NPR(Non-Programming Resource)とプログラミングをつなぐ取り組み
ードリトルによるタートルアルゴリズムー

青木 浩幸[†] ウラル ハリット[†] 崔 淑敬[†]
井戸坂 幸男[††] 兼宗 進[†††] 李 元揆[†]

情報社会の発展により，情報を専門としない非情報科学系学生への情報教育の重要性が大きくなっている。しかしながら，情報の基本的概念のひとつであるアルゴリズムの学習は，プログラミングによる方法が一般的であり，プログラミング経験のない非情報系の学生が取り組むには容易ではないという問題があった。近年では NPR (Non-Programming Resources) に代表されるような，プログラミングに依らず，カードのような実物の教具を用いた情報科学の学習方法が提案され，注目されている。本研究では，プログラミングに NPR のアイデアを導入することにより，アルゴリズムの学習を効果的に進めるための方法を提案し，タートルアルゴリズムと名付けた。評価は大学の非情報科学系学生対象のアルゴリズムの授業において行った。

The Activities Connecting between NPR and Programming
- Turtle Algorithm with Dolittle -

Hiroyuki Aoki[†] Halit Vural[†] SookKyoung Choi[†]
Yukio Idosaka[††] Susumu Kanemune[†††] WonGyu Lee[†]

Improving of information society makes information education for non-informatics students more important. One of basic concept of informatics is algorithm. When students learn algorithm, programming is mostly used. However it is challenging for non-informatics students having no programming experience.

Recently methods like the Non-Programming Resource (NPR) are suggested and taken notices. It is an educational resource to aim to teach informatics without programming but with materials such as cards as real things. This study proposes the method to make algorithm education effective with adding the idea of NPR to programming. It is named as Turtle Algorithm. The evaluation is taken at the algorithm classes for non-informatics university students.

1. はじめに

情報教育の広がりにより，情報を専門としない非情報科学系の学生にもアルゴリズムをはじめとする情報科学の基礎を学ぶ機会が増えてきた。アルゴリズムを学習するには従来プログラミングによる方法が一般的であったが，非情報科学系学生はプログラミングの経験がないため，従来とは異なる学習方法が求められている。

そのような中，Non-Programming Resources[1]（以下 NPR と記す）や Computer Science Unplugged[2]（以下 Unplugged と記す）に代表される，コンピュータを用いずにカードのような実物の教具を用いた情報科学の学習手法が注目されている。Unplugged は子どもを主な対象としているが，NPR は情報教育全般を広く対象としているため，本稿ではこれらをまとめて NPR と総称する。

本研究では，コンピュータの中でアルゴリズムがどのように役立っているかということを，プログラミングの体験を通して理解する教育を目指している。そこで我々は，プログラミングによるアルゴリズム学習に NPR の長所を取り入れ，プログラミング初心者でも取り組みやすくすることを考えた。本稿では２つの試みを紹介する。

これらの試みは，韓国高麗大学における非情報科学系の学生対象の「データで表現する世界」という授業の中で評価を行った。

2. NPR の利点とプログラミングへの統合

NPR は，2000 年にフィンランドで開催された ACM の ITiCSE 会議において提唱された，プログラムを使わない教育資源の活動である。これまでプログラミングなどの技術やスキルに偏りがちであった情報教育に対して，より本質的な問題解決能力や分析，設計，

† 高麗大学
Korea University, Seroul, Korea
†† 松阪市立飯南中学校
Iinan Junior High School, Matsusaka, Japan
††† 一橋大学
Hitotsubashi University, Tokyo, Japan

アルゴリズムに目を向けるべきであるとする考え方に基づいている。そのような学習を行うために，プログラミングを行う前に，紙に書くことや実物の道具による教材（=Non-Programming Resource）を用いることの有用性を提唱している。

