IDEAL *threadpool* SIZE

mahmoudyoussef0097@gmail.com LinkedIN:mahmoud-youssef97

What is the ideal size of a Thread-Pool?

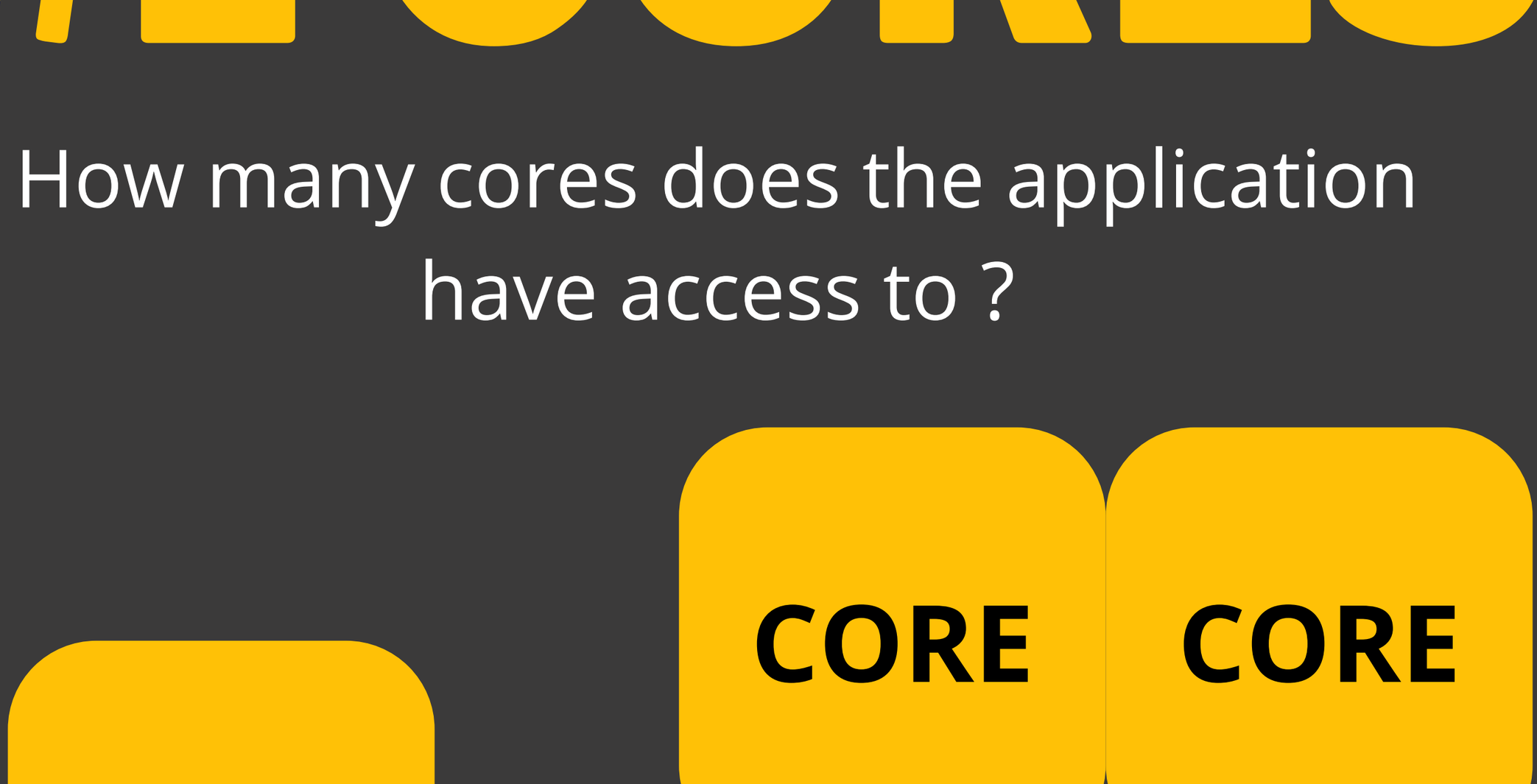
NO IDEAL SIZE !

This totally depends on the use case and trade-offs!

In order to understand how to choose the size of our thread-pool we need to understand the factors that affect our choice first!

#1 CORES

How many cores does the application have access to ?



