



deti

universidade de aveiro
departamento de eletrónica,
telecomunicações e informática

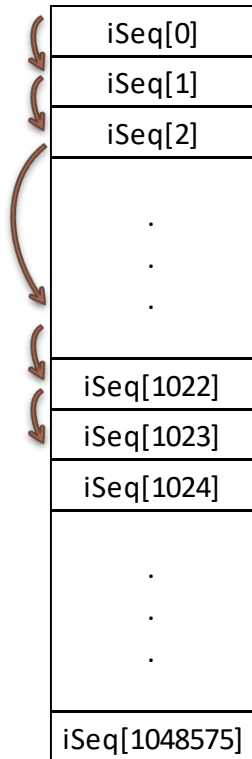
Assignment 3

- João Leite, nº 115041
- Luís Batista, nº 115279

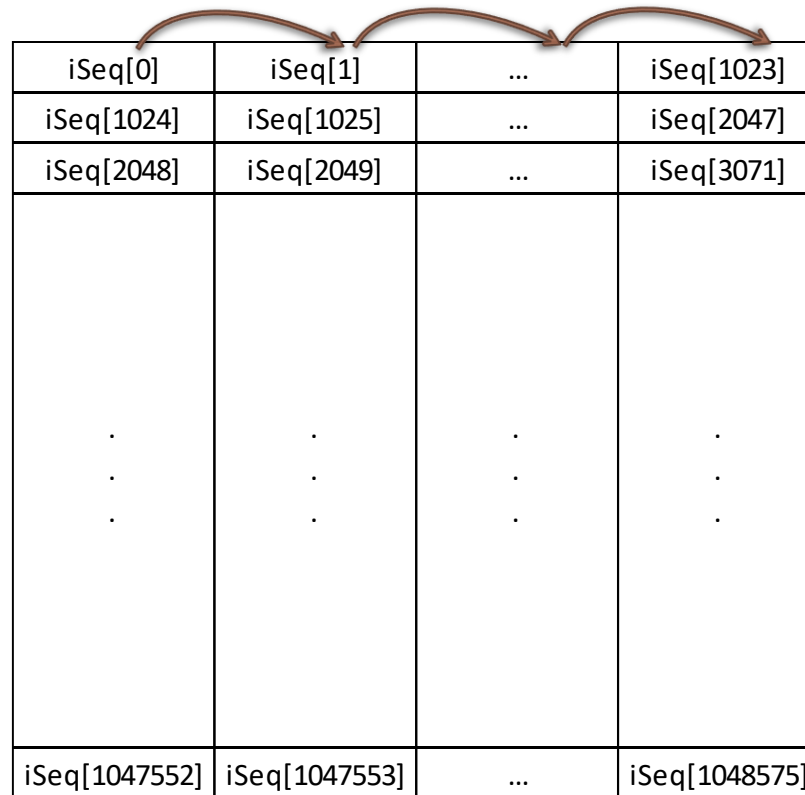
Program 1

Row processing

$\text{integerSequence}[N * (1 \ll \text{iter}) * \text{idx} + i]$, $0 \leq \text{idx} < (N \gg \text{iter})$
 $0 \leq i \leq (1 \ll \text{iter}) * N$
 $N = 1024$



iSeq[0]
iSeq[1]
iSeq[2]
⋮
iSeq[1022]
iSeq[1023]
iSeq[1024]
⋮
iSeq[1048575]



iSeq[0]	iSeq[1]	...	iSeq[1023]
iSeq[1024]	iSeq[1025]	...	iSeq[2047]
iSeq[2048]	iSeq[2049]	...	iSeq[3071]
⋮	⋮	⋮	⋮
iSeq[1047552]	iSeq[1047553]	...	iSeq[1048575]

Program 1

Results

Multithread solution (pthread)

PC1 Specs:

Lubuntu VM
4 cores

Binary file	2 workers
	Mean \pm Standard deviation
datSeq1M.bin	1.066 \pm 0.021 s

Multiprocess solution (MPI)

PC1 Specs:

Lubuntu VM
4 cores

Binary file	2 workers
	Mean \pm Standard deviation
datSeq1M.bin	178.7 \pm 23.2 ms

Single thread solution (CPU)

Remote PC
Specs:

GPU: Nvidia
GTX 1660 TI

Binary file	"1 worker"
	Mean \pm Standard deviation
datSeq1M.bin	763.4 \pm 19.9 ms

CUDA solution (GPU)

Remote PC
Specs:

GPU: Nvidia
GTX 1660 TI

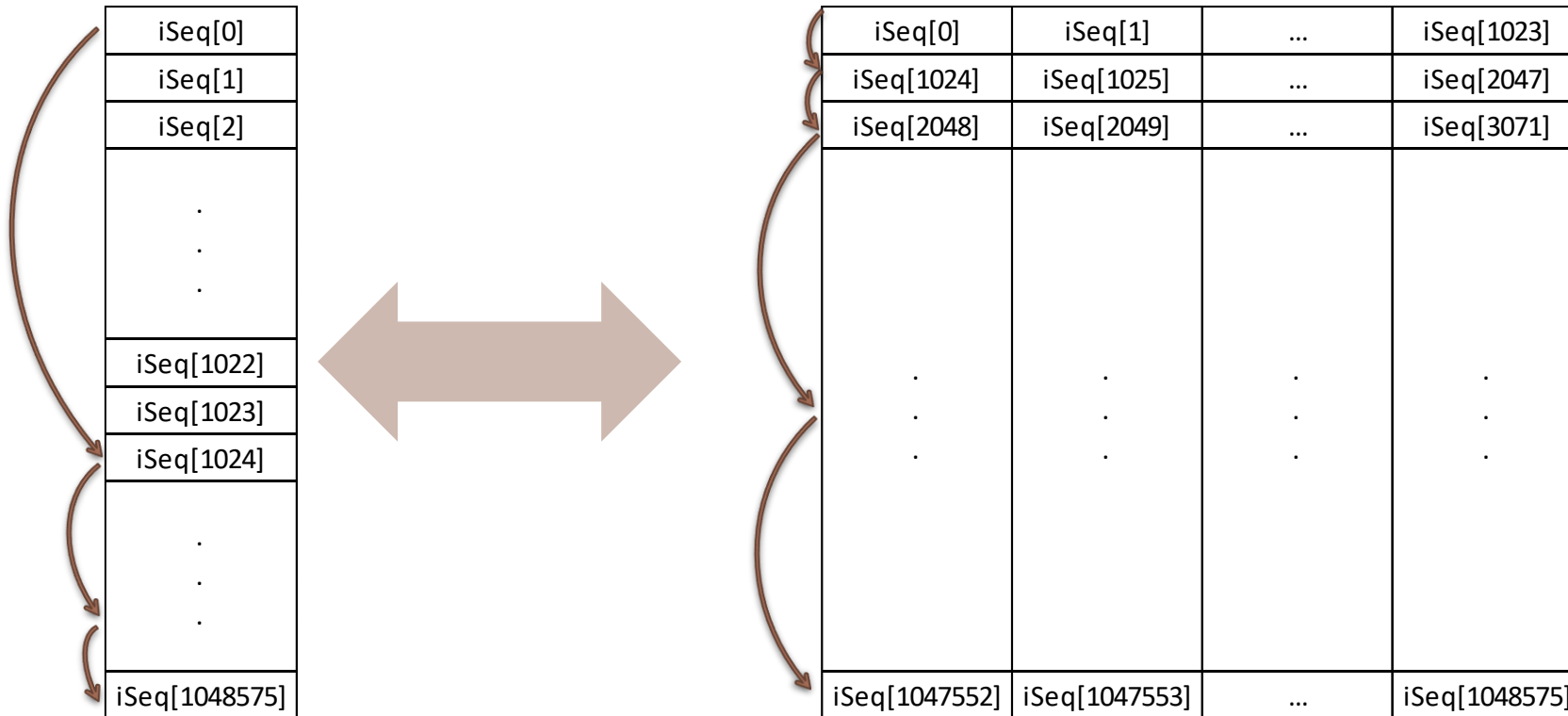
Binary file	Optimum config
	Mean \pm Standard deviation
datSeq1M.bin	2.502 \pm 0.001 s

(**NOTE**: both mean and standard deviation results were calculated by running the program 5 times)

Program 2

Column processing

`integerSequence` [$(1 \ll \text{iter}) * \text{idx} + N (i \bmod N) + (i \div N)$], $0 \leq \text{idx} < (N \gg \text{iter})$
 $0 \leq i \leq (1 \ll \text{iter}) * N$
 $N = 1024$



Program 2

Results

Multithread solution (pthread)

PC1 Specs:

Lubuntu VM
4 cores

Binary file	2 workers
	Mean \pm Standard deviation
datSeq1M.bin	1.066 \pm 0.021 s

Multiprocess solution (MPI)

PC1 Specs:

Lubuntu VM
4 cores

Binary file	2 workers
	Mean \pm Standard deviation
datSeq1M.bin	178.7 \pm 23.2 ms

Single thread solution (CPU)

Remote PC
Specs:

GPU: Nvidia
GTX 1660 TI

Binary file	"1 worker"
	Mean \pm Standard deviation
datSeq1M.bin	763.4 \pm 19.9 ms

CUDA solution (GPU)

Remote PC
Specs:

GPU: Nvidia
GTX 1660 TI

Binary file	Optimum config
	Mean \pm Standard deviation
datSeq1M.bin	4.2942 \pm 0.0005 ms

(**NOTE**: both mean and standard deviation results were calculated by running the program 5 times)

CUDA solution

Conclusions

- **Expected results**

- In CUDA solution, since there is normally a gain when using an higher number of workers at the same time, it would be expected to achieve better times using one of the CUDA programs.
- Comparing program 1 with program 2, it would be expected to get better performance in the first approach because there is a better cache use in a way there are less memory fetches. In program 2 this wouldn't happen so frequently because of the blocks size (multiple of 1024).

- **Outcome**

- Both CUDA approaches ended up being slower than any other type of solution. It could be related with the overhead of creating a large number of threads or the concurrency that exist between threads of different blocks.
- The row processing was indeed better (almost 2 times faster) than the column processing algorithm. This can be explained as there is a better mapping between the thread blocks and the memory zone which they will access, ending up accessing contiguous values.

- *Is it worthwhile to use the GPU to solve this kind of problem?*

∴ No. The large number of threads being created and parallelized could be adding unnecessary overhead to the program execution time, as well as not making the most optimized use of the caching mechanism.