# Sejtautomaták

Szőke Kálmán Benjamin - SZKRADT.ELTE

2012. május 17.

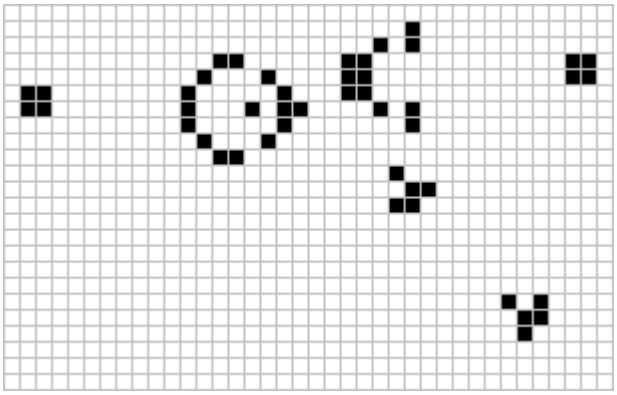
## 1. Bevezetés

A legismertebb sejtautomaták egyike a John Conway által kifejlesztett életjáték. Ennek a sejtmozaik háttere olyan, mint a négyzetrácsos füzetlap: ebben a szerkezetben minden sejtnek nyolc sejtszomszédja van. Az egyes sejtek kétféle állapotban lehetnek: élő vagy halott állapotban. Az idő, ahogy minden egyszerűbb sejtautomatában, diszkrét időegységekben telik, és a sejtek működése a következő:

- Egy olyan sejt helyére, amely halott, de három élő sejtszomszédja van, élő sejt születik.
- Egy olyan sejt, amely élő volt, és két vagy három szomszédja is élő volt, életben marad.
- Az összes többi, másmilyen környezetű sejt halott lesz a következő lépésben.

Az Életjáték sok érdekes szerkezet mozgását, gyarapodását, vagy elmúlását és sajátos alakzatok tartós fönnmaradását tudja szimulálni. A szimulációs képességének erőssége abban rejlik, hogy maga a játék Turing-teljes, vagyis bármit, amit ki lehet algoritmikusan számolni, az kiszámolható vele. Gardner írta róla, hogy: ,,A játék, amit Conway alkotott azonnal híres lett, de közben egy teljesen új matematikai kutatási terület felé is megnyitotta az utat, a sejtautomaták felé.” Conway egyik sejtése az volt, hogy a növekedésnek van egy felső korlátja, és 50 dolláros jutalmat ajánlott annak aki igazolni, vagy cáfolni tudja az állítását 1970 előtt. A sejtés megcáfolására az egyik út az, ha az ember felfedez egy mintát, egy úgynevezetett pisztolyt, amely tőle elfelé mozgó alakzatokat lő ki, úgynevezett siklókat. A másik módszerhez egy puffer vonatot kell megalkotni, amely, ahogy halad előre, hátrahagyja a füstjét. A díjat Bill Gosper által vezetett csapat nyerte el a massachusetts-i

egyetemről ugyanezen év novemberében. Mára már Gosper glider gun-ként ismert az általuk kifejlesztett minta.



1. ábra. Gosper glider gun

# 2. Algoritmus

A feladatok elvégzéséhez C++ nyelven írt programot készítettem, ami kimenetként számozott .dat fájlokat készít a diszkrét időegységekben kialakúlt sejtmozaikról. Formailag ezek a kimeneti fájlok mátrixoknak felelnek meg, 0 vagy 1-es értékekkel. Az animáció elkészítéséhez Matlab-ot használtam, amivel az elkészített mátrixokat animált .gif-be mentettem ki.