Single Core

Quad Core

1 Core = 1 Thread

More Cores = More Threads = More Parallelisation

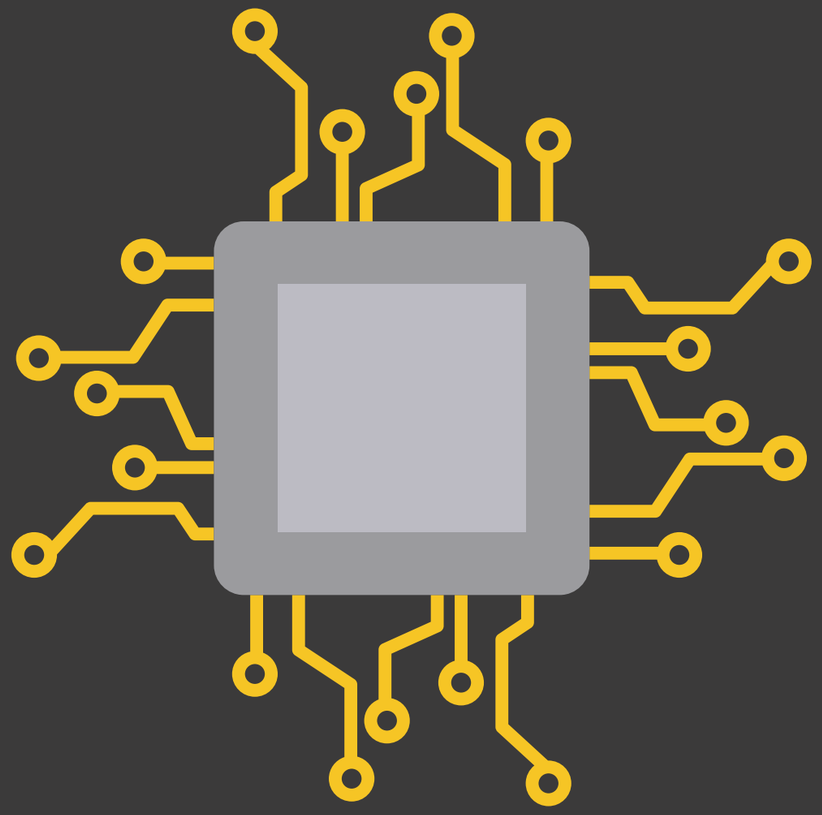
The second thing to understand is the type of the submitted task itself!

#2 TASKS

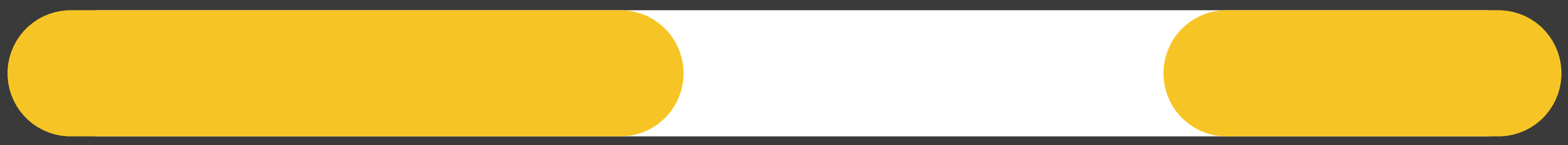
There is two types of the submitted tasks, either it's an I/O Bound Task or CPU Intensive Task

I/O Bound Task is mainly spending its time waiting for input and output operations to complete. (Anything which CPU can't perform). ex: reading/writing data to disk.

CPU Intensive Task is mainly spending its time waiting for computations to complete. ex: Sorting, searching and mathematical computations.



Let's assume we have
only ONE CPU CORE!



CPU is running a thread CPU is free

#OUR GOAL

Our goals is to maximise the CPU utilisation

2

Submitted Tasks
CPU Intensive Tasks

TH1

Threadpool

What would happen if we have #100 tasks and only one thread?

