Métodos eficientes para la cuantificación de muestras de ADN

Eloy Bardají Pujadas

08/03/2022

Resumen–
Este trabajo se ha centrado en el estudio y optimización de una aplicación bioinformática usada para obtener estimaciones cuantitativas de las especies contenidas en una muestra compleja de ADN. La aplicación es un WorkFlow lineal compuesto por varias etapas, en las cuáles se van procesando las diversas muestras.
A lo largo del proyecto se han realizado múltiples experimentos, en su mayoría relacionados con el paralelismo y la herramienta BWA, con el objetivo de mejorar el rendimiento del WorkFlow. También se crea una heurística con la que optimizar los tiempos de ejecución. Las ejecuciones del trabajo ser realizan en un clúster.
Los resultados finales usando la mejor configuración y la mejor heurística han permitido reducir los tiempos de ejecución significativamente frente a otras configuraciones y heurísticas usadas.

Palabras clave– Optimización, ADN, genomas, paralelización, heurística, BWA

Abstract–
This work has focused on the study and optimization of a bioinformatics application used to obtain quantitative estimates of the species contained in a complex DNA sample. The application is a linear workflow made up of various stages, in which the various samples are processed.
Throughout the project, multiple experiments have been carried out, mostly related to parallelism and the BWA tool, with the aim of improving the performance of workflow. A heuristic is also created with which to optimize execution times. Job executions are performed on a cluster.
The final results using the best configuration and the best heuristic have allowed to reduce execution times significantly compared to other configurations and heuristics used.

Keywords– Optimization, DNA, genomes, parallelize, heuristics, BWA

ÍNDICE

1 INTRODUCCIÓN

La optimización de programas y de WorkFlows[10] puede abordarse desde muchos puntos diferentes, es este caso se dispone de un clúster[11] para realizar los experimentos. Dicho clúster cuenta con 12 nodos y cada uno de ellos dispone de 6x2Cores. Además, este clúster cuenta con una cola de trabajos tipo SLURM.

El proyecto del que procede dicho WorkFlow se llama "Estimation of the relative abundance of species in artificial mixtures of insects using low-coverage shotgun metagenomics"[4] de Lidia Garrido-Sanz y consiste en el análisis y comparación de diferentes ADN para estudiar la población animal de un determinado territorio.

A partir de aquí se debe buscar la forma de acelerar el programa, ya que la base de datos que tiene el programa es de 22 muestras x 110 genomas, sin embargo, cuando esta base de datos crezca el tiempo de ejecución crecerá con ella, por lo que la optimización es una tarea clave para que el proyecto sea útil y realmente funcional.

El trabajo está organizado por los siguientes apartados:
- Objetivos: en este primer apartado se explican las metas del proyecto.
- Metodología y planificación: se expone el tipo de metodología que se utiliza a lo largo del proyecto, además de informar de la planificación de este y comprender el Workflow del problema.
- Análisis teórico: se realiza un análisis de la complejidad del problema de forma teórica.
- Mejoras realizadas: se exponen las mejoras que se han realizado para que el programa funcione más rápido.
- Heurísticas: se explican las dos heurísticas que se usan durante el estudio.
- Experimentos con BWA[5]: se explican y analizan los diferentes experimentos que se han realizado sobre la herramienta de mapeo BWA.
- Conclusiones: Reflexiones y conclusiones finales sobre los experimentos y el trabajo efectuado.

2 OBJETIVOS

El objetivo principal del proyecto es optimizar el WorkFlow al máximo, sin embargo, es difícil poder afirmar que un programa no pueda optimizarse más. Para cumplir este objetivo se utilizan diversas herramientas como scripts, heurísticas...

Como objetivo secundario: la obtención de un resultado final que pueda usarse de forma real por investigadores para analizar diferentes muestras y genomas.

3 METODOLOGÍA Y PLANIFICACIÓN

Se ha utilizado la metodología KANBAN[9] para poder gestionar y realizar tareas de forma ágil. Para llevar un correcto control y gestión del proyecto se ha utilizado la aplicación web Trello[6], la cual sirve de ayuda para la organización y gestión de las diferentes tareas que se han ido asignando en el proyecto.
Se va a utilizar una planificación flexible en la que varias tareas pueden verse activas simultáneamente, por ejemplo: Aunque se hubiere terminado de modificar la estructura del programa, si posteriormente al modificar el algoritmo principal se requiriese editar de nuevo la estructura de este se podría volver a modificar. Por ello se opta por una organización más dinámica de las tareas a realizar.

