Vol.2022-MPS-138 No.37
Vol.2022-BIO-70 No.37
2022/6/28

個体ベースモデルにおける生活史段階に基づいた追跡可能性を重視したシミュレーション環境

小田 朋宏[1,a)]　デュア ガエル[2,b)]

概要：自然界においては，多くの生物種が複数の生活史段階を持つ生活環を持ち，各生活史段階ごとに環境における振る舞いが異なる．エージェントベースモデル (ABM) において，個体ではなく生活史段階をエージェントとして表現する場合には，各個体の識別方法が必要になる．本研究では，個体ベースモデリング環境 re:mobidyc におけるエージェント生成追跡機能によって，個体群動態および進化的適応における個体識別と祖先追跡への応用を期待している．

Simulation Environment and Traceability for Individual-Based Modeling with Life History Stages

Abstract: Many species in nature have life history with multiple stages, and each individual behaves differently by its life stage. When we model a life hisotry stage as an agent, in ABM (Agent-Based Modeling), we need a method to identify individuals among different life history stages. In this paper, traceability of agent creations implented in Individual-Based Modelling environment named re:mobidyc. will be introduced and discussed. We expect to apply the agent creation traceability to individual identification and ancestor tracking in population dynamics and evolutionary adaptation.

1. はじめに

筆者らは，生物学や環境学での個体群動態に特化した ABM (Agent-Based Modelling) 環境として，re:mobidyc [1], [2] を開発している．ABM 環境として利用可能なシミュレーション環境は古くから開発され，実用化されている．特に LOGO 言語をベースにして拡張された ABM 環境が，NetLogo [3] を始め，数多く開発されてきた．それらの ABM 環境では，人間や動物の個体や群れなど移動可能な対象を，空間上の位置を持ち事前にプログラムされたプロシージャを持つタートルをエージェントとしてモデリングする．すなわち，複数の種類のエージェントについて，それぞれが持つ内部情報や，環境や他のタートルとの相互作用を定義する．

IBM (Individual-Based Modeling) は ABM でも特に，個体を対象にしてモデリングし，複数の個体の行動をシミュレーションすることにより個体の群れの特性を分析する手法である．筆者らは，プランクトン等を対象とした IBM 環境として，re:mobidyc を設計し開発してきた．自然界には，生活史段階と呼ばれる，成長段階によって体の構造や形状および行動が大きく変容する種が多く存在する．本稿では，生活史段階を持つ個体をエージェントとしてモデリングする際の問題点を示し，その解決としてエージェント生成の追跡可能性の確保を提案する．

2. IBM と生活史段階

本節では，IBM を生物学や環境学の研究への適用における，生活史段階の取り扱いの問題について説明する．生活史とは，生物の個体が生まれてから死ぬまでの一生にわたる変化を表す．生活史の把握は生物の研究の基礎であり，変態や生殖など個体の構造や習性を特徴的に変化させる生活史イベントによって，いくつかの生活史段階に分けられる．生活史段階の例として，**図 1** に動物プランクトンの一種である Eurytemora affinis の生活史段階を示す．Eurytemora affinis は卵から孵化すると，ノープリウス幼

[1] 株式会社 SRA
Software Research Associates, Inc.
[2] 静岡大学理学部
DGTalAqualab, Faculty of Science, Shizuoka University
[a)] tomohiro@sra.co.jp
[b)] dur.gael@shizuoka.ac.jp

Vol.2022-MPS-138 No.37
Vol.2022-BIO-70 No.37
2022/6/28

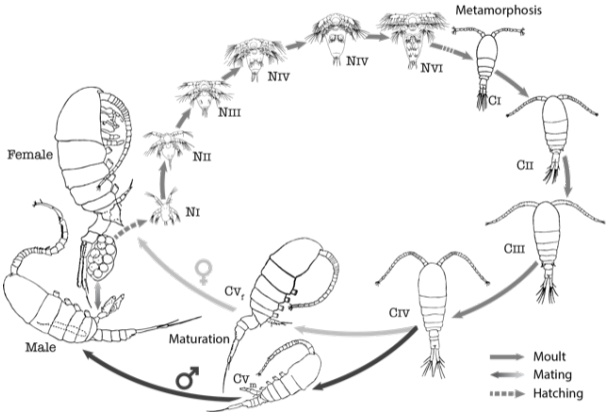

図 1 Eurytemora affinis の生活史段階 (参考文献 [1] より引用)
Fig. 1 Life history stages of Eurytemora affinis (quoted from [1])