ニュージーランドで開発された Unplugged は，コンピュータを使わなくても（プラグを抜く=Unplugged），小中学生の子どもが遊びながら情報科学を学べる可能性を示した。この取り組みは世界各国に広がりつつあり，日本でもテキストの翻訳が刊行された [3]。Unplugged は次のような特徴を持っている。

- ゲームを使い，楽しみながら学べる。
- 体を動かし，体験を通して学べる。
- グループで協力して問題を解決できる。

アルゴリズムを学習する過程では，「与えられた制約の中で解決方法を論理的に組み立てる」という考え方を理解することが重要である。制約の例としては，「比較によって大小を判定しなければならない」，「同時に比較できるのは2つの値だけである」といった，限定された「利用できる操作」がある。制約が緩すぎると論理的な思考を育てることができず，強すぎると課題に取り組むことができなくなるので，適切な制約を設定することが重要である。Unplugged はその制約を「ゲームのルール」として必然性をもって与え，学習者にパズルを解く感覚で論理的に考えさせることに成功している。

一方，プログラミングによりアルゴリズムを学ぶ従来の学習では，学習者はアルゴリズムの制約に加え，「2 つの値を交換するために一時的な変数を導入する」，「配列というデータ構造にインデックスでアクセスする」といった，プログラミング言語の制約を体験する。その結果，不要な制約の習得に時間と労力を費やされてしまい，本来のアルゴリズムの問題解決に集中できなくなるという問題が存在した。

そこで本研究では，プログラミングに NPR の考え方を取り入れることにより，プログラミングにおける不要な制約を軽減し，アルゴリズムをより理解しやすく提示することを試みた。

また，教育用プログラミング言語「ドリトル」[4]を採用することで，プログラミング言語の学習を短時間で行い，授業の本来の目的であるアルゴリズムの学習に注力できるようにした。ドリトルは日本で開発されたオブジェクト指向プログラミング言語であり，日本語や韓国語など母国語で記述できる特徴がある。

3. アルゴリズム可視化

3.1 メタファによるアルゴリズム表現

データの並べ替えのようなアルゴリズムにおいて，プログラム中の変数の値を見るだけではアルゴリズムの動作を理解することは容易ではない。個々の変数の値だけでは，コンピュータが行っている処理の全体像をイメージしにくいためである。

NPR では，値を並び替えるために，数値を書き込んだ実際のカードを用意し，それらを手で移動させて並べ替える学習を行う。そこでプログラムにおいてもデータを画面上にオブジェクトとして表示し，それを視覚的に表示しながら動作させることを考えてみる。

図1は，5台の「カートレーサー」を画面に表示し，それらを値によって並び替える画面である。

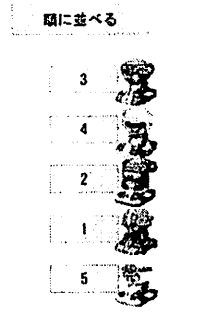

図 1 「カートレーサー」の並び替え

画面上のオブジェクトを並び替えるためには，値によって並び替えるアルゴリズムのプログラムに加え，それらの動きを画面上に表示するプログラムが必要になる。しかし，アルゴリズムのプログラムを記述することは学習の目的であるが，その動きを画面上で表現するプログラムを書くことは学習の目的ではない。

そこで本研究では，学習者が値を並べ替えるアルゴリズムを記述するだけで，並べ替えの様子が画面で視覚的に確認できるフレームワークを実現した。

このフレームワークの実現には，オブジェクト指向の継承を利用している。ドリトルはオブジェクト指向言語であるため，配列のメソッドをオーバーライドすることで，配列操作に画面上のオブジェクトの移動を反映するようにした。この仕組みは，プログラムの実行前に自動実行される startup.ini という初期化ファイルに定義した。学習者はこのフレームワークを意識せずに，アルゴリズムの処理だけを考えたプログラムを作成することができる。

