アルゴリズムの実装と可視化のためのMAS
—プログラミング教育の明日に向けて—

向　直人

プログラミング教育は日本の将来を担う重要な課題となっている．アルゴリズムはプログラムを実装するうえで必要不可欠な要素であるが，初学者にとっては理解が難しく躓くきっかけとなる．特に文系学部では，コンピュータに苦手意識をもつ学生も多く，図や擬似コードによる指導だけで，アルゴリズムを十分に理解させることは難しい．一方，マルチエージェントモデルは，タスクを複数のエージェントで分割することから，アルゴリズムの基本設計である分割統治法や動的計画法を表現しやすく，アルゴリズム教育に適している．本稿では，マルチエージェントモデルの開発環境である「artisoc」を採用し，アルゴリズムを実装・可視化した結果を報告する．

キーワード：アルゴリズム，可視化，マルチエージェント，プログラミング教育

1. プログラミング教育の現在

近年，初等教育からのプログラミング教育が注目され，学校教育における ICT 環境や教育教材のあり方が議論されている．平成 28 年 5 月には文科省主導で「プログラミング教育に関する有識者会議」が開かれ，小学校段階における効果的なプログラミング教育の実現に向けた検討が始まっている．2020 年度の学習指導要領からのプログラミングの必修化も示唆されており，プログラミング教育は日本の成長戦略の一つとして大きな役割を担っている．文献 [1] では，先行研究を基にしたプログラミング教育の学習効果の考察から，アルゴリズム的思考・論理的思考の向上が見込めると述べており，プログラミング教育が，コンピュータや情報という枠組みを超えて，一般的な問題解決力を向上させることを示唆している．

一方，現在の大学教育においては，情報科学や情報工学を専攻とする学部・学科ではプログラミングは必修の専門科目とされており，プログラミング教育のノウハウが蓄積されている．しかし，対象者である学生の多くはもとより理数系科目を得意とし，プログラミングの学習に必要な基礎知識や技術が備わっていると考えられ，使用されている教材など，これまでの指導方法をそのまま初等教育に適用することは困難であろう．また，文系学部においても，一般教養としてプログラミング教育を取り入れている大学もあるが，文献 [2, 3] によると，全体の 3 割程度に過ぎないことが報告されている．文献 [4] では，北海道大学の事例が紹介されている．同大学では，全学生必修で「情報学 I」を開講しており，約 20 人の少人数クラスの形式を採用している．情報に関する総合的な能力向上が目的となっており，ウェブページの作成やプログラミングなどが学習内容に含まれている．実施における最大の問題として，多数の指導者が必要であることを指摘しており，ICT を活用することで，20 人程度の少人数クラスを可能にしている．しかし，北海道大学のように大学全体のバックアップを得て，大規模に情報教育を実施できる組織は稀であろう．

上記に述べたように，初等教育また文系学部におけるプログラミング教育には，まだ多くの問題が残されており，文科省はもとより現場の教員にとって大きな課題である．そこで，本稿では，従来の図や擬似コードに加え，アルゴリズムを可視化することで，プログラミング教育を支援する試みについて述べる．文献 [5] では，データの可視化によって，新たな気づきを得たり，問題解決の糸口の発見が可能になると述べられており，可視化がアルゴリズム理解のための強力な手段であることは明白である．アルゴリズムの可視化には，ビジュアル・プログラミングなどのアプローチも存在するが，ここではマルチエージェントモデルを採用する．これまで，マルチエージェントモデルは，社会シミュレーションを目的として利用されることがほとんどであるが，複数の個体が自律的に行動する仕組みは，多くのアルゴリズムの振舞いを表現することに適している．マルチエージェントモデルの開発には構造計画研究所が提供する「artisoc」を利用し，最急勾配法や

