Számítógépes Szimulációs Labor

4. Komplex rendszerek

Készítette:
Sztreha Balázs
Informatikus fizikus

November 19, 2007

1 Sejt automaták

Neumann János nevéhez fűződik az elsők közötti sejt automata modell (önreprodukáló gépek), amely megpróbálja megértetni a biológiai reprodukciót. Egy sejtautomata modell alap összetevői a következők:
- a folytonos koordinátákat (x, y, z) lecseréljük véges sok, térben rögzített cellára, sejtre (általában tömb vagy rács)
- a folytonos dinamikai függvényeket is közelítjük, az értékek diszkrét megválasztásával, az egyes cellák, sejtek helyein
- a folytonos időt diszkretizáljuk (dt időlépés)
- a dinamikai egyenletek egy helybeli szabállyal cseréljük le: egyes időlépéseknél, a cellák új értéket kapnak, attól függően milyen értékek vannak a szomszédban
- a cella vagy sejt értékek egyszerre vagy szinkronizáltan frissülnek

2 Homok modell

OpenGL alkalmazást használunk, mert így látványosabban, és szemléletesebben tudjuk bemutatni a programok futását.

2.1 sand1.cpp

A letöltött sand1.cpp fileban a *takeStep()* és *pileIsStable()* függvények felelősek a homokszemcsék viselkedéséért. 1 dimenzióban vizsgáljuk a rendszert, ennek az állapotát a *height[L]* tömb tárolja, amiben a homok-oszlop magassága van. A *stable[L]* tömbnek az elemei logikai változók, amiben az egyes oszlopok stabil-instabil tartjuk számon. Az L egész szám adja meg az oszlopok számát. 0 és L-1 közötti szám generálásával adjuk meg, melyik oszlop magasságát növeljük 1-el. Az i-eik oszlop instabil, ha a következő feltétel teljesül rá:

$$h_{i+1} - h_i > criticalSlope,$$

ahol a *criticalSlope* a kritikus különbség (ami alapesetben 1, de ezt változtathatjuk közben).

Feladatunk, hogy kimérjük a lavinák skálázását, ami azt jelenti, hogy ábrázoljuk hányszor fordultak elő a lavinák hosszai. Egy lavina hossza azt jelenti hogy, ha keletkezett egy instabil oszlop, akkor az aktuális homokszemcse mennyi távolságot tett meg, amíg újra stabil lett a rendszer. Ehhez bevezettem egy *avalanche[]* tömböt, aminek az i-eik argumentuma megadja, hogy hányszor

fordult elő i hosszúságú lavina. A következőképpen egészítettem ki a programot (inicializálás és lenullázás nélkül):

```
void takeStep() {
    if (pileIsStable()) {
       ++avalanche[A];
       A=0;
       int i = int(rand() / double(RAND_MAX) * L);
       ++height[i];
    } else {
       for (int i = 0; i < L; i++)
          if (!stable[i]) {
             ++A;
             ...
}
void mouse(int button, int state, int x, int y) {
    switch (button) {
      case GLUT_LEFT_BUTTON:
        if (state == GLUT_DOWN) {
            if (running) {
                glutIdleFunc(NULL);
                running = false;
                for (int k=0;k<L;k++)
                  std::cout << k << " " << avalanche[k] << "\n";
                std::cout << "\n\n";
                ...
}
```

Majd ezt a tömböt L=100 esetén, körülbelül 2000-3000 lépés múlva ábrázoltam (1. ábra). Sajnos technikai okok miatt nem sikerült log-log skálán "egyenest" illesztenem, viszont a pontok 80%-a szemre egy egyenesen helyezkedett el. Az illesztett függvényem:

$$f(x) = ae^{b(x-c)} - d,$$

ahol a kapott paraméterek az illesztés után:

$$a = 5, b = -1.05 \pm 0.02, c = 4.5 \pm 0.1, d = -1.3$$

2.2 sand1m.cpp

A feladat szerint módosított sand1m.cpp-ben ugyanazzal a módszerrel mértem ki a lavinákat. A következő szabálynak kellett teljesülni h_i instabil oszlop