## 2.1. Conway féle Élet-játék

```
#include <fstream>
#include <algorithm>
#include <iostream>
#include <cmath>
#include <string>
#include <sstream>

using namespace std;

int main(int argc, char **argv)
{
  string matrix_file;
  unsigned int N = atoi(argv[2]); //matrx szélesség
  unsigned int M = atoi(argv[1]); //matrix magasság
  matrix_file = argv[3];
  int n;
  cout<< "Add meg n erteket: ";
  cin >> n;
  ifstream Mat_file(matrix_file.c_str());
```

```
int **matrix = new int*[M];
for ( unsigned int i=0; i<M; i++)
{
    matrix[i] = new int[N];
}
int **temp_matrix = new int*[M];
for ( unsigned int i=0; i<M; i++)
{
    temp_matrix[i] = new int[N];
}

for ( unsigned int m=0; m<M  ; m++)
{
    for ( unsigned int n=0; n<N  ; n++)
    {
        Mat_file >> matrix[m][n];
        temp_matrix[m][n]=matrix[m][n];
    }
}
Mat_file.close();

unsigned int suma; //szomszéd szám
for ( unsigned int i=0; i<10  ; i++)
{
    ostringstream file;
    file << i << ".dat";
    ofstream result(file.str().c_str());
    for ( unsigned int k=1; k<(M-1); k++)
    {
        for ( unsigned int j=1; j<(N-1); j++)
        {
        suma=(matrix[k-1][j-1]+matrix[k-1][j]+matrix[k-1][j-1]+matrix[k+1][j-1]+
        +matrix[k+1][j]+matrix[k+1][j-1]+matrix[k][j-1]+matrix[k][j+1]);
            if(suma==(n+1) && matrix[k][j]==0)
                temp_matrix[k][j]=1;
            if(suma==(n+1) || suma==n && matrix[k][j]==1)
                temp_matrix[k][j]=matrix[k][j];
            if(suma>(n+1) || suma<n)
                temp_matrix[k][j]=0;
        }
    }
    for ( unsigned int m=0; m<M  ; m++)
    {
        for ( unsigned int n=0; n<N  ; n++)
        {
            result << temp_matrix[m][n] << "\t";
            matrix[m][n]=temp_matrix[m][n];
        }
        result << endl;
    }
}


for(int i=0; i<N; ++i)
{
    delete[] matrix[i];
}
delete[] matrix;
for(int i=0; i<N; ++i)
{
    delete[] temp_matrix[i];
```

```
  }
  delete[] temp_matrix;
  return 0;
}
```