むかい　なおと
椙山女学園大学文化情報学部
〒 464–8662 愛知県名古屋市千種区星が丘元町 17–3
nmukai@sugiyama-u.ac.jp

図 1　スクラッチの実行画面

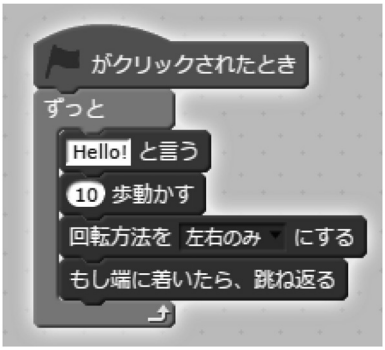

図 2　スクラッチのエディタ

シミュレーティッド・アニーリングなどの最適化アルゴリズムの可視化に加え，MAS コンペティションで発表した作品を紹介する．

本稿の構成は以下である．2 節では，ビジュアルプログラミングなど，プログラミング教育に関する技術や研究について述べる．3 節では，マルチエージェントモデルによるアルゴリズムの実装・可視化の特徴や利点に関して，最急勾配法などのアルゴリズムを例に挙げて解説する．4 節では，MAS コンペティションにおいて発表された，アルゴリズム可視化の事例を紹介する．最後に，5 節で本稿をまとめる．

2.　プログラミング教育の現在

本節では，初等教育や大学におけるプログラミング教育に関する最新の技術や研究の動向について述べる．

2.1　ビジュアルプログラミング

初学者を対象としたプログラミング教育の導入には，ビジュアルプログラミングがよく利用される．これは，プログラムのソースコードをテキストで入力するのではなく，視覚的な表現を用いて主にマウスなどの操作で入力することを意味する．最も有名なビジュアルプログラミング言語は「Scratch」[1]であろう．Scratch は，MIT メディアラボによって開発されており，入力したプログラムが図 1 のように，アニメーションで表現されることから，教育用の言語として人気が高い．また，ソースコードは，図 2 のように，ブロックを組み合わせて入力するため，プログラミング言語特有の文法規則などを学ぶ必要がない．

文献 [6] では，Scratch を小学校におけるプログラミング教育に導入した．学習用教材の「WeDo」[2]を利用し，モーターやセンサーを Scratch で制御することが生徒の課題となっている．検証の結果，Scratch を活用した授業は，生徒にとって簡単で楽しく，プログラミングに必要な一定の知識とスキルが習得できたと報告している．また，文献 [7] では，女子大学の文系学生を対象に，Scratch を利用したプログラミングの集中講義を実施した．このケースでは，学生の資質が椙山女学園大学と類似していることが予想される．アンケートの結果，受講者のプログラミングに対する興味が向上したことが示された．さらに，「コンピュータの仕組み」や「インタラクティブメディアの仕組み」に対する理解度も上昇したことを報告している．

このように，ビジュアルプログラミングは初学者に対して効果的に作用し，プログラミング教育の導入には最適だと考えられる．実行結果が視覚的にフィードバックされることから，ソースコードとその振舞いの対応関係を理解することが容易である．しかし，あらかじめ定義されているブロックしか利用できないなど，実装の制約も多く，多様なアルゴリズムを表現することは難しい．よって，アルゴリズムの理解を支援するには，より自由度の高い別のアプローチが必要となる．

2.2　アルゴリズム学習に特化したビジュアルプログラミング

Scratch はプログラミング学習には適しているが，アルゴリズム学習には不向きであると先に述べた．そこで，プログラミング学習とアルゴリズム学習を切り離し，「アルゴリズム的思考法」に着目した研究が文献 [8, 9] で報告されている．この研究では，アルゴリズムに必要な制御構文の入力のため，独自にビジュアルプログラミング可能なウェブアプリケーション「AT」を開発しており，ビジュアルプログラミングから始め，C 言語によるアルゴリズムの実装までを円滑に学習する仕組みを提供している．アプリケーションには，授業支援機能も組み込まれており，使用可能なブロックの制限や，課題提出・管理なども可能である．この手

