Començat el	dissabte, 13 de novembre 2021, 11:36
Estat	Acabat
Completat el	dissabte, 13 de novembre 2021, 11:50
Temps emprat	13 minuts 59 segons
Punts	5,00/5,00
Qualificació	**10,00** sobre 10,00 (**100**%)

Pregunta **1**

Correcte

Puntuació 1,00 sobre 1,00

What kind of task decomposition will you use for a countable loop like the one shown below? (assuming that you don't modify the sequential version of the code)

```
for (int i = 0; i < n; i++)  {
    C[i] = A[i] + B[i];
}
```

Trieu-ne una:

- (Linear) Iterative task decomposition ✔ Very good.
- Recursive task decomposition

La teva resposta és correcta.

Pregunta **2**

Correcte

Puntuació 1,00 sobre 1,00

What kind of task decomposition will you use for an uncountable loop like the one shown below? (assuming that you don't modify the sequential version of the code)

```
for (int i = 0, int final = 0; i < function(n) && !final; i++) {
    if (A[i] + B[i] > MAX) final = 1;
    else C[i] = A[i] + B[i];
}
```

Trieu-ne una:

- Recursive task decomposition
- (Linear) Iterative task decomposition ✔ Yes! Although the loop is uncountable the potential parallelism is found in the execution of iterations of the loop.

La teva resposta és correcta.

Pregunta **3**

Correcte

Puntuació 1,00 sobre 1,00

What kind of task decomposition will you use to parallelize the execution of the following function?

```
void
function_increment(int * vector, int n) {
   int n2= n/2;
   if (n==0) return;
   if (n==1) vector[0]++;
   else {
     function_increment(vector,n2);
     function_increment(vector+n2,n-n2);
   }
}
```

Trieu-ne una:

- Recursive task decomposition — ✔ Great. Now the question is how? :)
- (Linear) Iterative task decomposition

La teva resposta és correcta.

Pregunta **4**

Correcte

Puntuació 1,00 sobre 1,00

Let's remember the differences between Leaf and Tree Recursive Task Decompositions.

Trieu-ne una o més:

- Leaf Recursive task decompositions allow the exploitation of the parallelism among all the tasks that are created for the leaves in a tree recursive traversal. ✔ Right!
- Tree Recursive task decompositions parallelize the traversal of the tree, usually reducing the overall parallel execution time. ✔ Right!

La teva resposta és correcta.

Pregunta **5**

Correcte

Puntuació 1,00 sobre 1,00

What kind of task decomposition will you use to parallelize the execution of the following program?

```
#define N 1024
#define MIN 16

void doComputation (int * vector, int n) {
   int size = n / 4;
   for (int i = 0; i < n; i += 4)
      compute(&vector[i], size);
}

void partition (int * vector, int n) {
   if (n > MIN) { // MIN is multiple of 4
      int size = n / 4;
      for(int i=0: i<4: i++)
```

```
        partition(&vector[i*size], size);
      }
   else
      doComputation(vector, n);
   return;
}
void main() {
   ...
   partition (vector, N); // N is multiple of 4
   ...
}
```

Trieu-ne una:

- Iterative only, either applied to the loop inside «doComputation» or to the loop inside partition.
- This program cannot be parallelised using the two strategies (iterative or recursive) presented in this video lesson.
- Recursive only, either with a leaf or tree strategy depending on where tasks are specified.
- Depends on the granularity one wants to exploit, it could be iterative inside function «doComputation» to reach fine-grain tasks and it could be recursive to reach coarser-grain tasks, leaf if tasks were applied to the invocation of «doComputation» or tree if tasks were applied to each recursive invocation of «partition». ✔ Right!

La teva resposta és correcta.

◄ Video lesson 6: iterative vs. recursive task decompositions

Salta a...

Problem after video lesson 6 ►