Vol.2019-EIP-86 No.1
2019/11/22

JavaScript による知的財産の社会的生成流通の マルチエージェント シミュレーションの実験的システム

金子 格†

概要: JavaScript と HTML5 環境で知的財産生成流通のマルチエージェントシミュレーションの実験的システムを試作したのでその概要を報告する。ブラウザに標準的に備わる JavaScript と HTML5 を利用することでシミュレーションプログラムはブラウザ上で動作し、一般公開や、Web 環境上で様々な異なる条件でのシミュレーションを、容易に行うことができる。知的財産の流通管理方法の効果を示すツールとしての可能性についても考察する。
キーワード: マルチエージェント，知的財産，制度設計，JavaScript，コンテンツ流通，シミュレーション

1. 概要

著作物の利用においてパッケージ主体の流通からネットワーク配信を前提としたストリーミングやサブスクリプションへと利用形態の主体が移行して久しい。このような変化の中、著作権法も頻繁に改定されている。

もとより著作権は自然所有権ではなく著作物の利用の促進のために作られた人工的な制度である。利用の実情に合わせ「(著作物の)文化的所産の公正な利用に留意しつつ，著作者等の権利の保護を図り，もつて文化の発展に寄与すること」(平成 30 年 7 月 6 日改正著作権法第一条)を目的とすると著作権法にも明記されている通り、利用技術の変化にあわせてその適切な改定が必要であり、実際にネット利用の発達に応じて検討，修正することが行われてきた。

しかしネット上の新たな利用方法は社会制度としては、人類社会にとって未経験なものであることが多い。はたしてどのような法制度がどのような効果をもたらすかは常に議論が分かれるところである.

新たな法制度の効果を予想する道具としてマルチエージェントシミュレーションがある**エラー! 参照元が見つかりません。**．すでに選挙，入札，交通網などの研究に適用されている．従来， Java などのオブジェクト指向言語で開発されることが多かったが，我々は今回デジタル著作権の制度比較分析を目的として JavaScript と HTML5 環境でマルチエージェントシミュレーション環境を構築した．著作権のように広く関心を持たれ議論される分野では，広くシミュレーション結果を共有できることが望ましい．JavaScript+HTML5 が持つポータビリティが大きなメリットになると考える.

本システムは VS Code とブラウザのみで開発した．実行はブラウザのみで可能でスマートフォンでも問題なく実行できる. 様々なシミュレーションをほぼすべての携帯端末やパソコンで様々な条件で実行可能である.

2. 著作権下のエージェント行動ルール

マルチエージェントシミュレーションとはコンピュータ上にプログラムにより実現されたエージェントを多数準備し、それらを社会に属する公衆を模倣するようにプログラムで制御するシミュレーション手法である。建築物や工業部品の強度や振動を模倣する手法には有限要素法という手法があるが、その場合には各部がお互いに連結しているのに対しマルチエージェントでは各個人がそれぞれ独立した属性、状態、行動原理をもって行動する。そのようにして多くの公衆をシミュレートすることで、たとえば自由市場における価格や購買の挙動や、都市における交通網の中の人々の挙動などをシミュレーションするものである。

したがってマルチエージェントシミュレーションを行うには、まず個人が従うべき行動原則(法制度という意味のルールではなく、どういう原則によって各個人が意思決定を行い次の行動を選ぶかというルール)を設計する必要がある。

本シミュレーションは実験的システムであり、また教育目的でもあるために以下の方針でルール設計を行った。

I ルールは著作権に関する性質を確認できる最小限とする
II 著作者、利用者、海賊行為を行うものを模倣するエージェントを含める

このような方針にもとづいて最小源のルールを策定した。以下のルールを実装した.