[1] https://scratch.mit.edu/
[2] https://afrel.co.jp/product/wedo2.0-introduction

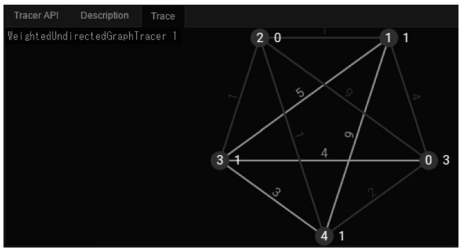

図 3　Algorithm Visualizer の実行画面

法では，アニメーション表現など冗長と思われる機能を制限することで，アルゴリズム学習への最適化を実現している．しかし，アンケート結果によると，プログラミング経験者にとっては，機能が制限されていることから「使いづらい」と感じることもあるようである．また，アルゴリズムを可視化するための機能が弱く，複雑なアルゴリズムの理解支援には適さないと考えられる．

2.3　アルゴリズムの可視化に特化したウェブサービス

アルゴリズムの可視化に特化したウェブサービスとして「Algorithm Visualizer」[3]がある．グラフ探索やソートなど多種多様なアルゴリズムに対応しており，JavaScript で記述されたソースコードを閲覧しながら，ウェブ上で実行結果を確認することが可能である．実行結果はアニメーションで表現され，直感的な理解が可能な仕組みとなっている．図 3 は，グラフ構造において，最短経路を導出するための「ダイクストラ法」を可視化した結果である．本稿の目的であるアルゴリズム理解の支援には適しているが，プログラミングに習熟した経験者でなければ，併記されているソースコードを理解し，改良することは難しいと思われる．たとえば，ダイクストラ法の初期ノードと目的ノードを変更するには，図 4 に示すソースを修正する必要がある．経験者であれば，変数 s に初期ノードの番号，変数 e に目的ノードの番号を代入すればよいとわかるが，初学者にはハードルが高いであろう．また，実行結果の可視化に関しては，あらかじめ実装されている仕様に従う必要があり，表現の自由度が低いことも挙げられる．

3.　MAS を利用したアルゴリズムの実装・可視化

本稿では，プログラミング学習の効果向上を目的として，マルチエージェントモデルを利用したアルゴリズムの実装と可視化に注目する．本節では，マルチエージェントモデルを用いることの利点について述べ，最急勾配法とシミュレーティッド・アニーリングを実装・可視化した例を紹介する．

[3] http://algo-visualizer.jasonpark.me

```
var s = Integer.random(0, G.length - 1);
var e;
do {
e = Integer.random(0, G.length - 1);
} while (s === e);
```

図 4　ダイクストラ法のソース（抜粋）

3.1　MAS を利用することの利点

これまでマルチエージェントモデルは，文献 [10] で報告されているように，主に社会科学分野の解析アプローチとして利用されてきた．たとえば，文献 [11] では，エージェントベースの交通シミュレーションを構築することで，現実に近いドライバーの意思決定を再現することを試みている．また，文献 [12] では，歩行者モデルと災害データを組み合わせ，津波避難や水害避難などの避難シミュレーションを構築する取り組みを紹介している．これらは，マルチエージェントモデルの特徴である「個々のルールに従うエージェントの相互作用により大局的な現象を理解する」ことを目的としている．上記の例では，エージェントは人や車をシミュレートし，最適な交通設計や避難誘導を導出することが狙いである．一方で，マルチエージェントモデルは，タスクを複数のエージェントで分担することから，多くのアルゴリズムの基本となっている分割統治法や動的計画法の考えに酷似している．加えて，文献 [13] で述べられるように，マルチエージェントモデルは，離散化された時間単位（ステップ）で，エージェントや環境が変化することを前提としており，アルゴリズムに必要な繰り返し処理の表現も容易である．つまり，マルチエージェントモデルに従って，アルゴリズムを実装することは，自然かつ効率的であるといえる．アルゴリズムの表現にマルチエージェントモデルが適している例を下記に列挙する．