## 2.2. Élet-játék (nyílt peremfeltétel)

```
#include <fstream>
#include <algorithm>
#include <iostream>
#include <cmath>
#include <string>
#include <sstream>

using namespace std;

int main(int argc, char **argv)
{
  string matrix_file;
  unsigned int N = atoi(argv[2]); //matrx szélesség
  unsigned int M = atoi(argv[1]); //matrix magasság
  matrix_file = argv[3];
  int n;
  cout<< "Add meg n erteket: ";
  cin >> n;
  ifstream Mat_file(matrix_file.c_str());

  int **matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      matrix[i] = new int[N];
  }
  int **temp_matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      temp_matrix[i] = new int[N];
  }

  for ( unsigned int m=0; m<M  ; m++)
  {
      for ( unsigned int n=0; n<N  ; n++)
      {
          Mat_file >> matrix[m][n];
          temp_matrix[m][n]=matrix[m][n];
      }
  }
  Mat_file.close();

  unsigned int suma; //szomszéd szám
  for ( unsigned int i=0; i<10  ; i++)
  {
      ostringstream file;
      file << i << ".dat";
      ofstream result(file.str().c_str());
      for ( unsigned int k=0; k<M; k++)
      {
          for ( unsigned int j=0; j<N; j++)
          {
              if (k==1 && j==1)
              {
```

```
                suma=matrix[k+1][j]+matrix[k+1][j+1]+matrix[k][j+1];
            }
            else if (k==M && j==N)
            {
                suma=matrix[k-1][j]+matrix[k-1][j-1]+matrix[k][j-1];
            }
            else if (k==1 && j==N)
            {
                suma=matrix[k+1][j]+matrix[k+1][j-1]+matrix[k][j-1];
            }
            else if (k==M && j==1)
            {
                suma=matrix[k-1][j]+matrix[k][j+1]+matrix[k-1][j+1];
            }
            else if (k==1)
            {
                suma=matrix[k+1][j+1]+matrix[k+1][j-1]+matrix[k][j-1]+matrix[k][j+1];
            }
            else if (k==M)
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k][j-1]+matrix[k][j+1];
            }
            else if (j==1)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k-1][j+1]+matrix[k+1][j+1];
            }
            else if (j==N)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k+1][j-1]+matrix[k-1][j-1];
            }
            else
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k+1][j+1]+matrix[k+1][j-1]+
                +matrix[k][j-1]+matrix[k][j+1];
            }

            if(suma==(n+1) && matrix[k][j]==0)
                temp_matrix[k][j]=1;
            if(suma==(n+1) || suma==n && matrix[k][j]==1)
                temp_matrix[k][j]=matrix[k][j];
            if(suma>(n+1) || suma<n)
                temp_matrix[k][j]=0;
        }
    }
    for ( unsigned int m=0; m<M  ; m++)
    {
        for ( unsigned int n=0; n<N  ; n++)
        {
            result << temp_matrix[m][n] << "\t";
            matrix[m][n]=temp_matrix[m][n];
        }
        result << endl;
    }
}


for(int i=0; i<N; ++i)
{
    delete[] matrix[i];
}
delete[] matrix;
for(int i=0; i<N; ++i)
```

```
  {
      delete[] temp_matrix[i];
  }
  delete[] temp_matrix;
  return 0;
}
```

## 2.3. Élet-játék (élő peremfeltétel)

```
#include <fstream>
#include <algorithm>
#include <iostream>
#include <cmath>
#include <string>
#include <sstream>

using namespace std;

int main(int argc, char **argv)
{
  string matrix_file;
  unsigned int N = atoi(argv[2]); //matrx szélesség
  unsigned int M = atoi(argv[1]); //matrix magasság
  matrix_file = argv[3];
  int n;
  cout<< "Add meg n erteket: ";
  cin >> n;
  ifstream Mat_file(matrix_file.c_str());

  int **matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      matrix[i] = new int[N];
  }
  int **temp_matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      temp_matrix[i] = new int[N];
  }

  for ( unsigned int m=0; m<M  ; m++)
  {
      for ( unsigned int n=0; n<N  ; n++)
      {
          Mat_file >> matrix[m][n];
          temp_matrix[m][n]=matrix[m][n];
      }
  }
  Mat_file.close();

  unsigned int suma; //szomszéd szám
  for ( unsigned int i=0; i<10  ; i++)
  {
      ostringstream file;
      file << i << ".dat";
      ofstream result(file.str().c_str());
      for ( unsigned int k=0; k<M; k++)
      {
          for ( unsigned int j=0; j<N; j++)
          {
```

```
            if (k==1 && j==1)
            {
                suma=matrix[k+1][j]+matrix[k+1][j+1]+matrix[k][j+1]+5;
            }
            else if (k==M && j==N)
            {
                suma=matrix[k-1][j]+matrix[k-1][j-1]+matrix[k][j-1]+5;
            }
            else if (k==1 && j==N)
            {
                suma=matrix[k+1][j]+matrix[k+1][j-1]+matrix[k][j-1]+5;
            }
            else if (k==M && j==1)
            {
                suma=matrix[k-1][j]+matrix[k][j+1]+matrix[k-1][j+1]+5;
            }
            else if (k==1)
            {
                suma=matrix[k+1][j+1]+matrix[k+1][j-1]+matrix[k][j-1]+matrix[k][j+1]+3;
            }
            else if (k==M)
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k][j-1]+matrix[k][j+1]+3;
            }
            else if (j==1)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k-1][j+1]+matrix[k+1][j+1]+3;
            }
            else if (j==N)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k+1][j-1]+matrix[k-1][j-1]+3;
            }
            else
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k+1][j+1]+matrix[k+1][j-1]+
                +matrix[k][j-1]+matrix[k][j+1];
            }

            if(suma==(n+1) && matrix[k][j]==0)
                temp_matrix[k][j]=1;
            if(suma==(n+1) || suma==n && matrix[k][j]==1)
                temp_matrix[k][j]=matrix[k][j];
            if(suma>(n+1) || suma<n)
                temp_matrix[k][j]=0;
        }
    }
    for ( unsigned int m=0; m<M  ; m++)
    {
        for ( unsigned int n=0; n<N  ; n++)
        {
            result << temp_matrix[m][n] << "\t";
            matrix[m][n]=temp_matrix[m][n];
        }
        result << endl;
    }
}


for(int i=0; i<N; ++i)
{
    delete[] matrix[i];
}
```

```
  delete[] matrix;
  for(int i=0; i<N; ++i)
  {
      delete[] temp_matrix[i];
  }
  delete[] temp_matrix;
  return 0;
}
```