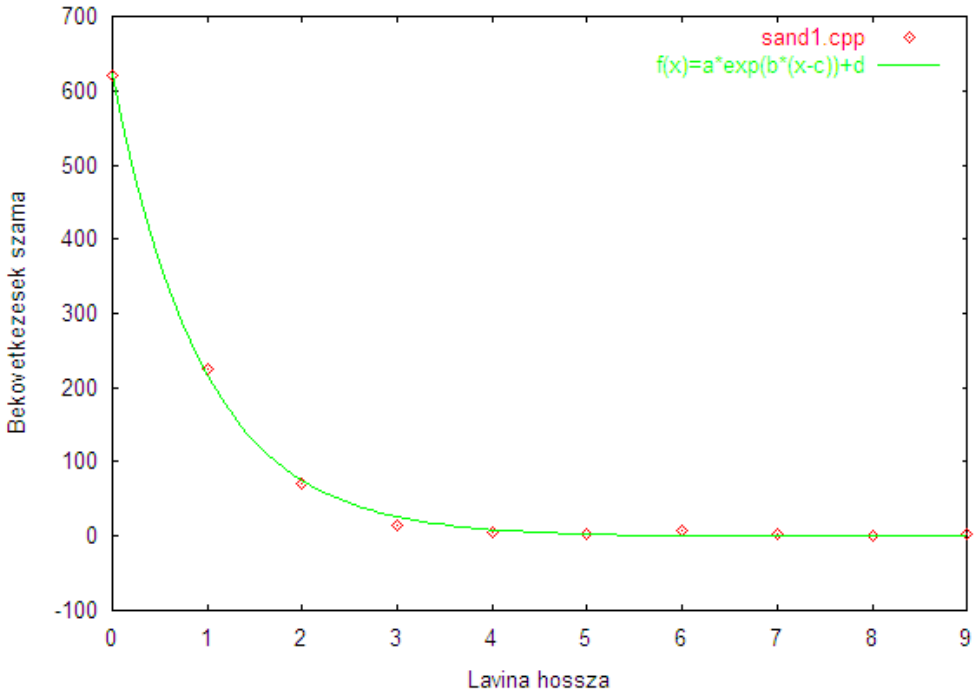

Figure 1: Lavinák skálázása (sand1.cpp)

esetén:

$$h_i \Rightarrow h_i - 2$$
$$h_{i+1} \Rightarrow h_{i+1} + 1$$
$$h_{i+2} \Rightarrow h_{i+2} + 1$$

A programban lévő szabályt a következőre kellett átírni:

```
void takeStep() {
   if (pileIsStable()) {
     ...
   } else {
     for (int i = 0; i < L; i++)
       if (!stable[i]) {
         ++A;
         --height[i];
         --height[i];
         if (i < L - 2){
           ++height[i + 1];
           ++height[i + 2];
```

```
        }
        else if(i < L - 1)
          ++height[i + 1];
      }
    }
}
```

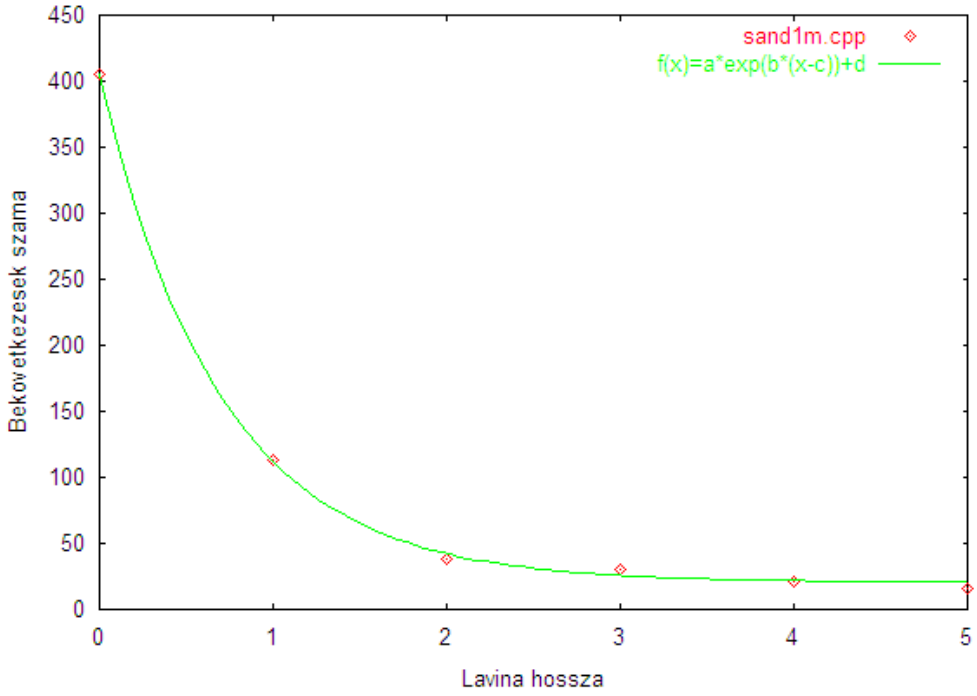

Figure 2: Lavinák skálázása (sand1m.cpp)

Az illesztett függvényem:

$$f(x) = ae^{b(x-c)} - d,$$

ahol a kapott paraméterek az illesztés után:

$$a = 5, b = -1.2 \pm 0.02, c = 4.64 \pm 0.1, d = 0$$

2.3 Összehasonlítás

Az első modell stabilabbnak tűnt olyan szempontból, hogy nem volt drasztikus lavina, ami sok oszlop stabil helyzetét megváltoztatta volna, viszont egy bizonyos mennyiségű homok (telítettség) után a random generátor határozta

meg a lavinák hosszúságát, ami nem volt a módosított modellben.

$$A = L - rand(),$$

ahol A a lavina hosszúság. Illesztésekben nem volt nagy különbség.