- 分割統治法に基づくアルゴリズム（マージソート，クイックソートなど）
- 動的計画法に基づくアルゴリズム（Viterbi アルゴリズム，ダイクストラ法など）
- 多点探索型のメタ戦略（多点最急勾配法，遺伝的アルゴリズムなど）

たとえば，マージソートは分割統治法に基づいたア

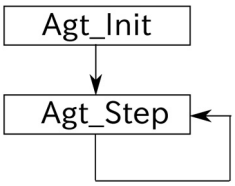

図 5　エージェントの関数

ルゴリズムであり，対象のデータを分割し，分割されたデータをソートした後に，マージするという手順でソートを実現する．ここで，分割されたデータをエージェントとみなすことで，「個々のエージェントによるソート」や「複数のエージェントのマージ」といった振舞いが明確になり，アルゴリズムの実装を助けることができる．また，文献 [14] にまとめられているように，最急勾配法は基本的に 1 点の探索点を用いるため，アルゴリズム自体はシングルエージェントで表現することが可能であるが，マルチエージェントモデルに従うことで，多点探索への拡張や，ほかの探索アルゴリズムとの比較が容易となる．

アルゴリズム教育においては，実装の簡便さに加え，可視化が理解を助ける重要な要素となる．そこで，マルチエージェントモデルの開発環境として，株式会社構造計画研究所が提供している「artisoc」[4] を採用し，マルチエージェント・シミュレーションの結果の出力・表示機能を利用する．上記のようにアルゴリズムをエージェントのルールとして定義しておけば，artisoc では GUI の操作だけで，アルゴリズムの振舞いを可視化することが可能である．これにより，学習者はアルゴリズムの実装に専念すればよく，煩雑な可視化という手順を簡略化できることに大きな意味がある．

3.2　MAS を利用した実装・可視化の例

ここでは，artisoc を使用して，マルチエージェントモデルに従って最急勾配法とシミュレーティッド・アニーリングを実装・可視化した結果を報告する．artisoc では，図 5 に示すように，エージェントの行動ルールを，Agt_Init 関数と Agt_Step 関数に記述する．Agt_Init 関数は，エージェントが生成されるときに 1 回だけ実行され，エージェントのパラメータの初期化に用いられる．また，Agt_Step 関数は，1 ステップごとに 1 回だけ実行され，エージェントの振舞いを記述する．エージェントの環境や可視化を定義するための関数・設定も存在するが，アルゴリズム学習を目的とする場合は，指導者側がこれらを事前に定義しておき，エージェントの実装のみを学習者に課すことが望ましいと考えられる．

[4] http://mas.kke.co.jp/

```
Agt_Step{
 My.X_plot = My.X_plot +
             Universe.Alpha * @gx(My.X_plot)
 My.Y_plot = @fx(My.X_plot)
 setScale()
}
```

図 6　最急勾配法の Agt_Step 関数

まずは最急勾配法を取り上げる．最急勾配法は連続空間（関数）の最適化問題に対するアルゴリズムの一つであり，反復法によって 1 点の探索点を更新しながら近似解を求める．このとき，探索点の更新には，関数の勾配を利用することが特徴である．ここでは，$f(x) = |\sin(x) \times x| + 2 \times x$ を空間とし，空間内の最大値の探索を目的とする．探索点の更新は下記の式 (1) に基づく．α は更新時の変化量の重みを決めるパラメータである．

$$x' = x + \alpha \frac{d}{dx} f(x) \tag{1}$$

学習者は，アルゴリズムの実装に際し，上記の探索点の更新のみを注視し，エージェントとして実装すればよい．図 6 が，最急勾配法を実装した Agt_Step の例である．My.X_plot，My.Y_plot は探索点（エージェント）の座標，@fx は対象となる関数，@gx は@fx の導関数である．また，setScale() は，探索点の描画のためにスケールを調整するために，指導者側が独自に実装した関数である．このように，artisoc を利用することで，環境設定や可視化などの煩雑な作業の隠蔽が可能で，学習者はアルゴリズムの本質を捉えることに専念できる．

