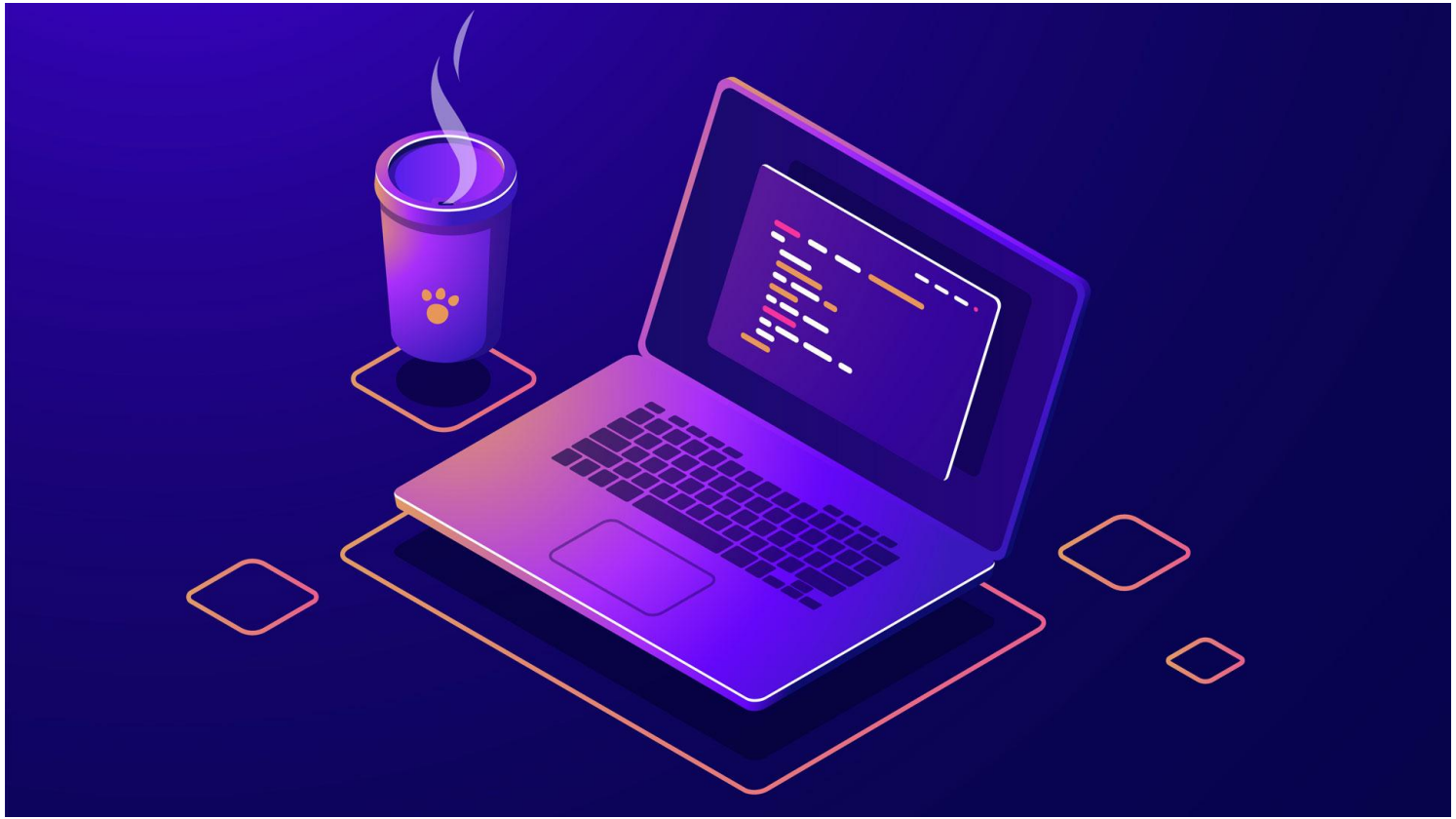
Task Definition



Task Definition

- Can the computation be divided in parts?[1]
  - Task decomposition: based on the processing to do (e.g. functions, loop iterations)
  - Data decomposition: based on the data to be processed (e.g. elements of a vector, rows of a matrix) (implies task decomposition)
  - There may be (data or control) dependencies between tasks
- Metrics to understand how our task/data decomposition can potentially behave
- Factors: granularity and overheads