## 2.4. Élet-játék (periódikus peremfeltétel)

```
#include <fstream>
#include <algorithm>
#include <iostream>
#include <cmath>
#include <string>
#include <sstream>

using namespace std;

int main(int argc, char **argv)
{
  string matrix_file;
  unsigned int N = atoi(argv[2]); //matrx szélesség
  unsigned int M = atoi(argv[1]); //matrix magasság
  matrix_file = argv[3];
  int n;
  cout<< "Add meg n erteket: ";
  cin >> n;
  ifstream Mat_file(matrix_file.c_str());

  int **matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      matrix[i] = new int[N];
  }
  int **temp_matrix = new int*[M];
  for ( unsigned int i=0; i<M; i++)
  {
      temp_matrix[i] = new int[N];
  }

  for ( unsigned int m=0; m<M  ; m++)
  {
      for ( unsigned int n=0; n<N  ; n++)
      {
          Mat_file >> matrix[m][n];
          temp_matrix[m][n]=matrix[m][n];
      }
  }
  Mat_file.close();

  unsigned int suma; //szomszéd szám
  for ( unsigned int i=0; i<10  ; i++)
  {
      ostringstream file;
      file << i << ".dat";
      ofstream result(file.str().c_str());
      for ( unsigned int k=0; k<M; k++)
      {
```

```
        for ( unsigned int j=0; j<N; j++)
        {
            if (k==1 && j==1)
            {
                suma=matrix[k+1][j]+matrix[k+1][j+1]+matrix[k][j+1]+5;
            }
            else if (k==M && j==N)
            {
                suma=matrix[k-1][j]+matrix[k-1][j-1]+matrix[k][j-1]+5;
            }
            else if (k==1 && j==N)
            {
                suma=matrix[k+1][j]+matrix[k+1][j-1]+matrix[k][j-1]+5;
            }
            else if (k==M && j==1)
            {
                suma=matrix[k-1][j]+matrix[k][j+1]+matrix[k-1][j+1]+5;
            }
            else if (k==1)
            {
                suma=matrix[k+1][j+1]+matrix[k+1][j-1]+matrix[k][j-1]+matrix[k][j+1]+3;
            }
            else if (k==M)
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k][j-1]+matrix[k][j+1]+3;
            }
            else if (j==1)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k-1][j+1]+matrix[k+1][j+1]+3;
            }
            else if (j==N)
            {
                suma=matrix[k-1][j]+matrix[k+1][j]+matrix[k+1][j-1]+matrix[k-1][j-1]+3;
            }
            else
            {
                suma=matrix[k-1][j+1]+matrix[k-1][j-1]+matrix[k+1][j+1]+matrix[k+1][j-1]+
                +matrix[k][j-1]+matrix[k][j+1];
            }

            if(suma==(n+1) && matrix[k][j]==0)
                temp_matrix[k][j]=1;
            if(suma==(n+1) || suma==n && matrix[k][j]==1)
                temp_matrix[k][j]=matrix[k][j];
            if(suma>(n+1) || suma<n)
                temp_matrix[k][j]=0;
        }
    }
    for ( unsigned int m=0; m<M  ; m++)
    {
        for ( unsigned int n=0; n<N  ; n++)
        {
            result << temp_matrix[m][n] << "\t";
            matrix[m][n]=temp_matrix[m][n];
        }
        result << endl;
    }
}


for(int i=0; i<N; ++i)
{
```

```
      delete[] matrix[i];
  }
  delete[] matrix;
  for(int i=0; i<N; ++i)
  {
      delete[] temp_matrix[i];
  }
  delete[] temp_matrix;
  return 0;
}
```