多点型の最急勾配法の振舞いを可視化した結果が図 7 と図 8 である．図 7 は探索の初期状態である．空間には異なる高さの峰が 3 カ所存在し，30 のエージェントがランダムに配置されている．エージェントは最急勾配法に従い，探索点を更新しながら，近傍の峰を目指して移動する様子がアニメーションで表示される．図 8 が収束状態であり，エージェントが 3 カ所の峰に集まっていることがわかる．artisoc では，このようなアニメーションが GUI の操作のみで実現でき，学習者の負担は極めて少ない．

次にシミュレーティッド・アニーリングを取り上げる．シミュレーティッド・アニーリングは，最急勾配法と同様に最適化問題に対するアルゴリズムの一つであり，反復法によって 1 点の探索点を更新しながら近

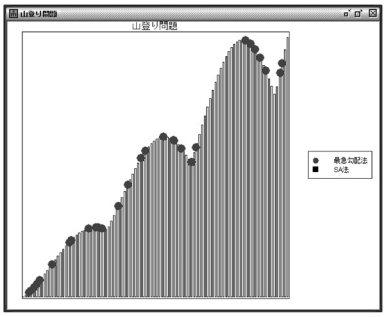

図 7　最急勾配法の初期状態

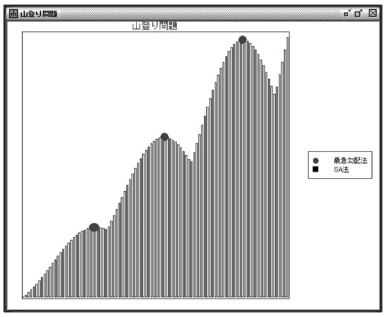

図 8　最急勾配法の収束状態

図 9　コントロールパネル

似解を求める．探索点の更新は，最急勾配法とは異なり，解が悪化する場合でも確率的に遷移するという特徴がある．新たな探索点への遷移確率 p は，式 (2) に示すように，温度パラメータ T を導入したメトロポリス法により決まる．

$$p = \exp\left(\frac{f(x') - f(x)}{T}\right) \tag{2}$$

ここで問題となるのはパラメータの調整である．シミュレーティッド・アニーリングは，最急勾配法とは異なり，先に述べた温度パラメータ T に結果が大きく依存する．学習者はパラメータを調整しながら，アルゴリズムの振舞いの変化を確認することが望ましい．artisoc では，図 9 に示すようなコントロールパネルの機能があるため，Agt_Step 関数などの修正なしに，GUI の操作だけでパラメータの調整が可能である．ここでは，探索方法やエージェント数などに加え，温度パラメータ T を 0 から 1 の範囲でスライダーバーで設定できる．この機能も，artisoc がアルゴリズムの可視化に適した特徴の一つである．

