Лабораторная работа № 4

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И КЛЕТОЧНЫЕ АВТОМАТЫ В GNU OCTAVE (MATLAB)

Цель работы: знакомство с применением клеточных автоматов в решении задач математического моделирования, настройка клеточного автомата.

Задание: изучить теоретическую часть и пример кода клеточного автомата, найти вкравшуюся в код ошибку. Изменяя входные параметры системы в соответствии с заданием, изучить их влияние на результат работы модели. Подготовить содержательный отчет, сделать выводы.

Примечание. Для выполнения настоящей работы используется бесплатная программа GNU Octave (http://www.octave.org), имеющая существенную совместимость с Matlab и близость с Octave.

1. Представление о клеточных автоматах

Клеточный автомат (КА) — дискретная модель, изучаемая в математике, теоретической биологии, физике, гидравлике и т.д. Включает регулярную решётку ячеек, каждая из которых может находиться в одном из конечного множества состояний, таких как 1 и 0.

Решетка может быть любой размерности (соотношение длины сторон). Для каждой ячейки определено множество ячеек, называемых окрестностью. К примеру, окрестность может быть определена как все ячейки на расстоянии не более 2 от текущей (*окрестность фон Неймана ранга 2*).

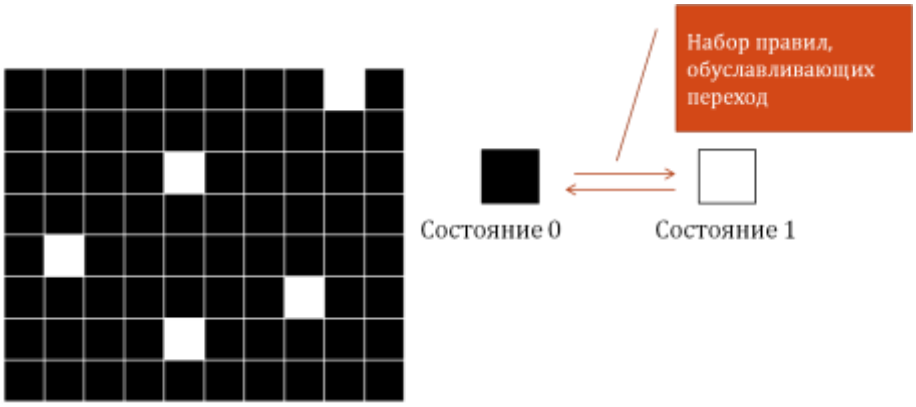

Рис. 1. *Схематическое представление простейшего клеточного автомата*

Для работы клеточного автомата требуется задание начального состояния всех ячеек, и правил перехода ячеек из одного состояния в другое. На каждой итерации (ходе), используя правила перехода и состояния соседних ячеек, определяется новое состояние каждой ячейки.

Обычно правила перехода одинаковы для всех ячеек и применяются сразу ко всей решётке.

Пример: игра «Жизнь» Дж Конвея. *Для моделирования поведения популяции одноклеточных простейших используется решетка (матрица) размерностью100x100 элементов. Решетка заселяется некоторым исходным количеством организмов. Затем, с течением времени для всей решетки выполняется ряд правил, управляющих поведением «организмов», т.е. элементов матрицы:*

1) Выживание. Каждая фишка, имеющая вокруг себя две или три соседние фишки, выживает и переходит в следующее поколение.

2) Гибель. Каждая фишка, у которой больше трёх соседей, погибает, то есть снимается с доски из-за перенаселённости.

Каждая фишка, вокруг которой свободны все соседние клетки или же занята всего одна клетка, погибает от одиночества.

3) Рождение. Если число фишек, с которыми граничат какая-нибудь пустая клетка, в точности равно трём (не больше и не меньше), то на этой клетке происходит рождение нового «организма», то есть следующим ходом на неё ставится одна фишка.

Код основного исполняемого модуля игры «Жизнь» convay.m (автор Iain Haslam, exolete.com):

```
% life.m - Conway's Game of Life
%
% A grid of dead and living cells is made.
% Cells are born to three adjacent parents,
% and die of overcrowding or loneliness.
%        Iain Haslam, December 2005

len=50; GRID=int8(rand(len,len));
up=[2:len 1]; down=[len 1:len-1]; %the world is round
colormap(gray(2));
for i=1:1000
    neighbours=GRID(up,:)+GRID(down,:)+GRID(:,up)+GRID(:,down)+...
        GRID(up,up)+GRID(up,down)+GRID(down,up)+GRID(down,down);
    GRID = neighbours==3 | GRID & neighbours==2;
    image(GRID*2); pause(0.02);
    axis tight;
end
```

После начала игры, «популяция» быстро начинает сокращаться, уступая место долгоживущим структурам – «глайдерам», которые способны двигаться. Их столкновения друг с другом приводят к разрушению стабильности.