図2に交換法（バブルソート）のプログラム例を示す。これは授業で使用したサンプルプログラムを翻訳したものである。

```
外回数= 1。
「
  内回数= 1。
  「
    走者= 走者リスト!（内回数）見る。
    次走者= 走者リスト!（内回数＋1）見る。

    「走者:値 ＞ 次走者:値」！ なら「
      走者リスト!（内回数＋1）（走者）上書き。
      走者リスト!（内回数）　（次走者）上書き。
    」実行。

    内回数= 内回数＋1。
  」！（5−外回数）繰り返す。

  外回数= 外回数＋1。
」！（5−1）繰り返す。
```

図 2 ドリトルによる交換法のプログラム

このプログラム中で“走者リスト”は配列であり，“**走者リスト!（番号）見る。**”で番号がインデックスの値を取り出す。“**走者リスト!（番号）（値）書く。**”で，番号の位置に値を格納し直す。

本プログラムでは走者を並び替えることが目的であるため，“走者リスト”配列には走者となるタートルオブジェクトを格納した。走者が持っている並べ替えに用いる値は，各走者の“値”というプロパティに格納されており，“走者：値”の形で値を取り出すことができる。

フレームワークを実現するために，配列を継承した可視化配列を定義している。そして，配列オブジェクトの以下のメソッドをオーバーライドすることで配列操作を画面上のオブジェクトであるメタファの動きとして可視化した。主な変更を次に示す。

(1) “入れる”メソッドのオーバーライド
　配列に要素を追加するメソッドである。要素の“値”プロパティの比較演算子を(3)で説明する機能のためにオーバーライドする。

(2) “上書き”メソッドのオーバーライド
　配列の要素を配列内で移動させるためのメソッドである。指定されたインデックスに新しい要素を代入するとともに，要素のオブジェクトを画面上で新しい位置へ移動するアニメーションを表示する。

(3) 配列要素に付加された“値”プロパティに対する比較演算子のオーバーライド
　比較結果が真か偽かを返す演算子である。比較の動作を表示するため，比較対象になっている要素のオブジェクトの隣に，それぞれの要素の“値”プロパティを表示し，一定時間待機する。

このフレームワークを用いることで，学習者は画面上のオブジェクトの操作を意識することなく，アルゴリズムの記述に集中することができる。実際，可視化配列を通常の配列に置き換えるだけで，オブジェクトを画面に表示させず，値の並び替えだけを行うことが可能である。これはアルゴリズムを理解した上で，データ数を増やして並べ替えのコストを比較する学習時に特に有効である。

アルゴリズムの動作を画面上のオブジェクトで表現したことで，配列のインデックスを間違えるといったプログラム上の間違いが，起きた時点で視覚的に確認できるようになった。これは，プログラムの実行完了後に結果を見てから原因を探すことに比べ，誤りの発見を容易にする利点があった。

3.2 アルゴリズム処理経過の記録

3.1 で述べたフレームワークにより，データの比較やデータ交換の操作のアルゴリズムの処理を画面上で見せることができるようになったが，見せただけでは学習者が理解できたことにはならない。理解できる速さは人によってまちまちであること，分かっていない学習者にとってアニメーションは流れていくだけで，何も印象に残らないこともあるからである。着実な理解を図るためには自分自身のペースで考えさせることが必要である。そのため，ただ見せるだけではなく，処理経過の記録を残し，学習者が自分で考察するための材料を提供することを試みた。

3.1 で取り上げたプログラムを発展させ，走者を時間経過とともに左に少しずつ移動させながら，並び替えの処理を含めて軌跡を記録させるようにした。図 3 に実行の様子を示す。あみだくじ状の線で並べ替えの処理が描かれている。

