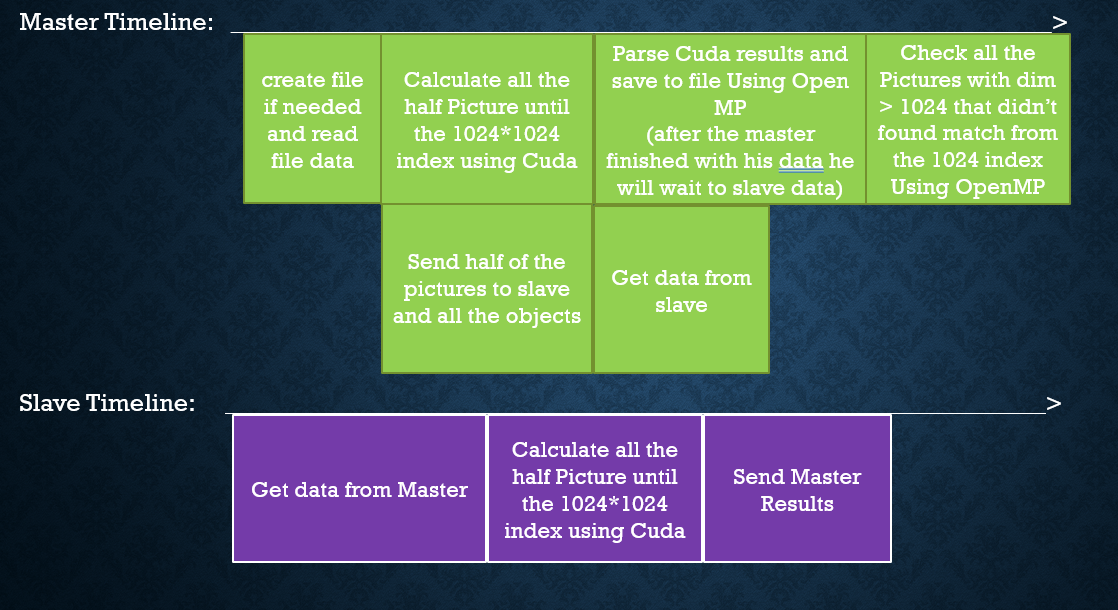
**ReadMe Contain:**

1. Architecture
2. About make file
3. Executing time on the difference programs option

**Architecture:**



**Master:**

1. creates file if needed (in case there isn't input file), and read the data
2. Two Tasks Parallel:
   1. Send to slave his Data (half of the Pictures and all the objects). **Each element type in the matrix is byte (not int)**, Its saves a lot of time during transferring the information.
   2. Check his own data using Cuda: The Cuda Searching matches is very fast and make a very good performance.

**Make Task Barrier:**

Why makes a barrier?

1. the two task later are:

Task3- wait for slave result.

Task4- parse all data and save to file.

During the tasks in step **2**, if we make task of "send slave data" and another task "get from slave data" on the same time, we lose a thread that will be stuck a lot of time in the task of getting data from slave (Wait for data before you pass the data).

The calculate using cuda task cannot be on the same time with the "parse result and save to file" task, because we don't have the results yet.

we don’t make it parallel with the last task "check all Picture from 1024\*1024 index", because it can be an unnecessary check in case it will be found in the 1024\*1024 first indexes,

and the best performance happen when all the threads can dill with "parse the results and save to file" all thogether.

1. New Two Tasks Parallel:
   1. parse result and save to file- parse all the master results and save it to file, after the master finished save is own data, he will wait in this task until he get all the slave results and then he will parse & save them too.
   2. Get results from slave
2. The last task: check all the Pictures with dim >1024 that didn't find a match yet, and save to file the results. Very quick because it's a small number of checks.

**In case of One PC run Parallel Option**, in situation with big data (big data defined in the program), the Slave and Master will make sure they didn't use Cuda together, because there is one GPU and when they use it together it will lower the efficiency, the Master will send the slave a MPI message when he finished to use Cuda, I choose that the Master use Cuda first because before he finished to save his own results to file, he will get already all the slave results.

**CUDA explanation:**

Get an empty Array (int\*) for results in size:

(number\_of\_pictures/2) \* // (each P have pictures /2 )

(number of objects) \*

(Picture max dim or 1024) //the smallest

**Picture …**

**Picture 0**

**Picture 1**

**Size of Picture max dim or 1024**

**Size of Picture max dim or 1024**

**Size of Picture max dim or 1024**

**Size of Picture max dim or 1024**

**One match result**

**OBJECT …**

**OBJECT 0**

**OBJECT …**

**OBJECT 1**

**OBJECT 0**

**OBJECT 1**

**There is a lot of Memory to allocate, the solution is to collapse every full result to a one cell (int):**

**Int: 32**

**In case of match:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**10 bits for the object ID**

**1 bit for telling its match (==1)**

**10 bits for the Y match**

**10 bits for the X match**

10 bits = 1023 -> it can be a problem in case there is more than 1024 object.

For index X and Y there isn’t a problem because the Max index we will get is

1024-object dim (we make a limit 1024 threads.

So, the biggest index can be 1023 (unless object dim = 0, but then you don't really have what to check).

If there isn't a match on cell we will put just a zero.

**Calculation complexity evaluation for each thread (the not parallel calculation in Cuda) :**

(Picture dim – object dim) \* Object dim \* Object dim

**Worst Case:**

(1024 – object dim) \* Object dim \* Object dim

**About make file:**

Make runSerial (for a serial program)

Make run (for a parallel program on one PC)

Make runOn2 (for a parallel program on two PC) – need to make "mf" file and fill PC Ips

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Parallel Two PC**  **Execute time** | **Parallel One PC**  **Execute time** | **Serial program**  **Execute time** | **Objects**  **Dim** | **Pictures**  **Dim** | **Number of**  **objects** | **Number of**  **Pictures** | **About input**  **file** |
| **0.459 sec** | **0.258 sec** | **1.958 sec** | **50** | **~1100** | **5** | **3** | **Boris's file** |
| **1.036 sec** | **1.387 sec** | **1 hour**  **and 6 minutes** | **50** | **1000** | **100** | **100** | **Program**  **Create**  **File** |

**Executing time in the different program options (not include read file time)**