## 2.5. 2D homokdomb (Matlab)

```
function [n_t]=sand(N,i)
n_t=zeros(i,1);
%In=randi(7,N);
In=zeros(N);
[m n]=size(In);
figure;
colormap(gray);
imagesc(In);
f = getframe;
[im,map] = rgb2ind(f.cdata,256,'nodither');
pause(5)
for kor=1:i
    %m_rand=randi(n,1);
    %n_rand=randi(n,1);
    %In(m_rand,n_rand)=In(m_rand,n_rand)+1;
    In(m/2,n/2)=In(m/2,n/2)+1;
    for k=2:m-1
        for j=2:n-1
            if In(k,j)>=4
                In(k+1,j)=In(k+1,j)+1;
                In(k-1,j)=In(k-1,j)+1;
                In(k,j+1)=In(k,j+1)+1;
                In(k,j-1)=In(k,j-1)+1;
                In(k,j)=In(k,j)-4;
                n_t(kor)=n_t(kor)+4;
            end
        end
    end
In(1,:)=0;
In(:,1)=0;
In(n,:)=0;
In(:,n)=0;
imagesc(In)
f = getframe;
im(:,:,1,kor) = rgb2ind(f.cdata,map,'nodither');
pause(0.01)
end
imwrite(im,map,'animation.gif','DelayTime',0.01,'LoopCount',inf)
end
```

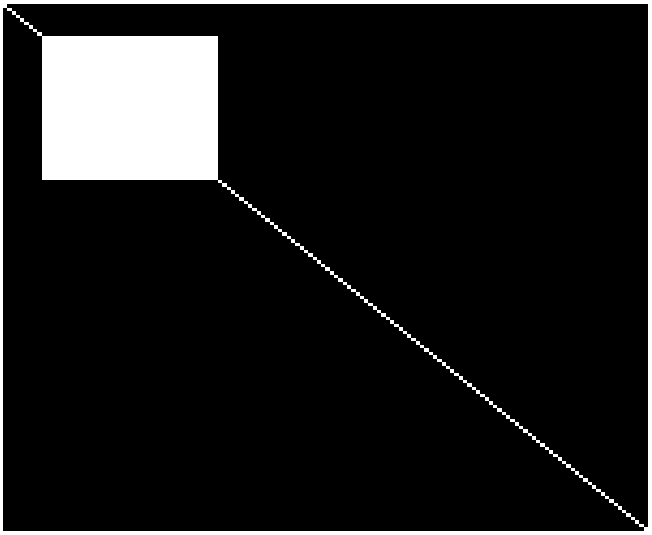
# 3. Eredmények

Az eredményeket az alábbi felsorolásokban írt .gif fájlokban animáltam. A képeken a fehéren látott részek értéke 1, a feketék pedig 0-ák.

## 3.1. 1. Feladat - Conway féle Élet-játék

`1fe_negyzet_n2.gif`

Az első feladatban kezdő sejtmozaiknak az alábbi elrendezést használtam.

