**Lygiagretus programavimas (GPU)**

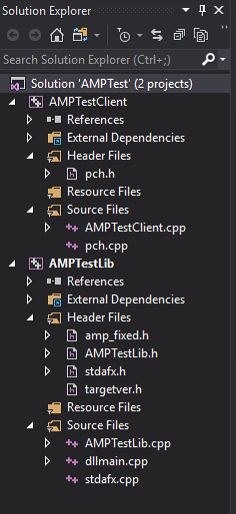
Antanas Valenčius ISI 1k. 2gr./2pogr.

**Užduotis**: parašyti programą, kuri atliktų lygiagrečius skaičiavimus naudojant GPU resursus (šiuo atveju, matricų daugybą).

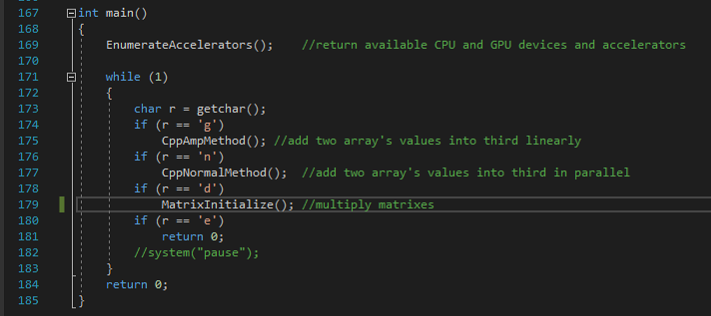
**Tikslas**: palyginti, kuo skiriasi lygiagretumas naudojant CPU nuo GPU.

Programa parašyta naudojant C++ AMP modelį (biblioteką). Labai į smulkmenas nelįsiu, bet pabandysiu paaiškinti, kas vyksta ir kaip pasileisti programą savo kompiutery.

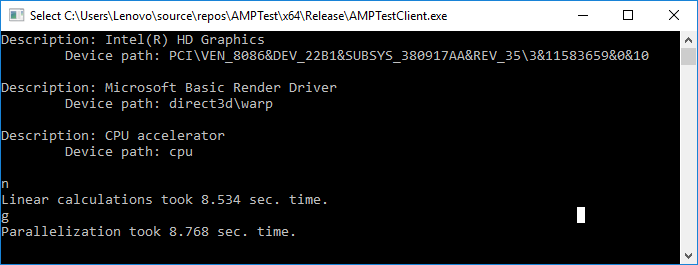
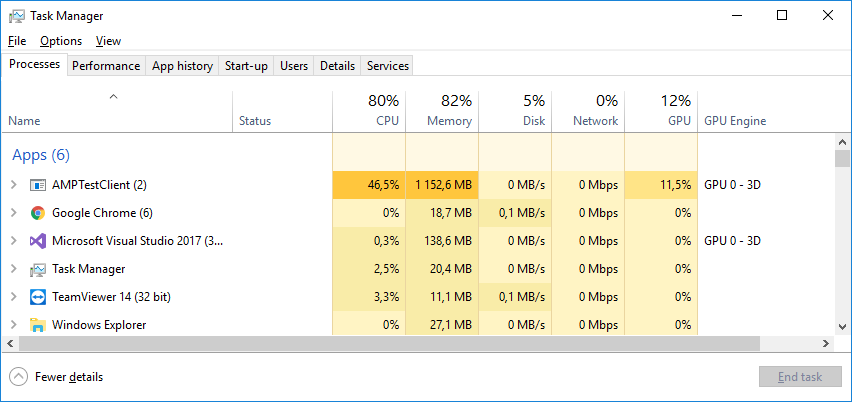
Norint atidaryt visą programos projektą per Visual Studio, reikia atidaryti AMPTEST/AMPTest.sln failą. Tenai yra 2 projektai: console application „AMPTest“ ir library „AMPTestLib“. Biblioteka yra tik tam, kad geriau išmokt naudotis failais programoje ir išmokt parašyt biblioteką, kuri sukompiliuojama .dll formatu.



„AMPTestClient.cpp“ Main funkcijoj vyksta loop‘as, kuriame galima pasirinkt funkciją, kurią norime iškviesti parašant tam tikrą raidę. Matricų daugybos funkcija iškviečiama parašant raidę „d“.