(1) 著作者，読者，海賊はエネルギーを取得，消費しながら活動し，エネルギーが不足すると停止する.
(2) 著作者は著作にエネルギーを使い読者が著作を利用する場合エネルギーを得る.
(3) 読者は著作物を取得すると対価として作者にエネルギーを与え，自身も著作物の利用からエネルギーを得る.
(4) 海賊版発行者は著作物に接すると自信の著作物に書き換え，以後著作物の利用対価は海賊版発行者が取得する

3. プログラム

プログラムは 2 段階に分けて開発した. 第一版は共著者 2(津江)が卒業研究テーマとして開発した**エラー! 参照元が見つかりません。**．プログラムサイズ 2000 行で様々な

† 名古屋市立大学
Nagoya City University

シミュレーションを行った．出力はテキスト出力とし，シミュレーション結果を excel でグラフ化して確認する方法をとった．

第 2 版は共著者 1 が第 1 版をもとにビジュアル化，コンパクト化とポータビリティの向上を図った．出力を 2D アニメーション化し，プログラムを 500 行に短縮し，HTML5 対応ブラウザのみシミュレーション結果を表示可能とした．．

表 1　開発したプログラム

	言語	機能	サイズ
第 1 版	JavaScript + excel	Text 出力	1000line
第 2 版	JavaScript	2DCG 出力	500line

4. 実行結果と考察

1.1 基本動作および著作者と読者の相乗効果

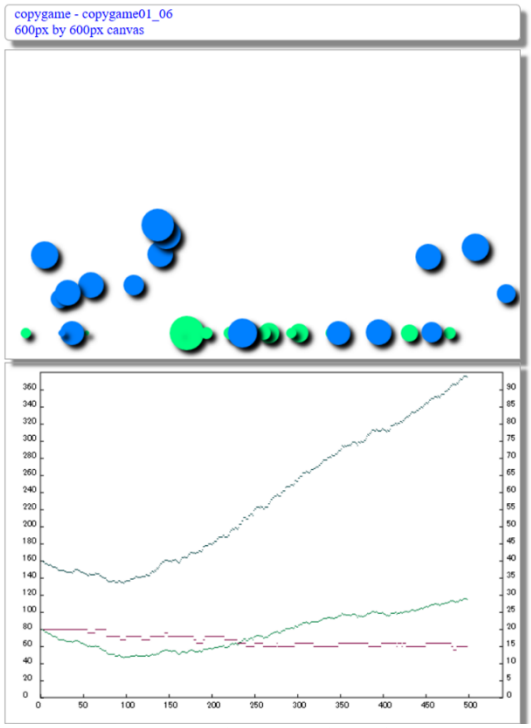

図 1　著作者 ＋ 読者の変化

図 1 はシミュレーション中の画面を示す．本図の上側のウインドウは，読者と著作者の挙動を，横約 600 ドットの 1 次元空間上の図形の位置と形状であらわしている．著作者，読者は左右に移動しながら活動する．著作者は著作を行うとその位置に著作物を残す．

読者は読者の位置に著作物があれば利用する．著作物を利用すると読者は上方向にジャンプして利用を示しジャンプ中は次の著作物は利用しない．

著者は著作をするとエネルギーを消費し，読者に著作を利用されるとエネルギーを獲得する．読者は定期的にエネルギーを失い，著作物を利用するとエネルギーを得ると同時に著作者にもエネルギーを与える．著者読者ともエネルギーを失と停止する．

図 1 の下半分では横軸を時間として，著作者，読者の総エネルギー，著作者のエネルギー，活動中の著作者の数の時間変化を示している．プログラムの一回の実行を 1 ターンとし，グラフ上では 1 ドットで表される．100 ターンまでは活動する著作者が減り著作者のエネルギーも減っている．これはまだ作成された著作物が少なく消耗が勝るためである．徐々に著作物が蓄積され 100 ターン以後は作成された著作物から得られる増加が消耗を上回り，著作者，読者とも総エネルギーは増加に転じる．これは現実社会で，著作物利用の好循環が生まれ文化の発展に寄与する状態に相当すると考えられる．