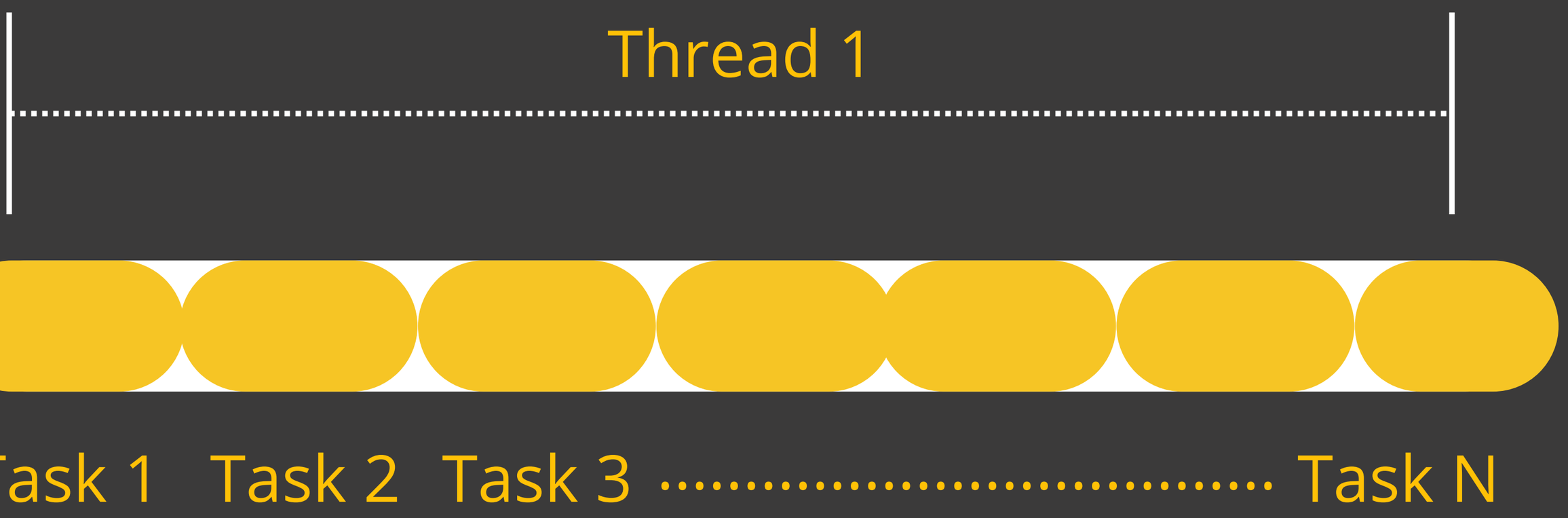
100

Submitted Tasks

CPU Intensive Tasks

TH1

Threadpool



Thread 1 is running on the CORE 1 and keep accepting and running the tasks. **We aimed to maximise the CPU utilisation.**

What would happen if we have #100 tasks and two threads?

100

Submitted Tasks

CPU Intensive Tasks

TH1, TH2

Threadpool

Task 1 Task 2 Task 3 ..... Task N

OS will schedule TH2 to take place over TH1, that could even happens before TH1 finishes the task. This is called :

#TIME SLICING

We aimed to maximise the CPU utilisation as well.

What would happen if we have #100 tasks and two threads? **But 2 CORES**

100

Submitted Tasks

CPU Intensive Tasks

TH1, TH2

Threadpool

CORE #1

Thread 1

Task 1 Task 2 Task 3 ..... Task N

CORE #2

Thread 2

Task 1 Task 2 Task 3 ..... Task N

So as we realised, no matter how many threads there are. **if we are doing a CPU Intensive Tasks, we are limited by the CORES.**

more **CORES** = more **PARALLELISATION!**

Many threads in this situation is so bad for both MEMORY AND PERFORMANCE!

CPU INTENSIVE TASKS

#CPU CORES

Let's assume we have only ONE CPU CORE! but we are dealing with I/O Bound Task.

