Számítógépes Szimulációs Labor

Jegyzőkönyv

1. Molekula dinamika

Sztreha Balázs
2007-09-30

1. Bevezető

A molekula dinamikát széles körben használják gázok, folyadékok, molekulák, galaxisok és csillagok elmosódásának szimulációjához. Jelen esetünkben, kiindulásunk egy L oldalú kockában, egymástól egyenlő távolságra elhelyezett Argon részecskék. Két részecske között a fellépő potenciált jól közelíthetjük a Lenard-Jones potenciállal:

$$V(r)=4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12}-\left(\frac{\sigma}{r}\right)^{6}\right] ,$$

ahol r a két részecske közepének a távolsága, $\varepsilon = 1.65\cdot 10^{-21}$ J és $\sigma = 3.4\cdot 10^{10}$ m az az r érték ahol az energia zérus. Az $1/r^{12}$ tag írja le az erős kölcsönhatást és az $1/r^{6}$ tag pedig a dipól-dipól kölcsönhatást (van der Waals). Ebből kiszámolható az erő:

$$F(r)=\frac{-dV(r)}{dr}=\frac{24\varepsilon}{\sigma}\left[2\left(\frac{\sigma}{r}\right)^{13}-\left(\frac{\sigma}{r}\right)^{7}\right]$$

2. Program felépítése

C++ nyelven írták, amit `g++` fordítóprogrammal lefordítunk futtatható állapotba.

`initialize()` függvény helyezi el a részecskéket, majd beállítja a kezdősebességeiket.

3. Feladatok

a, Átlagos Potenciál

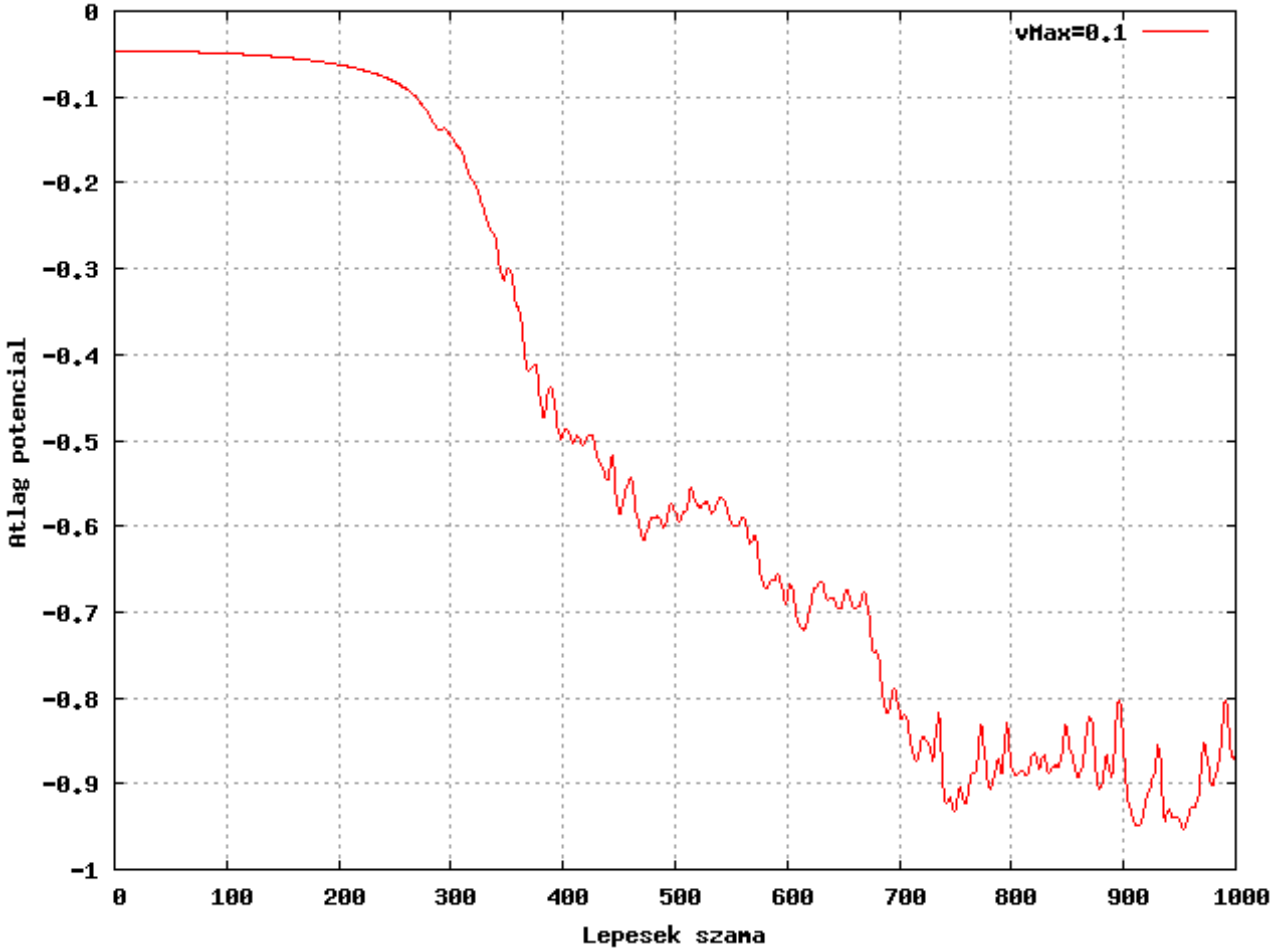
A következő kódrészlettel kell kiegészíteni, majd a *main*-ben kiiratni a fájl-ba az értékeit:

```
double atlagV() {
     double V=0;
      for (int i = 0; i < N-1; i++)        // loop over all
distinct pairs i,j
           for (int j = i+1; j < N; j++) {
           double rij[3];
           double rSqd=0;
                for (int k = 0; k < 3; k++) {
                    rij[k] = r[i][k] - r[j][k];
                    rSqd += rij[k] * rij[k];
                }
                V += 4 * ( pow(rSqd, -6) - pow(rSqd, -3));

               }
      return V/N;
}
```