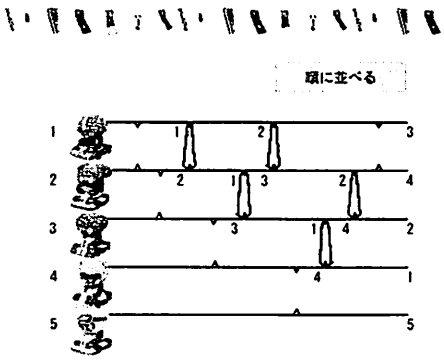

図 3 並び替えの軌跡

このように処理経過を画面に表示することにより並び替えの全体像が見えるようになり，同時に処理のコストについても考察する材料を与えることができた。

処理経過の時系列的な表現は，Unplugged における整列ネットワークの活動に類似している。Unplugged では床に図 4 のような図を描き，子どもたちがデータの役になって矢印の上を歩き，丸印で出会った場合，小さい方が左へ，大きい方が右へ分かれることで，並べ替えが行われる。

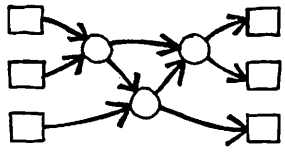

図 4 Unplugged における整列ネットワーク

学習者はこのようなネットワークを見ながら必ず正しい結果が得られるか，無駄のない処理であるかを考える。また，発展させてより大きなネットワークを自分で作る活動を行ったりする。

4. タートルアルゴリズム

初等中等のプログラミングでは以前から Logo によるタートルグラフィックスの実践が知られている[5]。この学習では絵を描く前に学習者自身が画面上のカメになった感覚を持ち，どのように画面上を動いたらよいかという視点でプログラムを作成する。その結果，単に離れた場所からカメに命令して動かすより，プログラムの動作を主体的に理解しやすくする効果があった。

また，小学校におけるドリトルの実践では[6]，いきなり手順に従って動くプログラムを組ませるのではなく，最初は画面上に個々の命令に対応するボタンを配置し，タートルを対話的に操作している。このような体験を通して「命令によりオブジェクトを操作する」という実行モデルや「どのような命令が用意されているか」ということをプログラミングの前段階として学ぶことができる。

これらの方法から，学習者が画面上のデータを検索するカーソルになったつもりでタートルを操作し，対話的に並び替えなどの作業を行う「タートルアルゴリズム」を考案した。

図 5 にタートルアルゴリズムによるプログラムの実行画面を示す。ここでは画面中央に並べられた四角形のカード列で配列を視覚的に表現し，その上を学習者の分身であるタートルオブジェクトが移動することで並び替えに必要な操作を行っていく。

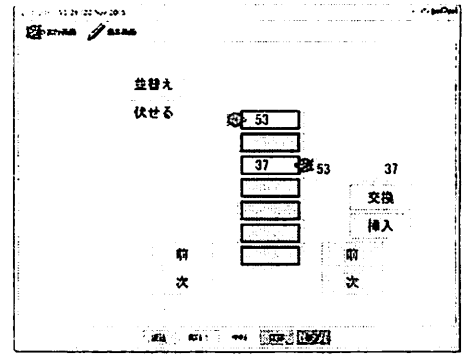

図 5 タートルによる並び替え操作の体験

カードは値が伏せられた状態で並んでいる。カード列の両側には配列のインデックス位置を示すタートルが存在し，タートルの位置にあるカードの値だけを見ることができる。これにより比較する 2 つの値を明示し，同時に他の値は操作対象にできないという制約を表している。

学習者は，「前」ボタンや「次」ボタンを使ってタートルを移動し，必要に応じて「交換」ボタンによって値を交換する操作を行いながら，試行錯誤を通して並び替えのアルゴリズムを学習していく。

学習者は，対話的に確認したアルゴリズムを左上の「並替え」ボタンの中にドリトルのプログラムとして定義することができる。

「並替え」ボタンには，初めはすべてのカードの値を見せる命令だけが定義されているので，並び替えの試行錯誤中に進行具合を見たいときなどに押して確認することができる。

学習者がプログラミングを行う際に用いる命令は，

並び替えに必要な操作そのものである（表 1）。そのうち4つはボタンを用いた操作で慣れ親しんでいる。並び替えアルゴリズムはこれら6つの操作命令と，繰り返し，条件分岐の制御構造だけで記述することができる。使える命令が小さな規模に限られていることで，学習者が感じる負担を小さくすることができる。

