Implementación de la tesis: "Multiple Sequence Alignment using Particle Swarm Optimization"

1st Vicente Machaca Arceda
DAISI
Universidad Nacional de San Agustín
Arequipa, Perú
vmachacaa@unsa.edu.pe

***Resumen*—Este reporte presenta la implementación y análisis de la tesis: "*Multiple Sequence Alignment using Particle Swarm Optimization*". Dicha tesis propone el uso de *Particle Swarn Optimization* (PSO) para solucionar el problema de *Multiple Sequence Alignment* (MSA). El autor propone la posición de cada *gap*, como vector para la representación de las partículas, luego define un *crossover* para simular el movimiento de una partícula hacia la partícula lider. Además, tambien se propone una mutación para diversificar la solución. En las pruebas realizadas, se comprobo que PSO obtiene muy buenos resultados e incluso similares a CLUSTALW.**

***Index Terms*—PSO, MSA, alineamiento de secuencias.**

I. Introducción

El area de Bioinformática ha tenido un auge en las ultimas decadas. Por ejemplo, el proyecto: *The Human Genome Project* (HGP) que inicio en 1990 y fue completado el 2003, donde participarón varios paises como EEUU, Reino Unido, Japón, Francia, Alemania, España y China [1]; el proyecto tenia como objetivo secuenciar todo el genoma humano, el cual resulto ser una secuencia de aproximadamente 3.2 billones de pares de bases [2]. En la actualidad exixten diversas areas de investigación como: la prediccón de estructura de proteinas, predicción de la función de una proteina a partir de estructuras de redes de proteinas, descubrimiento de medicamentos, predicción de enfermedades a partir del genoma, análisis de virus, etc. Es tan grande este campo de estudio que incluso se ha dividido en otras areas como *metagenomics*, *proteomics*, *chemical informatics*, etc.

I-A. Problema

Uno de los tantos problemas que existen en bioinformática, es el alineamiento multiple de secuencias. Se puede definir el alineamiento de secuencias como un método que permite determinar el grado de similitud entre dos o más secuencias [3], además el método puede insertar *gaps* dentro de las secuencias de consulta, con el objetivo de lograr la mayor cantidad de bases alineadas. Por ejemplo, en la Figura 1, se muestra el resultado luego de alinear varias secuencias de aminoacidos. Como podemos ver, se ha isertado *gaps* (-) para así maximizar la cantidad de aminoacidos que coinciden en la misma posición.

El alineamiento de secuencias puede dividirse en dos grupos: *pair-wise sequence alignment* (PSA) y *multiple sequence alignment* (MSA) [3]. La diferencia radica en que el primero alinea solo dos secuencias y el segundo puede alinear dos a mas secuencias. Además, el problema de MSA es considerado un problema NP completo [4]. Debido a esto es que se han planteado heuristicas que logran obtener una solución local; el algoritmo mas utilizado es CLUSTAL, fue propuesto por [5], este algoritmo ha sido mejorado con CLUSTALV [6], CLUSTALW [7] y CLUSTALX [8]. Otro algoritmo importante es MUSCLE [9].

El problema de los algoritmos mencionados anteriormente, es que a pesar de ser heurísticas, el tiempo de procesamiento es muy alto. Cada segundo los datos genómicos crecen exponencialmente [2] y los algoritmos utilizados para buscar información en estas bases de datos, son basados en alineamiento. Entonces, mientras mas crecen los datos, mas lentos se vuelven estos algoritmos [10].

I-B. Objetivo

El autor propone en su tesis aplicar *Particle Swarm Optimization* (PSO) como alternativa a CLUSTAL para solucionar el problema de MSA.

II. Conceptos básicos

Secuencia de ADN y aminoacidos: Si hablamos de ADN, nos referimos al conjunto de bases nitrogenadas Adenina (A), Citosina (C), Guanina (G) y Timina (T). Pero si nos referimos a aminoacidos, estos son 20 y son representados con letras desde la A hasta la V [3].

Secuenciamiento de ADN: Proceso por el cúal se determina el orden de cada base (A, T, C y G) presente en el genoma [2].

Figura 1. Ejemplo de *Multiple Sequence Alignment* (MSA). El método ha insertado *gaps* (-) en las secuecnias necesarias para maximizar la cantidad de letras (aminoacidos) que concidan en la misma posición.

Alineamiento de secuencias: Proceso por el cúal se comparan dos o más secuencias de ADN para determinar su grado de similitud [3]. Existen dos tipos: *pair-wise sequence alignment*, donde se alinean dos secuancias y *Multiple Sequence Alignment* (MSA), cuando alineamos mas de dos secuencias. Los algoritmos mas utilizados son: BLAST y CLUSTAL.

