Implementación de Plataforma de Servicios con una Agradable Experiencia de Usuario en la Interfaz web

Diego Hernández G. and others
Universidad Tecnológica Metropolitana, Chile
arratia391@gmail.com

Resumen

En función de la amplia necesidad de hoy en día por resolver problemas que requieren gran poder de cómputo se proporcionó mediante una plataforma web de servicios ambientada en un clúster computacional homogéneo y heterogéneo, solución a tres grandes problemas de esta envergadura estudiando a la vez su comportamiento en el clúster computacional. La plataforma se desarrolló en tres módulos y cada módulo proporciono una solución a un problema. Módulo 1: Búsqueda de patrones mediante algoritmo de código de la biblia. Módulo 2: Simulación de patrones de comportamiento mediante algoritmo de percolación. Módulo 3: Alineamiento de cadenas mediante procesamiento de secuencias de ADN en formato FASTA. Las soluciones proporcionadas por la plataforma se entregaron al usuario mediante correo electrónico en un tiempo directamente proporcional al tamaño del problema. Además durante la implementación de la solución se obtuvo que el comportamiento del clúster que procesaba la solución presentó una tendencia común a disminuir el rendimiento cuando se utilizaba su configuración heterogénea.

Palabras clave: Clúster, Clúster web, código torah, percolación, alineamiento de ADN.

Abstract

Following the widespread need of nowadays to solve problems that require high computing power it was provides a web service platform set in a cluster homogeneous and heterogeneous, solving three major problems of this scale and studying their behavior in the computational cluster. The platform was developed in three modules and each one of them provided a solution to a specific problem. Module 1: Search algorithm patterns using the bible code. Module 3: Simulation of behavioral patterns by percolation algorithm. Module 3: Alignment od strings by processing DNA sequences in FASTA format.The solutions provided by the platform were delivered to the user via email in a time proportional to the size of the problem. Also, during the implementation of the solution it was obtained the behavior of the cluster that processed the solution had a common tendency to degrade the performance when its heterogeneous configuration was used.

Key words: Cluster, Cluster web, Code torah, Percolation , DNA alignment.

I. Introducción

Hoy en día, los grandes volúmenes de datos que buscan ser convertidos en información han aumentado de manera considerable y exponencial, y para encontrarles soluciones a éstos grandes volúmenes de procesamiento se suelen implementar técnicas informáticas para poder obtener una respuesta rápida y como paradigma de solución se plantea el uso de procesamiento paralelo en un clúster computacional. En el contexto de la construcción y aplicación de plataformas WEB de alta eficiencia, en el presente documento se presentarán los aspectos teóricos para el desarrollo de un sistema integral el cual, a través de tres distintos módulos brindará solución a tres grandes problemas de cómputo. La plataforma web desarrollada se caracteriza por ser intuitiva, amigable y responsiva, y tiene por objetivo tomar los requerimientos del usuario mediante un formato específico por cada módulo; cadenas de ADN en formato FASTA para el alineamiento de ellas, encontrar una palabra con el código Torah o una simulación basada en percolación. Una vez reconocido el modulo a utilizar, se envían los procesos a la plataforma clúster. Esta plataforma está compuesta por nodos con distintas características (Clúster heterogéneo) permitiendo la escalabilidad gracias a su configuración en MPI. Finalmente, las respuestas serán enviadas por formato PDF al correo electrónico del usuario.

II. Materiales y Métodos

II.1. Plataforma

Se montó una plataforma web en un clúster que contenía 16 Nodos y por consecuencia 113 procesadores disponibles para el cómputo. Dentro de los 16 Nodos se caracterizaban tres tipos de nodo. Tipo A (Maestro): Procesador i3-3220cpu 3.30GHz x4 con una memoria de 3.8Gb, Gráfico Intel Integrado y S.O. Ubuntu 14.04 x64. Tipo B (Esclavos): Procesador i7-3730cpu 3.40GHz x8 con una memoria 7.7Gb, Gráfico Intel integrada y S.O. Ubuntu 14.04 x64. Tipo C (Esclavos): Procesador Pentium 4 Cpu 3.00GHz x2, Memoria 2.0Gb, Gráfico intel 945G x86/MMX/SSE2 y S.O. Ubuntu 14.04 x86, para darle un carácter heterogéneo. Se puede apreciar el diagrama de flujo del funcionamiento general que se propuso para la implementación de la web de servicios. Se debe asumir que cada problema especificado fue utilizado por un usuario específico y que tenía un conocimiento y manejo teórico de la problemática. Se puede intuir que el flujo de información es dependiente al conocimiento y el uso que le quiera asignar el usuario por ende se accede al home de la página, y a partir de ella se puede dirigir a la solución de la problemática a evaluar brindando ciertos parámetros de entrada inherentes al problema a resolver obteniendo una respuesta que será enviada mediante e-mail al remitente que realice la visita a la página.

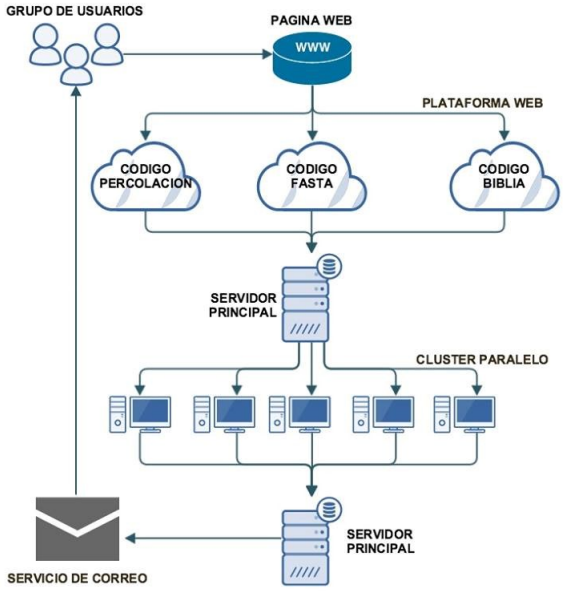

Figura 1: Modelo propuesto del funcionamiento del sistema. Fuente: Elaboración propia.

III. Módulo 1: Percolación

Aproximemos el bosque o la población por una red cuadrada en la que cada celda está ocupada por un árbol o persona con probabilidad p, o vacía con probabilidad p -1. Esta distribución de objetos y huecos será el estado inicial. Evidentemente, si p es pequeño es de esperar que los objetos estén en grupos bien distribuidos y poco conectados entre sí. En cambio, si p es próximo a 1, casi todas las celdas estarán ocupadas por un objeto, el bosque se extenderá de lado a lado de la red y su alta conectividad hará que acabe por propagar casi completamente. Se sabe que existe un valor crítico de p, (pc), tal que si p >pc y en el límite de red infinita, el fuego se extiende por toda la red (lo que no quiere decir que se propague por toda la región), mientras que si p <pc sólo afecta a una porción de él. Se define pues pc como aquel valor de p a partir del cual, en una red infinita, aparecen agregados de percolación, es decir, agregados que se extienden a toda la red. Primero se definirá el tamaño de la matriz, luego la cantidad de repeticiones de la prueba para obtener un pc promedio y así compararlo con las condiciones iniciales (pc mayor o menor que p). Se utilizara la percolación para encontrar cuáles serán los arboles afectados y hasta qué punto llegara dicho incendio forestal, los mismo sucederá con la propagación de enfermedades.

El algoritmo toma por parte del usuario los siguientes parámetros:

- Tipo de árbol.
- Tipo de suelo.
- Cantidad de metros.
- Distribución porcentual del espacio.

Con esta información, entra al algoritmo de percolación junto a los siguientes parámetros:

- Porcentaje de combustión del árbol.
- Porcentaje de combustión del suelo.

Figura 2: Proceso secuencial para la ejecución del algoritmo de Percolación. Fuente: Elaboración propia.

IV. Módulo 2: Código de la Biblia

El usuario entregó como parámetro de entrada al algoritmo el nombre del fichero con el texto a buscar (en formato PDF), el cual fue transformado en una lista de cadenas de texto con el contenido de cada una de sus páginas. Otro parámetro ingresado por el usuario es la palabra o cadena de palabras (frase) que se busca dentro del documento, cuyas ocurrencias se fueron registrando a lo largo de la ejecución. El último valor ingresado fue el del máximo salto que se ha probado sobre la extensión de la cadena ingresada. Al comenzar el algoritmo se dividió (dependiendo del número de procesadores esclavos disponibles) un intervalo de saltos el cual es asignado a cada nodo para que cada uno individualmente los procesaran de la misma forma que lo hace la búsqueda de coincidencias de forma secuencial, esto respondiendo a una arquitectura donde una misma instrucción se ejecuta sobre una gran cantidad de datos. Una vez terminada la lectura del texto completo, cada nodo devuelve al maestro las coincidencias encontradas en el texto si es que existiesen, teniendo en cuenta los saltos con el cual se localizaron y donde inicia y termina cada 'keyword'. El nodo maestro se encarga entonces de reunir todas las respuestas y llevarlas a un formato en serie (en un archivo JSON) para que estas sean integradas a la plataforma web.

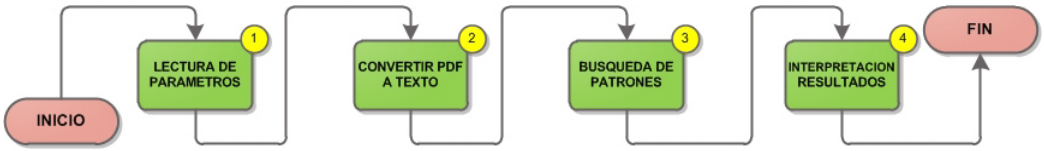

Figura 3: Proceso secuencial para la ejecución del algoritmo del código de la biblia. Fuente: Elaboración propia.

V. Módulo 3: FASTA

Establecida una base de datos de referencia, un archivo de entrada correspondiente a una cadena en formato .fasta, la matriz de sustitución requerida por el usuario y la penalización se procedió a realizar la comparación según el funcionamiento del algoritmo de alineación de secuencias de ADN mediante matrices de sustitución, utilizando como paradigma la programación dinámica y además una matriz de NxN, debido a las N letras que se comparan para cadenas de ADN, principalmente: A de adenina, G de guanina, C de citosina y T de Timina. Al terminar se comparó la cadena, el algoritmo asignó un puntaje (score) a la alineación (similitud de cadenas), el cual indicó que a mayor valor, mayor similitud existe entre la

comparación y la secuencia original. Cuando se encontró una puntuación más alta que la que ya se encontraba en ese momento, se guardaron las cadenas alineadas junto con la descripción de ambas hasta que finaliza la ejecución del algoritmo o se encuentre otra alineación con mayor puntaje. Terminando el proceso, la salida del programa entregó tanto las cadenas alineadas, el valor de similitud entregado por el algoritmo (Score) y el porcentaje de similitud en relación a la cadena que ha proporcionado el usuario para el análisis, todo esto, para las "n" secuencias con mayor similitud que estime conveniente el usuario para sus propósitos.

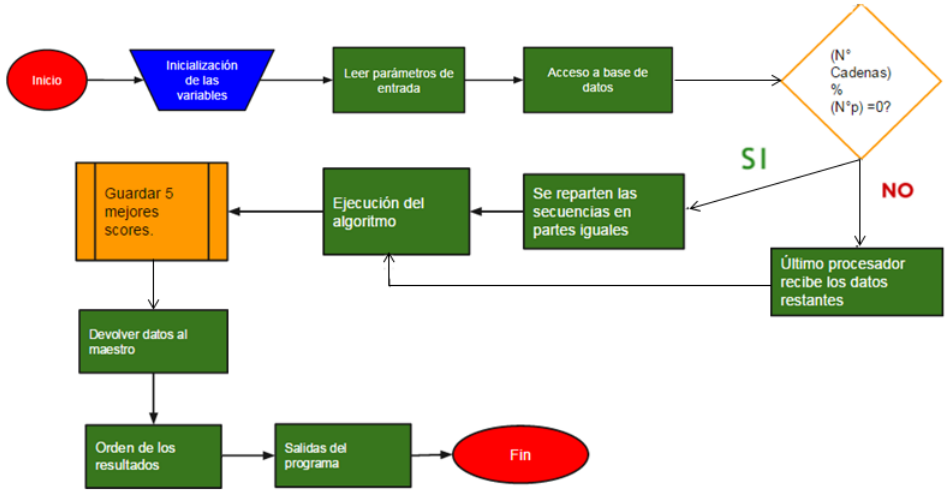

Figura 4: Proceso secuencial para la ejecución de FASTA. Fuente: Elaboración propia.

VI. Resultados

VI.1. Módulo 1: Percolación

VI.1.1. Enfermedades

Cantidad de nodos	Tiempo Ejecucion (Segundos)	Speed Up	Eficiencia	Costo
4	76,1611	3,8152	0,9538	19,0403
8	54,1648	5,3646	0,6706	6,7706
12	37,9948	7,6477	0,6373	3,1662
16	28,5167	10,1895	0,6368	1,7823
20	22,6772	12,8134	0,6407	1,1339
24	19,1442	15,1781	0,6324	0,7977
28	26,6243	10,9138	0,3898	0,9509
32	14,4210	20,1492	0,6297	0,4507
36	27,4237	10,5956	0,2943	0,7618
40	27,0315	10,7494	0,2687	0,6758
44	32,1247	9,0451	0,2056	0,7301
48	9,6373	30,1506	0,6281	0,2008
52	21,7816	13,3402	0,2565	0,4189
56	43,1588	6,7326	0,1202	0,7707
60	24,3106	11,9525	0,1992	0,4052
64	7,2139	40,2796	0,6294	0,1127
68	39,6916	7,3207	0,1077	0,5837
72	23,8112	12,2032	0,1695	0,3307
76	7,9127	36,7223	0,4832	0,1041
80	54,7275	5,3094	0,0664	0,6841
84	41,5834	6,9877	0,0832	0,4950
88	29,2113	9,9473	0,1130	0,3319
92	16,6444	17,4577	0,1898	0,1809
96	4,8269	60,1984	0,6271	0,0503
100	67,0743	4,3321	0,0433	0,6707
104	59,6688	4,8697	0,0468	0,5737
108	50,4095	5,7642	0,0534	0,4668
112	40,9301	7,0992	0,0634	0,3654
116	31,5916	9,1977	0,0793	0,2723
120	22,1915	13,0938	0,1091	0,1849
124	12,7681	22,7576	0,1835	0,1030

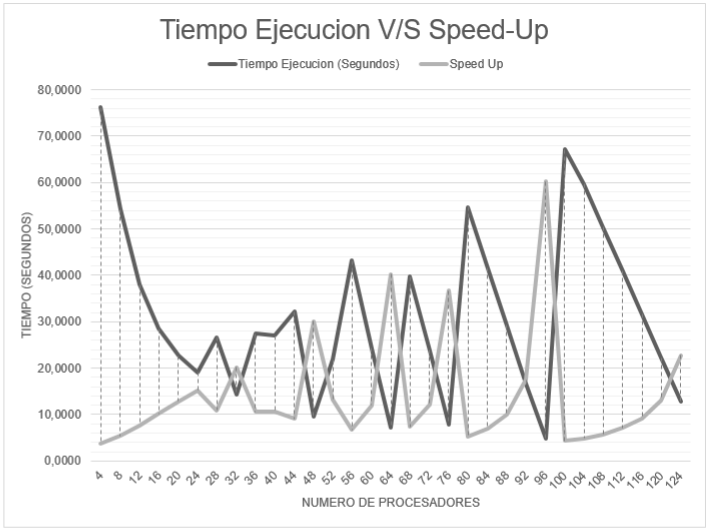

Figura 5: Tabla de mejor eficiencia del modulo percolación e inherente Gráfica de mejor SpeedUp vs mejor Tiempo de ejecución para el modulo, para la experiencia Enfermedades. Fuente: Elaboración propia.

VI.1.2. Incendios

Cantidad de nodos	Tiempo Ejecucion (Segundos)	Speed Up	Eficiencia	Costo
4	75,9536	3,8256	0,9564	18,9884
8	55,4358	5,2416	0,6552	6,9295
12	38,0428	7,6380	0,6365	3,1702
16	28,5394	10,1814	0,6363	1,7837
20	22,7206	12,7889	0,6394	1,1360
24	19,1668	15,1601	0,6317	0,7986
28	26,5165	10,9581	0,3914	0,9470
32	14,4309	20,1353	0,6292	0,4510
36	27,2841	10,6499	0,2958	0,7579
40	26,7829	10,8491	0,2712	0,6696
44	32,0321	9,0713	0,2062	0,7280
48	9,6229	30,1959	0,6291	0,2005
52	21,8677	13,2877	0,2555	0,4205
56	43,2171	6,7235	0,1201	0,7717
60	24,3316	11,9421	0,1990	0,4055
64	7,2413	40,1268	0,6270	0,1131
68	39,4552	7,3646	0,1083	0,5802
72	23,6992	12,2608	0,1703	0,3292
76	7,9243	36,6682	0,4825	0,1043
80	54,4283	5,3386	0,0667	0,6804
84	41,6916	6,9696	0,0830	0,4963
88	29,1747	9,9597	0,1132	0,3315
92	16,5460	17,5615	0,1909	0,1798
96	4,8278	60,1870	0,6269	0,0503
100	68,5586	4,2383	0,0424	0,6856
104	59,7290	4,8648	0,0468	0,5743
108	50,3338	5,7729	0,0535	0,4661
112	41,0614	7,0765	0,0632	0,3666
116	31,5745	9,2027	0,0793	0,2722
120	22,2396	13,0655	0,1089	0,1853
124	12,7649	22,7634	0,1836	0,1029

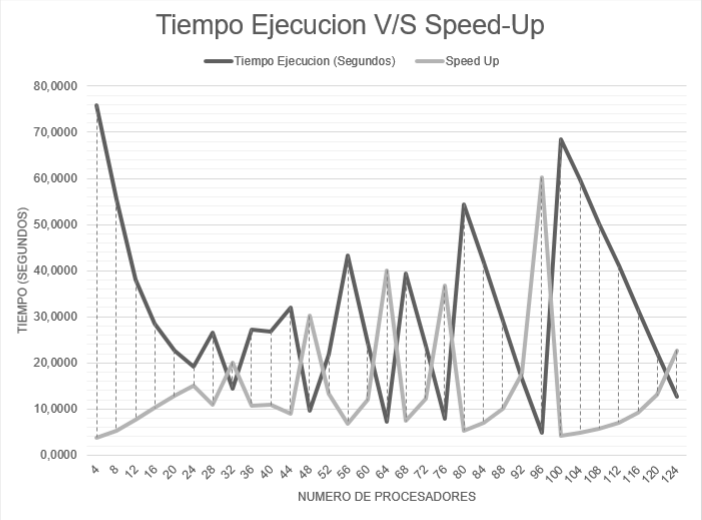

Figura 6: Tabla de mejor eficiencia del modulo Percolación e inherente Gráfica de mejor SpeedUp vs mejor Tiempo de ejecución para el modulo, para la experiencia Incendios. Fuente: Elaboración propia.

VI.2. Módulo 2: Código de la Biblia

VI.2.1. Explicito

Cantidad de nodos	Tiempo Ejecucion (Segundos)	Speed Up	Eficiencia	Costo
10	188.110	53.018	5.301	19
20	93.860	106.256	5.312	5
30	88.490	112.704	3.756	3
40	64.450	154.744	3.868	2
50	59.630	167.252	3.345	1
60	54.250	183.838	3.063	0.904
70	48.950	203.743	2.910	0.699
80	47.340	210.672	2.633	0.592
90	43.480	229.375	2.548	0.483
100	37.370	266.878	2.668	0.374
110	36.670	271.973	2.472	0.333
112	34.540	288.745	2.578	0.308
113	35.260	282.849	2.503	0.312

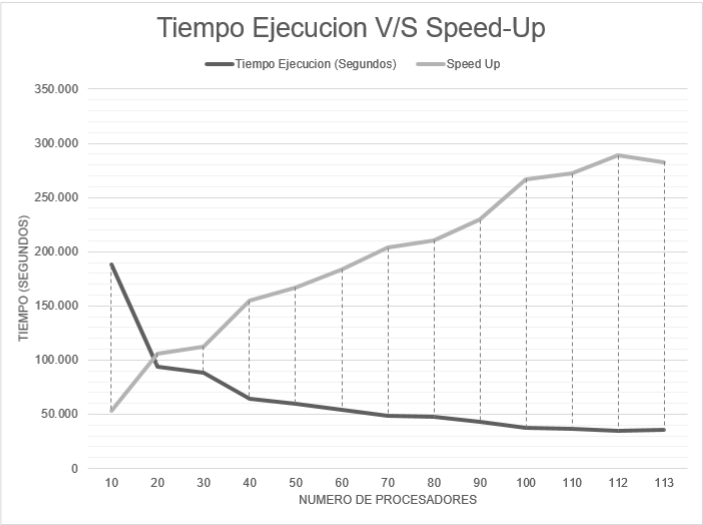

Figura 7: Tabla de mejor eficiencia del modulo Código de la Biblia e inherente Gráfica de mejor SpeedUp vs mejor Tiempo de ejecución para el modulo, para la experiencia Explicito. Fuente: Elaboración propia.

VI.3. Módulo 3: FASTA

VI.3.1. ADN

Eficiencia	Costo
1881,1000	0,5301
1877,2000	0,5312
2654,7000	0,3756
2578,0000	0,3868
2981,5000	0,3345
3255,0000	0,3063
3426,5000	0,2910
3787,2000	0,2633
3913,2000	0,2548
3737,0000	0,2668
4033,7000	0,2472
3868,4800	0,2578
3984,3800	0,2503

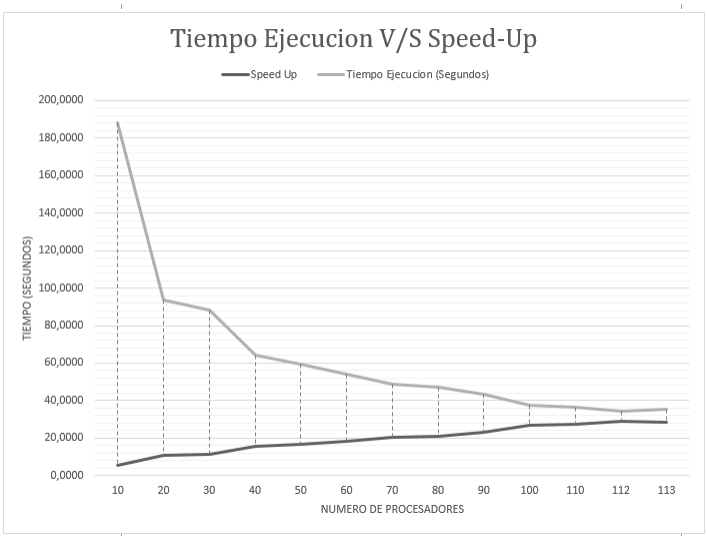

Figura 8: Tabla de mejor eficiencia del modulo Fasta e inherente Gráfica de mejor SpeedUp vs mejor Tiempo de ejecución para el modulo, para la experiencia ADN. Fuente: Elaboración propia.

VI.3.2. Proteínas

Cantidad de nodos	Tiempo Ejecucion (Segundos)	Speed Up	Eficiencia	Costo
5	8,3066	3,0246	1,6613	0,6049
10	8,3030	3,0259	0,8303	0,3025
15	6,7611	3,7160	0,4507	0,2477
20	5,9411	4,2288	0,2971	0,2114
25	5,3867	4,6641	0,2155	0,1865
30	5,1661	4,8633	0,1722	0,1621
35	5,0959	4,9302	0,1456	0,1408
40	4,5927	5,4704	0,1148	0,1367
50	4,3267	5,8068	0,0865	0,1161
60	4,4891	5,5967	0,0748	0,0932
70	4,3237	5,8108	0,0618	0,0830
80	4,1658	6,0310	0,0521	0,0753
90	4,0480	6,2065	0,0450	0,0689
100	4,0358	6,2253	0,0404	0,0622
110	4,0087	6,2674	0,0364	0,0569
113	3,8783	6,4781	0,0343	0,0573

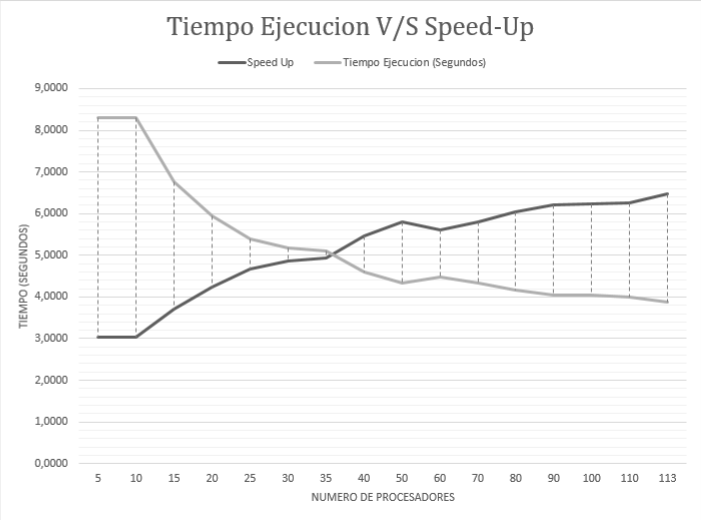

Figura 9: Tabla de mejor eficiencia del modulo Fasta e inherente Gráfica de mejor SpeedUp vs mejor Tiempo de ejecución para el modulo, para la experiencia Proteínas. Fuente: Elaboración propia.

VII. Discusión

En el desarrollo del Proyecto Plataforma Web de Servicios de Computación Paralela se logró implementar, mediante un clúster computacional de ambiente de proceso paralelo y secuencial, una plataforma web de interfaz intuitiva y amigable que permite al usuario final ingresar una problemática de las áreas de simulación de propagación de incendios forestales, cadenas de ADN en formato FASTA para el alineamiento de cadenas de ADN y búsqueda de patrones mediante Torah.

VII.1. Plataforma

- La configuración del entorno de desarrollo del proyecto en los aspectos de paralelización (MPI), servidor de dominio (DNS) y servidor de correo(MailGun) fue exitosa, siendo el acceso al dominio http://00-ironman.clustermarvel.utem/webParalela/ la cual es la interfaz gráfica del entorno de paralelización configurado y entrega la respuesta mediante correo.
- Finalmente se encontró una tendencia general: Un clúster heterogéneo en la práctica ralentiza las funciones principales de un clúster.

VII.2. Módulo 1: Percolación

- Al observar el gráfico de tiempos de ejecución en paralelo con matriz tamaño 100 se apreció que el menor tiempo de ejecución se encuentra al utilizar 96 procesadores.
- El gráfico de SpeedUp de matriz tamaño 100 muestra que el procesador óptimo es 96, esta información explica por qué al utilizar el nodo 96 se encontró el menor tiempo de ejecución.
- La eficiencia del gráfico de matriz tamaño 100 se encontró al utilizar 4 procesadores, pero también es posible encontrar un alto número de eficiencia en los procesadores 48,64 y 96.

- El costo del gráfico de matriz tamaño 100 se redujo bastante en el procesador 96, no se logró apreciar bien en el gráfico, pero al revisar la tabla, se observa que es el procesador con menor costo. Al analizar los tiempos de ejecución del gráfico con matriz tamaño 1000 de ejecución, se observó que nuevamente al usar 96 procesadores, se obtuvo el menor tiempo de ejecución.

- Con unas curvas más pronunciadas, se puede observar claramente que al usar 96 procesadores es donde se obtuvo mejor SpeedUp.

- Al comparar los resultados con las métricas tomadas al usar una matriz de tamaño 100, se observó que la eficiencia se encontró en los nodos 32, 48, 64,76 y 96.

VII.3. Módulo 2: Código de la Biblia

- A mayor cantidad de caracteres que tenga una palabra menos probable es encontrar coincidencias del mismo.

- A mayor cantidad de palabras en una frase también disminuye la probabilidad de éxito en la búsqueda de tales patrones, pero no tanto como la diferencia de la cantidad de caracteres.

- El éxito de encontrar coincidencias está condicionado por los caracteres con los cuales está constituida las palabras. Esto es debido a que en todo el texto hay diferentes proporciones de caracteres distintos. Por ese motivo si una palabra está formada por caracteres que tienen escasa participación en el texto, este tendrá pocas probabilidades de ser localizado debido a que ni siquiera puede existir.

- Para las pruebas bajo las condiciones entregadas de largo de palabra y las frecuencias de caracteres se observó que disminuyen las cantidades de coincidencias.

VII.4. Módulo 3: FASTA

- El problema es altamente paralelizable, debido a que los tiempos paralelos en comparación a los secuenciales son considerablemente menores.

- Un clúster heterogéneo para este problema (al menos uno de las características que se usó) resultó ser perjudicial para las aspiraciones de lograr reducir los tiempos de ejecución, debido a que las características de esos procesadores eran muy inferiores.

- La optimización al código paralelo resultó de forma exitosa, debido a que redujo los tiempos en algunos casos en 15 segundos.

- La diferencia de tamaño entre las cadenas a alinear definitivamente influye en los tiempos de ejecución. A mayor diferencia, más tiempo toma alinear las secuencias.

- Otro punto que influye en los tiempos de ejecución son la cantidad de alineaciones que realiza el algoritmo para determinar el óptimo, a mayor cantidad de alineaciones, más tiempo de ejecución toma el programa.

- A nivel de programación, Python resultó ser una herramienta bastante útil para desarrollar los códigos, gracias a su simplicidad y a la cantidad de librerías disponibles para su utilización.

- Añadir más procesadores al óptimo resultó traer consecuencias, ya que los tiempos de ejecución aumentaron. Esto se debe principalmente a los tiempos de comunicación que hay entre los equipos que fueron aumentando a medida que más procesadores se conectaban a la ejecución.

- La cantidad óptima de procesadores para la alineación de proteínas son 111 y para el alineamiento del ADN son 95. El algoritmo es altamente paralelizable.

VIII. Bibliografía

VIII.1. Fasta

Rehm, B.H.A. (2001). Bioinformatictools-for DNA/proteinsequenceanalysis, functional assignment of genes and proteinclassification. Appl. Microbiol. Biotechnol. 57(5-6):579-592.

Pearson, W. R. (2014). BLAST and FASTA similaritysearchingformultiplesequencealignment, BioinformaticsforBeginners, paginas 133-155 Retrievedfromscopus.com

Centro de Biología molecular y Biotecnología de la Universidad Tecnológica de Pereira.

La Jornada. 20 de octubre de 2007. Watson y Crick, los padres del AND [en línea]. México, D.F. Disponible desde Internet en: [citado en 7 de octubre de 2010].

Colombia Médica [en línea]. Universidad del Valle: Cali, 2008 – [citado en 7 de octubre de 2010]. Vol. 39, No. 001. (enero-marzo 2008). Disponible desde Internet en: ISSN 1657-9534. GenBank® es la base de datos de secuencias genéticas del Institutos Nacionales de Salud (en inglés National Institutes of Health, NIH), una colección anotada de todas las secuencias de ADN a disposición del público.

Attwood, T. K. (2002). «6». Introducción a la bioinformática. Prentice Hall. ISBN 84-205-3551-6.

VIII.2. Código de la Biblia

Witztum D., Rips E. and Rosenberg Y. . (1994). Equidistant Letter Sequences in the Book of Genesis. Starical Science, 9, pp.429-438.

Thomas D.. (1997). Hidden Messages and The Bible Code. 1997, de Skeptical Inquirer

Mckay B., Bar-Natan D., Bar-Hillel M., and Kalai G.. (1999). SOLVING THE BIBLE CODE PUZZLE. Statical Science, 14, pp. 50-173.

Drosnin M.. (1997). El código secreto de la Biblia. Estados Unidos: Planeta.

Drosnin M.. (2005). El nuevo código secreto de la Biblia. Estados Unidos: Planeta.

Drosnin M.. (2011). El código secreto de la Biblia III. Estados Unidos: Planeta.

VIII.3. Percolación

Návar, J.J. (2010). Modelación del contenido de agua de los suelos y su relación con los incendios forestales en la Sierra Madre Occidental de Durango, México. Mayo 14, 2015, de scielo Sitio.

Salat, R.S.. (2005). El fenómeno de la percolación. mayo 15, 2015, de Escuela Superior de Física y Matemáticas del IP.

Canals, M and Canals, A. (2010, mayo). Percolación de la epidemia de influenza AH1N1 en el mundo: Utilidad de los modelos predictivos basados en conectividad espacial. Revista Medica de Chile, 138, 573-580.

W. Lebrecht. (2012, septiembre 28). Umbral de percolacion en las redes de Kagomé y Dice . Revista Mexicana de Física, 59, 1-5.

W. Lebrecht. (2010, junio 28). Umbrales de percolacion exactos en redes duales. REVISTA MEXICANA DE FíSICA, 56, 190-196.

Cuestas, E. and Vilaró, M. and Serra, P.. (2011). Predictibilidad de la propagación espacial y temporal de la epidemia de influenza A H1N1 en la Argentina por el método de percolación. Revista argentina de microbiología, 43, 186-190.