#OUR GOAL

Our goals is to maximise the CPU utilisation

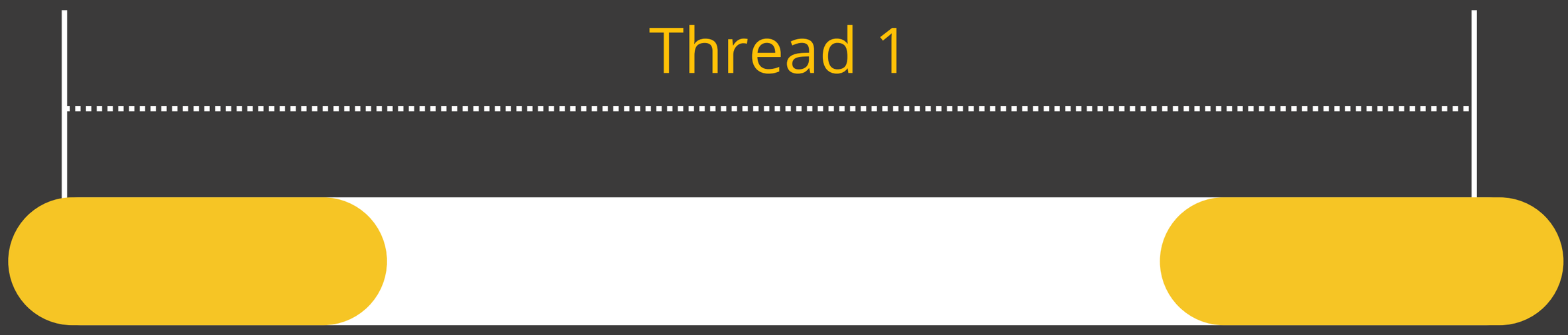
1

Submitted Tasks

TH1

I/O Bound Tasks

Threadpool



This is not EFFICIENT at all!

In order to achieve the maximum CPU utilisation as we aim.

1

Submitted Tasks

I/O Bound Tasks

TH1

Threadpool

Thread 1

Task 1

OS schedules other threads to run till the result is ready!

Task 1

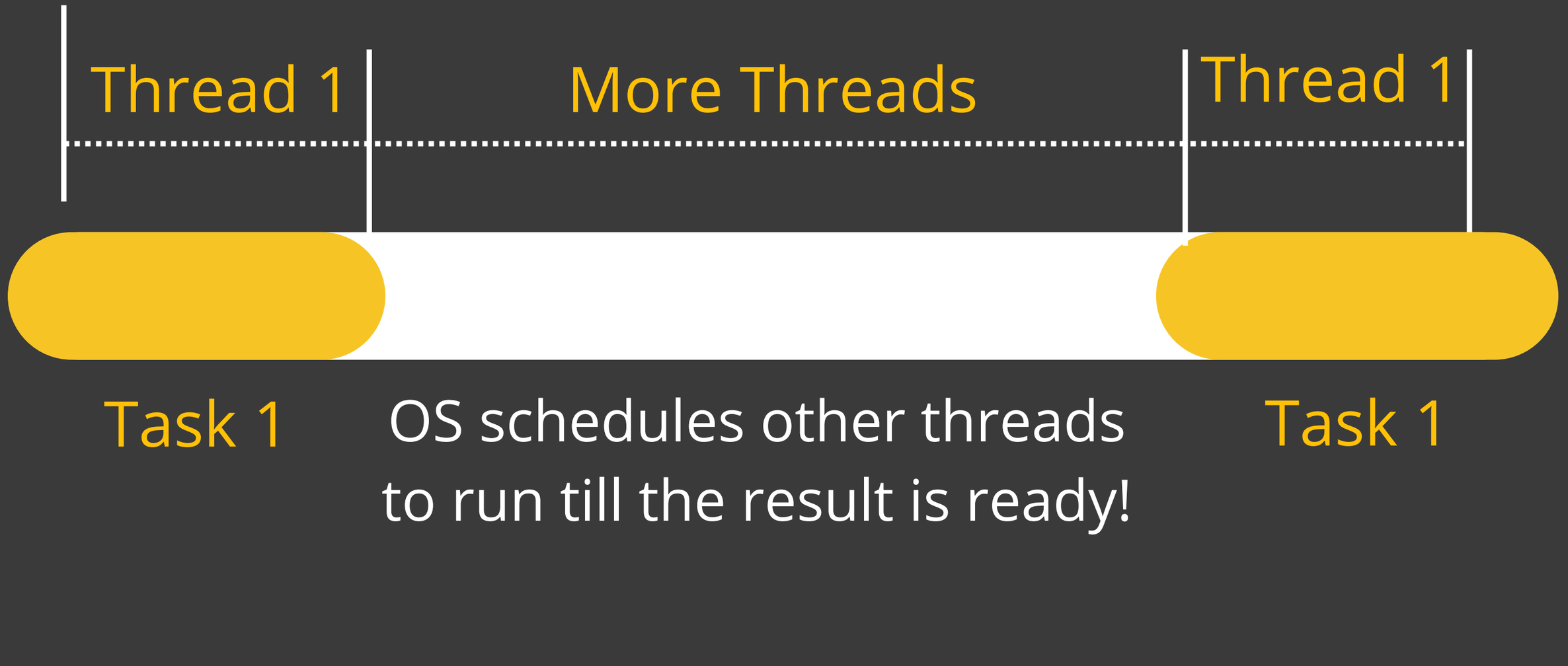
t1: the TH1 is taken out of the CPU to **Waiting State**

TH1

Waiting State

t2: OS schedules TH1 to **CONTINUE**!

