**Granularity / Grain Size (GITHUB) (test)**

Granularity nedir

In [parallel computing](https://www.wikiwand.com/en/Parallel_computing), granularity (or [grain](https://www.wikiwand.com/en/Granulation) size) of a [task](https://www.wikiwand.com/en/Task_(computing)) is a measure of the amount of work (or computation) which is performed by that task.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

Another definition of granularity takes into account the communication [overhead](https://www.wikiwand.com/en/Overhead_(computing)) between multiple processors or processing elements. It defines granularity as the ratio of computation time to communication time, wherein, computation time is the time required to perform the computation of a task and communication time is the time required to exchange data between processors.

Granularity = Computation time / Communication time

Granularity is usually measured in terms of the number of instructions executed in a particular task.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1) Alternately, granularity can also be specified in terms of the execution time of a program, combining the computation time and communication time.

**Types of parallelism**

Depending on the amount of work which is performed by a parallel task, parallelism can be classified into three categories: fine-grained, medium-grained and coarse-grained parallelism.

**Fine-grained parallelism**

In fine-grained parallelism, a program is broken down to a large number of small tasks. These tasks are assigned individually to many processors. The amount of work associated with a parallel task is low and the work is evenly distributed among the processors. Hence, fine-grained parallelism facilitates [load balancing](https://www.wikiwand.com/en/Load_balancing_(computing)).

As each task processes less data, the number of processors required to perform the complete processing is high. This in turn, increases the communication and synchronization overhead.

Fine-grained parallelism is best exploited in architectures which support fast communication. [Shared memory](https://www.wikiwand.com/en/Shared_memory) architecture which has a low communication overhead is most suitable for fine-grained parallelism.

It is difficult for programmers to detect parallelism in a program, therefore, it is usually the [compilers'](https://www.wikiwand.com/en/Compiler) responsibility to detect fine-grained parallelism.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

An example of a fine-grained system (from outside the parallel computing domain) is the system of [neurons](https://www.wikiwand.com/en/Neurons) in our [brain](https://www.wikiwand.com/en/Brain).[[4]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenoteRuss4)

[Connection Machine (CM-2)](https://www.wikiwand.com/en/Connection_Machine) and [J-Machine](https://www.wikiwand.com/en/J-Machine) are examples of fine-grain parallel computers that have grain size in the range of 4-5 μs.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

**Coarse-grained parallelism**

In coarse-grained parallelism, a program is split into large tasks. Due to this, a large amount of computation takes place in processors. This might result in load imbalance, wherein certain tasks process the bulk of the data while others might be idle. Further, coarse-grained parallelism fails to exploit the parallelism in the program as most of the computation is performed sequentially on a processor. The advantage of this type of parallelism is low communication and synchronization overhead.

[Message-passing](https://www.wikiwand.com/en/Message-passing) architecture takes a long time to communicate data among processes which makes it suitable for coarse-grained parallelism.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

[Cray Y-MP](https://www.wikiwand.com/en/Cray_Y-MP) is an example of coarse-grained parallel computer which has a grain size of about 20s.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

**Medium-grained parallelism**

Medium-grained parallelism is used relatively to fine-grained and coarse-grained parallelism. Medium-grained parallelism is a compromise between fine-grained and coarse-grained parallelism, where we have task size and communication time greater than fine-grained parallelism and lower than coarse-grained parallelism. Most general-purpose parallel computers fall in this category.[[4]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenoteRuss4)

[Intel iPSC](https://www.wikiwand.com/en/Intel_iPSC) is an example of medium-grained parallel computer which has a grain size of about 10ms.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

**Example**

Consider a 10\*10 image that needs to be processed, given that, processing of the 100 pixels is independent of each other.

Fine-grained parallelism: Assume there are 100 processors that are responsible for processing the 10\*10 image. Ignoring the communication overhead, the 100 processors can process the 10\*10 image in 1 clock cycle. Each processor is working on 1 pixel of the image and then communicates the output to other processors. This is an example of fine-grained parallelism.

Medium-grained parallelism: Consider that there are 25 processors processing the 10\*10 image. The processing of the image will now take 4 clock cycles. This is an example of medium-grained parallelism.

Coarse-grained parallelism: Further, if we reduce the processors to 2, then the processing will take 50 clock cycles. Each processor need to process 50 elements which increases the computation time, but the communication overhead decreases as the number of processors which share data decreases. This case illustrates coarse-grained parallelism.

|  |  |  |
| --- | --- | --- |
| **Fine-grain** : Pseudocode for 100 processors | **Medium-grain** : Pseudocode for 25 processors | **Coarse-grain** : Pseudocode for 2 processors |
| void main()  {  switch (Processor\_ID)  {  case 1: Compute element 1; break;  case 2: Compute element 2; break;  case 3: Compute element 3; break;  .  .  .  .  case 100: Compute element 100;  break;  }  } | void main()  {  switch (Processor\_ID)  {  case 1: Compute elements 1-4; break;  case 2: Compute elements 5-8; break;  case 3: Compute elements 9-12; break;  .  .  case 25: Compute elements 97-100;  break;  }  } | void main()  {  switch (Processor\_ID)  {  case 1: Compute elements 1-50;  break;  case 2: Compute elements 51-100;  break;  }  } |
| Computation time - 1 clock cycle | Computation time - 4 clock cycles | Computation time - 50 clock cycles |

**Levels of parallelism**

Granularity is closely tied to the level of processing. A program can be broken down into 4 levels of parallelism -

**Instruction level.**

**Loop level**

**Sub-routine level and**

**Program-level**

The highest amount of parallelism is achieved at instruction level, followed by loop-level parallelism. At instruction and loop level, fine-grained parallelism is achieved. Typical grain size at instruction-level is 20 instructions, while the grain-size at loop-level is 500 instructions.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

At the sub-routine (or procedure) level the grain size is typically a few thousand instructions. Medium-grained parallelism is achieved at sub-routine level.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1)

At program-level, parallel execution of programs takes place. Granularity can be in the range of tens of thousands of instructions.[[1]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotehwang1) Coarse-grained parallelism is used at this level.

The below table shows the relationship between levels of parallelism, grain size and degree of parallelism

|  |  |  |
| --- | --- | --- |
| Levels | Grain Size | Parallelism |
| Instruction level | Fine | Highest |
| Loop level | Fine | Moderate |
| Sub-routine level | Medium | Moderate |
| Program level | Coarse | Least |

**Impact of granularity on performance**

Granularity affects the performance of parallel computers. Using fine grains or small tasks results in more parallelism and hence increases the [speedup](https://www.wikiwand.com/en/Speedup). However, synchronization overhead, [scheduling](https://www.wikiwand.com/en/Scheduling) strategies etc. can negatively impact the performance of fine-grained tasks. Increasing parallelism alone cannot give the best performance.[[5]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotechen5)

In order to reduce the communication overhead, granularity can be increased. Coarse grained tasks have less communication overhead but they often cause load imbalance. Hence optimal performance is achieved between the two extremes of fine-grained and coarse-grained parallelism.[[6]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenote6)

Various studies[[5]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenotechen5)[[7]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenote7)[[8]](https://www.wikiwand.com/en/Granularity_(parallel_computing)#citenote8) have proposed their solution to help determine the best granularity to aid parallel processing. Finding the best grain size, depends on a number of factors and varies greatly from problem-to-problem.

-

-