生 (N_I〜N_{VI})，コペポディド幼体 (C_I〜C_V) を経て，成体になる．それぞれの生活史段階は脱皮により次の段階に進行する．C_V で雌雄の分化がおこり，それぞれ C_{Vf} および C_{Vm} として，成体にも雌雄がある．それぞれの生活史段階ごとに，捕食行動や生殖などの行動や，付属肢の場所や数，卵胞の有無などの体の構造や，生存条件，体長，代謝量，移動速度などの能力が変化する．ここでは Eurytemora affinis を例にしたが，生物種によって生活史段階の数やその進行が異なるが，生物の研究のための個体のモデリングにおいて，生活史段階の役割は重要である．

IBM の適用領域の 1 つに，生態系に対する金属汚染の影響の分析が挙げられる．Kadiene らが動物プランクトンの一種 Pseudodiaptomus annandalei に対するカドミウムの影響について生活史段階ごとの影響を報告している [4]．これらを IBM でモデリングする場合には，生活史段階ごとに水や食餌の摂取経路などを，エージェントが持つべき内部状態と行動として記述する．生活史段階により持つべき内部状態や行動が変化する場合に，エージェントを複数の生活史段階を含めた個体として定義するか，あるいは生活史段階ごとに異なる種類のエージェントとして分けて定義するかについて，モデリング言語に応じて異なるアプローチが考えられる．

生活史段階を既存の ABM でモデリングする例として，NetLogo での生活史段階の表現を説明する．NetLogo は汎用の ABM モデリング環境であり，生活史段階を直接記述するための専用の言語機能は持っていない．NetLogo では一般に品種を `breed` という特殊な変数で扱う．それぞれの個体について，`breed` として生活史段階を設定することで，個体ごとの生活史段階を保持し識別することができる．また，`breed` を再設定することで，生活史段階の進行を表現することができる．エージェントの内部状態を保持するエージェント変数は生活史段階を通して共通であり，生活史段階ごとに行動を変化させるためには，`breed` に対する

```
breed [nauplii nauplius] ;; ノープリウス幼生
breed [copepodids copepodid] ;; コペポディド幼生
to move
  if breed = nauplii [;; ノープリウス幼生の遊泳
    set heading random 360
    fd 1 ]
  if breed = copepodids [;; コペポディド幼生の遊泳
    left random 10
    right random 10
    fd 5]
end
to metamorphosis
  if breed = nauplii [
    set breed copepodids] ;; コペポディド幼生に変態
end
```

図 2 生活史段階を netlogo でモデリングした例
Fig. 2 Example model of lifehistory stages in netlogo

条件分岐で生活史段階ごとに振る舞いを記述する．**図 2** に netlogo での生活史段階の定義と行動の記述例を示す．1-2 行目の `breed` 文でそれぞれ生活史段階としてノープリウス幼生（単数形 `nauplius` 複数形 `nauplii`）とコペポディド幼生（単数形 `copepodid` 複数形 `copepodids`）を定義している．3-12 行目で手続き `move` を 2 つの `if` 文によって定義している．1 つ目の `if` 文ではノープリウス幼生の `move` 行動を定義し，2 つ目の `if` 文でコペポディド幼生の `move` 行動を定義している．13 行目以降で，ノープリウス幼生からコペポディド幼生に変態する `metamorphosis` 行動を `breed` 変数への代入によって定義している．

3. re:mobidyc における生活史段階と個体の扱い

本節では，re:mobidyc の概要と，生活史段階をどのように扱っているかを説明する．re:mobidyc は生物学や環境学への IBM の適用に特化した ABM 環境である [1], [2]．NetLogo が命令型プログラミング言語をベースにした ABM 環境であるのに対して，re:mobidyc はモデル記述者に逐次実行やループなど命令的なプログラミング知識をできるだけ要求しないようにモデリング言語が設計されている．

re:mobidyc では，1 つのシミュレーションモデルに複数種類のエージェントをユーザ定義し，それぞれのエージェントについて内部に持つ属性とエージェントの行動を定義するタスクを記述する．re:mobidyc のエージェントは生活史段階ごとに定義する．したがって，1 つの個体は生活史段階ごとに異なるエージェントインスタンスとして表現される．シミュレーション結果を個体単位で分析する場合には，1 つの個体に複数のエージェントインスタンスが対応するため，あるエージェントインスタンスがどの個体に対応するのかを識別可能であること，あるいは逆に，ある

```
Nauplius is Animat with
    age [day].
Copepodid is Animat with
    heading [degree].
```