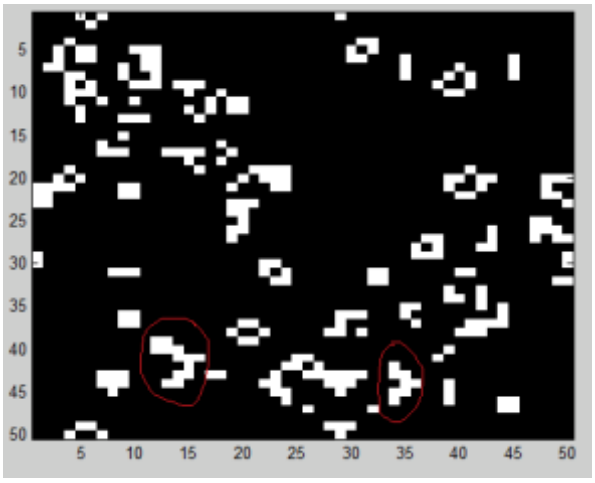

Рис. 2. *Долгоживущие самодвижущиеся структуры игры «Жизнь» – «глайдеры» (обведены красным)*

Самостоятельно. *Разобрать структуру программы «Жизнь». Изучить поведение системы при изменении входных параметров –* ***размеров поля***, ***времени жизни клеток*** *и других.*

Рекомендации. В Matlab/Octave для вывода строковых сообщений используются следующие функции. Преобразование числа в строку, выполняемое в Octave функцией *string(Число),* в Matlab/Octave требует применения функции num2str(Число). Объединение строк, осуществляемое в Octave оператором сложения, в Matlab/Octave требует strcat(Строка1, Строка2, …Строка n).

Существуют и другие несовместимости. При возникновении технических затруднений, необходимо читать справку по функциям, применяемым в Matlab/GNU Octave.

Прервать выполнение программы можно, нажав в командном окне комбинацию клавиш «Ctrl+C».

Внимание! Начало координат изображения в Matlab/Octave/Octave – левый верхний угол. Столбцы и строки графического изображения развернуты на 90 относительно его матричного представления.

4.2. Создание клеточного автомата-модели в GNU Octave / Matlab

Для создания клеточного автомата возьмем следующую ситуацию. В резервуаре с жидкостью на дно падает дробинка. На нее воздействуют Архимедова сила F_a, стремящаяся вытолкнуть дробинку и сила тяжести F_g, направленная ко дну сосуда. Если плотность жидкости больше плотности дробинки, последняя плавает, в противном случае – тонет (рис. 3). Сопротивлением жидкости и ее вязкостью при падении в ней дробинки следует пренебречь.

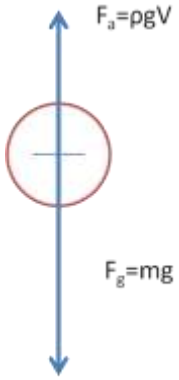

Рис. 3. *Схема сил, действующих на падающих в жидкости дробинку. Fg – сила тяжести, Fa – Архимедова сила*

Модель, реализуемая в виде программы, должна удовлетворять следующим условиям:

1. Дробинка движется с ускорением;
2. Если $F_a > F_g$ дробинка не тонет, выдается сообщение о плавании дробинки;
3. Максимальный срок выполнения программы 100 кадров (шагов) или

момент касания дна;

4. По завершении работы модели выдается график изменения глубины дробинки и ее ускорения (рис. 4).

Простой клеточный автомат, реализующий падение дробинок в воде, показан ниже (pellet_ca.m). Его необходимо запустить, понять как работает и отразить в отчете особенности его работы:

```
%pellets_ca.m
clear; close all;
len=50; GRID=rand(len,len)>0.99;
colormap(gray(2));
g=9.81;  %gravity constant
w_Rho=1000; %density of fluid
p_Rho=11000; %density
p_m=0.050; %mass
p_V=p_m/p_Rho; %volume

for i=1:50
    Fp=p_m*g.*GRID;
    Fa=w_Rho*g*p_V.*GRID;
    dF=Fp-Fa;
    [r c]=find(dF>0);
    r=r+1;
    r(r>=len)=len;
    ind=sub2ind(size(GRID),r,c);
    GRID=zeros(len,len);
    GRID(ind)=1;
    imagesc(GRID); pause(0.05);
end;
```

Код Matlab *с ошибками* для программы, реализующей падение дробинки в резервуар дан в приложении Б. Эта программа, выводит аналитический результат, однако, строго говоря, клеточным автоматом не является (почему?).

