

CUDA a paralelní výpočty

Paralelní výpočty jsou typem výpočtů, při kterých se mnoho instrukcí provádí současně. Vycházejí z principu, že velké problémy lze často rozdělit na menší, které se pak řeší souběžně. **CUDA** (Compute Unified Device Architecture) je platforma od společnosti NVIDIA, která tento princip aplikuje na grafické procesory.

1. Sekvenční vs. Paralelní zpracování

Abychom pochopili přínos CUDA, musíme srovnat přístup CPU a GPU:

- **CPU (Sekvenční):** Jako mistr kuchař. Dokáže připravit i velmi složitý recept, ale dělá věci postupně (nebo v několika málo vláknech). Je ideální pro logiku, větvení programu a operační systémy.
- **GPU (Paralelní):** Jako tisíc pomocných kuchařů. Nejsou tak chytří, ale pokud jim zadáte jednoduchý úkol (např. „nakrájejte cibuli“), udělají ho všichni naráz a extrémně rychle.

2. Architektura CUDA výpočtu

Program v CUDA se skládá ze dvou částí: kódu pro **Host** (CPU) a kódu pro **Device** (GPU).

A. Kernel

Kernel je speciální funkce napsaná v C/C++, která se spustí na GPU. Kouzlo spočívá v tom, že se tato funkce nevykoná jednou, ale **N-krát paralelně** v N různých CUDA vláknech.

B. Hierarchie vláken (Threads, Blocks, Grids)

CUDA organizuje výpočetní zdroje do logických celků:

- **Thread (Vlákno):** Základní jednotka výpočtu.
- **Block (Blok):** Skupina vláken, která mohou sdílet data a synchronizovat se.
- **Grid (Mřížka):** Soubor bloků, které tvoří celý výpočetní úkol (např. zpracování celého obrázku).

3. Paměťový model v CUDA

Efektivita paralelních výpočtů závisí na tom, jak rychle se data dostanou k jádrům. CUDA využívá několik vrstev paměti:

- **Global Memory (VRAM):** Velká, ale relativně pomalá. Přístupná všem vláknům.
- **Shared Memory:** Velmi rychlá paměť uvnitř jednoho bloku. Klíč k vysokému výkonu (funguje jako programovatelná cache).

- **Registers:** Nejrychlejší, soukromá pro každé vlákno.

4. Proč je to důležité? (Škálovatelnost)

Klíčovou vlastností CUDA je **škálovatelnost**. Pokud napíšete program pro GPU se 128 jádry, stejný kód poběží automaticky rychleji na nové kartě s 5000 jádry. Hardware si sám rozdělí bloky vláken mezi dostupné výpočetní jednotky (Streaming Multiprocessors).

5. Příklad: Sčítání polí (Vector Addition)

V běžném C (CPU) byste použili cyklus `for`. V CUDA (GPU) napíšete pouze operaci pro jeden prvek a systém ji spustí pro všechny indexy naráz:

```
// CUDA Kernel pro sčítání dvou polí
__global__ void addVectors(float *A, float *B, float *C, int n) {
    int i = threadIdx.x + blockIdx.x * blockDim.x;
    if (i < n) {
        C[i] = A[i] + B[i];
    }
}
```

Související články:

- [Grafické procesory \(GPU\)](#)
- [Deep Learning a neuronové sítě](#)
- [Programování v C++](#)

Tagy: *cuda parallel_computing gpu programming nvidia*

From:
<https://www.serviceit.cz/> - IT ENCYKLOPEDIE

Permanent link:
<https://www.serviceit.cz/doku.php?id=it:lang:cuda>

Last update: **2026/01/02 13:20**