図 3 re:mobidyc でのエージェント定義例
Fig. 3 Example agent definitions in re:mobidyc

シミュレーション時刻において，ある個体を表すエージェントインスタンスを特定可能であることが求められる．以下，re:mobidyc のモデリング言語の概要を説明した後に，re:mobidyc における個体とエージェントインスタンスの追跡可能性を確保する仕組みを説明する．

3.1 re:mobidyc によるモデル記述の概要

re:mobidyc では，移動可能な個体を表す Animat，二次元空間を分割した局所的環境を表す Patch，および帯域的環境を表す World の 3 種類のエージェントを定義することができる．ただし，Animat は抽象クラスであり，具体的なエージェント種別をユーザが定義する．

図 3 に re:mobidyc でのエージェント定義の例を示す．このエージェント定義は図 2 での NetLogo での定義と同様のモデルを構成するために，ノープリウス幼生（Nauplius）および小ピポディド幼生（Copepodid）の 2 つの生活史段階を定義している．Nauplius エージェントには属性として age が宣言されていて，その単位は日である．re:mobidyc は属性や直値に単位を付けることで，式に次元の誤りがないか検査する．age 属性は，後に Copepodid に変態するための条件に使われる．Copepodid エージェントは移動方向を表す heading 属性を持ち，単位は度である．ノープリウス幼生は移動能力が低く，ランダムな方向に移動するため，Nauplius エージェントには heading 属性は定義されていない．以上のように，re:mobidyc では生活史段階ごとに，必要な属性を単位とともに定義することができる．

次に，re:mobidyc での行動の記述を説明する．re:mobidyc では，行動を動作定義とタスク定義の 2 段階に分けて定義する．動作定義で動詞（自動詞または他動詞）を定義して，タスク定義ではそれぞれ S-V または S-V-O の構文で，どのエージェントがどの動作をするのかを宣言する．ある Animat が単独または環境とのみ相互作用を行う場合は自動詞を定義し，他の Animat との相互作用を行う場合には他動詞を定義する．**図 4** に move，turn，age，および metamorphosis の 4 つの自動詞を定義した例を示す．各動詞の定義は，エージェントの属性への更新を行う式の列挙と，必要な場合に where 以降の局所定義の列挙からなる．属性への更新は，新しい値を与える my <属性名>’ = <表現式> 形式と，差分値を与える my Δ <属性名>’ = <表現式>形式と，単位時間当たりの差分値を与える my d/dt <属性名>’ = <表現式>形式の 3 種類がある．また，

```
to move is
    my d/dt x' = cos(theta)*r
    my d/dt y' = sin(theta)*r
where
    theta = the heading
    r = the speed.
to turn is
    my Δheading' = the heading.
to age is
    my Δage' = Δtime.
to metamorphosis is
    when my age > 40 [hour]
    with stage Copepodid
    new heading' =
      uniform 0 [degree] to 360 [degree].
```

図 4 re:mobidyc での動作定義例
Fig. 4 Example action definitions in re:mobidyc

the heading および the speed などの接頭辞 the がついた識別子は，タスク定義で具体的な表現式に置き換えるための，プレースホルダーである．re:mobidyc では，プレースホルダーを使ったマクロ置換によって，move のような汎用的な動詞を定義する．

図 4 の 11 行目以降で変態 metamorphosis 動作が定義されているが，13 行目の with stage Copepodid はこの動作を実行したエージェントは次のシミュレーションステップいから Copepodid エージェントになることが定義され，その heading 属性の初期値としてランダムな方向が一様分布により与えられる．

タスク定義の例を**図 5** に示す．1 行目ではノープリウス幼生 Nauplius は move 動作をすることを宣言し，2 行目から 5 行目まででプレースホルダーから置き換わるべき表現式を与えている．この置換を図 4 の 1 行目から 6 行目までで定義された動作 move に適用することで，ランダムな方向に 0.1[cm/s] 進むタスクが定義される．以下，同様に Nauplius エージェントの age タスクと metamorphosis 動作，および，Copepodid エージェントの move 動作と turn 動作が定義されている．