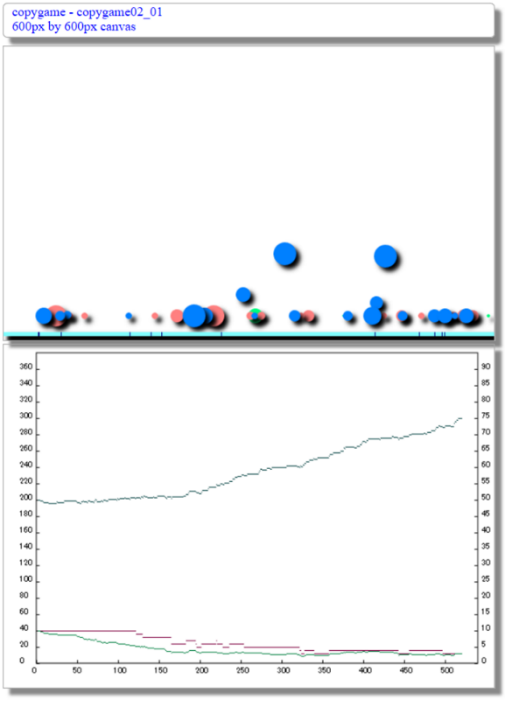

図 2　著作者+読者+海賊

1.2 海賊の成長モデル

図 2 は海賊行為を行うエージェントが 20 個，著作者が 10 個，読者が 20 個でのシミュレーションを行った結果である．全体のエネルギーはゆるやかに増加するが，図の上半分のウインドウにおいて途中赤色で示す海賊が増加し，著作物の収益が海賊に奪われ，活動可能な著作者の数が減少する．　500 ターンで著作者は半分以下になり全体のエネルギー増加が抑制される．

図の上半分では緑色の著作者が縮小し赤色の海賊が大きな円となり活発に活動していることがわかる．また著作物が少数のうちは海賊の活動レベルも低く，著作者と著作物が豊富になった後で海賊が増加し著作者の活動が抑制される．現実社会で海賊行為が著作者の活動を阻害している状況に相当すると考えられる．

1.3 海賊の禁止の効果

次に図 3 に海賊の活動を禁止した場合を示す．著作者，読者，海賊の数は同一で海賊の活動だけを禁止する．著作者，読者のエネルギーが増加し，全体のエネルギーも順調に増加する．特に，活動を停止する著作者がほとんどいない点が大きく異なる．

このシミュレーションでは，著作者，読者，全体としてもエネルギーが増加する．現実社会で海賊の排除が文化の発展に寄与する状況に相当すると考えられる．

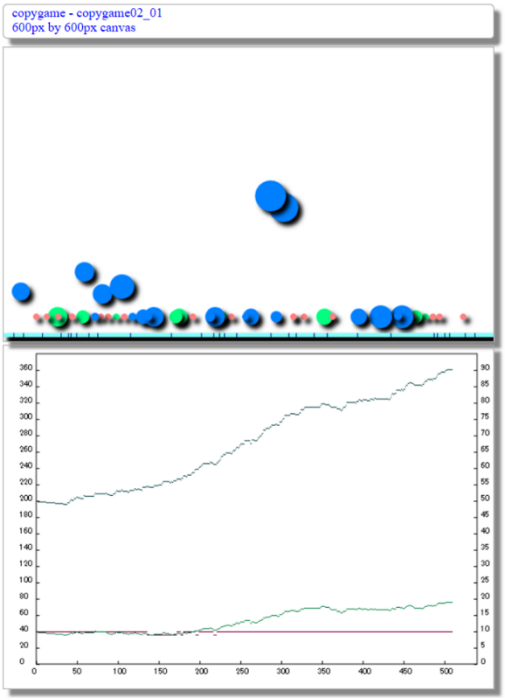

図 3　海賊の活動を禁止した場合

5. 考察とまとめ

JavaScript により実用的なエージェントシミュレーションが実現できることが示された. 特に大規模なシミュレーションでなければブラウザ上で十分実装, 実行が可能だった. これらの結果から, JavaScript によるマルチエージェントシミュレーションは開発も容易であり、プレゼンテーションや教育利用に適していると考えられる.

今回実装したモデルでは著作者と読者の相乗効果が確認できた. また海賊による悪影響, 海賊の活動を抑制することによる創作活動の活発化などが確認できた. これらのことから, 簡単なシミュレーションモデルで著作権の社会への影響分析が可能と考えられる.

これらのシミュレーションの挙動はもちろん様々な条件によって変化する. どのような場合にどのような著作権制度が全体的な文化の発展に寄与するかを分析する上で, 手軽に利用できポータビリティの高い JavaScript 上のマルチエージェントシミュレーションは効果的なツールとなると考えられる.

References:

[1] 山影 進, 服部 正田, "コンピュータの中の人工社会 -- マルチエージェントシミュレーションモデルと複雑系", 構造計画研究所(2002).