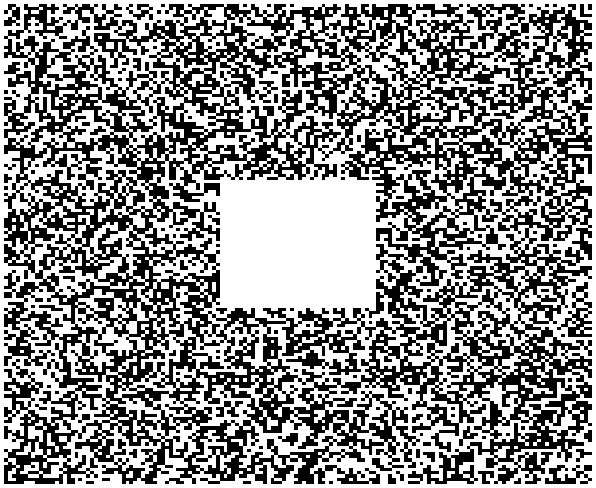


2. ábra. Négyzet és átló

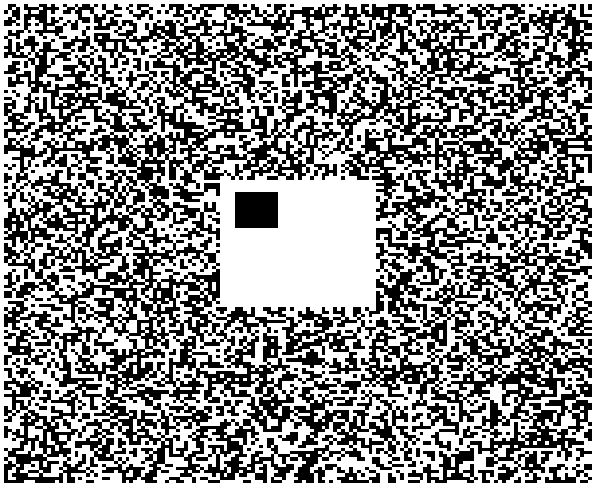
## 3.2. 2. Feladat - Módosított szabályok

`2fe_negyzet_n1.gif`
`2fe_n3.gif`
`2fe_n4.gif`
`2fe_n5.gif`
`2fe_n6.gif`
`2fe_n7.gif`
`2fe_n8.gif`

Kezdő sejtmozaiknak az alábbi elrendezéseket használtam.



3. ábra. Négyzet, és random háttér



4. ábra. Négyzet réssel, és random háttér

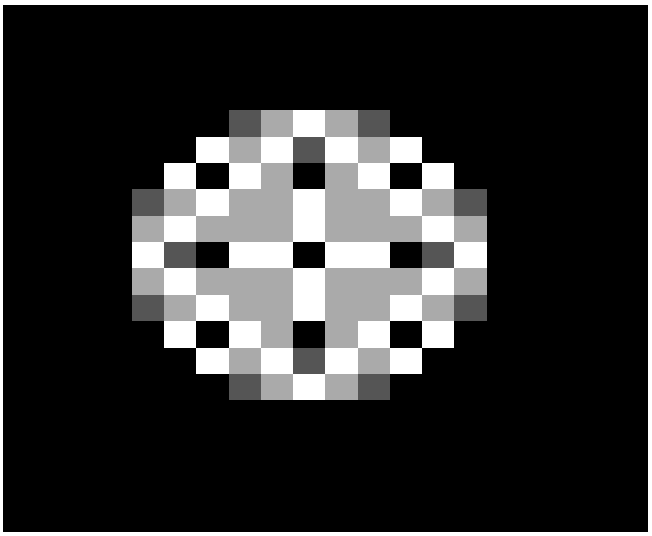
## 3.3. 3. Feladat - Peremfeltételek

`3fe_negyzet_elo.gif`
`3fe_negyzet_nyilt.gif`
`3fe_negyzet_nyilt_n3.gif`
`3fe_negyzet_peri.gif`