表 1 並び替えに必要なメソッド

メソッド	動作
番地	データオブジェクト上の番号を指定することでその位置に移動する。
前	順序的に前のデータ要素の位置に移動する
次	順序的に次のデータ要素の位置に移動する
見る	自分が持っている（自分の位置にある）数値を読み込む。可視化のために一定時間停止する。
交換	2 匹のカメの間で，それぞれが持っている数値を交換する。アニメーションを行う。
挿入	自分が持っている数値を，もう一匹の「カメ」のいる位置に挿入する。アニメーションを行う。

完成したプログラム例を図6に示す。これは選択法のプログラムである。

```
緑カメ！(数値リスト)1 番地。

「|外回数|
  赤カメ！(数値リスト)(外回数＋1)番地。

  「|内回数|
    「(緑カメ！見る) ＞ (赤カメ！見る)」
    ！なら「緑カメ！(赤カメ)交換」実行。

    赤カメ！次。
  」！(7−外回数)繰り返す。

  緑カメ！次。
」！(7−1)繰り返す。

数値リスト！見せる。
```

図 6 タートルアルゴリズムによる選択法

このプログラムでは，配列の役割をするオブジェクト “数値リスト” がカード列を表わしている。配列のインデックスを示す “緑カメ” と “赤カメ” はアルゴリズム操作用に拡張されたタートルオブジェクトである。

5. 授業での評価

5.1 授業概要

今回検討した内容を大学の授業で評価した。授業は韓国高麗大学の「データで表現する世界」という科目であり，その中でアルゴリズムを学習する6時間を担当した。

受講者は非情報科学系の学生で，1 年生が多くプログラミングの経験はほとんどなかった。授業は2クラスあり，学生数はAクラスが43 人，Bクラスが23 人であった。授業は60分で週2コマ行った。授業はウラルハリットが担当し，講義は英語で行われた。

評価の目的は次の2点を調査することである。

- アルゴリズム学習における画面上のカード操作と実物のカード操作の実用性の違い
- 扱うデータの視覚化や画面上でのオブジェクト操作といった，画面内の NPR 的活動がアルゴリズムとプログラミング学習に与える学習効果

5.2 授業展開

以下に授業の流れを示す。また，最終回の授業の終了後に評価のためのアンケートを行った。

1. アルゴリズムの概念の説明。
2. バブルソートのアルゴリズムが描かれたフローチャートを配布し，値の書かれたカードを使ってアルゴリズムを体験的に理解する。A クラスでは紙のカードを，B クラスでは図 5 のコンピュータ画面上のカードを使用した。
3. プログラミングによる処理の実装。3 章で扱ったフレームワークにより，処理の内容が画面上に可視化される。
4. アルゴリズムの違いによる処理のコストの違いを考察。

5.3 観察による評価

カードを使った学習の様子を観察した。Aクラスでは実際にカードを作った学生は 1/3 程度であった。教具を作らせることは学習者の負担が大きいようである。カードを作った学生たちは，机上にカードを並べて配列を作り，ペンなどの文房具をインデックスとして動かしながら理解を深めていた。カードを作らなかった残りの 2/3 の学生は，フローチャートの読解だけを行っていた。

Bクラスではほぼ全員が画面上のカードを使い学習を進めていた。マウスの操作により試行錯誤しながら学習する様子が観察された。

5.4 質問紙による評価

質問紙の結果を表2に示す。回収できた質問紙は，Aクラス 32 名，Bクラス 14 名分である。

表 2 質問紙の結果：平均 (標準偏差)