[2] 津江 銀河, “著作権のルールにおける創作活動等の影響分析シミュレータ”, 平成 30 年東京工芸大学工学部コンピュータ応用学科卒業論文(201)

[付録]

以下は本システムの全ソースコードである. すべてを同一の web サイトに設置市, HTML5 対応ブラウザで実行可能である。本プログラムの改変, 利用は自由である.

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <link rel="stylesheet" type="text/css"
href="stylesheet.css">
        <script src="copygame_agent.js">    </script>
        <script src="script.js">    </script>
        <title>copygame02_01</title>
    </head>
    <body onload = 'draw_canvas()'>
        <div>
            <div id='div1'>
                copygame - copygame02_01 <br/>
                600px by 600px canvas
            </div>
            <div id='div2'></div>
            <canvas id="canvas_tag_1" width="600"
height="360"></canvas>
            <canvas id="canvas_tag_2" width="600"
height="420"></canvas>
        </div>

    </body>
</html>
```

stylesheet.css

```
#div1 {
    float: top;
    padding-left: 10px;
    width:590px;
    height: 40px;
    border-radius: 5px;
    border-style: solid;
    border-width: 1px;
    border-color: #aaa;
    color: #0000ff;
    box-shadow: 8px 8px 5px #888;
  }
#div2 {
    float: top;
    height: 10px;
  }
#canvas_tag_1 {
    float: top;
    border-color: #aaa;
    border-style: solid;
    border-width: 1px;
    box-shadow: 8px 8px 5px #888;
  }
#canvas_tag_2 {
    float: top;
    border-color: #aaa;
    border-style: solid;
    border-width: 1px;
    box-shadow: 8px 8px 5px #888;
  }
```

script.js

```
// プログラム全体で使用する変数
var c1;       // ゲームボード描画コンテキスト
var c2;       // グラフ描画コンテキスト
var copygame_agents;    // 全 copygame_agent を格納する配列
var number_of_copygame_agents; // copygame_agent の数
var number_of_creators; // copygame_agent の数
var number_of_pirates; // pirates の数
var gb;       // game bord, sixze 600
var tick_count1;     // tick count 1



// 現状を plot する

function plot_status(){
  var x1 = Math.floor(tick_count1+1);
  if (x1>599) x1=599;
  esum=0; // 全体のエネルギー
  ncnt=0; // 全体の生きているエージェントの数
  aena=0; // 著作者のエネルギー
  var n;
  for (n = 0; n < number_of_copygame_agents; n++) {
    esum=esum + copygame_agents[n].ep;
    if (copygame_agents[n].ep > 0 &&
copygame_agents[n].type==1) {
      ncnt=ncnt + 1;
      aena=aena + copygame_agents[n].ep;
    }
  }
  c2.beginPath();
  c2.rect(x1 + 40,390-esum, 1,1);
  c2.fillStyle = 'rgb(0,64,64)'; // 紺色
  c2.fill();
  c2.beginPath();
  c2.rect(x1 + 40,390-ncnt*4, 1,1);
  c2.fillStyle = 'rgb(128,0,64)'; // 紺色
  c2.fill();
  c2.beginPath();
  c2.rect(x1 + 40,390-aena, 1,1);
```

```
  c2.fillStyle = 'rgb(0,128,64)'; // 紺色
  c2.fill();
}

function plot_axis(){
  c2.beginPath();
  c2.fillStyle = 'rgb(0,0,0)'; // 紺色
  c2.rect(40,10,540,380);
  c2.stroke();
  var n;
  for (n=0; n<540; n+=50) {
    c2.beginPath();
    c2.fillStyle = 'rgb(0,0,0)'; // 紺色
    c2.rect(n+40,386,1,4);
    c2.fill();
    c2.font = "10px 'ＭＳ Ｐゴシック'";
    c2.strokeStyle = "blue";
    c2.fillText(n.toString(10),n+35,402);
  }
  for (n=0; n<380; n+=20) {
    // 左側の目盛
    c2.beginPath();
    c2.fillStyle = 'rgb(0,0,0)'; // 黒
    c2.rect(40,390-n,4,1);
    c2.fill();
    c2.font = "10px 'ＭＳ Ｐゴシック'";
    c2.strokeStyle = "blue";
    c2.fillText(n.toString(10),20,390 - n);
    // 右側の目盛
    c2.beginPath();
    c2.fillStyle = 'rgb(0,0,0)'; // 黒
    c2.rect(576,390-n,4,1);
    c2.fill();
    c2.font = "10px 'ＭＳ Ｐゴシック'";
    c2.strokeStyle = "blue";
    var n1 = n / 4;
    c2.fillText(n1.toString(10),585,390 - n);
  }
  //c2.filltext("tick",210,220)
}