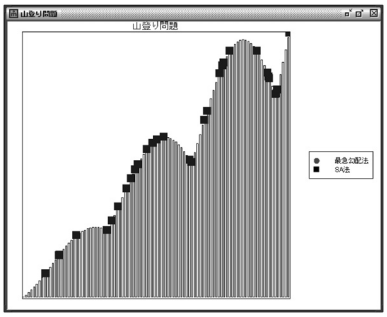

図 10　シミュレーティッド・アニーリングの初期状態

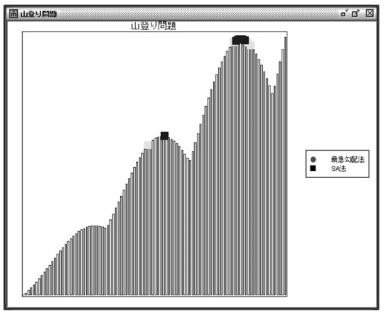

図 11　シミュレーティッド・アニーリングの収束状態

多点型のシミュレーティッド・アニーリングの振舞いを可視化した結果が図 10 と図 11 である．図 10 は探索の初期状態であり，最急勾配法と同様に，30 のエージェントがランダムに配置されている．エージェントは，確率的に探索点を更新しながら，近傍の峰を目指す．図 11 が収束状態である，最急勾配法とは異なり，エージェントは左にある低い峰を越えて，右の 2 カ所の峰に到達していることがわかる．また，収束後も峰中央に留まることなく，確率的に広く探索していることが視覚的にわかる．このように，アルゴリズムをエージェントとして実装することで，同一環境におけるアルゴリズムの比較が容易であり，学習者のメリットは大きい．また，異なるアルゴリズムのエージェントを混在させるなど，発展的なシミュレーションも可能である．

4.　MAS コンペティションでの事例

ここでは，artisoc を利用してアルゴリズムを可視化した事例について述べる．いずれの事例も，筆者の研究グループが構造計画研究所が主催する MAS コンペ

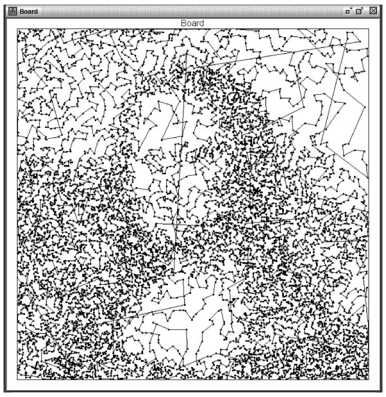

図 12　貪欲法の可視化

図 13　パーティクルフィルタの可視化

ティション[5]において発表したものである．

4.1　TSP Art

巡回セールスマン問題 (TSP) の応用である「Mona Lisa TSP Challange」[6] が題材である．公開されているモナリザのテストデータは 10 万点で構成されているが，ここでは 1 万点に間引いたデータを対象とした．アルゴリズムには貪欲法を採用し，その振舞いを可視化した．貪欲法は，局所的に最適な経路を繰り返し選択するため，最適解は保証されないが，高速に解の導出が可能である．図 12 が貪欲法の振舞いを可視化した結果である．エージェントは，点を結びながら移動を繰り返し，最終的にモナリザの絵が浮かび上がる．

4.2　パーティクルフィルタ

物体追跡に用いられるパーティクルフィルタを可視化した．追跡する物体は図 13 に示すボールであり，図中の人物がボールを移動させたり，紙コップで隠すという行為を繰り返す．パーティクルはエージェントとして実装され，尤度の高いエージェントの状態を基に，ボールの位置が推定される．

[5] http://mas.kke.co.jp/modules/tinyd3/
[6] http://www.math.uwaterloo.ca/tsp/data/ml/monalisa.html

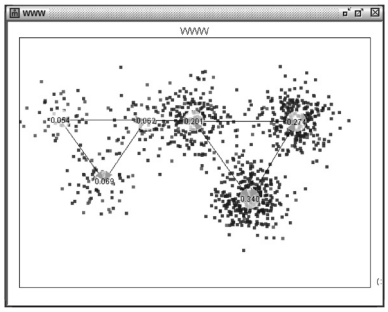

図 14　ページランクの可視化

4.3　ページランク

Google の検索エンジンの評価基準の一つであるページランクを可視化した．ページランクは，各ウェブページがもつ得点を，リンク構造に応じてほかのウェブページに伝搬することで決まる．この振舞いは，確率的にウェブページを遷移するランダムサーファーに例えられ，あるウェブページにランダムサーファーが存在する確率がページランクとなる．そこで，このランダムサーファーをエージェントとして実装・可視化した結果が図 14 である．エージェントがウェブページを遷移する様子がアニメーションで表現される．

