

TP 2

Master 2 SID

Benoist GASTON

benoist.gaston@univ-rouen.fr

OpenMP Fibonacci

- Considérer le programme `fib.c` (<https://github.com/benoistgaston/m2sid-2020.git>) qui calcul de manière récursive la suite de fibonacci.

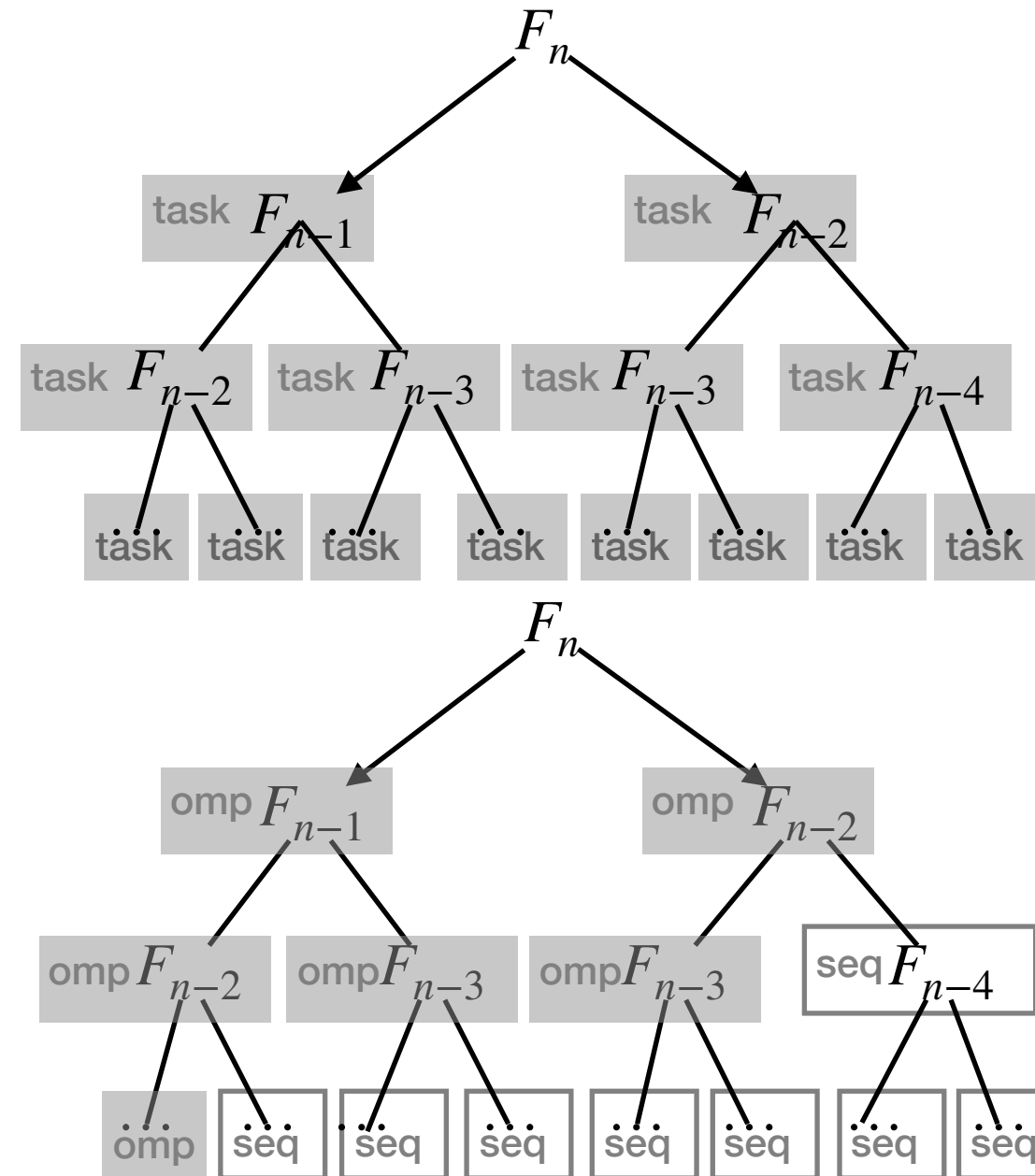
$$F_n = n, n = 0, 1$$

$$F_n = F_{n-1} + F_{n-2}, n \geq 2$$

- On se propose de paralléliser la fonction `fib_rec()` avec openmp sur le paradigme de distribution de tâches. Ce paradigme reprend l'exemple divide and conquer présenté en cours.

• Questions

1. Prendre en main le code ; le compiler à l'aide du makefile. Faire tourner pour des valeurs de n 10, 20, 30 40.
2. Sur la base de la fonction `fib_rec()`, écrire une fonction `fib_omp()` parallélisant à l'aide de tâche.
3. Observer les performances de la version parallèle (utiliser la commande système `time`).
4. Pour résoudre le problème de performance constaté, définir dans `fib_omp()` un seuil en dessous duquel la fonction `fib_rec()` sera appelée à la place de `fib_omp()`.
5. Observer les performances de cette version hybride.



OpenMP Bubble Sort

- Considérer le programme `b-sort.c` (<https://github.com/benoistgaston/m2sid-2020.git>) qui propose une implémentation du tri à bulle.

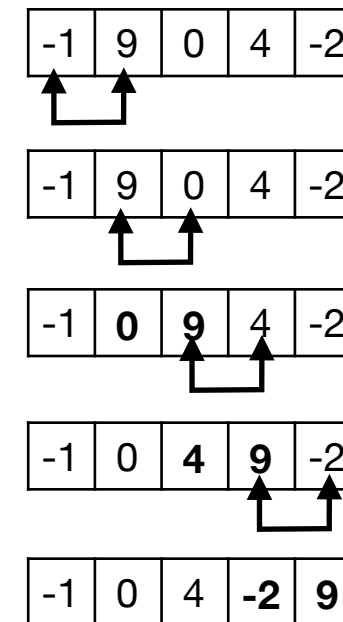
```

tri_à_bulles(Tableau T)
    pour i allant 1 de (taille de T)-1
        pour j allant de 0 à i-1
            si T[j+1] < T[j]
                échanger(T[j+1], T[j])
    
```

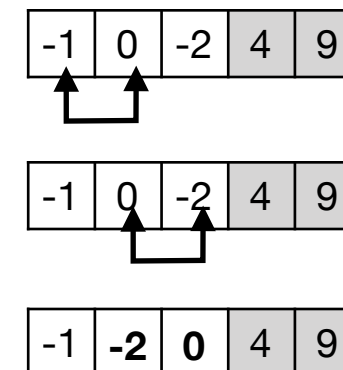
- On se propose de paralléliser la fonction `b-sort()` avec openmp.

Questions

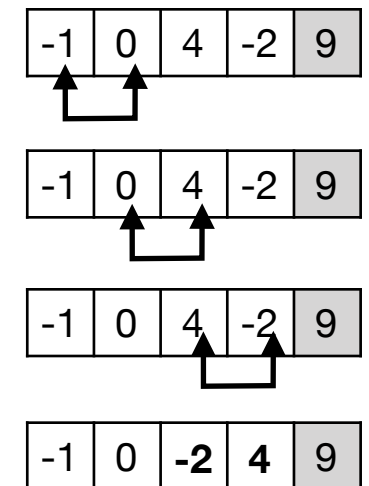
- Prendre en main le code ; le compiler à l'aide du makefile.
- Tenter une directive `parallel for` sur la boucle interne.
- Y a-t-il une erreur à la compilation ? À l'exécution (tester plusieurs fois) ?
- Comment peut-on modifier l'algorithme pour la boucle `for` parallélisable ?
- Modifier l'algorithme et le paralléliser.



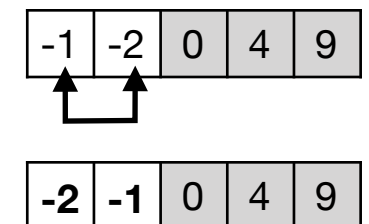
Étape 1



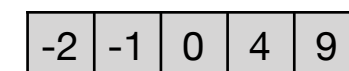
Étape 3



Étape 2



Étape 4



Final