// copygame_agent クラスを使ったアニメーションの本体
// 毎秒 30 回実行する関数
function tick1() {
  tick_count1=tick_count1+1;
  // 描画領域をいったんクリアする
  c1.clearRect(0, 0, 600, 600);

  // 20 個の円についてのループ
  var n;
  for (n = 0; n < number_of_copygame_agents; n++) {
    // copygame_agent を移動し、描画する
    copygame_agents[n].progress();
    copygame_agents[n].move();
    copygame_agents[n].show();
  }
  for (n = 0; n < 600; n++){
    gb[n].show();
  }
  plot_status();
}

// 初期化
function draw_canvas() {
  // c1 = 2d コンテキスト、を用意する
  var canv1 = document.getElementById('canvas_tag_1');
  c1 = canv1.getContext('2d');
  if (!canv1 || !canv1.getContext) {
     return false;
  }
  var canv2 = document.getElementById('canvas_tag_2');
  c2 = canv2.getContext('2d');
  if (!canv2 || !canv2.getContext) {
     return false;
  }
  tick_count1=0; // tick count をゼロリセット
  gb = new Array(600);
  // game bord のクリア
  for (var n=0; n<600; n++) {
    gb[n]=new game_cell(n,0);
  }
  // copygame_agent 数の設定
  number_of_copygame_agents = 50;
  number_of_pirates = 20;
  number_of_creators = 10;

  // 全 copygame_agent を格納する配列の準備
  copygame_agents = new Array(number_of_copygame_agents);
  // 全 copygame_agent の初期化
  for (var n = 0; n < number_of_copygame_agents; n++) {
    if(n < number_of_creators) {
    copygame_agents[n]=new copygame_agent(n,gb,1);
    }
    else if (n < number_of_creators + number_of_pirates ) {
      copygame_agents[n]=new copygame_agent(n,gb,3);
    }
    else {
      copygame_agents[n]=new copygame_agent(n,gb,2);
    }
  }

  // tick1 を毎秒 30 回実行するための設定
  plot_axis();
  setInterval(tick1, 100);
}
```

copygame_agent.js -- Experiment 1

```
// copygame01_06
// 2019/6/17
// copygame_agent のクラスの定義
// 初期化
// gb 長さ 600 の配列
// copygame01_03 ルール
// エージェントは 2 タイプ
//  type 1 creator
//    ep 初期値 = 4
//    1%の確率で行う。
//    エネルギーが 2 以上なら自分を author として記録する
//    エネルギーは 1 減らす
//  type 2 consumer
//   Author が登録されている場合
//     author のエネルギーを +1 増加する
//     ジャンプする
//   マークされていない場合
//     自由運する
//

// class definition : game_cell
// ゲームのます目の定義
function game_cell(x1,y1) {
  this.x=x1;
  this.y=y1;
  this.st = 0; // status = 0
  this.author = null;    // copy game agent marking here
}

game_cell.prototype.show=function(){
   // rect を描画
    // y 座標を整数に変換してから描画する
    var x1,y1,h1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);
    c1.beginPath();
    h1 = this.st*5;
    c1.rect(x1,380-y1-h1, 10,h1);
    if(this.st==1){
      c1.fillStyle = 'rgb(255,0,128)'; // 黄色
    }else{
      c1.fillStyle = 'rgb(255,128,0)'; // 黄色
    }
    c1.fill();
}

// (ix,iy)は (x.y)の整数値
// 初期化により x,vx はランダムに
// 10<x<580, -25<x<25
// y, vy は 0 に初期化

function copygame_agent(aid1,gb1,ty1) {
  this.aid=aid1;
  this.gb = gb1;    // game board
  this.type = ty1;    // agent type = 2
  this.x = Math.random() * 570 + 10;
  this.y = 0;
  this.vx = Math.random() * 50 - 25;
  this.vy = 0;
  this.ep = 4;  //  energy point = 0
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);
}