本節での定義例では，生活史段階を推移する動作として，with stage Copepodid を使った metamorphosis 動作を定義した．re:mobidyc では，これら生活史に直接変化を与える命令として，new，change，die および kill の 4 種類の生活史命令 を提供している．new は，生殖行動として新たな個体を生成する生活史命令であり，change は同一個体が次の生活史段階に移行する生活史命令である．die および kill は個体を死亡させる生活史命令であり，die 命令はタスクを遂行するエージェントを，kill 命令はタスク定義の S-V-O 構文の O に相当するエージェントを停止させる．

```
Nauplius move
where
    the heading ->
      uniform 0 [degree] to 360 [degree]
    the speed -> 0.1 [cm/s].
Nauplius age.
Nauplius metamorphosis.
Copepodid move
where
    the heading -> my heading
    the speed -> 0.5 [cm/s].
Copepodid turn
where
    the heading ->
      uniform -10 [degree] to 10 [degree].
```

図 5 re:mobidyc でのタスク定義例
Fig. 5 Example task definitions in re:mobidyc

IBM では個体単位の振る舞いを中心にエージェントを使ってモデリングし分析する．生活史段階を移行させる `stage` 命令は，移行後の生活史段階を表すエージェントを生成して，タスクを実行したエージェントを消滅させることから，`stage` 命令によって置き換わったエージェントに関しても，個体としての一貫した識別を維持することが求められる．この問題についての，re:mobidyc での解決を次節で説明する．

3.2 インスタンス生成リンクによる個体および祖先の識別

re:mobidyc の処理系は，シミュレーション時刻ごとにエージェントの属性値を管理する時系列用のメモリ領域 [1] と，エージェントインスタンスを管理するメタ領域の 2 種類のメモリ領域を管理している．メタ領域では，エージェントクラスとエージェントインスタンスの対応関係と，インスタンスを生成したインスタンスの対応関係を管理している．4 つの生活史命令 `new`，`stage`，`die` および `kill` は，メタ領域に対する操作を行うための命令であり，インスタンス生成の対応関係に対する変更を行う生活史命令は `new` および `stage` である．

re:mobidyc では，インスタンス生成の対応関係として，`reproductionLink` と `stageLink` の 2 種類を区別して管理する．`new` 命令は `reproductionLink`，`stage` 命令は `stageLink` にそれぞれ新たな対応関係を追加する．`stageLink` に生活史段階の移行関係を記録することで，異なる生活史段階の間の個体の識別性を確保することができる．また，`reproductionLink` に記録されたインスタンス生成関係から，親子関係を追跡することが可能になる．この親子関係は，複数世代にわたる汚染物質の影響の追跡や，特定の環境の変化に対する進化適応の分析に適用することが想定されている．

4. おわりに

多くの ABM のモデリング言語はプログラミング言語の言語機能を取り入れている．プログラミング言語の言語機能は計算の記述として強力な機能を持つが，一方でプログラムの停止性などの問題を持ち込む可能性も含んでいる．また，本稿で論じたように，生活史段階などモデリング対象に特有な概念に対して，必ずしも十分な表現力を持たない場合がある．ABM 向けのモデリング言語の設計では，プログラミング言語とは異なる要請があり，モデリング対象に応じて適切な言語機能を設計し，処理系を実装する必要がある．re:mobidyc の開発はまだ初期段階であり，今後も re:mobidyc の開発を継続する．長期的な課題として，プログラミング言語やその処理系の流用ではない，対象領域に特化したモデリング言語を設計し処理系を実装するための手法の確立を目指したい．

謝辞 筆者らの所属は，Pharo コンソーシアム会員として，re:mobidyc の開発への支援を得た．ここに感謝の意を記す．

参考文献

[1] 小田 朋宏, デュア ガエル: 個体群動態論に特化したマルチエージェントシミュレーション基盤 Re:Mobidyc, ソフトウェアシンポジウム 2021 論文集, pp. 35-44, 2021.
[2] Oda, T., Dur, G., Ducasse, S. and Souissi, S.: *re:Mobidyc - Reconstructing MOdeling Based on Individual for the DYnamics of Community*, in Proc. of International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'21), pp. 367-371, 2021.
[3] Tisue, S., and Wilensky, U. : *Netlogo: A simple environment for modeling complexity.*, in Proc. of International Conference on Complex Systems, Vol. 21, pp. 16-21, 2004.
[4] Kadiene, E. U., et al. : *Acute and chronic toxicity of cadmium on the copepod Pseudodiaptomus annandalei: A life history traits approach*, Chemosphere Vol. 233, pp. 396-404, 2019.