料を提供している。そこで，「言語処理系」における活動において，学生が発見した「事実」について，学生が各自で保有している「事実」を共有するために，「lookbacks」という共有用チャンネルを Slack 上に資料の教材ごとに作成している[2]。また，「事実」だけでなく，それらを含む，探究過程で導出された個々の「気づき」を，図 5 のように書き出させている[3]。その際，課題の解法については lookbacks への記入を禁じており，これまでの実践では，結果としてそれを記入した学生はいなかった。このように，このプロセスが図 1 右の「主体以外の Agent」に対して主体が行うる循環的因果性における活動に相等する。

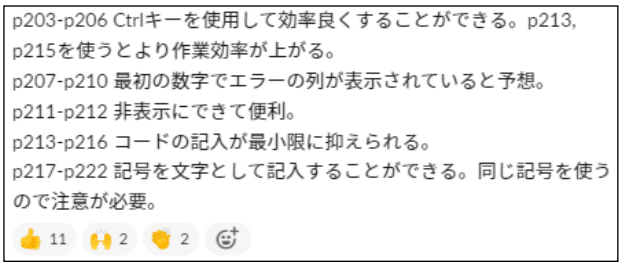

図 5 Slack への書き出しとフィードバック

作問システムとは，Web ブラウザ上で動作し，作成された問題（クイズ）と正解の登録，解答者の自動割り当て，正誤判定などの機能を実装したシステムである[11]。授業中，まず，学生は事前に準備してきた問題（クイズ）とその正解を，図 6 のように登録する。次に，個人で問題（クイズ）を解く場面で，他の学生が作成した問題（クイズ）がランダムに割り当てられ，それらに解答する。最後に，正誤が示され，出題者に対して問題（クイズ）についてのコメントや誤りの指摘，別解の提案など，相互フィードバックを行うのである。

作問システムを通して学生には，解答スコアと出題スコアが与えられる。解答スコアは，正解であれば「score=1」であり，正解でなければ「score=0」となる。また，出題スコアは，正解率（正解数/有効解答数）に応じて変動し，2/3 に近づくほど高いスコアが与えられるため，難し過ぎても，易しすぎてもスコアは低くなる。そのため，学生

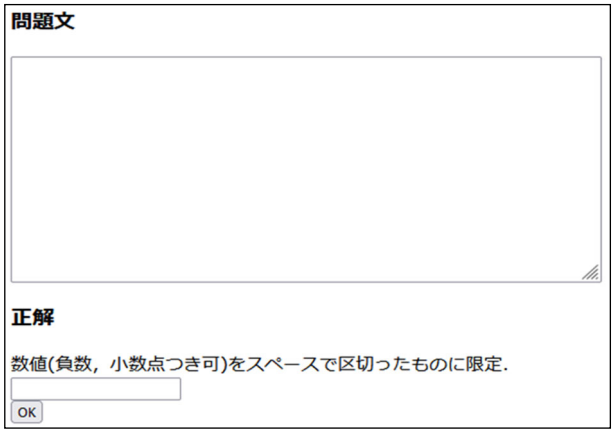

図 6 作問システムへの問題（クイズ）の登録

は適度な難易度の問題 (クイズ) を考案する必要がある。

これらの活動をあわせて，コンピュータ概論における実践は，図 7 のように，「同期の学び」の S1～S3 と，「非同期の学び」の A1～A3 からなり，それぞれ，括弧内に示したツールを使用して行う。

［同期の学び］
S1：書き出しに対し相互フィードバックを行う（Slack）
S2：作成した問題（クイズ）を解き合う（作問システム）
S3：コードを共有し実行して確かめる（Bit Arrow）
［非同期の学び］
A1：コードを書き実行して確かめていく（Bit Arrow）
A2：気づいたことを書き出す（Slack へ）
A3：問題（クイズ）を作る（作問システムへ）

その際，学生は図 2 のサイクルで一連の活動を行い，まず LMS からダウンロードした教材をもとに A1～A3 を個々に行うことから始め，次時で授業中に S1～S3 をそれぞれ同期しながら行うことで一巡する。

そのため，最初の非同期の学びは第 1 回から第 2 回の授業までとなる。非同期の学びは，個々に A1 で課題（体験して試そう）に取り組むことから始まり，その取り組みの中で気づいたことなどを Slack に書き出し，それが A2 となる。それらを踏まえて，問題（クイズ）を作成し，それが A3 となる。次に，A1 で課題 B に取り組み始めることで，次のサイクルが始まる[3]。