// move : 移動
// 一番上 (iy=0)
//   gb[this.ix].st=0 なら gb[this.ix].st を 1 に
//   gb[this.ix].st=1 なら
// y 座標を 1, vy を 10 にする

copygame_agent.prototype.progress = function() {
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);

 if(this.type==1){
    // creator type
    if(this.iy==0 && this.ep>0){
      // エネルギーが残っており、iy=0 の場合
      var rr=Math.random();
      if (rr < 0.02){
        // 2% の確率でエネルギーを減らす
        this.ep = this.ep - 1;
      }
      if (rr < 0.01){
        // 1%の確率で作品を残す
        this.gb[this.ix].st = 1;
        this.gb[this.ix].author=this;
      }
    }
  }
  else if(this.type==2) {
    if (this.iy==0) {
      if (this.gb[this.ix].st==1) {
        var author1;
        // マーク済の位置にあれば、ジャンプ
        this.y = 1;
        this.vy = this.ep; // エネルギー分ジャンプ
        author1 = gb[this.ix].author;
        author1.ep = author1.ep + 1;
        this.ep = this.ep + 1;
      }
      else {
        // マークがなければそのままの位置でマーク
```

```
      }
    }
  }
}
copygame_agent.prototype.move = function() {
  // consumer type
  if(this.ix < 0 || this.ix>=600) {
    console("heelo")
  }
  if (this.x < 10  || this.x > 580) {
     this.vx = - this.vx;
  }   // gravity
  if (this.y > 0) {
    this.vy -= 1;
  }
  // motion
  this.x += this.vx;
  if (this.x <0) this.x=0;
  if (this.x>=600) this.x=599;
  this.y += this.vy;
  // collision with ground
  if (this.y<0) {
    this.y=0;
    this.vy=0;
  }
}

// show : 表示
copygame_agent.prototype.show = function() {
  // 円を描画
    // y 座標を整数に変換してから描画する
    var x1,y1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);


    c1.beginPath();
    c1.arc(x1,330-y1, this.ep, 0, Math.PI * 2);
    if(this.type==1){
      c1.fillStyle = 'rgb(0,255,128)'; // 紺色
    }else{
      c1.fillStyle = 'rgb(0,128,255)'; // 紺色
    }
    c1.shadowColor = 'rgb(0,0,0)';   // 影
    c1.shadowOffsetX = 5;
    c1.shadowOffsetY = 5;
    c1.shadowBlur = 5;
    c1.fill();
}
```

copygame_agent.js -- experiment 2

```
// copygame02_01
// 2019/6/18
// 概要 海賊版業者の導入
// copygame_agent のクラスの定義
// 初期化
// gb 長さ 600 の配列
// copygame02_01 ルール
// エージェントは 3 タイプ
//  type 1 creator
//    ep 初期値 = 4
//    1%の確率で行う。
//    エネルギーが 2 以上なら自分を author として記録する
//    エネルギーは 1 減らす
//  type 2 consumer
//   Author が登録されている場合
//     author のエネルギーを +1 増加する
//     ジャンプする
//   マークされていない場合
//     自由運する
//  type 3 pirates
//     もし st=1 ならインデックスを自分に書き換える
//     エネルギーは減らさない
//

// class definition : game_cell
// ゲームのます目の定義
function game_cell(x1,y1) {
  this.x=x1;
  this.y=y1;
  this.st = 0; // status = 0
  this.author = null;    // copy game agent marking here
}

game_cell.prototype.show=function(){
   // rect を描画
    // y 座標を整数に変換してから描画する
    var x1,y1,h1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);
    c1.beginPath();
    h1 = this.st*5;
    c1.rect(x1,380-y1-h1, 10,h1);
    if(this.st==1){
      c1.fillStyle = 'rgb(255,0,128)'; // 黄色
    }else{
      c1.fillStyle = 'rgb(255,128,0)'; // 黄色
    }
    c1.fill();
}

// (ix,iy)は (x.y)の整数値
// 初期化により x,vx はランダムに
// 10<x<580, -25<x<25
// y, vy は 0 に初期化

function copygame_agent(aid1,gb1,ty1) {
  this.aid=aid1;
  this.gb = gb1;    // game board
  this.type = ty1;    // agent type = 2
  this.x = Math.random() * 570 + 10;
  this.y = 0;
  this.vx = Math.random() * 50 - 25;
  this.vy = 0;
  this.ep = 4;  //  energy point = 0
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);
}