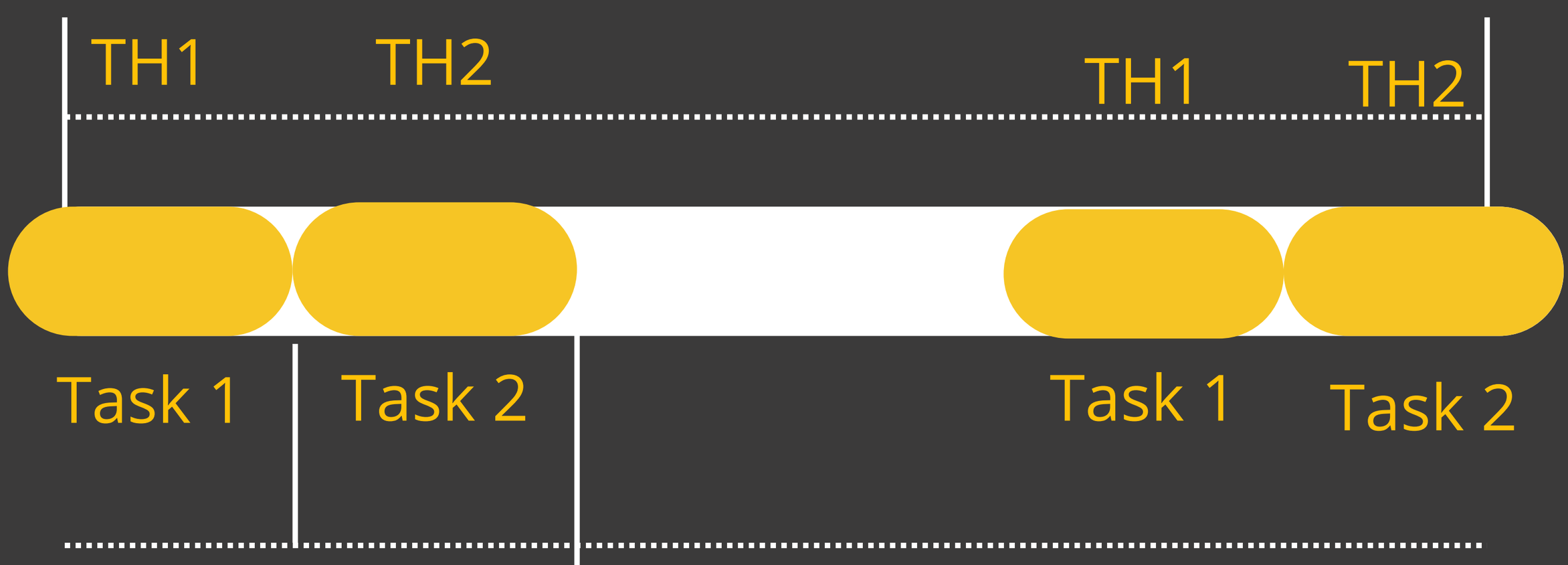
So this is the opportunity we have to run multiple-threads even on a single core to maximise the CPU utilisation!



So increasing Threads in I/O Bound Tasks would speed up the performance and maximise the CPU utilisation leveraging the Waiting Time for tasks completion.

Many threads will reduce the (free) CPU time which can be utilised

2	TH1, TH2
Submitted Tasks **I/O Bound Tasks**	Threadpool



The TH1 is taken out of the CPU to **Waiting State and TH2 is scheduled while TH1 completes.**

The TH2 is taken out of the CPU to **Waiting State** and **OS schedules other threads till any runnable thread completes.**

I/O BOUND TASKS

Threads that can be added for each **CPU CORE depends on the time for Single I/O Task Operation to complete!**

GENERAL FORMULA

With **CPU Intensive Tasks**, the waiting time is **ZERO** hence the result would be **Number of CORES**

With **I/O Bound Tasks**, the waiting time is Non-Zero hence the result would be **GREATER THAN OR EQUAL Number of CORES**

mahmoudyoussef0097@gmail.com

LinkedIN:mahmoud-youssef97