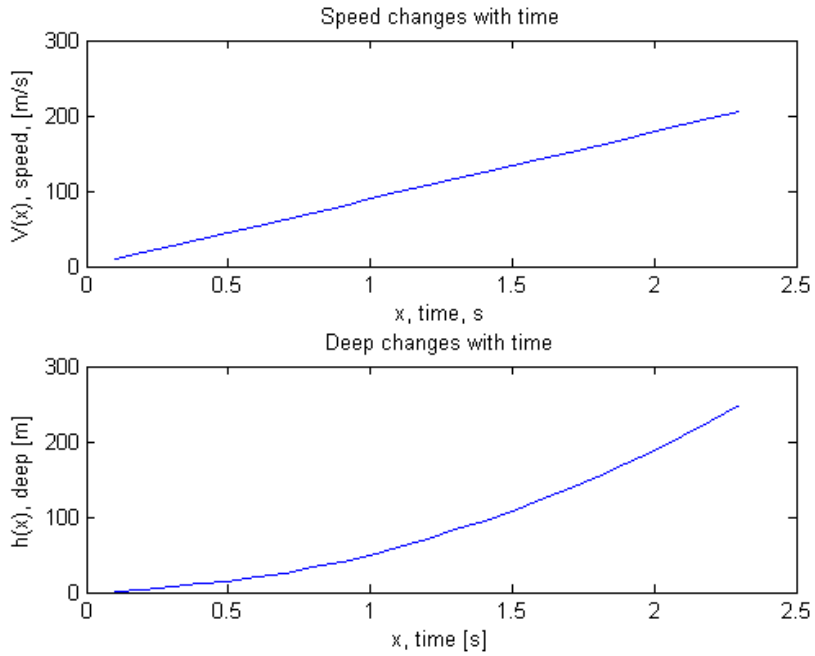

Рис. 4. Графики, создаваемые программой по завершении ее выполнения. Верхний – изменения скорости со временем, нижний – увеличение глубины со временем.

Задание для самостоятельной работы

- Исправить код Matlab, найдя ошибки, препятствующие его нормальной работе (*pellet_b.m*);
- Используя комментарии в коде программ, детально выяснить порядок их работы, значение переменных и алгоритмы применяемые в функциях;
- Понять, и отразить в отчете работу программы *pellet_ca.m.* Усовершенствовать программу, добавив возможность накопления частиц в несколько слоев;
- Использовать программу для исследования падения дробинки плотности, заданной в варианте (Приложение А), в жидкости №1 и №2;
- Описать проводимый эксперимент, сравнить результаты и обосновать выводы;
- Ответить на вопрос: «Как поведет себя дробинка из свинца в ртути?». Обосновать ответ аналитически.
- **Составить блок-схему работы программы «Падение дробинки»**

Требования к отчету

Отчет о выполнении лабораторной работы должен содержать:

- тему и цель лабораторной работы на титульной странице;
- оглавление;
- необходимые теоретические сведения по теме решаемых задач;
- исходные данные и последовательность их обработки (преобразования);
- поэтапный расчет в виде вставок кода Octave/Matlab в объектах «Подпись»;
- результаты выполнения кода Octave/Matlab в виде вывода в командном окне;
- графики, выполненные в Octave/Matlab;
- интерпретацию результатов работы моделей, выполненные самостоятельные задания;
- отметку преподавателя о выполнении лабораторной работы.

Рекомендуемая литература

1. Алексеев Е.Р. Octave: Решение инженерных и математических задач / Е.Р. Алексеев, О.В.Чеснокова, Е. А.Рудченко // М. : ALT Linux ; БИНОМ. Лаборатория знаний, 2008. 260 с.
2. Павлова М.И. Руководство по работе с пакетом Octave 2.6. 2003. 200 с.
3. Поршнев С. В. Компьютерное моделирование физических процессов в пакете MATLAB / М.:, 2003. 593 с.
4. Эдвардс Ч., Пенни Д. Дифференциальные уравнения и краевые задачи: моделирование и вычисление с помощью *Mathematica, Mapple и Matlab* / М.: 2008, 1104 с.

