Inteligencia Computacional para datos de alta dimensionalidad

Práctica 2: Spark

Find a Highest-Scoring Fitting Alignment of Two Strings

Say that we wish to compare the approximately 20,000 amino acid-long NRP synthetase from Bacillus brevis with the approximately 600 amino acid-long A-domain from Streptomyces roseosporus, the bacterium that produces the powerful antibiotic Daptomycin. We hope to find a region within the longer protein sequence v that has high similarity with all of the shorter sequence w. Global alignment will not work because it tries to align all of v to all of w; local alignment will not work because it tries to align substrings of both v and w. Thus, we have a distinct alignment application called the Fitting Alignment Problem.

"Fitting" w to v requires finding a substring v' of v that maximizes the global alignment score between v' and w among all substrings of v.

Objetivos:

1. Aprender a programar y ejecutar códigos paralelos con Spark
2. Dominar la consola de Spark
3. Entender los retos a los que nos enfrentamos al paralelizar un código secuencial.

¿Qué hay que hacer?

1. Implementar el algoritmo secuencialmente (o buscar una implementación de otra asignatura y asegurarse de que funciona) (http://rosalind.info/problems/ba5h/)
2. Utilizar Spark para, dada la cadena de referencia del fichero cadena.txt, encontrar el highest-scoring fitting aligment de todas las cadenas incluidas en el fichero dataset.txt. Al acabar se debe devolver, para la cadena que consigue la mayor puntuación y para la que consigue la mínima puntuación, el score y las cadenas modificadas (conteniendo gaps), tal como se muestran en Rosalind.
3. Utilizando la consola de Spark detecta los principales cuellos de botella de tu código (las instrucciones que consumen más tiempo de ejecución). ¿Cuáles son y por qué?
4. Ejecuta el código paralelo utilizando 1, 2, 4 y 8 workers y dibuja una gráfica con los tiempos de ejecución. Si está bien paralelizado, a mayor número de workers el tiempo de ejecución debería ser menor, aunque siempre con un límite (*ley de Amhdal*). Además, va a depender del hardware disponible así que es posible que en tu máquina virtual no se observe mejora. Comenta la gráfica indicando si hay mejoras o no y por qué.
5. **[Difícil]** Mejora el rendimiento de tu código Spark utilizando caches.
6. El fichero schizophrenia.fasta se ha descargado de la web del NCBI y contiene secuencias de mRNA asociadas a la esquizofrenia humana. Sin embargo, el formato del fichero no es adecuado para procesar directamente con Spark, porque cada elemento ocupa más de una línea y Spark construye los RDDs a partir de ficheros partiéndolos por líneas. Es, por tanto, necesario un paso de preprocesado para poder utilizarlo. El script fastaReader.py lee este fichero y devuelve una lista de tuplas en la que cada tupla contiene por una parte un texto que identificación a cadena y por otra la propia cadena. Describe cómo funciona cada paso de este script. Enfócate en describir qué forma tiene cada elemento del RDD tras cada paso.

7. Modifica tu código Spark para utilizar como entrada el RDD que produce fastaReader.py en vez del obtenido a partir fichero dataset.txt. La respuesta deberá contener, además de lo indicado en el apartado 2, el texto identificativo de la cadena.

Entrega:

La entrega se realizará a través del Moodle. Se debe entregar el código de la versión Spark que lee el fichero dataset.txt (el punto 2) y de la versión Spark utilizando fastaReader.py (punto 7). También se entregará una memoria describiendo los códigos entregados y con las respuestas a las preguntas formuladas.

Recuerda que la práctica es por parejas, preferiblemente alguien de informática con alguien de ciencias, para ayudarse mutuamente (equipo multidisciplinar). Por eso en la memoria o en un fichero aparte se debe indicar quienes componen la pareja.