Gap: Caracter "-" insertado durante el alineamiento de secuencias para alinear cada base y maximizar el grado de similitud.

III. Trabajos relacionados

El algoritmo mas utilizado para alinear dos secuencias de ADN o aminoacidos es BLAST [11]. Este método a tenido varias mejoras como TurboBLAST [12], ScalaBLAST [13], Usearch [14] y ScalaBLAST 2.0 [15]. Pero si hablamos de alinear mas de dos secuencias, el algoritmo mas utilizado es CLUSTAL [5], con mejoras en CLUSTALV [6], CLUSTALW [7] y CLUSTALX [8].

El problema de MSA, tambien ha sido tratado de solucionarse con métodos *alignment-free*. Uno de los primeros métodos fue propuesto por Xu [16] y Zhang [17]. Pero, recientemente, se ha propuesto PSO en dos niveles [18], [19]. Tambien, se ha presentado una versión llamada *chaotic-PSO* [20] y otras versiones híbridas [21], [22].

IV. Metodología aplicada

En esta sección describimos la metodología planteada por el autor para solucionar el problema de MSA utilizando PSO.

IV-A. Representación de las partículas

Un de los primeros pasos para solucionar un problema utilizando PSO, es la representación de cada particula. El autor propone utilizar la posición de cada *gap* como un vector. Por ejemplo en al Figura 2, tenemos la representación del la partícula *leader* y una partícula cualquiera (ambas representan una posible solución al problema). Ahora en la Figura 3, tenemos otra forma de representar dichas partículas, en este caso solo estamos considerando la posición de cada *gap* insertado. La opción mas facil de controlar es la correspondiente a la Figura 3.

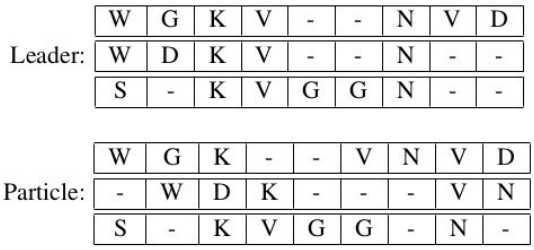

Figura 2. Ejemplo de la partícula *leader* y una partícula cualquiera. Ambas representan una posible solución a un problema de alineamiento de 3 secuencias de aminoacidos.

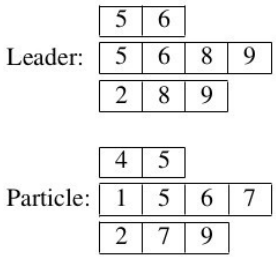

Figura 3. Ejemplo de la partícula *leader* y una partícula cualquiera. En este caso solo estamos registrando la posición de cada *gap* insertado.

IV-B. Movimiento de las partículas

Según el algoritmo de PSO, cada partícula debe acercarse al *leader* en cada iteración. Este acercamiento será implementado haciendo un *crossover*. El *crossover* permitirá que la nueva partícula (partícula en movimiento) tenga información de ambas partículas (*leader* y la partícula en movimiento).

Para aplicar un *crossover* entre dos partículas, debemos calcular primero la distancia entre ellas aplicando la formula de la Ecuación 1. Luego debemos calcular el punto de cruce de ambas partículas, para esto aplicamos la formula de la Ecuación 2, en este caso $length$ representa la longitud de las secuencias.

$$distance = \frac{matchingGaps}{totalGaps} \quad (1)$$

$$crossPoint = rand(1, distance * length) \quad (2)$$

Por ejemplo, dada las partículas de la Figura 3 y suponiendo que hemos obtenido un *crossPoint = 5*, utilizando la Ecuación 2. La partícula en movimiento se desplazaría y su nueva representación sería el resultado de aplicar un *crossover*. En la Figura 4 mostramos el resultado de aplicar el *crossover*. Para lograr esto, por cada secuencia del leader insertamos sus gaps en la nueva partícula que son menores o iguales al $crossPoint$, luego completamos insertando los *gaps* de la partícula en movimiento que tengan *gaps* mayores al $crossPoint$.

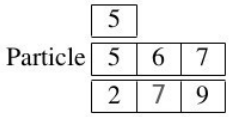

Figura 4. Resultado del movimiento de una partícula luego de aplicar *crossover*.

IV-C. Función objetivo

La función objetivo propuesta, es la misma utilizada para medir el desempeño de los algoritmos basados en alineamiento como CLUSTAL para el problema de MSA. Por ejemplo en la Figura 5, se presenta un ejemplo del cálculo del *score* del alineamiento de dos secuencias. En este ejemplo se utilizo un *match score* : +1, *mismatch score*: 0, *openning gap*: -1 y *gap penalty*: -1. Para el caso de MSA, se procesa este *score* por cada posible conbinación de dos secuencias. Para los experimentos se usaron estos mismos valores.

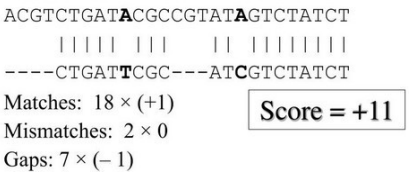

Figura 5. Ejemplo del cálculo del *score* del alineamiento de dos secuencias de ADN. En este caso se utilizo: *match score* : +1, *mismatch score*: 0, *openning gap*: -1 y *gap penalty*: -1

IV-D. Mutaciones

Para diversificar al algoritmo de PSO, Se propuso hacer mutaciones a las partículas. El proceso de mutación consistia en escoger de manera aleatoria una partícula, luego dentro de dicha partícula se escogia una secuencia y se insertaba un *gap* en una posición aleatoria. Luego, para que todas las secuencias dentro de la partícula tengan la misma longitud, se insertaba un *gap* al principio o al final según un valor generado aleatoriamente.

V. Experimentos

En esta sección detallaremos las bases de datos utilizadas y los parametros del algoritmo PSO para poder replicar los resultados.

V-A. Bases de datos

El autor propone un conjunto de 7 bases de datos. S1, S2, S3, S4, S5, S6 y S7 que el construyo a partir de un conjunto de secuencias de ADN. En la Tabla I, presentamos el *ascension code* de cada secuencia utilizada. Ademas, el autor propuso un conjunto pequeño de secuencias para hacer pruebas rapidas, a este conjunto lo llamo S8. En nuestro caso, al ser una tesis un poco antigua, algunas secuencias ya no estaban disponibles en NCBI, y solo logramos obtener las secuencias de S6, S7 y S8.

Cuadro I
BASES DE DATOS UTILIZADOS POR [10]. S1, S2, S3, S4, S5, S6 Y S7 ES EL NOMBRE QUE EL AUTOR DEFINIO PARA CADA CONJUNTO DE SECUENCIAS. LA SEGUNDA COLUMNA REPRESENTA EL *ascension code* DE CADA SECUENCIA.

Dataset	**Secuencias**
S1	HCV2L1A10 HCV2L3A5 HCV2L3C1 HCV2L3C8 HCV2L3D4 HCV2L3E6 HCV2L3A7 HCV2L3A9 HCV2L3B2 HCV2L3B1
S2	HS06674 HS06675 HS06676 HS06677 HS06679
S3	TPAHISIN TNIHISIN TNHISIN TMIHISIN TMHISIN THHISIN TFHISIN TEHISIN TCUHISIN TCHISIN TBHISIN TAUHISIN TAHISIN TTHISIN TSHISIN TRHISIN TPYHISIN TPIHISIN TPHISIN TCAHISIN TLHISIN
S4	HI1U16764 HI1U16766 HI1U16768 HI1U16776 HI1U16778 HI1U16770 HI1U16774 HI1U16772
S5	HI1U16765 HI1U16767 HI1U16769 HI1U16771 HI1U16773 HI1U16775 HI1U16777 HI1U16779
S6	PP59651 PP59652 PP59653 PP59654 PP59655 PP59656
S7	AB023287 AB023286 AB023285 AB023284 AB023283 AB023279 AB023278 AB023276

Como se menciono anterioriormente, se utilizaron las bases de datos S6, S7 y S8. En la Tabla II, detallamos datos adicionales de las bases de datos utilizadas. Por ejemplo la base de datos mas sencilla es S8, en la Figura 6, mostramos este conjunto de secuencias.

Cuadro II
BASES DE DATOS UTILIZADOS EN LOS EXPERIMENTOS.

Dataset	**Longitud mínima**	**Longitud máxima**	**Cantidad de secuencias**
S6	8	17801	153
S7	457	457	8
S8	7	10	5

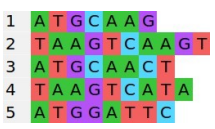

Figura 6. Conjunto de secuencias de la base de datos S8. Se utilizo Mega para colorear cada base.

V-B. Parametros

Los parametros son descritos en la Tabla III. Para reducir el tiempo de procesamiento reducimos el valor de algunos parametros como la cantidad de iteraciones, el autor proponia en este caso 1000 iteraciones. Luego la cantidad de veces que se replico cada experimento era de 30 y nosotros solo lo replicamos 10 veces.

Cuadro III
PARAMETROS UTILIZADOS EN LOS EXPERIMENTOS. ALGUNOS PARAMETROS FUERON DISTINTOS A LOS PLANTEADOS POR [10] (ITERACIONES), CON EL OBJETIVO DE REDUCIR EL TIEMPO DE PROCESAMIENTO.

Parametro	**Valor**
Iteraciones	30
Cantidad de partículas	25
Probabilidad de mutación	0.2
Gaps permitidos	30 %
Cantidad de veces que se replico el experimento	10

VI. RESULTADOS

En la Tabla IV, se muestra el *score* (resultado de la función objetivo) obtenido en cada base de datos. Como se puede apreciar, el *score* depende de la complejidad del problema, por ejemplo en el conjunto de secuencias S8 obtenemos un score bajo, pero para las secuencias de S6 (la mas compleja), obtenemos un *score* mucho mas grande. La idea es que mientras mayor sea el *score*, es mejor.

Cuadro IV
RESULTADOS OBTENIDOS EN CADA BASE DE DATOS. SE HICIERON PRUEBAS INCLUYENDO Y EXCLUYENDO LA MUTACIÓN.

Dataset	***PSO* con mutación**	***PSO* sin mutación**	**CLUSTALW**
S6	12678	10012	18045
S7	11105	9054	12564
S8	32	28	49

Para visualizar mejor el resultado de los alineamientos se utilizo la herramieta Mega, esta herramienta colorea cada base según su posición para ver mejor su alineamiento. En la Figura 7, mostramos el resultado luego de alinear el conjunto de secuencias de S8 utilizando PSO y CLUSTALW. Como podemos ver, existen muchas soluciones para cada problema, e incluso podemos llegar a tener un diferentes soluciones con un mismo *score*. En la Figura 8, mostramos un parte del alineamiento de las secuencias de S7, recordemos que estas secuencias tienen una longitud de 457 bases.

VII. DISCUSIÓN

De la Tabla IV, comprobamos que el desempeño de PSO es bueno y casi llega a una solución similar a CLUSTAW. Además, uno de los problemas de PSO es que llega muy pronto a optimos locales, debido a eso, la versión de PSO con mutación tiene mejores resultados. Otro problema de PSO y otros algoritmos similares, es la

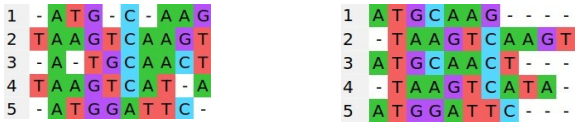

Figura 7. **Izquierda**: Resultado luego de alinear las secuencias de la base de datos S8 utilizando PSO.
Derecha: Resultado luego de alinear las secuencias de la base de datos S8 utilizando CLUSTALW.

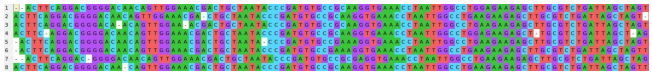

Figura 8. Resultado luego de alinear las secuencias de la base de datos S7 utilizando PSO. Solo se esta mostrando un extracto, porque las secuencias tienen una longitud de 457 bases.

forma de definir el desplazamiento de cada partícula, en este trabajo se planteo utilizar un *crossover*, pero existen trabajos recientes propuestos por [23], [24] y [25] donde obtienen mejores resultados modificando la función de movimiento de cada partícula y combinando PSO con las cadenas de Markov.

Una prueba visual del buen desempeño de PSO es presentado en las Figuras 7 y 8. Como vemos en las imágenes, se aprecia que se ha obtenido un alineamiento aceptable. Cada base (letra) es representada con un color distinto y esta forma de visualización ayuda bastante para determinar el alineamiento entre dos o mas secuencias.

VIII. CONCLUSIONES

La tesis propone el uso de PSO para solucionar el problema de MSA. El autor evalua los resultados en 8 bases de datos, que representan un conjunto de secuencias de diferentes longitudes. El autor obtuvo dichas secuencias de NCBI, pero al ser una tesis del 2009, algunas secuencias ya no estaban presentes.

El autor tambien propone aplicar una mutación a las partículas para evitar caer en optimos locales, esta mutación se basaba en insertar un *gap* dentro de una secuencia de una partícula. Esta diversificación (mutación) mejoro el desempeño de PSO.

Para comprobar el desempeño de PSO, se comparó sus resultados con CLUSTALW. El desempeño de PSO fue bueno y logro un *score* similar a CLUSTALW. En la actualidad existen varias modificaciones en la forma de como se define una partícula y la función de movimientno de esta con muy buenos resultados.

REFERENCIAS

[1] NIH, "The human genome project."
[2] J. M. Archibald, *Genomics: A Very Short Introduction*, vol. 559. Oxford University Press, 2018.
[3] J. Xiong, *Essential bioinformatics*. Cambridge University Press, 2006.

[4] L. Wang and T. Jiang, "On the complexity of multiple sequence alignment," *Journal of computational biology*, vol. 1, no. 4, pp. 337–348, 1994.

[5] D. G. Higgins and P. M. Sharp, "Clustal: a package for performing multiple sequence alignment on a microcomputer," *Gene*, vol. 73, no. 1, pp. 237–244, 1988.

[6] D. G. Higgins, A. J. Bleasby, and R. Fuchs, "Clustal v: improved software for multiple sequence alignment," *Bioinformatics*, vol. 8, no. 2, pp. 189–191, 1992.

[7] J. D. Thompson, D. G. Higgins, and T. J. Gibson, "Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic acids research*, vol. 22, no. 22, pp. 4673–4680, 1994.

[8] F. Jeanmougin, J. D. Thompson, M. Gouy, D. G. Higgins, and T. J. Gibson, "Multiple sequence alignment with clustal x," *Trends in biochemical sciences*, vol. 23, no. 10, pp. 403–405, 1998.

[9] R. C. Edgar, "Muscle: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[10] F. B. R. Zablocki *et al.*, *Multiple sequence alignment using Particle swarm optimization*. PhD thesis, University of Pretoria, 2009.

[11] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped blast and psi-blast: a new generation of protein database search programs," *Nucleic acids research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[12] R. D. Bjornson, A. Sherman, S. B. Weston, N. Willard, and J. Wing, "Turboblast (r): A parallel implementation of blast built on the turbohub," in *ipdps*, p. 0183, IEEE, 2002.

[13] C. Oehmen and J. Nieplocha, "Scalablast: a scalable implementation of blast for high-performance data-intensive bioinformatics analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 8, pp. 740–749, 2006.

[14] R. C. Edgar, "Search and clustering orders of magnitude faster than blast," *Bioinformatics*, vol. 26, no. 19, pp. 2460–2461, 2010.

[15] C. S. Oehmen and D. J. Baxter, "Scalablast 2.0: rapid and robust blast calculations on multiprocessor systems," *Bioinformatics*, vol. 29, no. 6, pp. 797–798, 2013.

[16] F. Xu and Y. Chen, "A method for multiple sequence alignment based on particle swarm optimization," in *International Conference on Intelligent Computing*, pp. 965–973, Springer, 2009.

[17] Z. P. H. Hongwei, "Pso for multiple sequences alignment [j]," *Computer Engineering and Applications*, vol. 18, 2005.

[18] S. Lalwani, R. Kumar, and N. Gupta, "Efficient two-level swarm intelligence approach for multiple sequence alignment," *Computing and Informatics*, vol. 35, no. 4, pp. 963–985, 2017.

[19] S. Lalwani, H. Sharma, M. K. Mohan, and K. Deep, "An efficient bi-level discrete pso variant for multiple sequence alignment," in *Harmony Search and Nature Inspired Optimization Algorithms*, pp. 797–807, Springer, 2019.

[20] X.-j. Lei, J.-j. Sun, and Q.-z. Ma, "Multiple sequence alignment based on chaotic pso," in *International Symposium on Intelligence Computation and Applications*, pp. 351–360, Springer, 2009.

[21] T. K. Rasmussen and T. Krink, "Improved hidden markov model training for multiple sequence alignment by a particle swarm optimization—evolutionary algorithm hybrid," *Biosystems*, vol. 72, no. 1-2, pp. 5–17, 2003.

[22] L. Chaabane, "A hybrid solver for protein multiple sequence alignment problem," *Journal of bioinformatics and computational biology*, vol. 16, no. 04, p. 1850015, 2018.

[23] Q. Zhan, N. Wang, S. Jin, R. Tan, Q. Jiang, and Y. Wang, "Probpfp: a multiple sequence alignment algorithm combining hidden markov model optimized by particle swarm optimization with partition function," *BMC bioinformatics*, vol. 20, no. 18, pp. 1–10, 2019.

[24] S. Lalwani and H. Sharma, "Multi-objective three level parallel pso algorithm for structural alignment of complex rna sequences," *Evolutionary Intelligence*, pp. 1–9, 2019.

[25] N. Moustafa, M. Elhosseini, T. H. Taha, and M. Salem, "Fragmented protein sequence alignment using two-layer particle swarm optimization (ftlpso)," *Journal of King Saud University-Science*, vol. 29, no. 2, pp. 191–205, 2017.