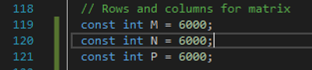
Taip pat, yra EnumerateAccelerators funkcija, kuri yra aprašyta „AMPTestLib.cpp“ faile. Ji išrašo devices, kuriuos gali naudoti AMP.

CppNormalMethod funkcija atlieka O(N) sudėtingumo algoritmą, kuris atlieka dviejų masyvų skaičių sudėtis ir įkelia į trečią masyvą. Tą patį atlieka ir CppAmpMethod funkcija, tačiau lygiagrečiai.

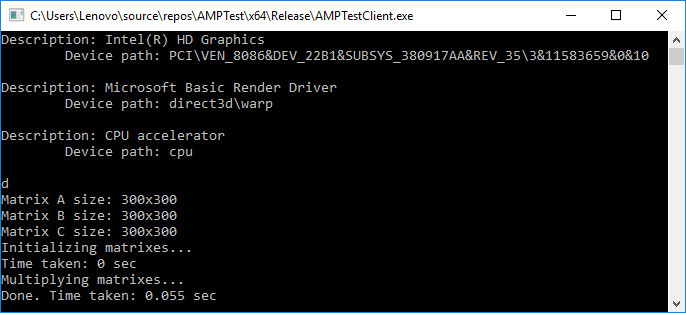


Lygiagrečių skaičiavimų veikimo metu, GPU naudojo vos 12% viso savo pajėgumo (masyvai inicializuojami naudojant CPU, tačiau sudedami naudojant GPU). Rezultatas toks pat, kaip tik naudojant CPU.

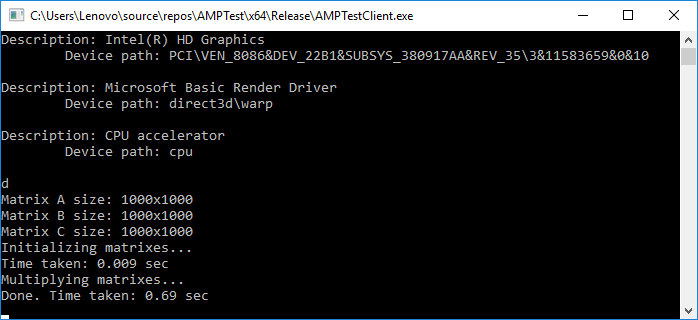
Matricų inicializavimas nuosekliai vyksta MatrixInitialize funkcijoj. Vėliau, funkcijoj MatrixMultiply jos yra sudauginamos lygiagrečiai. Matricų daugybos sudėtingumas yra O(N3).



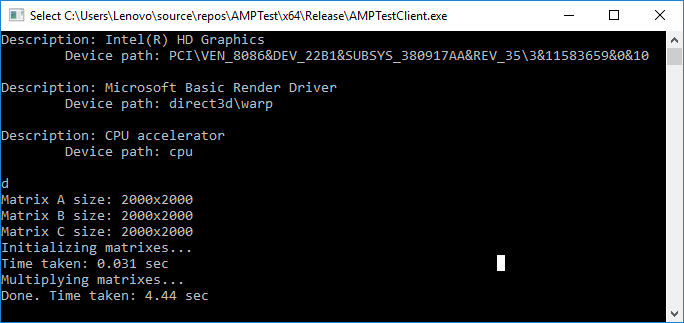
Matricų dydžius galima keisti šiose „AMPTestClient.cpp“ failo eilutėse.