Варианты исходных данных для решения самостоятельной задачи.

Параметры: $\rho_{f,}$ №1 – плотность жидкости №1, $\rho_{f,}$ №2 – плотность жидкости №2, ρ_p – плотность дробинки

Вариант	$\rho_{f,}$ №1	$\rho_{f,}$ №2	ρ_p
1	550	2500	9000
2	825	4480	10083
3	597	3999	9912
4	526	3205	8012
5	425	2324	8561
6	756	2293	10051
7	582	4066	9714
8	408	3282	7919
9	579	2143	9800
10	368	4026	9507
11	581	2392	10306
12	386	4149	8249
13	464	2877	7142
14	678	2407	9185
15	513	2863	10175
16	683	4312	9278
17	308	4112	7740
18	658	4433	8567
19	383	3930	7323
20	637	4784	10749
21	309	2688	9958
22	464	4020	11745
23	740	2085	7778
24	726	2170	11316
25	309	4993	7446
26	357	4620	11329
27	593	3250	9555
28	941	4828	10039
29	310	4840	11481
30	433	3626	10787
31	489	2761	7316
32	863	2301	7539
33	570	2547	10720
34	493	3379	10313
35	587	2128	11173
36	964	3426	8776
37	477	3655	8119
38	716	2634	10668
39	446	4891	10074
40	399	3931	10552
41	873	4258	8187
42	312	2434	7582
43	427	2897	11802
44	875	4457	9419
45	546	3897	8065
46	519	3353	11014
47	677	4999	10135
48	896	4985	8429
49	540	4982	9134
50	923	2894	8946

Программа «Падение дробинки»

(с ошибками!)

```
%высота "стакана" h=250 м. Масса дробинки m=0.010 кг. Rho [kg/m3]
%settings
glass=zeros(250,50); %reservoir size of 250x50 px
%environment, gravity
g=9.81;          %gravity constant
%fluid init
w_Rho=1000;      %density of fluid

%pellet initial state and position
p_Rho=11000; %density
p_xy=[25 1]; %coordinates
p_sp=0;      %initian speed
p_sz=2E-6;   %size
p_m=0.050;   %mass
p_V=p_m/p_Rho; %volume

%action
imshow(glass); %output on start

%data logging
sec_x=[]; %arrays for pellet state logging
speed_y=[];
deep_y=[];

%2 forces act in that model
%Fp=mg  Fa=Rho_g_V - gravity and Archimeds forces description
Fp=p_m*g;
Fa=w_Rho*g*p_V;
delta_F=Fa-Fp; %<0 sinks, >0 floats
p_a=abs(delta_F)/p_m; %pellet's acceleration, nonzero, abs(delta_F)
t=0.1; %discretisation of time in experiment 0.1 second
size_glass=size(glass); %reservoirs height and width

for i=1:100 %100 falling of 100 frames max
 p_sp=p_sp+p_a; %speed acceleration
 dS=p_sp*t;  %speed delta

 if (delta_F<0) %pellet sinks, because Fa<Fg
 p_xy=p_xy+[0 round(dS)]; %enlarge depth
 end;
 if (delta_F>0) %pellet sinks, because Fa<Fg
     disp('Pellet is floating!)

 if p_xy(2)>size_glass(1) %checking of bottom reaching
     disp ('bottom is reached');
     break; %break up the cycle in that case
 end;

 %preparation of animated frame before output
 pict_out=glass;
 pict_out(p_xy(2),p_xy(1))=1;

 %changing frame in visual picture
 imagesc(pict_out);

 %data logging
 sec_x=[sec_x (i*t)];
 speed_y=[speed_y p_sp];
 deep_y=[deep_y p_xy(2)];
 %pict heading
 heading=strcat('Frame ' num2str(i),' of 100; ', 'T [sec] =', num2str((i*t)));
 title(heading);
 pause (1);
end;
%final graphics output
figure;
subplot(2,1,1);
plot(sec_x,speed_y);
title ('Speed changes with time');
xlabel('x, time, s');
ylabel('V(x), speed, [m/s]');
subplot(2,1,2);
plot(sec_x,deep_y);
title ('Deep changes with time');
xlabel('x, time [s]');
ylabel('h(x), deep [m]');
```