El proyecto no ha contado con una planificación del todo eficaz debido a problemas externos, sin embargo, la planificación y las tareas que se realizan son las siguientes:
- Elaboración de un estudio teórico como análisis inicial
- Creación de scripts para la gestión y administración de tareas
- Optimización utilizando paralelización en las ejecuciones con BWA
- Elaboración de heurísticas
- Pruebas con diferentes heurísticas

A continuación se muestra el WorkFlow del programa (Figura 1) y se observa el funcionamiento que tiene el programa. En primer lugar, existe una fase de filtrado que no se ve reflejada en el diagrama, una vez filtrados los datos de entrada se ejecuta BWA, que mapea las muestras con los genomas. Una vez está mapeada la muestra con todos los genomas se utiliza la herramienta SamTools, que no aparece en el diagrama. Finalmente, se ejecuta el algoritmo alfa-delta, el cual decide si los resultados obtenidos se pueden asignar a una especie o, por el contrario, se descartan.

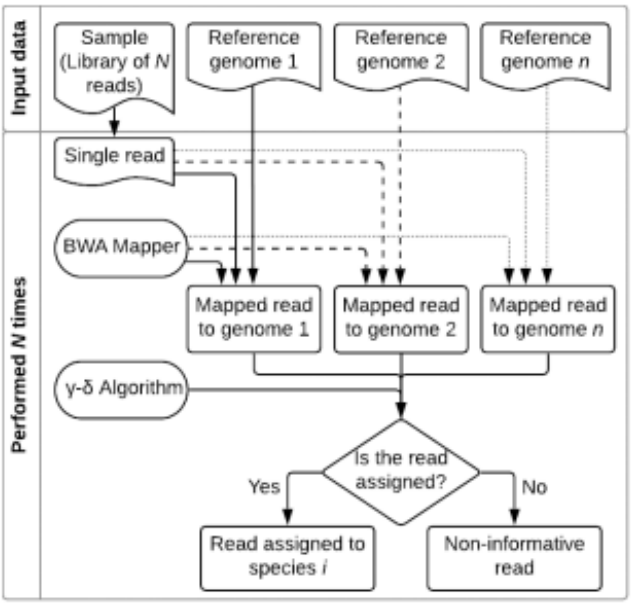

Fig. 1: Workflow del pipeline completo (Extraido de la referencia [4])

Además, se realiza una ejecución inicial de todo este WorkFlow en la que se observa que el peso del mapeo por parte de BWA ocupa más del 95 % del tiempo total de la ejecución. Por ello se va a focalizar la atención en dicho proceso, BWA, para realizar en él las mejoras y los análisis pertinentes.

4 ANÁLISIS TEÓRICO

En primer lugar, se ha realizado un análisis de forma teórica del código y la complejidad de este, además de la

interconexión de los diferentes procesos que se ejecutan en el programa.

Esto deja ver que el problema principal de ejecución y donde más tiempo se puede ganar es en la ejecución de BWA(BWA es un software utilizado para mapear secuencias poco divergentes contra un genoma de referencia), por lo que la atención principal recaerá en este proceso.

Antes de optimizar nada se ejecuta un análisis de la complejidad del programa y de cada uno de sus procesos, pudiendo así determinar como se enfoca la parte de optimización. En este caso se ha determinado que la complejidad de BWA es polinómica, donde encontramos que el tiempo de ejecución depende de los tamaños de las muestras y de los genomas. Además, tras una investigación exhaustiva del comportamiento de BWA según las diferentes muestras y genomas se ha podido llegar a la conclusión de que el tiempo de ejecución está correlacionado con dichos tamaños.

Se intenta conseguir una función con la que poder estimar los tiempos de ejecución de BWA solo con conocer los tamaños de la muestra y el genoma a comparar, sin embargo, no se consigue llegar a una fórmula como tal, ya que la desviación que se ha obtenido con estos cálculos teóricos es demasiado elevada. Todo esto se expone con datos en el apartado 7.3.

Aunque no se ha podido llegar a una fórmula general para todo BWA, si se pueden obtener buenas aproximaciones en torno a cada muestra, es decir, conociendo una muestra con anterioridad podemos extraer una fórmula en la que según el tamaño del genoma obtengamos un tiempo aproximado de cuanto tardará la ejecución con esa combinación muestra/genoma. Esto puede resultar útil para hacer unas aproximaciones iniciales.

A continuación vemos en esta gráfica (Figura 2) como se comporta una muestra determinada respecto al tiempo/tamaño de cada uno de los genomas que se han comparado aquí(Un total de 14 genomas). Además, se muestra la fórmula polinómica mencionada anteriormente para esta determinada muestra (5-LH).

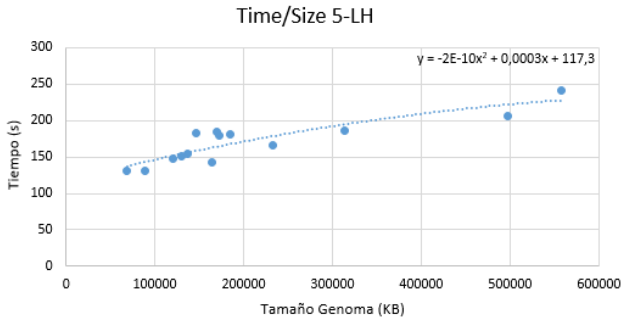

Fig. 2: 5-LH comparando tiempo y tamaño de los genomas

Una vez realizados los análisis iniciales la principal prioridad es optimizar el proceso BWA, ya que es el cuello de botella de este programa.

Adicionalmente, se estudia si BWA puede ser paralelizable y como realizar las ejecuciones para que lo sea.

5 Estudios individualizados

Tras haber efectuado un previo estudio ya se conoce que el mayor tiempo de ejecución del programa se produce en el proceso BWA, por ello las mejoras propuestas son principalmente para optimizar el tiempo de ejecución de este proceso.

Una vez se conoce que BWA es el cuello de botella del WorkFlow se debe estudiar si BWA es paralelizable y de que forma lo es. Esta herramienta es paralelizable de múltiples formas, por lo que los experimentos que se realicen con ella también variarán. Tiene la capacidad de ser ejecutado por la cantidad de threads que se necesite, puede ejecutarse con varias instancias a la vez... Por todo esto se realizan experimentos diversos en el apartado 7, con la intención de obtener la mejor solución posible.

5.1. Paralelismo en BWA

Como primera opción para mejorar el programa se ha planteado la utilización de herramientas de paralelismo, para así optimizar los procesos. En primer lugar, se realizan pruebas acerca del número de cores y threads óptimos para el programa (Estudio en apartado 7.1). Debe tenerse en cuenta que la mejor configuración para un clúster puede no serlo para otro, por ello se intentará buscar la mejor configuración para el clúster que se está utilizando actualmente.

Como pruebas iniciales se realizan pruebas en las que se varía el número de Cores asignados a cada tarea de BWA y con ello se podrá comparar el rendimiento de las diferentes configuraciones. De tal forma que se pueda afirmar cuál es la mejor configuración, ya que será la que se utilice en próximas mejoras.

5.2. Script general de control

Esta mejora no se centra tanto en la optimización de tiempo en sí sino en el ahorro de tiempo en la gestión del lanzamiento de tareas y procesos. Se crea un script en el que sea fácil manipular y lanzar las tareas de una forma sencilla. Además de esto que tenga la capacidad de utilizar paralelismo, como hemos comentado anteriormente, que será lo que haga que esta mejora haga reducir el tiempo de ejecución de los procesos.

Dentro del script general de control tenemos un script genérico de ejecuciones de BWA, el cual sirve para seleccionar y controlar como se hacen las ejecuciones de BWA, controlando número de Cores, Heurística seleccionada...

5.3. Scripts de gestión

Además de los scripts creados anteriormente, se necesitan otros dos scripts, los cuales tienen como funcionalidad la gestión de datos y de carpetas.

El primero se utiliza para crear los directorios necesarios para la ejecución de BWA y lo hace de forma ordenada y limpia, para que se pueda trabajar de forma adecuada.

El segundo script lo que hace es: una vez se ha realizado una ejecución o el número deseado de ejecuciones, se configurará este script para que extraiga los tiempos de ejecución de forma simple a un fichero para poder exportarlos a una herramienta que nos permita un mejor procesamiento de los datos.

6 HEURÍSTICAS

Se utilizan dos heurísticas diferentes con la idea de reducir el tiempo de ejecución inicial. A continuación se habla detalladamente acerca de esto, ya que la heurística es un aspecto importante a la hora de optimizar un programa como este.

6.1. Heurística Random (Orden Alfabético)

La heurística utilizada por defecto en el programa es una heurítica de tipo aleatorio, en este caso es por orden alfabético, pero a modo de heurística esto sería totalmente aleatorio, ya que el ordén alfabético no ofrece ningún tipo de orden correlativo a métricas. Por ello nos referiremos a esta heurística como aleatoria/random. Tras hacer un estudio teórico del funcionamiento del programa se puede deducir que no es la mejor heurística posible. Por ello se plantea la siguiente solución.

6.2. Heurística Big-Big

Esta heurística planteada se trata de una heurística simple pero eficaz. Esta heurística se basa en la realización de los trabajos más grandes al principio y de los más cortos al final, es una heurística que se usa frecuentemente y que ofrece buenos resultados. En este caso lo que se hace es ejecutar los procesos de BWA con muestras más grandes en primer lugar, también se hará lo propio con los genomas que se ordenarán de mayor a menor, por lo que se consigue un uso más eficiente de los recursos del sistema, además de una reducción de tiempos de ejecución. (como se puede ver en apartado 7.1)

7 RESULTADOS EXPERIMENTALES CON BWA

Como se ha comentado anteriormente, los análisis y experimentos se centran en el proceso BWA, ya que ocupa casi la totalidad del tiempo de ejecución del pipeline.

7.1. Análisis Iniciales

En primer lugar, se analiza el programa original, de esta forma se toman unos tiempos iniciales y se extrae información relevante. En la siguiente tabla (Tabla 1) podemos observar como la ejecución completa de dicho proceso tarda cerca de 51 horas, donde más del 95 % del tiempo se está ejecutando BWA.

N* de muestras	N* de genomas	N* de cores	N* de Threads	Tiempo obtenido	Tiempo por muestra
3	110	6	12	6:07:00	2:02
6	110	6	12	12:44:00	2:07
11	110	6	12	24:19:00	2:12
22	110	6	12	50:38:00	2:18

Tabla 1: BWA tabla inicial

Se lanzan ejecuciones parciales para observar como el tiempo por muestra se ve afectado. Las muestras se seleccionan de forma que el tamaño de estas sea similar en todas las ejecuciones, ya que hay muestras que tienen un tamaño muy superior a otras. Por ello no se puede asegurar que el tiempo por muestra evolucione exactamente como se ha proyectado en la tabla. También podemos ver como el tiempo por muestra crece de forma prácticamente lineal.

A continuación vemos una tabla que nos muestra la comparación entre diferentes configuraciones, que varían en el número de Cores utilizados para la ejecución del proceso BWA. En este caso se utiliza la muestra 0-PM comparada con todos los genomas que se tienen a disposición (110). Se utiliza el modo Exclusive que permite la utilización de un Nodo de forma que no haya ruido por parte de otros usuarios del sistema, ya que se está utilizando una cola SLURM.

Tiempo según número de cores Utilizado (UTILIZADO DE FORMA EXCLUSIVE)

N* de muestras	N* de genomas	N* de cores	N* de Threads	Tiempo obtenido (min)	SpeedUp	T*core	Eficiencia
1 (0-PM)	110	1	1	110,29	1	110,29	1
1 (0-PM)	110	2	2	66,02	1,670554377	132,04	0,835277189
1 (0-PM)	110	3	3	50,43	2,18699187	151,29	0,72899729
1 (0-PM)	110	4	4	39,51	2,791445204	158,04	0,697861301
1 (0-PM)	110	6	6	29,11	3,788732394	174,66	0,631455399
1 (0-PM)	110	8	8	28,34	3,891672548	226,72	0,486459068
1 (0-PM)	110	12	12	26,22	4,206331045	314,64	0,350527587

Tabla 2: Tabla inicial según número de cores

Tras ver la tabla se pueden sacar conclusiones adicionales. Tenemos una mejora de tiempos respecto a un mayor uso de nodos y por contra tenemos que la eficiencia disminuye al utilizar más nodos, como se ve en las siguientes gráficas. (Figura 3 y Figura 4)

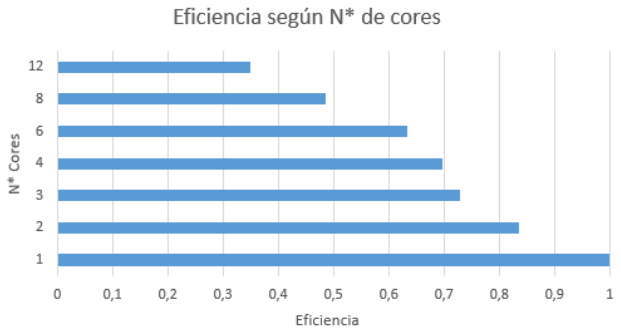

Fig. 3: Eficiencia según número de cores

La eficiencia puede variar entre [0-1] como valores mínimos y máximos. Viendo el gráfico anterior se puede destacar la pérdida de eficiencia con las configuraciones de 8 y 12 Cores.

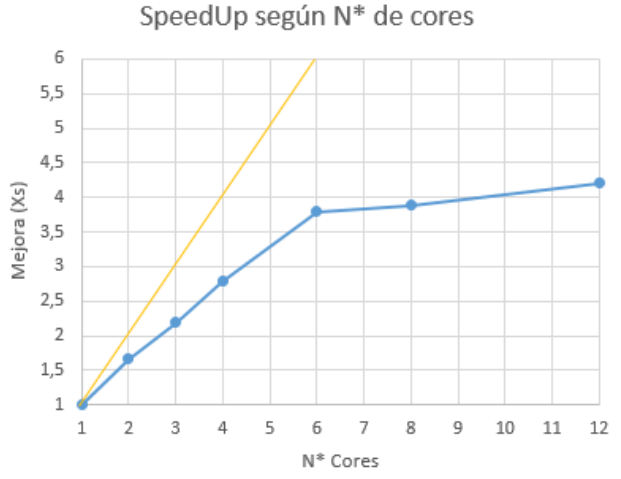

Fig. 4: SpeedUp según número de cores

En esta última gráfica de SpeedUp la recta amarilla (Figura 4) representa el SpeedUp ideal que se podría obtener. Se extraen varias conclusiones de este experimento, en primer lugar, se observa como el SpeedUp con el uso de 8 y 12 Cores mejora mucho menos que con el de 2, 3, 4 y 6 cores. Esto está relacionado directamente con la eficiencia analizada en las figuras anteriores, por lo que a priori se descartan como mejores configuraciones para la optimización de BWA el uso de 8 y 12 Cores.

7.2. Estudio BWA con 1 y 6 Cores

A continuación se analizan los datos obtenidos en las pruebas de ejecución del archivo adjunto Cálculos (Hojas 6 Cores y 1 Core) en las que se obtienen los datos de ejecutar 4 diferentes muestras con 14 genomas, aunque en las figuras 5 y 6 solo se muestra la ejecución de 4 genomas diferentes, con la intención de que sea más visual. La finalidad de este experimento es la comprobación de sí el tiempo de ejecución de BWA está estrictamente relacionado con el tamaño de las muestras y de los genomas que se analicen en él. Adicionalmente, se realiza el experimento utilizando 1 y 6 Cores, ya que de esta forma se puede asegurar si la relación existe o no.

En las siguientes figuras (Figuras 5 y 6) se observa de forma gráfica como los tamaños de los genomas(línea gris) están correlacionados con el tiempo de ejecución que tarda una determinada ejecución con una muestra y genoma. También se observa que el tamaño de la muestra es determinante para el tiempo de ejecución del proceso, ya que hay ejecuciones que tardan más de 5 veces más que otras solo por el tamaño de la muestra, como podemos ver a continuación.

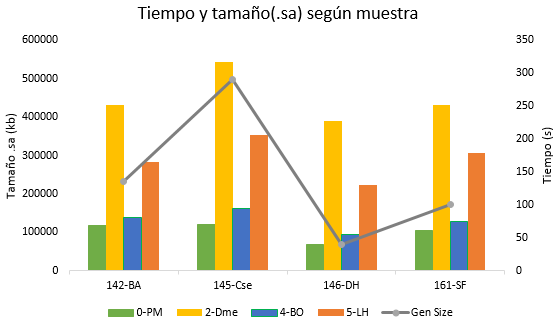

Fig. 5: Tiempo/Tamaño de los genomas con 1 Core

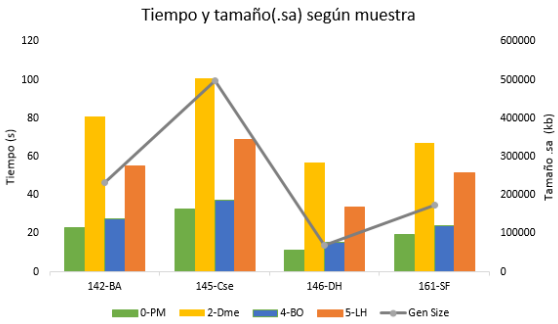

Fig. 6: Tiempo/Tamaño de los genomas con 6 Cores

En ambos gráficos podemos ver similitudes, ya que finalmente el uso de 1 o 6 cores no es significativo para conocer que los tamaños y los tiempos están relacionados. A continuación se muestra una pequeña tabla (Tabla 3) con los tamaños de las muestras comparadas en las gráficas anteriores para que se puedan valorar mejor las gráficas sin necesidad de consultar el documento con los datos completos (Cálculos (Hojas 6 Cores y 1 Core)).

Info tamaños muestras	
Muestras	Tamaño .fastq
0-PM	56709
4-BO	75880
2-Dme	297903
5-LH	183730

Tabla 3: Tamaño muestras

Finalmente, se puede observar como a tamaños menores de las muestras menores son los tiempos y viceversa, remarcar de nuevo que los tamaños de los genomas hacen que el tiempo varíe notablemente. Gracias a este análisis en el próximo apartado se intenta conseguir un análisis de forma más teórica acerca de estas relaciones tamaño/tiempo de las muestras y los genomas.

7.3. Estudio relación Tamaño y Tiempo

Una vez se ha concluido que el tiempo que tarda BWA en ejecutarse está relacionado con los tamaños de las muestras y los genomas que se analizan en él, se procede a realizar un análisis más preciso acerca de esto, con la intención de obtener una fórmula que permita estimar el tiempo que tardará BWA (en las máquinas que se están utilizando) según el tamaño de la muestra y del genoma que se vaya a analizar.

A continuación se observan un conjunto de gráficas(Figuras 7, 8, 9 y 10) que nos muestran los patrones con relación a los tamaños de un genoma y el tiempo que tarda la ejecución de BWA sobre una determinada muestra. Además, se indica una fórmula con la que estimar los tiempos de ejecución para esa determinada muestra.

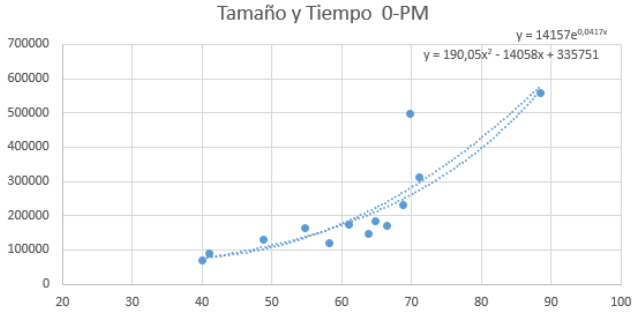

Fig. 7: Tiempo y tamaño de los genomas para la muestra 0-PM

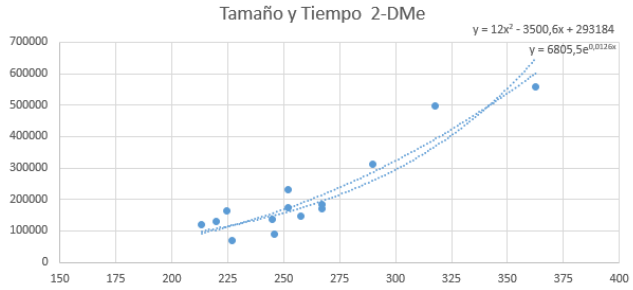

Fig. 8: Tiempo y tamaño de los genomas para la muestra 2-DMe

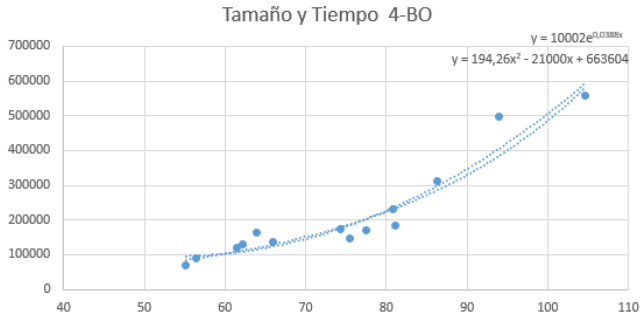

Fig. 9: Tiempo y tamaño de los genomas para la muestra 4-BO

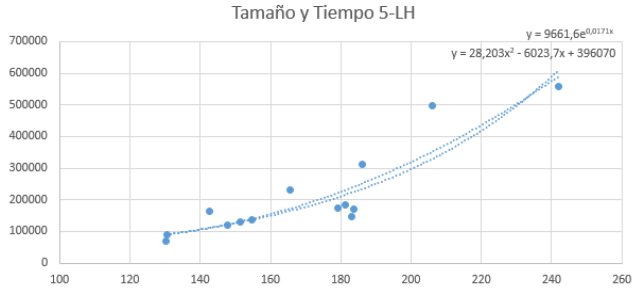

Fig. 10: Tiempo y tamaño de los genomas para la muestra 5-LH

Se observa una similitud en la relación entre estas muestras, por lo que un análisis extrapolando dichos datos para obtener una fórmula general que no dependa del tamaño de la muestra tiene sentido.
Tras intentar conseguir una fórmula que cubra todas las muestras y genomas se ha llegado a la conclusión de que el tiempo que tarda BWA en ejecutarse está totalmente relacionado con los tamaños tanto de la muestra como del genoma, pero que a su vez existen más factores que hacen que el tiempo de BWA varíe, por ello no se puede deducir una fórmula general que englobe todos los casos de forma precisa.

A continuación se muestra el estudio de una muestra concreta como es 2-DMe utilizando como parámetros un coeficiente que se obtiene de la división del tiempo de la ejecución de la muestra con un determinado genoma y el tamaño de la muestra (Figura 11). Esto se observa en la figura que tenemos a continuación.

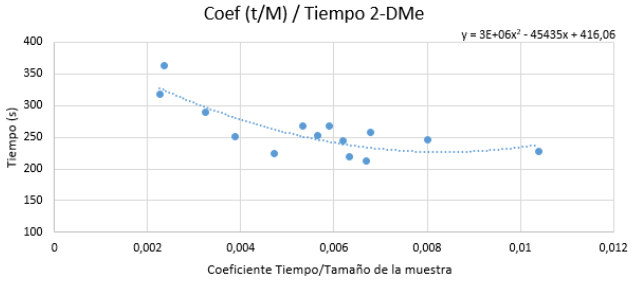

Fig. 11: Comparación entre coeficiente tiempo/Tamaño de la muestra para 2-DMe

Además, se observa la fórmula de tipo polinómica que se ajusta correctamente a los datos obtenidos, como se ha comentado con anterioridad.

Con este estudio se comparan los tiempos obtenidos de forma práctica con los tiempos que se obtendrían de forma teórica para la muestra 2-DMe, además se evalúa la desviación que tienen esas estimaciones respecto a la realidad que se presentan en la siguiente tabla (Tabla 4).

Tiempos reales	Estimaciones	Desviación
245,007	244,1507	0,35%
251,766	276,3902	9,78%
317,868	319,3520	0,47%
226,996	212,7342	6,28%
245,649	228,0133	7,18%
224,679	262,4091	16,79%
289,967	290,2646	0,10%
266,964	253,9266	4,88%
257,619	238,1834	7,54%
267,108	247,2842	7,42%
362,65	315,3793	13,03%
219,663	242,7555	10,51%
213,338	239,0618	12,06%
252,096	250,0465	0,81%

Tabla 4: Tiempos y estimaciones y su desviación

Finalmente, se puede analizar la desviación que se obtiene en esta muestra. Se observa que la desviación oscila hasta un 16,79 %, lo cual es una desviación relativamente alta, por ello esta fórmula solo debería de utilizarse a modo de estimación y no para utilizarla en casos que se requiera una alta precisión.

7.4. Estudio BWA con nodos saturados

En este estudio se busca analizar y comprender como la saturación y el trabajo afectan al tiempo de ejecución de la herramienta BWA, ya que como se ha analizado previamente además del tamaño de las muestras y de los genomas, existen otros factores que hacen que la ejecución tarde más o menos tiempo.

Por ello se llenarán los nodos que se utilicen al máximo de sus posibilidades, en este caso se dispone de nodos con 12 Cores, por lo que se lanzan ejecuciones que llenen el nodo. Por ejemplo, se ponen 12 trabajos que utilicen un core cada uno para llenar todo el nodo. En este caso se lanzan trabajos suficientes para saturar los nodos con las siguientes configuraciones:

- 1 Core por trabajo (12 Trabajos por nodo)
- 2 Cores por trabajo (6 Trabajos por nodo)
- 3 Cores por trabajo (4 Trabajos por nodo)
- 4 Cores por trabajo (3 Trabajos por nodo)
- 6 Cores por trabajo (2 Trabajos por nodo)
- 12 Cores por trabajo (1 Trabajo por nodo)

Una vez se ejecutan las configuraciones que se consideran interesantes para realizar este estudio, nos disponemos a analizar los diferentes comportamientos de estas ejecuciones, a continuación podemos observar un gráfico (Figura 12) con los resultados obtenidos para cada configuración.

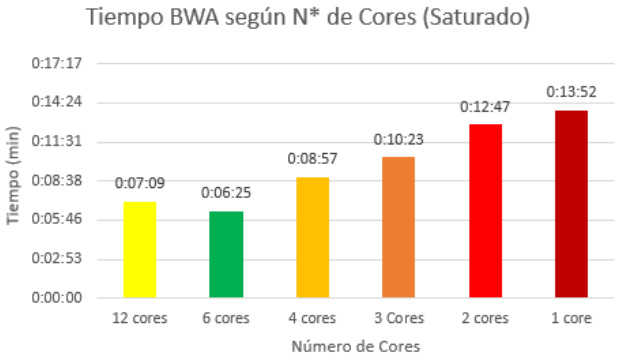

Fig. 12: Diferentes configuraciones y sus tiempos en nodos saturados

Analizando el gráfico anterior podemos llegar a la conclusión de que las mejores configuraciones para ejecutar un proceso completo en el que podamos utilizar el máximo de nodos disponibles debe de ser la configuración de 6 cores.

7.5. Estudio ejecuciones completas

Como último estudio se realizan una serie de ejecuciones completas de todo el proceso de BWA del programa, es decir, comparar las 22 muestras con sus 110 respectivos genomas. Para realizar este estudio se utilizan las mejores configuraciones obtenidas en el apartado anterior, 6 y 12 cores, además de una ejecución con 1 core para conocer el tiempo de referencia. También se realizan ejecuciones con las diferentes heurísticas desarrolladas, tanto con la RANDOM como con la heurística Big-Big, para así poder analizar las mejoras que aportan estas heurísticas.

En el siguiente gráfico se observan los diferentes tiempos de ejecución según la configuracion utilizada, además de la heurística.

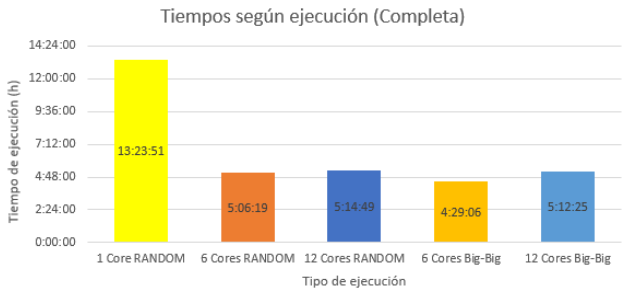

Fig. 13: Tiempos según el tipo de ejecución (Ejecución completa)

El gráfico nos confirma los resultados obtenidos en el experimento anterior, es decir, corrobora que la mejor configuración para ejecutar el programa será la de 6 Cores por trabajo. Se observa que la diferencia entre 6 y 12 cores no es demasiado grande para el caso de la heurística RANDOM. Para el caso de la heurística Big-Big la diferencia entre 6 y 12 cores sí es más sinificativa como veremos en la siguiente tabla (Tabla 5). El tiempo de ejecución con 12 cores es de 5:12:25 y con 6 cores es de 4:29:06, por lo que esta diferencia sí es más significativa. La mejora respecto al uso de la configuración de 1 Core por trabajo es notable, ya que se ha conseguido un SpeedUp cercano a 3x.

Versión	Tiempo(h)	SpeedUP vs 1 Core	SpeedUP RANDOM vs Big-Big	%
1 Core RANDOM	13:23:51	1	x	x
6 Cores RANDOM	5:06:19	2,6242	x	x
12 Cores RANDOM	5:14:49	2,5534	x	x
6 Cores Big-Big	4:29:06	2,9872	1,1383	13,83%
12 Cores Big-Big	5:12:25	2,5730	1,0077	0,77%

Tabla 5: Comparación versiones final y su SpeedUp

En esta tabla se pueden ver los resultados comentados previamente, en los que cabe resaltar la mejora sustancial que se obtiene utilizando la heurística Big-Big respecto a la heurística RANDOM, que en el caso de 12 Cores es una mejora casi inexistente, pero que, sin embargo, con la configuración de 6 Cores se obtiene un SpeedUp cercano al 14 %. Como se ha comentado durante el estudio, existe un cierto ruido en las ejecuciones que se han realizado, ya que al realizarse en un clúster existe la posibilidad de que otros usuarios estén utilizando ciertos nodos al mismo tiempo que se ejecuta BWA, por ello no se puede determinar un SpeedUp concreto para la heurística Big-Big, simplemente se estima una mejora de entre un 0-14 % respecto a la heurística RANDOM.

8 Conclusiones

Tras la realización de todos los experimentos anteriores y la visión global de toda la investigación se pueden sacar algunas conclusiones sobre cuáles son los cuellos de botella del Workflow, que tipos de configuraciones funcionan mejor en unas condiciones u otras, como afectan las heurísticas al tiempo de ejecución del programa...

Con todos los datos al alcance somos capaces de aplicar la solución óptima para la necesidad específica de BWA, podemos ejecutar con un nodo o con varios, utilizando una CPU o varias... Todo ello lo podemos resolver y adaptar de forma óptima gracias a los estudios realizados previamente.

Se ha visto como inicialmente al ejecutar BWA utilizando un nodo en modo exclusive se obtienen unos datos que pueden hacer pensar que las mejores configuraciones pueden ser las que menos cores utilicen, sin embargo, una vez se va avanzando en la investigación y los experimentos descubrimos que las mejores configuraciones son las que utilizan más cores. Todo esto puede ser debido a otros factores como conflictos en la memoria, recalentamiento de las máquinas del clúster, etc.

Finalmente, se elabora la heurística Big-Big con la que se optimiza el tiempo de ejecución de BWA entre un 0-14 %, y con ello, deducimos que para ejecutar lo más rápido posible BWA en el clúster al que se tiene acceso se utilizan 6 cores por trabajo y además se aplica la heurística Big-Big y con ello ganar cerca de un SpeedUp de x3.

9 AGRADECIMIENTOS

En primer lugar, dar las gracias a la autora del programa, Lidia Garrido-Sanz por permitirme trabajar sobre su proyecto.
Agradecer también a mi tutor Miquel Angel Senar su ayuda a la hora de plantear y gestionar las diferentes fases y complicaciones del proyecto.
Agradecer a mi amigo Allen Pedersen su ayuda con la revisión y corrección de los textos en Inglés.
Por último agradecer a la Universidad Autónoma de Barcelona los servicios y prestaciones que se me han ofrecido para la realización de este proyecto.

REFERENCIAS

[1] Código genético [Online] Available: https://www.genome.gov/es/genetics-glossary/Codigo-genetico

[2] Milton H. Saier, (2019) .Understanding the Genetic Code, Journal of Bacteriology, Volume 201, Issue 15. Available: https://journals.asm.org/doi/10.1128/JB.00091-19

[3] Christopher P. Austin, M.D. ADN (Ácido Desoxirribonucleico) [Online] Avalaible: https://www.genome.gov/es/genetics-glossary/ADN-acido-Desoxirribonucleico

[4] Garrido-Sanz, L., Senar, M. À., Piñol, J. (2020). Estimation of the relative abundance of species in artificial mixtures of insects using low-coverage shotgun metagenomics. Metabarcoding and Metagenomics, 4, 1– 18. Avalaible: https://mbmg.pensoft.net/article/48281/

[5] Manual Reference Pages - BWA. [Online] Available: http://bio-bwa.sourceforge.net/bwa.shtml

[6] Herramienta de trabajo adicional. [Online] Available: https://trello.com/

[7] Sh/Bash Cheatsheet. [Online] Available: https://devhints.io/bash

[8] Slurm Cheatsheet. [Online] Available: https://slurm.schedmd.com/pdfs/summary.pdf

[9] ¿Qué es Kanban? [Online] Available: https://kanbanize.com/es/recursos-de-kanban/primeros-pasos/que-es-kanban

[10] ¿Qué es un WorkFlow? [Online] Available: https://www.integradoc.com/que-es-un-workflow/

[11] ¿Qué es un cluster?. [Online] Available: https://www.idepa.es/innovacion/clusteres/que-es-un-cluster

• E-mail de contacto: eloybardaji@gmail.com
• Mención realizada: Enginyeria de Computadors
• Trabajo tutorado por: Miquel Angel Senar
• Curso 2021/22