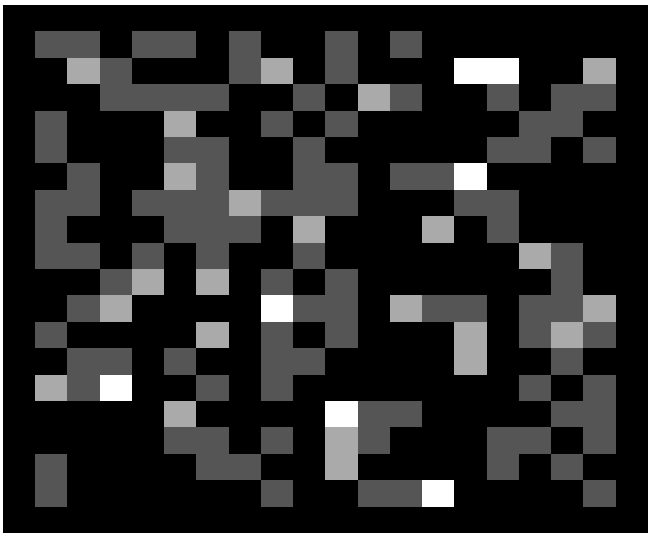
Kezdő sejtmozaiknak az első feladatban már bemutatott elrendezést használtam.

## 3.4. 4. Feladat - 2D homokdomb

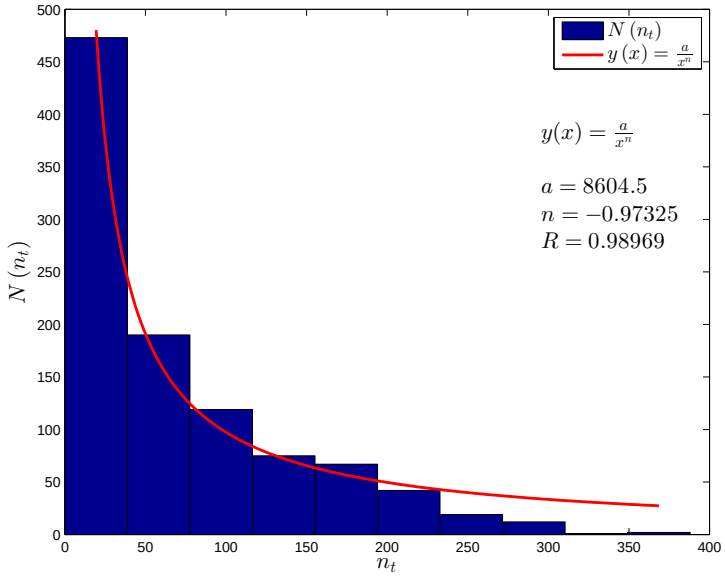
`sand_2d_center.gif`
`sand_2d_rand.gif`



5. ábra. Fejlődés középre ejtett homokszemekkel



6. ábra. Fejlődés véletlen helyre ejtett homokszemekkel




7. ábra. Power-law függvény illesztés

# Tartalomjegyzék

**1. Bevezetés** **1**

**2. Algoritmus** **2**
2.1. Conway féle Élet-játék . . . . . 2
2.2. Élet-játék (nyílt peremfeltétel) . . . . . 4
2.3. Élet-játék (élő peremfeltétel) . . . . . 6
2.4. Élet-játék (periódikus peremfeltétel) . . . . . 8
2.5. 2D homokdomb (Matlab) . . . . . 10

**3. Eredmények** **11**
3.1. 1. Feladat - Conway féle Élet-játék . . . . . 11
3.2. 2. Feladat - Módosított szabályok . . . . . 11
3.3. 3. Feladat - Peremfeltételek . . . . . 13
3.4. 4. Feladat - 2D homokdomb . . . . . 13

# Hivatkozások

[1] Jegyzet
http://complex.elte.hu/∼csabai/szamszim/lecture7/