質問項目	点数※と時間	
	A クラス（紙のカード）	B クラス（画面上のカード）
1．ドリトルの可視化が役に立った	3.6 (±0.9)	4.0 (±0.7)
2．プログラミングは難しい	3.6 (±1.0)	4.1 (±0.9)
3．カード操作がアルゴリズムの理解に役立った	4.0 (±0.9)	3.4 (±1.1)
4．カード操作は複雑で理解できなかった	2.3 (±1.2)	2.8 (±0.8)
5．カード操作学習に掛けた時間	53 分 (±42 分)	33 分 (±26 分)

※点数： 1=全然そう思わない 2=そう思わない 3=どちらでもない 4=そう思う 5=大変そう思う

5.5 考察

ドリトルによるアルゴリズム可視化は全体的に評判が良かった。アルゴリズム処理の理解に効果があったと考えられる。

ソートのプログラミングは初心者に取って簡単なものではないため，難しいと感じている学生が多い。

カード操作がアルゴリズムの理解に役立ったと感じている割合は，紙のカードを使った学習が有為に多かった(t 検定により 5%水準で検定)。画面上のカードに関しては，役立ったという評価とそれほどでないという評価に分かれた。

カード操作の複雑さに関しては，それほど複雑と感じていない。

作業時間の調査では，紙のカードを利用した学習の平均時間が長かったが，統計的に有為な差はなかった。しかし，最終的な課題の提出率は紙のカードを利用したAクラスが高く，学習の達成度が高かったことが伺える。

6. まとめ

教具を使って情報科学を学ぶ NPR のアイデアをプログラミング教育に取り入れる試みとして，アルゴリズムの可視化と，画面上で仮想的な教具を操作するタートルアルゴリズムを提案した。

アルゴリズムの可視化では，並び替えに用いられる操作をフレームワーク側でオーバーライドして自動的に画面上に表現することで，学生が記述したプログラムの処理経過を自分で確認できるようにした。非情報科学系の大学生を対象に行った評価では，この視覚化は効果があるという回答が多かった。

タートルを操作してアルゴリズムについて考えるタートルアルゴリズムについては，コンピュータ画面上でのカード操作の活動部分について評価した。結果，紙のカードを使う NPR の学習のほうが効果が高いことがわかった。今後は紙のカードと画面上のカードの違いを分析しつつ，今回提案したタートルアルゴリズム全体の評価を行っていきたい。

今回は紙のカードなど実物の教具を使う NPR の学習方法がアルゴリズムの学習に効果があることを確認できた。

また，ドリトルを用いたオブジェクト指向プログラミングを採用することで，アルゴリズムの記述に必要なデータ構造や命令だけを選択して与えることが可能であった。このような，学習に適した制約を考慮したプログラミング学習環境について，検討を進めていきたいと考えている。

参考文献

1) Joseph Bergin, Myles McNally, Mike Goldweber, Charles Kelemen, Tom Naps, Chris Power, Stephen Hartley : Non-Programming Resources for an Introduction to CS, http://csis.pace.edu/~bergin/iticse2000/

2) Tim Bell : A low-cost high-impact computer science show for family audiences, proceedings of Australian Computer Science Conference, p.10-16, 2000.

3) 兼宗進監訳:『コンピュータを使わない情報教育 アンプラグドコンピュータサイエンス』イーテキスト研究所, 2007.

4) 兼宗進, 中谷多哉子, 御手洗理英, 福井眞吾, 久野靖：初等中等教育におけるオブジェクト指向プログラミングの実践と評価, 情報処理学会論文誌「プログラミング」Vol.44, No.SIG13, pp.58-71, 2003.

5) シーモア・パパート：『マインドストーム—子供, コンピュータ, そして強力なアイデア』未來社, 1982.

6) 佐藤和浩, 紅林秀治, 青木浩幸, 西ヶ谷浩史, 井戸坂幸男, 鎌田敏之, 原久太郎, 久野靖, 兼宗進：IT クラフトマンシッププロジェクト～小中学生によるドリトルプログラミング～, 情報処理学会研究報告 CE(83), pp.173-180, 2006.