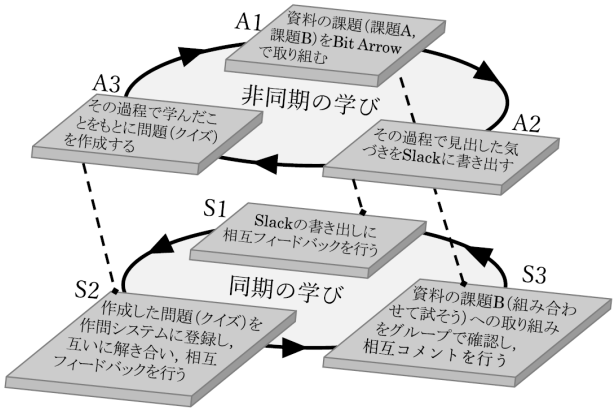

図 7 同期の学び・非同期の学び

図 7 では，A1 の取り組みが S3 の活動の前提となる。そして，A2 で書き出された気づきに対し S1 で相互フィードバックを行う。また，A3 で作成した問題（クイズ）について S2 で互いに解き合い相互フィードバックを行う。そのため，それらを点線で結び表している[3]。

授業は，S1 の活動から始まり，開始のタイミングを同期させて相互フィードバックを行い，授業者はそれに対するフィードバックを適宜行う。次に，S2 では，授業者は，問題（クイズ）の登録・確認と出題・解き合い・答え合わせと相互フィードバックについて，作問システム側でモードを切り替えていき，それぞれについて開始のタイミングを同期する。S3 では，開始のタイミングを同期し，課題 B についてアイディアの共有や相互コメントを行う。そして，次時は S1 の活動から始まるのである[3]。

3. データログの分析とその結果

サーバに残された Bit Arrow のログから，2020 年度と 2021 年度について学生の実行回数を算出したものが表 1 であり，2022 年度と 2023 年度について学生の実行回数を算出したものが表 2 である。

表 1 2020 年度と 2021 年度のプログラムの実行回数

	2020 年度	2021 年度
全員の総実行回数	39,319	48,040
コンパイルエラーとなった実行の割合	.29	.23
一人の平均実行回数	983.0	1455.8
授業時間内	386.2	408.5
授業時間外	596.8	1047.3

表 2 2022 年度と 2023 年度のプログラムの実行回数

	2022 年度	2023 年度
全員の総実行回数	53,711	74,039
コンパイルエラーとなった実行の割合	.28	.26
一人の平均実行回数	1451.6	1805.8
授業時間内	237.4	186.9
授業時間外	1214.2	1618.9

ただし，2020 年度と 2021 年度では，授業回数が異なるため，回数について平均を求めて比較すると， 授業時間内における一人の平均実行回数は，2020 年度が 32.2 回，2021 年度が 31.4 回である。それに対し，授業時間外では，2020 年度が 49.7 回，2021 年度が 80.6 回である。授業時間外は少なくとも非同期の学びであるため，2020 年度に比べ 2021 年度は，非同期の学びにおける学生のプログラミングの実行回数は向上している。なお，2021 年度と 2022 年度，2023 年度の授業回数は同じである。

2020 年度の実践に対し，2021 年度の実践ではパフォーマンス・クライテリアが導入されている。学生は表 3 のようなパフォーマンス・クライテリアを作成することや，それを用いて自らのパフォーマンスを自己評価するなど，評価活動に参画することを通して，自己調整学習を行っているといえる[3]。また，2022 年度の実践では，パフォーマンス・クライテリアをもとに，成功したパフォーマンスを示すチェックリストであるプロダクト・クライテリアが表 4 のように導入された。なお，表 4 のチェックリストにおいて，A111 から A113 は「コードを書き実行して

表3 作成されたパフォーマンス・クライテリアの例

A1：コードを書き実行して確かめていく（Bit Arrow）
● やり方が複数あるので，いろいろな方法で実践してみる。 ● どうしてそのコードで出力できるのか，コードを読み取る。 ● 気づいたこと，疑問に思ったことをメモしておく。
A2：気づいたことを書き出す（Slack へ）
● 1問ごとに気づきを書き出すのではなく，その回全体を通しての気づきやどのような仕組みになっているのかを書き込む。 ● 他の人の気づきを見ない。 ● 自分が難しいと思った問題を共有し，何が難しかったのかを明確にする。
A3：問題（クイズ）を作る（作問システムへ）
● 新しく発見したことを問題に取り入れてみる。 ● Bit Arrow の作成時に書いていたメモを基に，自分がつまずいた箇所を復習できるような問題も作ってみる。