Task Definition

- TDG: directed acyclic graph to represent tasks and dependencies between them
- Metrics:
  - $T_1 = \sum_{i=1}^{nodes}(work\_node_i)$
  - $T_\infty = \sum_{i \in criticalpath}(work\_node_i)$, assuming sufficient (infinite) resources
  - $Parallelism = T_1/T_\infty$
  - P_{min} is the minimum number of processors necessary to achieve $Parallelism$
- Task granularity vs. number of tasks

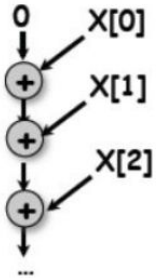
Example 1: Vector Sum

Compute the sum of elements `X[0] ... X[n-1]` of a vector `X`

```
sum = 0; for ( i=0 ; i< n ; i++ ) sum += X[i];
```

Task definition: each iteration of the `i` loop is a task.

- **TDG** (with input data):



- **Metrics:**

$T_1 \propto n$

$T_\infty \propto n$

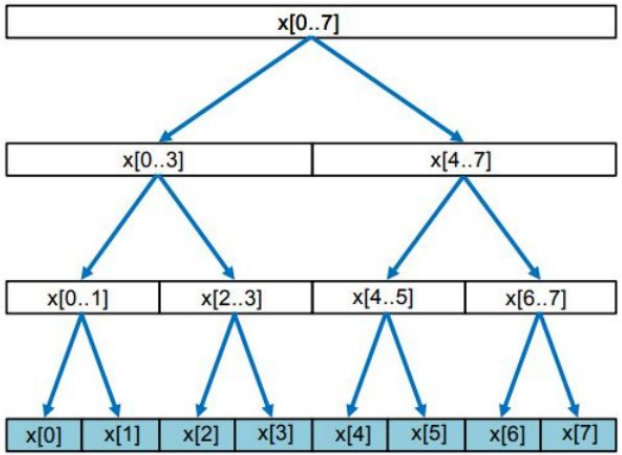
$Parallelism = 1$

How can we design an algorithm which leads to a TDG with more parallelism?

Example 1: Vector Sum

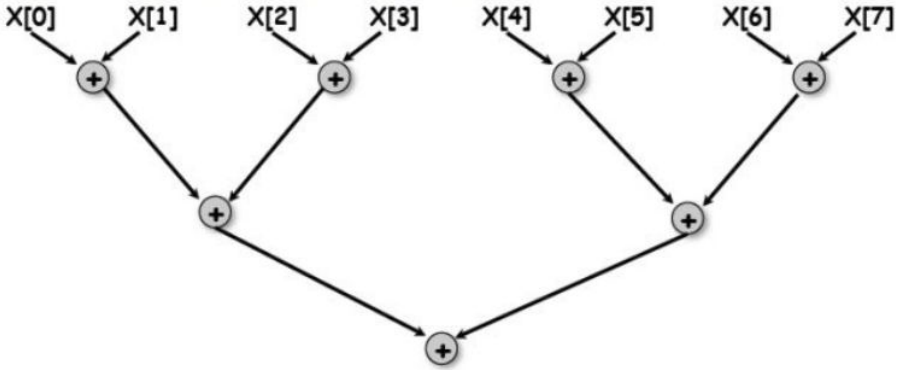
Writing a **recursive version** of the sequential program to compute the sum of elements `X[0] ... X[n-1]` of a vector `X`, following a *divide–and–conquer* strategy:

```
int recursive_sum(int *X, int n) {
   int ndiv2 = n/2;
   int sum=0;

   if (n==1) return X[0];

   sum1 = recursive_sum(X, ndiv2);
   sum2 = recursive_sum(X+ndiv2, n-ndiv2);
   return sum1+sum2;
}

void main() {
   int sum, X[N];
   ...
   sum = recursive_sum(X,N);
   ...
}
```



Example 1: Vector Sum

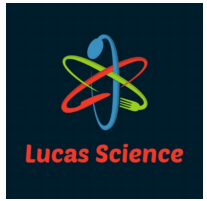
- **Task definition:** each invocation to `recursive_sum`
- **TDG** (with input data):



- **Metrics**:
  $T_1 \propto n$; $T_\infty \propto \log_2(n)$; $Parallelism \propto (n \div \log_2(n))$
- Same problem can be expressed with different algorithms/implementations leading to different metrics

Instructor Social Media

Youtube: Lucas Science



Instagram: lucaasbazilio

Twitter: lucasebazilio