A programot a következő paraméterekkel lefuttatva:

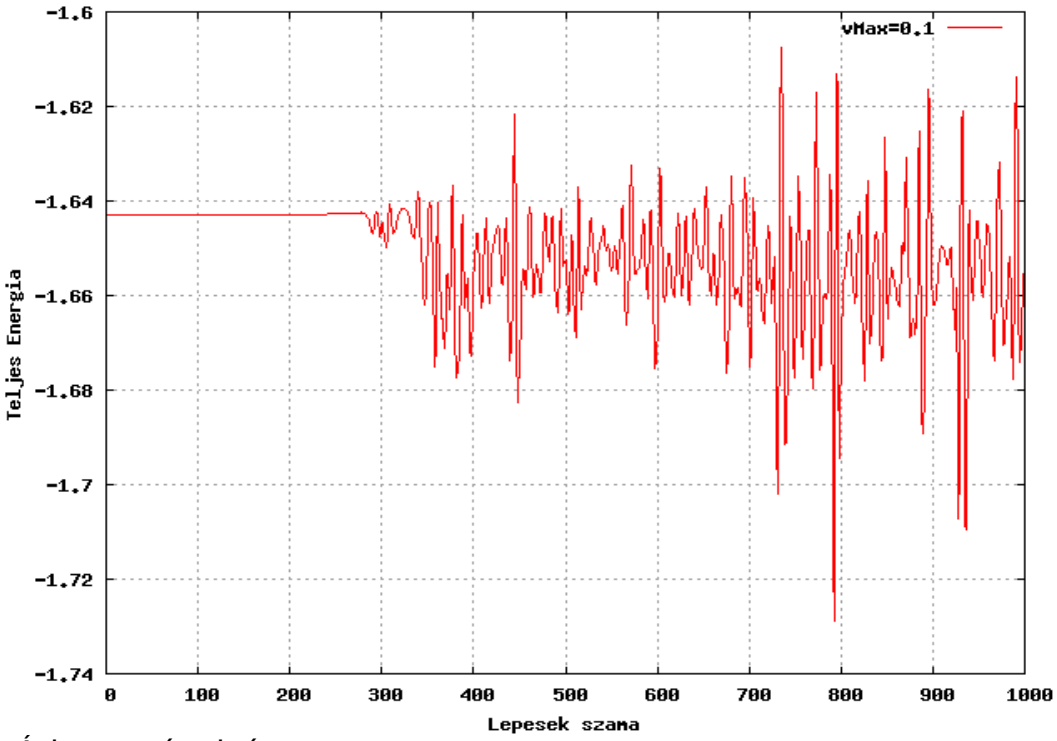
dt=0.01
vMax=0.1
1000 lépés



b, Teljes energia

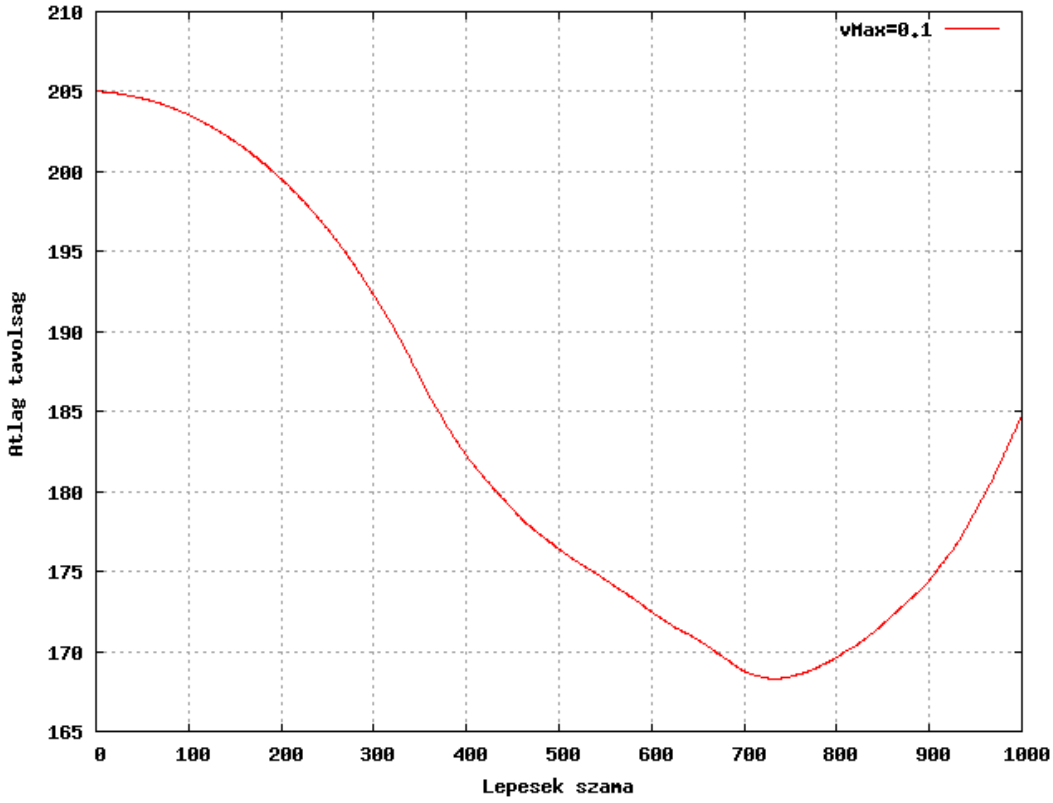
```
double TeljesE(){
    double E=0;
    for (int i = 0; i < N; i++)
       E += 0.5 * (v[i][0] * v[i][0] + v[i][1] * v[i][1] +
v[i][2] * v[i][2]);
    return E+N*atlagV();
}
```

Itt egyszerűbbnek láttam felhasználni az előbb kiszámolt átlag potenciált. Az ábrán látszik, hogy az elején még konstans az össz. energia, majd kb. a 300 lépésnél elkezdt fluktuálni, ami nem számábrázolási hiba, hanem az algoritmus pontatlansága.