表4 開発されたチェックリスト

A111：既習などから自分なりに仮説を立てコードを書き，エラーを恐れず実行しながら考える。 A112：正しく出力されなかった場合，エラーの原因を探り，都度修正し実行して確認するなど，エラーから学ぶ。 A113：結果ではなく過程から学べるよう，正しく出力された場合も，たまたま正しく出力された可能性もあるため，部分的にコードを変更して試してみる。
A121：気づいたことや疑問に思ったことをメモしておき，観たことと気づいたことの違いを明確にし，言語化し，整理する。 A122：解答ではなくヒントとなるよう工夫し，発見したことは間違っている可能性があっても積極的に書き出すようにする。 A123：他者の書き出しを見る前に自分の考えをまとめ，その回の内容を振り返り，つながりや仕組みを書き出すようにする。
A131：コードを実行する中で迷ったり間違えそうだと感じたりした箇所や面白かったことをメモしておき，作問に生かす。 A132：直前に学んだ内容だけでなく既習やこれまでの正答率，コメントを活かし，正答率 2/3 になるよう工夫して作問する。 A133：この問題で何に気づいて欲しいのか，用途や意味を意識し，当てずっぽうでは解けないような問題を作るようにする。

確かめる（BitArrow）」についてのものであり，A121 から A123 は「気づいたことを書き出す（Slack）」についてのものである。A131 から A133 は「学んだことをもとに問題作りをする（作問システム）」についてのものである。

パフォーマンス・クライテリアやプロダクト・クライテリアは，ルーブリックのようにレベルのあるチェックリストではないため，学生にとって使いやすかったようである。そのため，2022 年度や 2023 年度の実践においても，学生のプログラミングの実行回数は 2021 年度と比べて向上しており，特に，非同期の学びである授業時間外におけるプログラミングの実行回数が増加する傾向にある。

そこで，2023 年度の A111 から A133 までのチェックリストと学生個々人のプログラムの実行回数，コンパイルエラーとなった実行の割合，作問システムにおける解答スコアと出題スコアについて，欠測値を多重代入法で補完したのちに総和をとり，偏相関係数の推定値を求めた。その結果，チェックリスト間に有意な正の相関があった。

4. おわりに

能動的推論を活用したプログラミング学習環境において，学生がチェックリストを用いた評価活動に参画しながらプログラミングを学習することで，非同期の学びにおけるパフォーマンスが向上することが明らかになった。また，パフォーマンスを向上させる可能性のあるクライテリアについて検証していくことが今後の課題である。

参考文献

(1) 鷹觜莉子，山中脩也，長慎也，北島茂樹，丸山農：“自由エネルギー原理に従う循環的因果性の構成を支援するプログラミング学習環境の構築”，明星大学研究紀要. 情報学部，12，pp.21-30（2022）.

(2) 北島茂樹，山中脩也，長慎也，今野貴之：“オンラインプログラミング演習環境における対話の実践と評価”，コンピュータ＆エデュケーション，50，pp.40-43（2021）.

(3) 北島茂樹：“オンラインプログラミング演習環境におけるパフォーマンス・クライテリアの役割”，情報教育ジャーナル，4(2)，pp.17-31（2022）.

(4) 長慎也，山中脩也，北島茂樹，今野貴之：“学習者による書き出しを支援する SlackBot の開発と運用”，情報教育シンポジウム論文集，2023，pp.175-182.（2023）.

(5) 乾敏郎：“自由エネルギー原理”，認知科学，26(3)，pp.366-386，（2019）.

(6) Friston, K., Kilner, J., & Harrison, L. A free energy principle for the brain. *Journal of Physiology-Paris*, 100, pp.70-87 (2006).

(7) Friston, K. The free-energy principle: a unified brain theory? Nature Reviews Neuroscience, 11, pp.127–138 (2010).

(8) 吉田正俊，田口茂：“自由エネルギー原理と視覚的意識”，日本神経回路学会誌，25(3)，pp.53-70，（2018）.

(9) 長信也，長島和平，兼宗進，並木美太郎：“チュートリアル：Bit Arrow を用いた，つまずいている学習者の探し方”，情報処理学会研究報告，Vol.2022-CE-165, No.2, pp.1-8 (2022).

(10) 山中脩也，北島茂樹，長慎也，今野貴之：“コンピュータを用いた C.S. Peirce の探究過程の実現”，しごと能力研究 2020 特集号，pp.71-83（2020）.

(11) 長慎也，山中脩也，北島 茂樹，今野貴之：“情報系初年次のプログラミング演習における，コンピュータとの対話を重視したコースデザインと支援システム”，研究報告教育学習支援情報システム（CLE），2019-CE-152(20)，pp.1-9（2019）.