5.　まとめ

本稿では日本の成長戦略の一つであるプログラミング教育に注目した．情報科学や情報工学を専攻とする学部・学科では，プログラミングはすでに必修科目としてノウハウが集積されているが，初等教育や教養教育にプログラミングを導入するには，まだ多くの課題が残っている．そこで，マルチエージェントモデルに基づいたアルゴリズムの実装・可視化が，プログラミング学習者にとって効果的であることを，最急勾配法などの具体例を示しながら明らかにした．マルチエージェントモデルの開発環境として採用した artisoc は，現状では教育用に開発されているわけではないが，ビジュアルプログラミングの仕組みを導入することで，教育用途への応用が十分に可能であろう．今後，情報を指導する教員はプログラミング教育という壁に直面することは避けられない．この壁を乗り越え，日本が技術立国として世界を牽引する日が来ることを期待したい．

参考文献

[1] 山本利一，本郷健，本村猛能，永井克昇，“初等中等教育におけるプログラミング教育の教育的意義の考察，”教育情報研究，**32**(2), pp. 3–11, 2016.

[2] 岡部成玄, “ぺた語義――般情報教育の全国実態調査 (2)―,” 情報処理, **56**(1), pp. 94–97, 2014.
[3] 岡部成玄, “ぺた語義――般情報教育の全国実態調査 (1)―,” 情報処理, **55**(12), pp. 1400–1403, 2014.
[4] 布施泉, 岡部成玄, “ぺた語義―北海道大学における全学教育としての情報教育―,” 情報処理, **52**(10), pp. 1341–1345, 2011.
[5] 鈴木雅彦, 鈴村嘉右, “データ可視化の必要性と意義―データビジュアライゼーションとは―,” 情報の科学と技術, **65**(11), pp. 470–475, 2015.
[6] 山本利一, 鳩貝拓也, 弘中一誠, 佐藤正直, “Scratch と WeDo を活用した小学校におけるプログラム学習の提案,” 教育情報研究, **30**(2), pp. 21–29, 2014.
[7] 森秀樹, “Scratch を用いた文系大学生向けプログラミング教育,” 日本教育工学会論文誌, **34**, pp. 141–144, 2010.
[8] 小林慶, 國宗永佳, 山本樹, 香山瑞恵, 新村正明, “アルゴリズム的思考法教育を支援するビジュアルプログラミング環境の運用と評価,” 電子情報通信学会技術研究報告（教育工学）, **113**(229), pp. 87–92, 2013.
[9] 大浦真暉, 國宗永佳, 山本樹, 新村正明, “プログラミング学習の導入を支援するビジュアルプログラミング環境の開発と評価,” 電子情報通信学会技術研究報告（教育工学）, **114**(260), pp. 73–78, 2014.
[10] 森俊勝, “マルチエージェントシミュレーション：8. 日本におけるマルチエージェントシミュレーション活用の動向,” 情報処理, **55**(6), pp. 585–590, 2014.
[11] 水田秀行, 牟田英正, 今道貴司, “マルチエージェントシミュレーション：7. 都市計画のための交通シミュレーション―スマートな都市運営のためのデータ解析と What-if シミュレーション―,” 情報処理, **55**(6), pp. 579–584, 2014.
[12] 山下倫央, 野田五十樹, “マルチエージェントシミュレーション：6. 避難シミュレーションの実社会への応用,” 情報処理, **55**(6), pp. 572–578, 2014.
[13] 鳥海不二夫, 山本仁志, “マルチエージェントシミュレーション：1. マルチエージェントシミュレーションの基本設計,” 情報処理, **55**(6), pp. 530–538, 2014.
[14] 永田裕一, “多点探索型アルゴリズムの基礎と最前線,” オペレーションズ・リサーチ : 経営の科学, **58**(12), pp. 708–715, 2013.