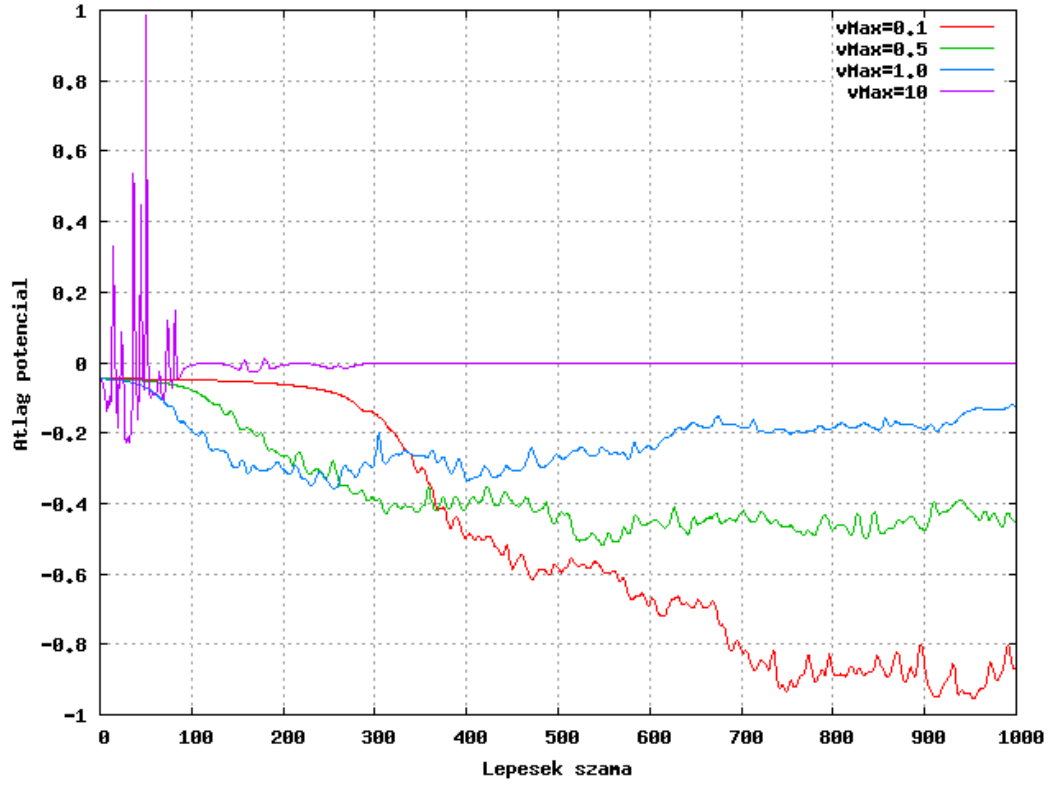
c, Átlagos távolság

```
double atlagR(){
    double R=0;
    for (int i = 0; i < N-1; i++)
        for (int j = i+1; j < N; j++) {
            R += pow((r[i][0] - r[j][0])*(r[i][0] -
r[j][0])+(r[i][1] - r[j][1])*(r[i][1] - r[j][1])+(r[i][2] -
r[j][2])*(r[i][2] - r[j][2]),0.5);
        }
    return R/N;
}
```



Az ábráról látszik, hogy a részecskék az elején még egymás felé mozognak, mivel a sebességnek véletlen értéket adtunk, tehát vannak akik egymás fele indulnak el. Ezt a hatást erősíti az is, hogy a potenciál hosszútávon vonzó. Az idő előrehaladtával azonban vannak, akik kiszabadulnak és egyenes vonalú egyenletes mozgást folytatnak, így végtelenbe nő a távolság.

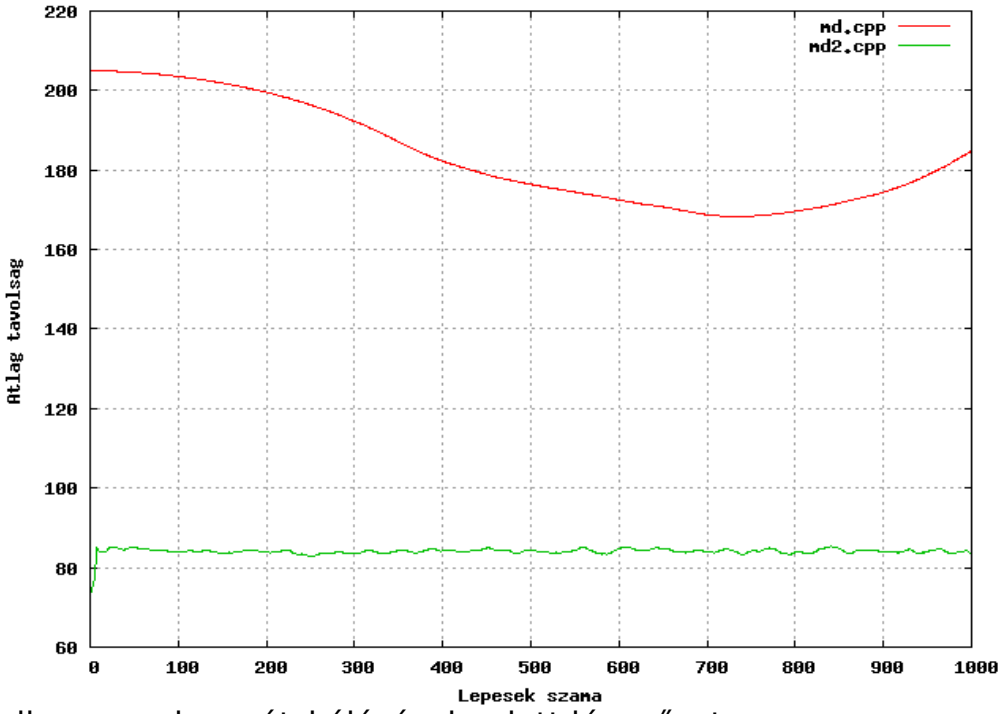
4. Kezdőfeltételek változtatása



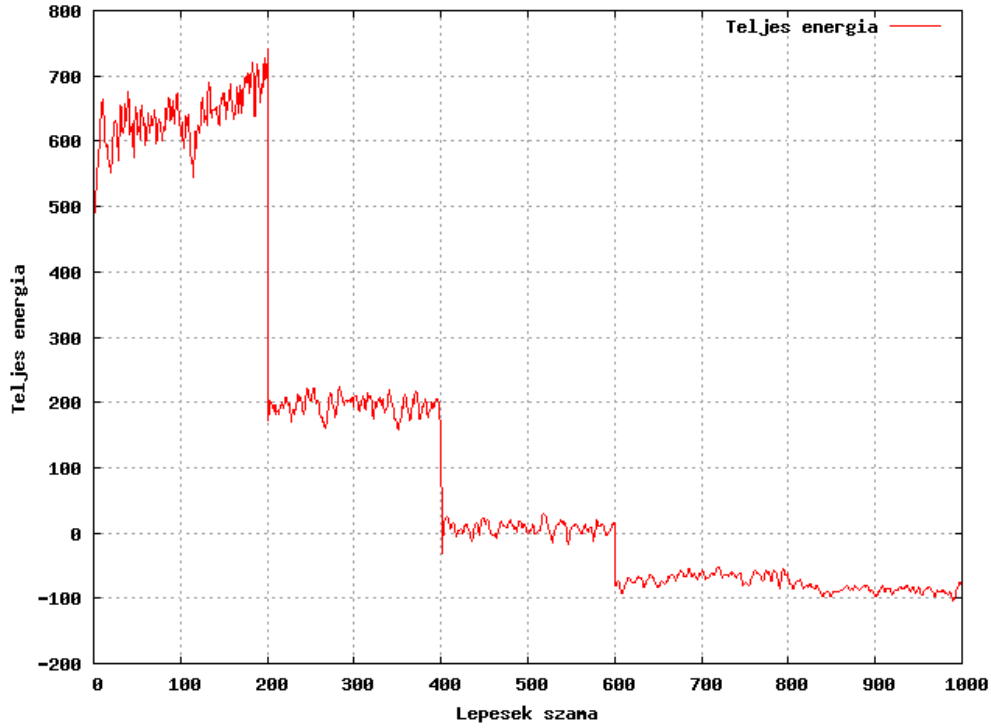
Ezen az ábrán az látszik, hogy ha a sebességet (ezáltal a hőmérsékletet) növelem, akkor már 100 lépés után az egy részecskére jutó átlagpotenciál zérus lesz, tehát elszakadtak egymástól.

5. md2.cpp

Az *md2.cpp* több ponton is közelebb áll a valósághoz. Egyrészt az atomok lapcentrált köbös rácsban helyezkednek el, ami a pontenciálhoz jobban illeszkedik, másrészt a kezdeti Maxwell-Boltzmann sebességeloszlás is jobban közelíti az egyensúlyi állapotot.
Összehasonlítva az *md.cpp*-vel az átlagos távolsággal:



A teljes energia az átskálázások miatt lépcsőzetes:



Ha az újra skálázásokat sűrítjük minden 10. lépésre, akkor máris gyorsabban konvergál konstans értékre az energia: