Ideas for Low Celluclast Performance

13.11.2019

Basic Architecture

- Accesses different memory than CPU
- Starting thousands of threads at low costs

Memory Management

```
typedef int dtype;
class Storage {
    public:
     explicit Storage(const std::vector<int>&);

    private:
     std::vector<int> _data;
     dtype* _cpu_pointer;
     dtype* _gpu_pointer;
     void initialize_gpu_memory();
};
```

- Memory pool, takes ownership
- Initializes the gpu memory as copy
- two differetn pointer, to cpu/gpu locations

```
typedef int dtype;
class Storage {
    public:
     explicit Storage(const std::vector<int>&);
     const dtype* cpu_pointer_const();
     const dtype* gpu_pointer_const();
     dtype* cpu_pointer();
     dtype* gpu_pointer();

    private:
     std::vector<int> _data;
     dtype* _cpu_pointer;
     dtype* _gpu_pointer;
     void initialize_gpu_memory();
     std::string recent_head;
     void sync_to_cpu();
     void sync_to_gpu();
};
```

- accesses pointers

Merge

Write about CPU merge

How to spwan to many threads?

Paper that does that, show as example, the cutting approach
Naiv approach : 2 Threads Cut both a and b into half,

$$A = 0000 B \qquad = 1111 C = ????????$$

$$A = \underbrace{00}_{\text{Thread 1}} \mid \underbrace{00}_{\text{Thread 2}} B \quad = \underbrace{11}_{\text{Thread 1}} \mid \underbrace{11}_{\text{Thread 2}} C = \underbrace{????}_{\text{Thread 1}} \mid \underbrace{????}_{\text{Thread 2}}$$

$$C = \underbrace{0011}_{\text{Thread 1}} \mid \underbrace{0011}_{\text{Thread 2}}$$

How to allocated work?

Here are the mergepath pictures

Comutation Procedure

```
__global__ void paralleMerge3(int* a, int sz_a, int
                              int length) {
    int diag = threadIdx.x * length;
    int a_start = mergepath(a, sz_a, b, sz_b, diag)
    int b_start = diag - a_start;
    merge2(a, a_start, sz_a, b, b_start, sz_b, c, d
}
```

- Each tread works on one part
- Thread calculates the value A_lower for itself
- Calcultes the also the B_lower (Why does that work again?)
- merges the two arrays

Problem: Slow as a Snail

show the growth rates vs std::mergesort

Reason: So much global meory access

50 Percent of the global memory traffic is caused by 3 Percent of the values
Corrobation: Cuda performance tool

Memory Hirachy of CUDA

The different memories and their sizes

show the plot of the different memories and their relative size on my card

Merging with local memory

Describe the shared memory

show the plot of the different memories and their relative size on my card

Show the results

show the plot of the different memories and their relative size on my card