Alessandro Bianchi

Esericizio 1

Utilizzando degli stream non parallelizzati (prima versione richiesta) si ottiene un tempo di esecuzione di 5080 ms, con un tempo effettivo di computazione di 686 ms.

Utilizzando invece i parallelStream (quindi parallelizzati), si ottiene un tempo di esecuzione pari a 4674 ms, con un tempo di computazione effettivo di 471 ms.

I paralleStream risultano quindi, come intuibili, più veloci della loro versione sequenziali.

Esercizio 2

Utilizzando le CompletableFutures, l'applicazione ha eseguito tutte le istruzioni in un tempo di 4775 ms, senza alcuna distinzione tra tempo totale e tempo di computazione (risultano infatti uguali).

Rispetto all'esecuzione con gli stream, le CompletableFutures risultano più veloci degli stream non parallelizzati ma su tempi pressoché identici rispetto ai parallelStream.

Esercizio 3

Analizzando l'esecuzione delle versioni con parallelStream e CompletableFutures con Java Mission Control, è possibile trovare delle differenze in quali thread si dividono il compito di processare i dati: utilizzando le parallelStream è possibile notare che il compito di eseguire le varie ricerca è a carico del main thread, mentre utilizzando le CompletableFutures questo compito viene passato al ForkJoinPool dell'applicazione.

Thread	Profiling Samples
main	330
ForkJoinPool.commonPool-worker-15	5
ForkJoinPool.commonPool-worker-13	4
RMI TCP Connection(1)-192.168.138.1	4
ForkJoinPool.commonPool-worker-9	4
ForkJoinPool.commonPool-worker-2	4
ForkJoinPool.commonPool-worker-4	3
ForkJoinPool.commonPool-worker-11	3
ForkJoinPool.commonPool-worker-6	2

parallelStream

Thread	Profiling Samples
ForkJoinPool.commonPool-worker-9	315
RMI TCP Connection(1)-192.168.138.1	9
ForkJoinPool.commonPool-worker-13	5
ForkJoinPool.commonPool-worker-4	4
ForkJoinPool.commonPool-worker-2	4
ForkJoinPool.commonPool-worker-11	3
main	3
ForkJoinPool.commonPool-worker-6	2
Reference Handler	1

CompletableFuture