Algoritmusok és adatszerkezetek hatékony implementálása C nyelven

Wittmann Mátyás (OXHER7)

Kis Házi Feladat

Bevezetés

A villamosmérnöki szakmában, azon belül az áramkörtervezésben használunk hőszimulációs programokat, hogy láthassuk a különböző alkatrészek termikus tulajdonságait. Ez a program a CPU és GPU teljesítményét hasonlítja össze, kihasználva a CUDA párhuzamos feldolgozás előnyeit. Az alapötlet az, hogy egy adott rácsos felületen megfigyeljük a hővezetést, különböző hősugárzó objektumok elhelyezése mellett, és mérjük, hogyan alakul a számítási idő az egyes megvalósításokban.

A program rövid áttekintése

Ez a hővezetési szimulációs program C és CUDA (Nvidia C) alapú algoritmusokat használ a hőmérséklet-eloszlás szimulációjához. A program célja, hogy összehasonlítsa a CPU (egyszálas C) és a GPU (CUDA) számítási sebességét, valamint hatékonyságát egy iteratív folyamat során. A hősugárzó objektumokat egy poligonfájl definiálja, amely a felület különböző pontjain létrejövő hőmérsékleti teljesítményeket írja le.

A program feladata

A program célja a hőmérséklet-eloszlás pontos meghatározása és az időigényes számítások optimalizálása a GPU segítségével. A program feladata, hogy:

1. Beolvassa a hősugárzó objektumok tulajdonságait egy poligonfájlból.
2. Számítsa ki a hőmérsékleti eloszlást (diszkrét hővezetési modellt felhasználva) meghatározott rácspontokon, a CPU és a GPU használatával is.
3. Mérje és rögzítse a futási időket, majd az adatokat egy CSV fájlba mentse.
4. A hőmérséklet-eloszlást vizuálisan ábrázolja egy hőtérképen.

A program működésének főbb elemei

1. **Adatbeolvasás**

   A program a felhasználó által megadott poligonfájlt használja, amelyben a különböző hősugárzó objektumok helyzete és sugárzási intenzitása található. Ez a fájl határozza meg, hogy a szimulációban hol és milyen erősségű hőhatások vannak jelen a szimulált területen.

2. **CPU és GPU alapú hővezetési számítás**

   - **CPU (C program)**: Az egyszálas C-alapú CPU szimuláció szekvenciálisan számolja ki a hőmérsékleti értékeket minden rácsponton. Ez egy hosszabb ideig tartó számítás, mivel minden pont számítása sorban történik.
   - **GPU (Nvidia C program)**: A GPU-alapú szimuláció párhuzamosan számítja ki a hőmérséklet-eloszlást, a CUDA párhuzamos feldolgozását kihasználva. Az iterációk során az algoritmus gyorsabban képes feldolgozni az adatokat, különösen nagyobb rácsok esetében.

3. **Eredmények kiértékelése és összehasonlítása**

   A program a futási időt méri mind a CPU, mind a GPU verzió esetében, és összehasonlítja azokat. Az eredményeket egy results.csv fájlban rögzíti, amely tartalmazza a hőmérsékleti adatokat.

4. **Eredmények vizualizálása**

   Egy Python szkript segítségével a hőmérsékleti adatokat hőtérképen ábrázolja, így vizuálisan is láthatóvá válik, hogy kimenetben nincs különbség a CPU és GPU eredmények között. (Két hőtérképet jelenítünk meg a CPU és GPU esetében is, egyiken interpolált a hőtérkép, míg a másik a nyers adatokat jeleníti meg).

Felhasználási területek és eredmények

A hővezetési szimulációs program alkalmazási területei közé tartoznak a hőtechnikai elemzések és kutatások, például a következő területeken:

- **Épületgépészet**: Hővezetési viszonyok elemzése különböző anyagok és szerkezetek között.
- **Elektromos áramkörök hőelemzése**: A komponensek hőmérsékleti eloszlásának vizsgálata, különösen GPU-kban és CPU-kban.
- **Gyártási folyamatok**: Hőkezelési folyamatok, ahol az egyenletes hőeloszlás biztosítása fontos.

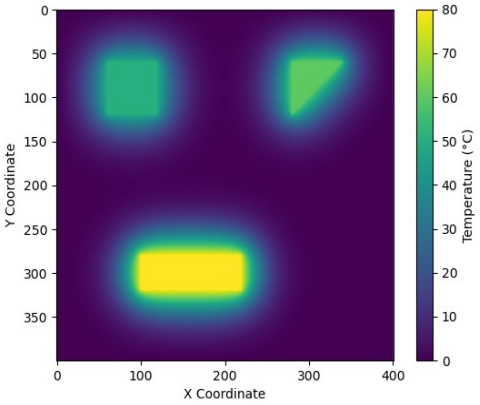
A mért eredmények segítenek az optimális hardverválasztásban és a számítási teljesítmény növelésében is, hiszen a szimuláció alapján jól látható, hogy a GPU-alapú feldolgozás sokkal gyorsabb lehet nagyobb adathalmazok esetén.

Összegzés és kiértékelés

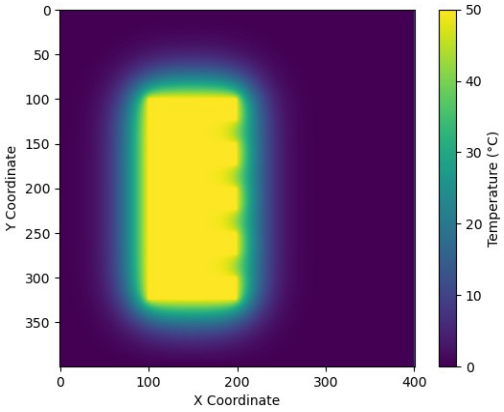
A Házi feladat keretein belül 3 tesztet végeztem és mind a 3 esetben sokkal kifizetődőbb volt a GPU használata. Minden teszt 5000 iterációt tett meg egy 400x400-as szimulációs felületen, csak a polygonok mennyiségét és elhelyezkedését változtattam. A GPU programban az a meglepő, hogy az adatok másolása, illetve a CUDA kernel indítása és meghívása megközelítőleg 1.4 másodpercet vesz igénybe, adatmérettől függetlenül, de ezektől eltekinve még kisebb adathalmazokra is sokkal gyorsabb a központi processzorral szemben.

A tesztek eredményei:

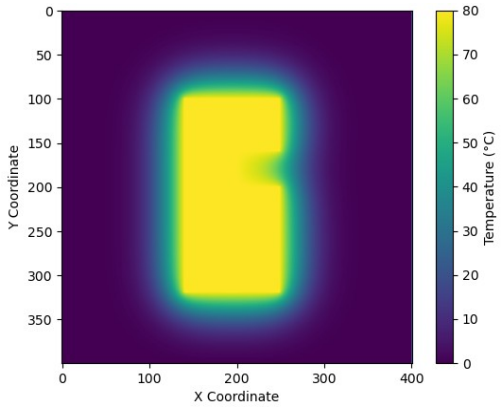
- **#1: polygons_1.txt**, ebben a polygon fileban egy **négyzet**, egy **háromszög** és egy **téglalap** van, a teszt egy **400x400**-as rácson futott **5000 iteráción**.
  - CPU futási idő: 1m48,416s
  - GPU futási idő: 2,038s



- **#2: polygons_2.txt**, ebben a polygon fileban egy **hűtőborda** modellje található, a teszt egy **400x400**-as rácson futott **5000 iteráción**.
  - CPU futási idő: 2m2,396s
  - GPU futási idő: 2,852s



- **#3: polygons_3.txt**, ebben a polygon fileban egy **SMD ellenállás** keresztmetszeti modellje látható, a teszt egy **400x400**-as rácson futott **5000** iteráción
  - CPU futási idő: 1m19,581s
  - GPU futási idő: 2,137s



A projekt elárhetőségei és egyéb források

- GitHub: matyi697 – thermo-simulation
- A diszkrét és nem diszkrét hőterjedésről források:
  - Wikipédia
  - YouTube