// move : 移動
// 一番上（iy=0)
//   gb[this.ix].st=0 なら gb[this.ix].st を 1 に
//   gb[this.ix].st=1 なら
// y 座標を 1, vy を 10 にする

copygame_agent.prototype.progress = function() {
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);

 if(this.type==1){
    // creator type
    if(this.iy==0 && this.ep>0){
      // エネルギーが残っており、iy=0 の場合
      var rr=Math.random();
      if (rr < 0.02){
        // 2% の確率でエネルギーを減らす
        this.ep = this.ep - 1;
      }
      if (rr < 0.01){
        // 1%の確率で作品を残す
        this.gb[this.ix].st = 1;
        this.gb[this.ix].author=this;
      }
    }
  }
  else if(this.type==2) {
    if (this.iy==0) {
      if (this.gb[this.ix].st==1) {
        var author1;
        // マーク済の位置にあれば、ジャンプ
        this.y = 1;
        this.vy = this.ep; // エネルギー分ジャンプ
        author1 = gb[this.ix].author;
        author1.ep = author1.ep + 1;
        this.ep = this.ep + 1;
      }
      else {
        // マークがなければそのままの位置でマーク

      }
    }
  }
  else if(this.type==3){
    // pirate type
    if (this.iy==0) {
      if (this.gb[this.ix].st==1) {
        var author1;
        // マーク済の位置にあれば、マークを自分に変更
        gb[this.ix].author = this;
      }
      else {
        // マークがなければそのままの位置でマーク

      }
    }
  }

}
copygame_agent.prototype.move = function() {
  // consumer type
  if(this.ix < 0 || this.ix>=600) {
    console("heelo")
  }
  if (this.x < 10  || this.x > 580) {
     this.vx = - this.vx;
  }   // gravity
  if (this.y > 0) {
    this.vy -= 1;
  }
  // motion
  this.x += this.vx;
  if (this.x <0) this.x=0;
  if (this.x>=600) this.x=599;
  this.y += this.vy;
  // collision with ground
  if (this.y<0) {
    this.y=0;
    this.vy=0;
  }
}

// show : 表示
copygame_agent.prototype.show = function() {
  // 円を描画
    // y 座標を整数に変換してから描画する
    var x1,y1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);


    c1.beginPath();
    c1.arc(x1,330-y1, this.ep, 0, Math.PI * 2);
```

```
    if(this.type==1){
      c1.fillStyle = 'rgb(0,255,128)'; // 緋色
    }else if (this.type == 2){
      c1.fillStyle = 'rgb(0,128,255)'; // 緋色
    }else{
      c1.fillStyle = 'rgb(255,128,128)'; // 緋色
    }
    c1.shadowColor = 'rgb(0,0,0)';   // 影
    c1.shadowOffsetX = 5;
    c1.shadowOffsetY = 5;
    c1.shadowBlur = 5;
    c1.fill();
}
```

copygame_agent.js - Experiment 3

```
// copygame02_01
// 2019/6/18
// 概要 海賊版業者の導入
// copygame_agent のクラスの定義
// 初期化
// gb 長さ 600 の配列
//   use count を使用することに 1 増加する
//   5 回に達したら消去
// copygame02_01 ルール
// エージェントは 2 タイプ
//  type 1 creator
//    ep 初期値 = 4
//    1%の確率で行う。
//    エネルギーが 2 以上なら自分を author として記録する
//    エネルギーは 1 減らす
//  type 2 consumer
//   Author が登録されている場合
//     author のエネルギーを +1 増加する
//     ジャンプする
//   マークされていない場合
//     自由運する
//  type 3 pirates
//     もし st=1 ならインデックスを自分に書き換える
//     エネルギーは減らさない
//

// class definition : game_cell
// ゲームのます目の定義
function game_cell(x1,y1) {
  this.x=x1;
  this.y=y1;
  this.st = 0; // status = 0
  this.author = null;    // copy game agent marking here
  this.ucount = 0;  // use count
}

game_cell.prototype.show=function(){
   // rect を描画
    // y 座標を整数に変換してから描画する
    var x1,y1,h1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);
    c1.beginPath();
    h1 = this.st*5;
    if(this.st==1){
      c1.fillStyle = 'rgb(0,0,128)'; // 黄色
    }else{
      c1.fillStyle = 'rgb(128,255,255)'; // 水色
    }
    c1.rect(x1,350, 5,5);
    c1.fill();
}

game_cell.prototype.use=function(){
  // このゲームセルのコンテンツを使う
  author1 = this.author;
  author1.ep = author1.ep + 1;
  this.ucount = this.ucount + 1;
  if (this.ucount >=5) {
    this.author = null;
    this.st = 0;
    this.ucount=0;
  }

}

// (ix,iy)は (x.y)の整数値
// 初期化により x,vx はランダムに
// 10<x<580, -25<x<25
// y, vy は 0 に初期化

function copygame_agent(aid1,gb1,ty1) {
  this.aid=aid1;
  this.gb = gb1;    // game board
  this.type = ty1;    // agent type = 2
  this.x = Math.random() * 570 + 10;
  this.y = 0;
  this.vx = Math.random() * 50 - 25;
  this.vy = 0;
  this.ep = 4;  //  energy point = 0
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);
}


// move : 移動
// 一番上（iy=0)
//   gb[this.ix].st=0 なら gb[this.ix].st を 1 に
//   gb[this.ix].st=1 なら
// y 座標を 1, vy を 10 にする

copygame_agent.prototype.progress = function() {
  this.ix=Math.floor(this.x);
  this.iy=Math.floor(this.y);

 if(this.type==1){
    // creator type
    if(this.iy==0 && this.ep>0){
      // エネルギーが残っており、iy=0 の場合
      var rr=Math.random();
      if (rr < 0.02){
        // 2% の確率でエネルギーを減らす
        this.ep = this.ep - 1;
      }
      if (rr < 0.01){
        // 1%の確率で作品を残す
        this.gb[this.ix].st = 1;
        this.gb[this.ix].author=this;
      }
    }
  }
  else if(this.type==2) {
    if (this.iy==0) {
      if (this.gb[this.ix].st==1) {
        var author1;
        // マーク済の位置にあれば、ジャンプ
        this.y = 1;
        this.vy = this.ep; // エネルギー分ジャンプ

        gb[this.ix].use();
        // author1 = gb[this.ix].author;
        // author1.ep = author1.ep + 1;
        this.ep = this.ep + 1;
      }
      else {
        // マークがなければそのままの位置でマーク

      }
    }
  }
  else if(this.type==4){
    // pirate type
    if (this.iy==0) {
      if (this.gb[this.ix].st==1) {
        var author1;
        // マーク済の位置にあれば、マークを自分に変更
        gb[this.ix].author = this;
      }
      else {
        // マークがなければそのままの位置でマーク

      }
    }
  }

}
copygame_agent.prototype.move = function() {
  // consumer type
  if(this.ix < 0 || this.ix>=600) {
    console("heelo")
  }
  if (this.x < 10  || this.x > 580) {
     this.vx = - this.vx;
  }   // gravity
  if (this.y > 0) {
    this.vy -= 1;
  }
  // motion
  this.x += this.vx;
  if (this.x <0) this.x=0;
  if (this.x>=600) this.x=599;
  this.y += this.vy;
  // collision with ground
  if (this.y<0) {
    this.y=0;
    this.vy=0;
  }
}

// show : 表示
copygame_agent.prototype.show = function() {
  // 円を描画
    // y 座標を整数に変換してから描画する
    var x1,y1;
    x1 = Math.floor(this.x);
    y1 = Math.floor(this.y);


    c1.beginPath();
    c1.arc(x1,330-y1, this.ep, 0, Math.PI * 2);
    if(this.type==1){
      c1.fillStyle = 'rgb(0,255,128)'; // 緋色
    }else if (this.type == 2){
      c1.fillStyle = 'rgb(0,128,255)'; // 緋色
    }else{
      c1.fillStyle = 'rgb(255,128,128)'; // 緋色
    }
    c1.shadowColor = 'rgb(0,0,0)';   // 影
    c1.shadowOffsetX = 5;
    c1.shadowOffsetY = 5;
    c1.shadowBlur = 5;
    c1.fill();
}
```