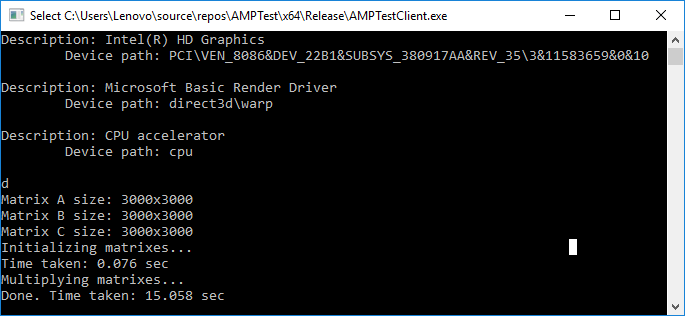
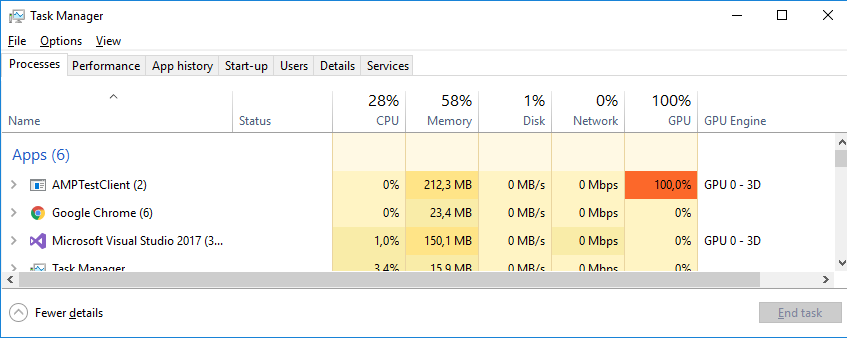
300x300 daugyba užtrunka dalį sekundės.



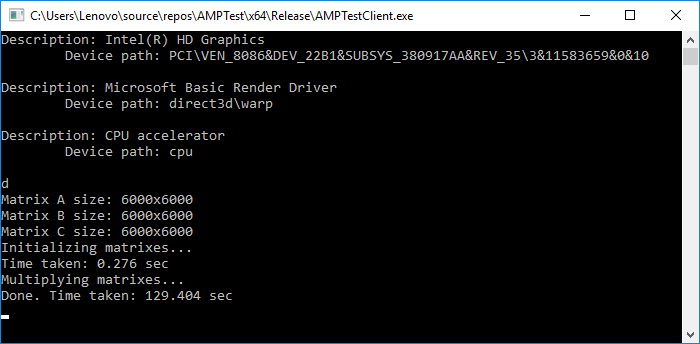
Padidinus matricas 3,332 = 11 kartų, programos laikas padidėjo irgi tik 11 kartų, tačiau, teoriškai turėjo per 3,333 = 37 kartus. Todėl greičiausiai, trumpiems skaičiavimams, nėra naudingas lygiagretumas naudojant GPU.



Padidinus matricas 22 = 4 kartus, šį kartą programos trukmė padidėjo 23 = 8 kartus nuo praeito kitimo. Iš karto matosi GPU suteikiama sparta tokiems skaičiavimams: openMP bei MPI metodais, reikėtų naudoti apie 13 procesoriaus branduolių, norint pasiekti tokį patį rezultatą, kaip 1 GPU.



Dar padidinus matricą 1,52 = 2,25 kartų, programos trukmė atitinkamai pailgėjo 1,53 = 3,375 kartų. Naudojama 100% GPU pajėgumo matricoms sudauginti.



Sudauginti daug didesnes matricas naudojant GPU uztruko neperdaugiausiai laiko.

**Kaip pasileisti?**

Iš pradžių, pabandykite tiesiog pasileisti „AMPTest\x64\Release\AMPTestClient.exe“ failą (32bit‘ų kompiuteriuose neveiks).

Atsidarę „AMPTest\AMPTest.sln“ failą per Visual Studio, galėsite redaguoti visą programą. Svarbiausios funkcijos yra aprašytos „AMPTest\AMPTestClient\AMPTestClient.cpp“ failę. Kompiliuojama 64bit‘ų architektūroje, nes ji yra greitesnė už 32bit‘ų. Taip pat, turi būti nustatyta Release režimu, nes Debug režimu nėra sukuriami tam tikri failai (nežinau kodėl). Programa sukūriama yra „AMPTest\x64\Release\AMPTestClient.exe“. Paleidus, ir parašius „d“, yra atliekama matricų daugyba. Tikiuosi veiks bent mano sukompiliuotas failas, todėl siūlau iš pradžių jį ir pasileisti.

**Išvados:**

GPU skaičiavimai, kurie yra sudėtingesni nei O(N), šiuo atvėjų matosi daug greitesni, nei CPU atliekami skaičiavimai. Todėl GPU yra naudojamas pastoviems sudėtingiems 2D, 3D skaičiavimams (žaidimai, VR). Kompiuterio ekrano veikimas yra tiesiog nuolatinė matricų daugyba ir jų atnaujinimas.