Párhuzamos eszközök beadandó bemutatása

Harmadfokú egyenlet megoldóképlet

A program a hozzá tartozó *.bat* fájl segítségével könnyen futtaható, a futtatáshoz MinGW szükséges.

A képen szöveg, fehér tábla látható

Automatikusan generált leírásA harmadfokú egyenlethez tartozó képlet amit felhasználtam a következő:

A rövidített verzióját könnyebb volt alkalmazni:

A képen diagram, szöveg, levél látható

Automatikusan generált leírás

Ennek az alkalmazása a programban a következő:

|  |
| --- |
| typedef struct cubic\_data{  float a;  float b;  float c;  float d;  float x1;  float x2;  float x3;  }Cubic\_data;  \_\_kernel void cubic\_equation(\_\_global Cubic\_data\* buffer) {  int idx = get\_global\_id(0);  float M\_PI = 3.14159265359;  float p, q, r;  p = buffer[idx].b / buffer[idx].a;  q = buffer[idx].c / buffer[idx].a;  r = buffer[idx].d / buffer[idx].a;  float Q = (pow(p, 2) - 3 \* q) / 9;  float R = (2 \* pow(p, 3) - 9 \* p \* q + 27 \* r) / 54;  float D = pow(Q, 3) - pow(R, 2);  if (D > 0) {  float theta = acos(R / sqrt(pow(Q, 3)));  buffer[idx].x1 = -2 \* sqrt(Q) \* cos(theta / 3) - p / 3;  buffer[idx].x2 = -2 \* sqrt(Q) \* cos((theta + 2 \* M\_PI) / 3) - p / 3;  buffer[idx].x3 = -2 \* sqrt(Q) \* cos((theta - 2 \* M\_PI) / 3) - p / 3;  } else if (D == 0) {  buffer[idx].x1 = -2 \* cbrt(R) - p / 3;  buffer[idx].x2 = cbrt(R) - p / 3;  } else {  float alpha = cbrt(sqrt(pow(R, 2) - pow(Q, 3)) + fabs(R));  float beta = Q / alpha;  if (R > 0) {  buffer[idx].x1 = -p / 3 + alpha + beta;  buffer[idx].x2 = -p / 3 - 0.5 \* (alpha + beta);  buffer[idx].x3 = buffer[idx].x2;  } else {  buffer[idx].x1 = -p / 3 + alpha + beta;  buffer[idx].x2 = -p / 3 - alpha - beta;  buffer[idx].x3 = buffer[idx].x2;  }  }  } |

Tanár úr kérésére átváltottam a több különböző adatot egy struktúrává, hogy egy nagy bufferként tudja átadni a program az adatot.  
  
A Laptopom problémája volt az, hogy nem ismerte a double típust. Ezt javítottam a float típusra, így már működött is, egy dolog kivételével. Az M\_PI – t nem ismerte a laptopom szintén, úgyhogy manuálisan definiáltam.

A refaktorálásokon kívül változatlan maradt a program, a kódrész előtt egy komment magyarázza, ami éppen ott történik. Élőben megbeszéltük tanár úrral.

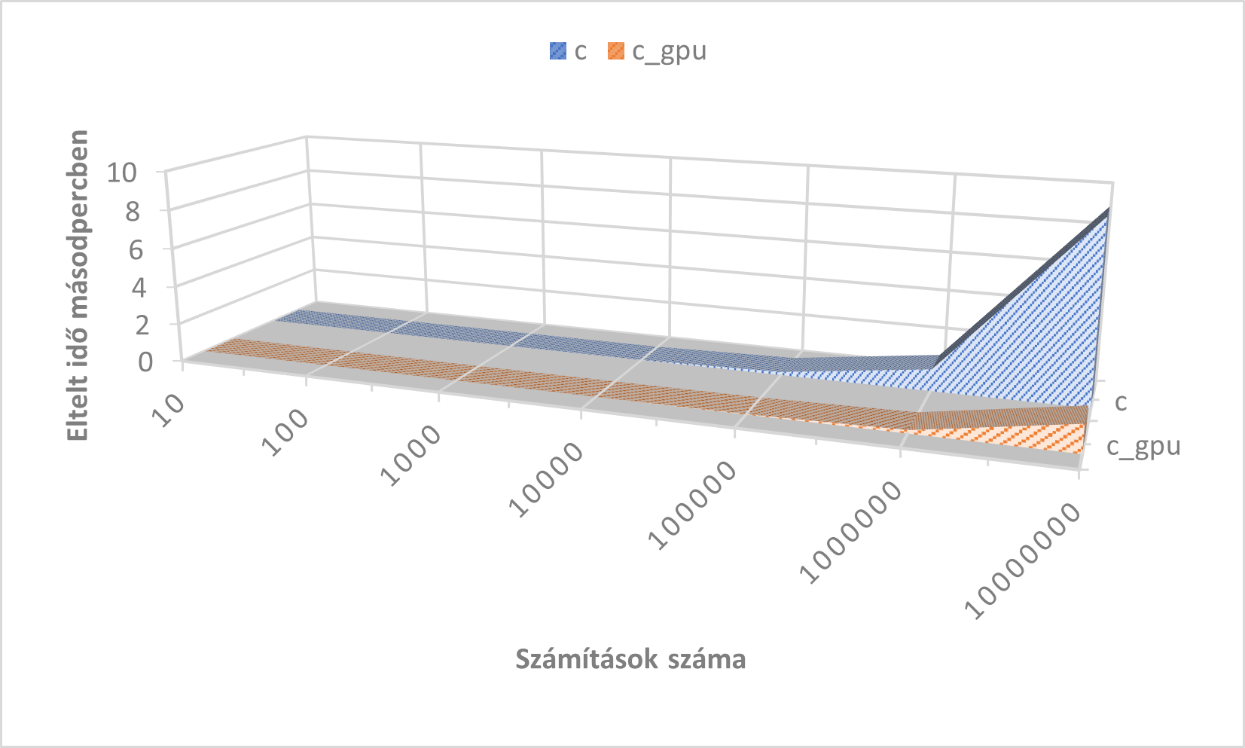
A sima és a GPU-s program közötti különbség látványosan nagy. A bal oldalon látható *harmadfoku.c* program végeredménye látszik a jobb oldalon pedig a *harmadfoku\_gpu.c*  végeredménye látszik. Az első adat az a számolások mennyisége a második pedig a futási ideje az említett számolásnak. Tehát itt a példában a bal oldali ábrán 1 millió számítás 1,11 másodperc alatt futott le.

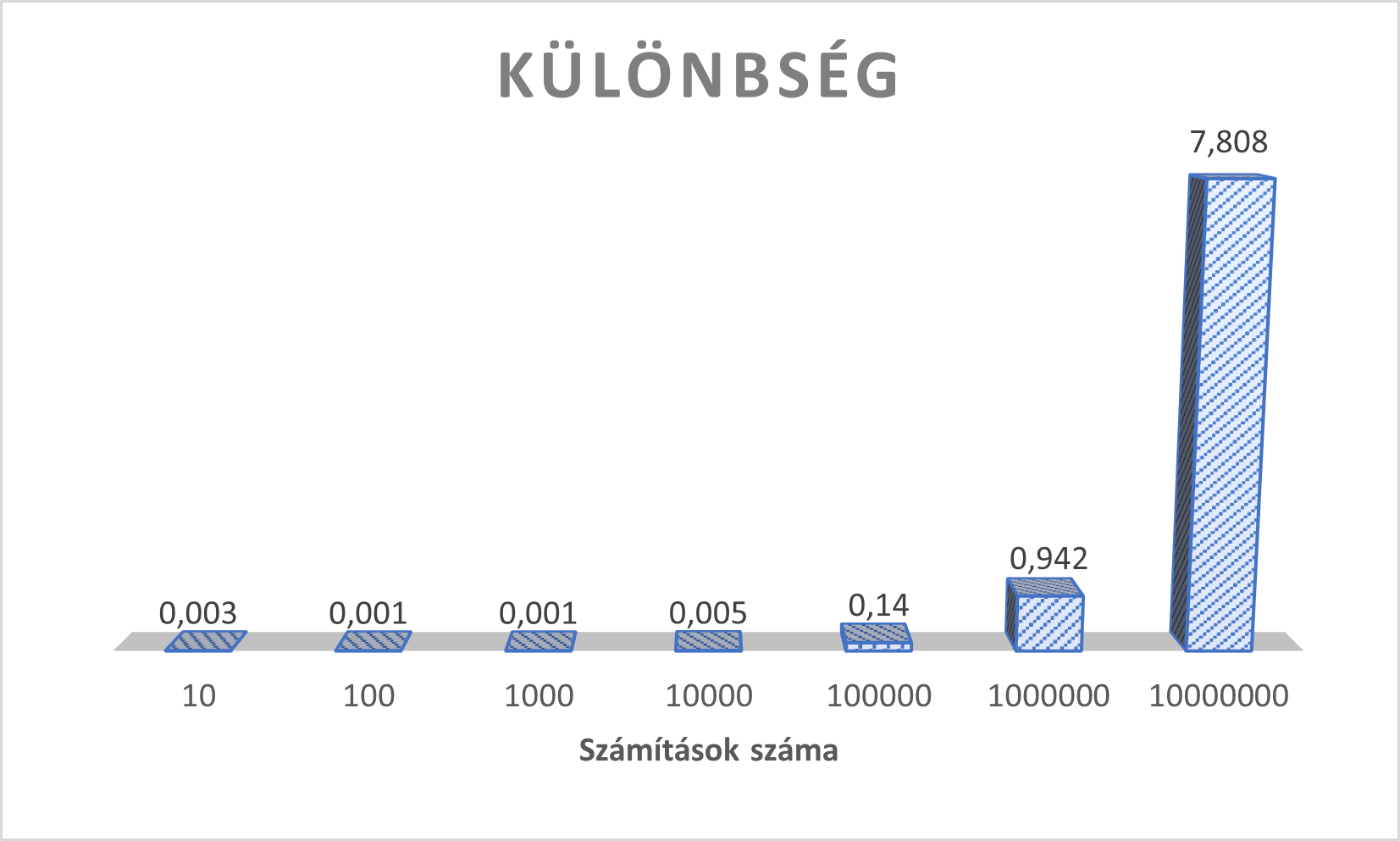
A képen szöveg látható

Automatikusan generált leírásA képen szöveg látható

Automatikusan generált leírás

A jobb oldalon lévő ami már igénybe vette a GPU-t ott 1 millió számítás 0,16 másodperc alatt futott le.

Számok használatával nem annyira egyértelmű ez a különbség az emberi agynak, ezért létrehoztam egy gráfot a fenti adatok segítségével.

A két futtatás közötti különbség: