Revista de Matemática: Teoría y Aplicaciones 2011 **18**(1) : 121–136
cimpa – ucr issn: 1409-2433

HEURÍSTICA PARA SOLUCIONAR EL PROBLEMA DE ALINEAMIENTO MÚLTIPLE DE SECUENCIAS

HEURISTIC FOR SOLVING THE MULTIPLE ALIGNMENT SEQUENCE PROBLEM

Roman Anselmo Mora–Gutiérrez*
Javier Ramírez–Rodríguez† Mayra Elizondo–Cortés‡

Received: 18 Feb 2010; Revised: 11 Nov 2010; Accepted: 19 Nov 2010

---

*Posgrado en Ingeniería de Sistemas, Facultad de Ingeniería, Universidad Nacional Autónoma de México, Avenida Insurgentes Sur sin número, Col. Copilco Universidad, Del. Coyoacán, 04360, México D. F. E-Mail: ing.romanmora@gmail.com

†Departamento de Sistemas, Universidad Autónoma Metropolitana - Azcapotzalco. Avenida San Pablo 180, Colonia Reynosa Tamaulipas, 02200 México D. F. E-Mail: jararo@correo.azc.uam.mx

‡Posgrado en Ingeniería de Sistemas, Facultad de Ingeniería, Universidad Nacional Autónoma de México, Misma dirección que/*same address as* Mora-Gutiérrez, E-Mail: mayra.elizondo@hotmail.com

Resumen

En el presente trabajo se desarrolló un nuevo algoritmo para resolver el problema de alineamiento múltiple de secuencias (AMS) que es un híbrido basado en las metaheurísticas de búsqueda de armonía (HS) y recocido simulado (RS). Éste fue validado con la metodología de Julie Thompson. Es un algoritmo básico y los resultados obtenidos durante esta etapa son alentadores.

Palabras clave: alineamiento múltiple de secuencias, búsqueda de la armonía, metaheurística híbrida, recocido simulado.

Abstract

In this paper we developed a new algorithm for solving the problem of multiple sequence alignment (AMS), which is a hybrid metaheuristic based on harmony search and simulated annealing. The hybrid was validated with the methodology of Julie Thompson. This is a basic algorithm and and results obtained during this stage are encouraging.

Keywords: multiple sequence alignment, harmony search, hybrid metaheuristic, simulated annealing.

Mathematics Subject Classification: 90C59.

1 Introducción

Una secuencia es una sucesión ordenada de objetos (símbolos o letras que proviene de un conjunto finito llamado alfabeto $\mathcal{A} = \{a_1, a_2, a_3, \ldots, a_n\}$) y se construye por la permutación de objetos. En la literatura frecuentemente se utilizan como sinónimos de secuencia, los términos sentencia o cadena (Chan et al., 1992).

El alineamiento de secuencias es un procedimiento para la comparación de dos o más secuencias, e involucra generar un matriz M, donde los renglones de la matriz tienen la misma longitud y el $i-$ésimo reglón se obtiene a partir de insertar un número finito de celdas vacías a la secuencia correspondiente s_i (Manthey, 2003)y ninguna columna conste completamente de espacios vacíos. Para el caso que un elemento $u \in s_a$ sea alineado con un espacio vacío $(-)$ insertado en la cadena s_b, esta acción se interpretará como la eliminación del elemento u de la sentencia s_a, o bien, como la inserción de u en la cadena s_b (Gusfieeld, 1993). El problema de alinear dos cadenas se maneja en la literatura como el problema de alineamiento pareado de secuencias.

Con base en lo anterior, se define al alineamiento múltiple de secuencias (AMS) de un conjunto de $N - secuencia \mid N = \{s_1, s_2, \ldots, s_x\} \wedge x \geq 3$

como la generación de una matriz M_{N*k} donde $\max(|s_1|, |s_2|, \ldots, |s_k|) \leq k \leq |s_1| + |s_2| + \ldots + |s_k|$, de tal manera que ninguna columna de M conste completamente de caracteres vacíos y el resultado de eliminar dichos espacios vacíos del $i-$ésimo renglón sea la secuencia $s_i \in N$ correspondiente (Thompson, et al., 1999). El AMS es un problema de optimización ya que se busca maximizar el número de coincidencias entre los elementos contenidos en $j-$ésima columna de M, y es un problema del tipo NP-duro (Elias, 2003).

El AMS se utiliza en diversas áreas, sin embargo, es en la biología, donde se le implementa con mayor frecuencia, debido a que es una herramienta eficaz para el analisis y la comparación de secuencias biológicas (proteicas y ácidos nucleidos), que sirve para encontrar las relaciones estructurales entre dichas cadenas, así como para determinar la historia evolutiva entre las sentencias (Dong, et al., 2008), además predecir estructuras secundarias y terciarias de las secuencias (Wang, et al., 2004), y pronosticar la función de dichas sentencias los organismos, ya que por lo general estructuras similares tienen funciones afines, un ejemplo, de lo anterior es la insulina en los mamíferos.

En la literatura no se ha generalizado el uso de una función única para cuantificar la eficacia del alineamiento en $N - secuencias$ (Ma, et al., 2007 & Gusfieeld, 1993), las funciones más utilizadas son: A) funciones de puntaje (penalización por espacios vacíos y suma por pares de las diferencias) B) alineamiento por templete y C) alineamiento por secuencia media. Como criterio de decisión al comparar dos o más alineamientos factibles de un mismo conjunto de $N - secuencia$, generalmenre se emplean las métricas de la similitud o la distancia total del conjunto de cadenas ya alineadas.

Dada la importancia del AMS, se han desarrollado una gran cantidad de métodos para resolverle, los cuales, se clasifican de acuerdo con tipo de búsqueda que utilizan en exhaustivos y aproximados.

Los procedimientos exhaustivos se basan en programación dinámica y en técnicas de ramificación y acotamiento; estos garantizan encontrar el alineamiento óptimo de un conjunto de $N - sentencias$, sin embargo, tienen una complejidad del orden $O(2^n \prod_{i=1,\ldots,n} |s_i|)$ (Chan, et al., 1992).

La importancia de los procedimientos exhaustivos radica en que se utilizan como subrutinas por los procedimientos aproximados para solucionar el AMS, algunos ejemplos son: A) Needleman-Wunsch (1969) B) Gotoh (1982) C) Murata et al. (1985), en el cual se compara simultáneamente tres secuencias. D) Gotoh-Altschul (1986) cuya aportación es la representación en una red del alineamiento pareado de secuencias, E) Carrillo-Lipman (1988), donde se introduce el esquema de acotación del par sabio (pairwise en inglés) para limitar el volumen del espacio dimensional de búsqueda,

por lo que se puede alinear simultáneamente 10 secuencias y F) Smith-Waterman (1981) es un método de alineamiento local.

Los métodos de búsqueda aproximada a su vez se dividen en procedimientos progresivos e iterativos.

Los procedimientos progresivos, también se conocen como jerárquicos o de árbol, se caracterizan por: alinear las secuencias más parecidas entre sí, para después ir incorporando el resto de las cadenas, a razón del grado de similitud; constan de tres fases básicas (Rech & Pilatti, 2004). El alineamiento resultante depende de la clasificación inicial de las secuencias en grupos relacionados. Algunos ejemplos son: A) Sankoff et al. (1982), B) Barton y Sterberg (1987), C) algoritmo Clustal, D) algoritmo FATA (1985), E) algoritmo BLAST (1990) y F) algoritmo T-COFFEE (1998), este método utiliza la función COFFEE (función para la evaluación objetivo del alineamiento) y dos librerías primarias generadas por los métodos Clustal y FASTA.

Por su parte los procedimientos iterativos son algoritmos que utilizan un alineamiento inicial, el cual, se va perfeccionando a través de una serie de ciclos (iteraciones) hasta el punto donde no es posible realizar mejora alguna, son de carácter estocástico algunos ejemplos son: el modelo oculto de Markov (HMM), recocido simulado, computación evolutiva, etc.

En este trabajo se desarrolló un nuevo procedimiento iterativo para resolver el AMS a través de un algoritmo híbrido entre las metaheurísticas búsqueda de armonía (HS por sus siglas en íngles)[1] y recocido simulado (RS). El algoritmo desarrollado utiliza el vigor híbrido resultado de la combinación de las metaheurísticas padres donde la estrategia HS se utiliza como procedimiento rector, mientras que el RS se implementa como estrategia para escapar de óptimos locales y a su vez mantener un alto grado de diversificación en la memoria armónica.

2 Algoritmo híbrido de HS y RS para solucionar el AMS

La elección de los algoritmos HS y RS como padres del híbrido se basó en los siguientes criterios: A)ambos procedimientos son metaheurísticas sencillas y coherentes que se basan en principios simples y claros; B) son procedimientos generales y adaptables ya que en la literatura se han reportado un gran número de aplicaciones en diferentes problemas para contextos muy heterogéneos; C) son procedimientos efectivos y eficientes, con

[1]Metaheurística desarrollada por Zong Woo Geem en 2001, la cual imita el proceso de perfeccionamiento del estado de armónico en la producción musical.

base en las aplicaciones reportadas en la literatura D) el RS ha sido utilizada para solucionar el AMS con excelentes resultados; E) la combinación de métodos heurísticos permiten tomar mejores decisiones y proporcionan una mayor flexibilidad a la hora de definir diferentes estrategias de elección y F) el algoritmo HS es una técnica reciente, por lo que no existe reportado una aplicación para resolver el AMS en la literatura, por ende la adaptación de este procedimiento para solucionar el AMS implica un aporte al conocimiento.

Para el diseño del algoritmo híbrido se utilizo la formulación del problema AMS considerando como función objetivo a la función COFFEE

$$\max z = \frac{\sum_{a=1}^{x-1}\sum_{a=a+1}^{x} w_{s_a,s_b} * similitud(A_{\hat{s_a},\hat{s_b}})}{\sum_{a=1}^{x-1}\sum_{a=a+1}^{x} w_{s_a,s_b} * l} \tag{1}$$

donde: $A_{\hat{s_a},\hat{s_b}}$:proyección de los alineamientos pareados de las secuencias; $similitud(A_{\hat{s_a},\hat{s_b}})$: valor del alineamiento entre $\hat{s_a}, \hat{s_b}$; w_{s_a,s_b}: ponderación del alineamiento entre$\hat{s_a}, \hat{s_b}$; l:número de columnas de la matriz de alineamiento múltiple.

El uso de la función COFFEE se basa en que ésta ha sido utilizada para el diseño de otras técnicas metaheurísticas destinadas a resolver el AMS, como son: A) T-COFFEE (Notredame, 2000), B) algoritmo iterativo y adaptativo para el mejoramiento de AMS (Wang, et al., 2004) y C) algoritmo genético con colonia de hormigas para el alineamiento múltiple de secuencia (Zne-Jung, et al., 2008).

El algoritmo híbrido desarrollado es un procedimiento iterativo, que realiza una búsqueda estocástica en el conjunto factible. A continuación se muestra el pseudocódigo del algoritmo híbrido HS-RS para solucionar el AMS .

Algoritmo 1. Procedimiento híbrido HS-RS para solucionar el AMS.

Input: Conjunto de secuencias ($N = \{s_1, s_2, \ldots, s_x\}$).
Output: Alineamiento múltiple del conjunto de secuencias.

1. Inicializar proceso de optimización.
   (a) Ingreso del conjunto de secuencias.
   (b) Asignación de los parámetros de entrada.
   (c) Construcción de la matriz W de ponderación
2. Generar la memoria armónica (HM) inicial.
3. Mientras no se satisfaga el criterio de paro hacer lo siguiente.

(a) Generar una nueva armonía.

(b) If valor del nuevo alineamiento ($V_{Auxiliar}$) mejor que el valor del peor alineamiento en memoria (V_{peor}).

- Actualizar memoria armónica.

(c) Else $If(rand \leq e^{\frac{V_{Auxiliar}-V_{peor}}{T}})$.

- Actualizar memoria armónica.

End If

End Mientras.

La complejidad del algoritmo anterior es del orden $O(n^n * l_M^2)$. Para mayor información consúltese Mora-Gutiérrez 2009. A continuación se describen cada una de las fases del algoritmo híbrido $HS-RS$ para el AMS.

2.1 Inicializar el proceso de optimización

En esta fase se ingresa la información sobre el conjunto de $N-secuencias$ a alinear, se asignan los valores a de los parámetros necesarios para la ejecución del algoritmo y se construye la matriz de ponderación entre los alineamientos pareados de las secuencias.

El conjunto de $N-secuencias$ debe cumplir las siguientes restricciones: A) dicho conjunto debe conformarse de al menos 3 secuencias ($N = \{s_1, s_2, \ldots, s_x\} \wedge x \geq 3$) y B)todas las cadenas del conjunto N provienen de la misma familia de sentencias $s_i \in \mathcal{F}\ \forall i = 1, 2, \ldots, x$.

En la tabla 1 se muestra el valor y/o la forma de estimación de los parámetros de entrada para el algoritmo.

Una vez ingresado el conjunto de secuencias y los parámetros de entrada al algoritmo, se procede a calcular la matriz de ponderación W. En esta matriz el elemento $w_{a,b}$ representa el porcentaje de similitud entre la secuencia s_a y la cadena s_b, dicho valor oscila entre 1 y 0, si el valor tiende a cero representa poca similitud entre la secuencias, mientras que si es más cercano simboliza una mayor similitud entre secuencias. La matriz W, es simétrica, con elementos en la diagonal igual a cero. Para la construcción de dicha matriz de ponderación se utiliza la siguiente subrutina.

Parámetros	**Valor**	
Tamaño de la memoria de armonía ($HMS \geq 1$)	5	
Criterio de paro ($MaxImp$)	800	
Consideración del ritmo en la memoria de armonía $HMCR = HMCRI + \left((HMCRF - HMCRI) * \frac{v}{MaxImp}\right)$ (2) donde: $HMCR$: Adaptación del ritmo en la memoria de armonía. v: Número de iteración. $MaxImp$: Máximo número de iteraciones. $HMCRI$: Consideración inicial del ritmo en la memoria de armonía $0 \leq HMCRI \leq HMCRF$ $HMCRF$: Consideración final del ritmo en la memoria de armonía $HMCRI \leq HMCRF \leq 1$	$HMCRI$ $HMCRF$	0.7 0.9
Parámetro de ajuste del ritmo (PAR) $PAR = PARI + \left((PARF - PARI) * \frac{v}{MaxImp}\right)$ (3) Ingram & Zhang, 2009 donde: PAR: Parámetro de ajuste del ritmo. v: Número de iteración. $MaxImp$: Máximo número de iteraciones. $PARI$: Parámetro de ajuste del ritmo inicial $0 \leq PARI \leq PARF$ $PARF$: Parámetro de ajuste del ritmo final $PARI \leq PARF \leq 1$	$PARI$ $PARF$	0 0.5
Temperatura inicial $0 \leq t_0$	10	
Grado de disminución de temperatura (α)	0.7	

Tabla 1: Parámetros de entrada para el algoritmo.

Ancho de banda para el ajuste del ritmo (b) $b = \eta(l^u_{M_i} - l^l_{M_i})(4)$ donde: η: Radio máximo de búsqueda de las vecindades. En la literatura se recomienda 0.1 sin embargo durante pruebas preliminares se obtuvo un mejor desempeño con 0.3 b: Ancho de banda. $l^u_{M_i}$: Límite superior del número de columnas en la matriz de alineamiento $l^l_{M_i}$: Límite unferior del número de columnas en la matriz de alineamiento	$0.3(caracteres - l_{max})$

Tabla 2: (Continuación de Tabla 1).

Algoritmo 2. Subrutina 1 Construcción de la matriz de ponderación W. .

Input: Un conjunto de secuencias ($N = \{s_1, s_2, \ldots, s_x\}$).

Output: Matriz de ponderación W.

1. For $a = 1 : 1 : x$
   (a) For $b = 1 : 1 : x$
      i. If $(a < b)$
         - $contador = 0$
         - $M_{s_a,s_b} \leftarrow$ Matriz de alineamiento de las secuencias s_a, s_b generado por el algoritmo de Needleman - Wunsch variante de Vinuesa.
         - $len \leftarrow$Número de columnas de la matriz M_{s_a,s_b}
         - For $j = 1 : 1 : len$
           – If $(M_{1,j} = M_{2,j})$
             ∗ $contador = contador + 1$
           – Else
             ∗ $contador = contador + 0$
             End If
           End For
         - $W_{a,b} \leftarrow \frac{contador}{len}$
      ii. Else If $(a > b)$
         - $W_{a,b} \leftarrow W_{b,a}$

iii. Else

- $W_{a,b} \leftarrow 0$

End If

End For

End For

2.2 Construcción de la memoria armónica inicial (HM)

En esta fase se genera aleatoriamente un conjunto de soluciones factibles, el cual será contenido en la matriz de memoria armónica (HM). La construcción de la HM se realiza mediante la subrutina 2.

Algoritmo 3. Subrutina 2 Construcción de la memoria armónica inicial.
Input: Un conjunto de secuencias ($N = \{s_1, s_2, \ldots, s_x\}$).
Output: Memoria armónica.

1. $q = 0$
2. For $P = 1 : 1 : HMS$
   (a) $l_M^P = int(l_{max} + ((caracteres - l_{max}) * rand))$
   (b) Construir una matriz auxiliar P
   (c) Actualizar el valor de l_M^P
   (d) $HM_{P,1} \leftarrow l_M^P$
   (e) $HM_{P,2} \leftarrow V(M^v)$

   End For
3. $\sum_{p=1}^{H} MSHM_{P,2}$
4. For $P = 1 : 1 : HMS$
   (a) $HM_{P,3} = \frac{q - HM_{P,2}}{q}$

   End For

En el algoritmo anterior se muestra que cada hilera de la HM constituye un alineamiento factible y se le asocia una matriz auxiliar P que contiene al conjunto de secuencia ya alineadas (submatriz M^P), el valor de alineamiento y de aptitud de cada secuencia (submatriz G^P), como se muestra en la siguiente relación

$$A = \left(\begin{array}{cccc|cc} M^P_{\widehat{s_1},1} & M^P_{\widehat{s_1},2} & \dots & M^P_{\widehat{s_1},l_{\overline{s_1}}} & COFFEE_{\widehat{s_1}} & fitness_{\widehat{s_1}} \\ M^P_{\widehat{s_2},1} & M^P_{\widehat{s_2},2} & \dots & M^P_{\widehat{s_2},l_{\overline{s_2}}} & COFFEE_{\widehat{s_2}} & fitness_{\widehat{s_2}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ M^P_{\widehat{s_x},1} & M^P_{\widehat{s_x},2} & \dots & M^P_{\widehat{s_x},l_{\overline{s_x}}} & COFFEE_{\widehat{s_x}} & fitness_{\widehat{s_x}} \end{array} \right) \quad (5)$$

donde: P: matriz auxiliar P; $M^P_{\widehat{s_a},j}$: $j-$ésimo elemento de la secuencia $\widehat{s_a}$; $COFFEE_{\widehat{s_a}}$: valor de la función COFFEE la secuencia $\widehat{s_a}$; $fitness_{\widehat{s_a}}$: valor de la función de aptitud de la secuencia $\widehat{s_a}$.

La submatriz M^P almacena el alineamiento múltiple factible para el conjunto de $N - secuencias$ para su cálculo se utiliza la subrutina 3, mientras que la submatriz G^P contiene estimadores de eficiencia sobre el alineamiento propuesto para lo que se utiliza la subrutina 4.

Algoritmo 4. Subrutina 3 Construcción de la submatriz M^P.
Input: Un conjunto de secuencias ($N = \{s_1, s_2, \dots, s_x\}$).
Output: Submatriz M^P.

1. $x \leftarrow$ Número de secuencias que integran al conjunto N.
2. l_M Número de columnas factibles para la matriz M^P.
3. For $i = 1 : 1 : x$
   (a) For $j = 1 : 1 : l_M$
      i. $M^P_{i,j} \leftarrow$ Colocar aleatoriamente un carácter o un espacio vacio.
   End For
   End For
4. Eliminar columnas que consten exclusivamente de guiones.
5. Actualizar el número de columnas de la submatriz M^P.

Algoritmo 5. Subrutina 4 Construcción de la submatriz G^P.
Input: Submatriz M^P.
Output: Submatriz G^P.

1. Calcular una matriz D de distancia entre los pares de secuencias ya alineados.
2. $x \leftarrow$ Número de secuencia en el conjunto N.
3. For $i = 1 : 1 : x$

(a) $G^P_{i,1} = \sum_{j=1}^{x} D_{i,j}$

End For

4. $V(M^P) = \frac{\sum_{i=1}^{x} G^P_{i,1}}{2}$
5. For $i = 1:1:x$

   (a) $G^P_{i,2} = \frac{G^P_{i,1}}{2 * V*(M^P)}$

   End For

2.3 Generación de un nuevo vector armónico

Una vez construida la memoria armónica inicial, se procede a generar una nueva improvisación, con el objeto de mejorar los alineamientos contenidos en HM, para lo cual se utiliza la subrutina 5.

Algoritmo 6. Subrutina 5 Generación de un nuevo vector armónico.

Input: Conjunto de secuencias N, vector de longitudes de las secuencias l, HM, $HMCRI$,
$HMCRF$,$PARI$,$PARF$,l_M,b, Matrices auxiliares asociadas $(P^1, P^2, , P^{HMS})$.

Output: Nueva armonía.

1. Actualizar parámetros PAR y $HMCR$.
2. If $rand \leq HMCR$

   (a) Utilizar un elemento en HM para generar el nuevo vector armónico.

      i. If $rand \leq PAR$

         • Variar en vecindades el elementos HM.

      End If

   (b) Construir una matriz auxiliar P acorde obtenida HM.

   (c) If es factible.

      i. Evaluar función objetivo.

      ii. Construir nuevo vector armónico.

   End If

   End IF
3. Else If

   (a) Construir aleatoriamente una matriz auxiliar P acorde obtenida HM.

   (b) If es factible.

      i. Evaluar función objetivo.

ii. Construir nuevo vector armónico.

End If

4. $V_{Auxiliar} \leftarrow$ valor de función objetivo.

2.4 Actualizar HM

En esta fase se toma la decisión de remplazar o no un elemento de la HM por el nuevo vector armónico.

Algoritmo 7. Subrutina 6 Actualización HM.

Input: Una nueva improvisación(matriz auxiliar), t_0, α, $V_{Auxiliar}$ y la memoria armónica HM.

Output: Memoria armónica actualizada.

1. $V_{peor} \leftarrow$ Peor valor en la columna 2 de HM.
2. $HM_i \leftarrow$ Vector HM correspondiente a peor valor.
3. If $V_{Auxiliar}$ mejor a V_{peor}
   (a) Remplazar HM_i por nuevo vector armónico.
4. Else
   (a) $P(a) = e^{\frac{V_{Auxiliar} - V_{peor}}{T}}$
   (b) $T = \alpha * T$
   (c) If $rand \leq P(a)$
      i. Remplazar HM_i por nuevo vector armónico.

   End If

   End IF
5. $q = \sum_{p=1}^{HMS} HM_{p,2}$
6. For $P = 1 : 1 : HMS$
   (a) $HM_{i,3} \leftarrow \frac{q - HM_{i,2}}{q}$

   End For

2.5 Satisfacer criterio de paro

El criterio que se utiliza en este algoritmo es del número máximo de iteraciones (nuevas improvisaciones), repitiendo iterativamente el algoritmo (y subrutinas) para la generación de una nueva armonía y para la actualización de memoria armónica.

3 Validación del algoritmo

Para realizar la prueba de validación se utilizó la base de datos BALiBASE (Thompson, et al. 1999) utilizada para la evaluación y comparación de programas y algoritmos destinados para resolver el AMS.

BALiBASE versión 1 es una colección referencial de 141 alineamientos proteicos que contiene más de 1000 secuencias (Notredame 2000;Thompson et al. 1999 & Wang y Li 2004). Para mayor información sobre BALiBASE consúltese (Thompson et al 1999).

La prueba de validación consistió en determinar el porcentaje de diferencia existente entre el alineamiento obtenido por el algoritmo híbrido y el alineamiento referencia mediante el programa bali_score para la cuantificación del parámetro SSP y la comparación de los resultados obtenidos por otras técnicas reportados en datos históricos.

4 Resultados

El algoritmo híbrido se programo en MATLAB Ra2008 y se corrieron en una computadora Inspiron 1720 procesador Intel centro Duo.

Referencia	Características de las secuencias del conjunto [2]		Promedio
1	Cortas	Menor 25%	0.2
		20-40%	0.37
		Mayor 35%	0.50
	Medianas	Menor 25%	0.13
		20-40%	0.28
		Mayor 35%	0.40
	Largas	Menor 25%	0.07
		20-40%	0.24
		Mayor 35%	0.26
2		Cortas	0.23
		Medianas	0.15
		Largas	0.16
3		Cortas	0.17
		Medianas	0.08
		Largas	0.11
4			0.01
5			0.07

[2]Longitud y porcentaje de similitud promedio entre las cadenas del conjunto.

Tabla 3: Valor de la similitud entre los alineamientos obtenidos por el algoritmo híbrido y los alineamientos referenciales cuantificando la métrica SSP.

Para cada uno de los 141 alineamientos referenciales se generaron tres alineamientos por el algoritmo híbrido $HS-RS$ y para cada uno de ellos se cuantifico el parámetro SSP, promediando el resultado de las tres replicas. En la tabla 3 se muestra un concentrado de los resultados obtenidos.

En la tabla 3 se observa que el mejor desempeño del la heurística desarrollada se da al trabajar en conjuntos pequeños de secuencias equidistantes (referencia 2), donde el comportamiento del algoritmo híbrido en comparación con otros métodos de solución para el AMS es en promedio una mejor alternativa a los métodos DIALIGN, HMMT, SB_PIMA y ML_PIMAS, ya que ofrece una mayor cercanía al alinamento referencia; sin embargo, los resultados obtenidos por el híbrido son inferiores a los encontrados por SAGA, Clustal X y Multialing.

En contraste, al trabajar con secuencias poco emparentadas y de tamaños variados su desempeño debe ser perfeccionado, (referencias 4 y 5).

Con respecto a la complejidad temporal el algoritmo híbrido $HS-RS$ resulta atractivo, dado que el orden de su complejidad es similar al reportado por los métodos Clustal y T-COFFEE, que actualmente son los más utilizados para resolver el AMS. Lo anterior se muestra en la tabla 4 (ver página 135).

5 Conclusiones

El algoritmo híbrido $HS-RS$ resulta una estrategia atractiva para el alineamiento de secuencias homólogas y estrechamente relacionada, además de poder alinear de manera simultánea a un número grande de secuencias en virtud de que este valor queda limitado por las características del procesador donde se implemente y que su complejidad ($O(n^2 * l^2)$) es similar a los métodos que actualmente son más utilizados para resolver el AMS. En contraste debe ser perfeccionado para alinear secuencias heterogéneas de longitud variable.

Referencias

[1] Agüero, F. (2004) "Alineamiento de secuencias. Búsqueda de secuencias en bases de datos", en:
`http://brucella.unsam.edu.ar/bioinformatica2004/sss-msa.ppt`, Instituto de Investigaciones Biotecnológicas, Universidad Nacional de General San Martín. Consultado 1/10/2008 8:30 p.m.

[2] Altschul, S.F.; Erickson, B.W. (1986) "Optimal sequence alignment using affine gap cost", *Bulletin of Mathematical Biology* bf 48(5-6): 603–616.

[3] Barton, G.J.; Sternberg, M.J.E. (1987) "A strategy for the rapid multiple alignment of protein sequences", *Journal Molecular Biology* **198**(2): 327–337.

[4] Chan, S.C.; Wong, K.C.; Chiu, D.K. (1992) "A survey of multiple sequence comparison methods", *Bulletin of Mathematical Biology* **54**(4): 563–598.

[5] Dong, E.; Smith, J.; Heinze, S.; Nathan, A.; Meiler, J. (2008) "BCL: Align-Sequence alignment and fold recognition with a custom scoring function online", *Gene* **422**(1-2): 41–46.

Nombre del algoritmo	Nombre del algoritmo	Nombre del algoritmo
Matriz de puntos	$O(l_a * l_b)$	2
Distacia de Leveshein	$O(l_a * l_b)$	2
Distancia Indel	$O(l_a * l_b)$	2
Distancia de Damerau	$O(l_a * l_b)$	2
Needleman-Wunsch	$O(l_a * l_b)$	2
Murata et al	$O(l_a * l_b * l_c)$	3
Gotoh	$O(l_a * l_b)$	2
Gotoh-Altschul	$O(l_a * l_b)$	2
Carrillo-Lipman	$O(l^n)$	10
Smith-Waterman	$O(l_a * l_b)$	2
Fredman	$O(l^3)$	3
Sankoff	$O(n_i(2l^n))$	5
Sankoff-Cedergren	$O(n_i(2l^n))$	3
Fitch	$O(n_i(2l^n))$	15
Johnson-Doolittle	$O(n(l - l_r)l_r^{n-1})$	3 o más
Karlin	$O(\|\mathcal{L}^2\|)$	2 o más
Waterman-Jones	$O(n * l_r^2 * l * B)$	2 o más
Waterman-Perlwitz	$O(nl^2)$	3 o más
Barton-Sternberg	$O(n!)$	hasta 10
Subbiah-Harrison	$O(n!)$	hasta 10
Clustal	$O(n^2l^2)$	3 o más
T-COFFEE	$O((n^2l^2) + (n^3))$	3 o más
Algoritmo A^*	$O(n^l)$	3 o más
Algoritmo de Viterbi	$O(n^3P)$	3 o más
Híbrido de $HS - RS$	$O(n^2l^2)$	3 o más

Tabla 4: Comparativo sobre la complejidad de algunos de los métodos para la solución del AMS, donde: n_i: número de nodos internos en el árbol filogenético; l: longitud de la secuenica más larga; n: número de secuencias; l_r: longitud del residuo; $\mathcal{L}^2$: tamaño de la lista de regiones; B: valor de función de acuerdo con la naturaleza de las cadenas.

[6] Elias, I. (2003) "Settling the intractability of multiple alignment", in: T. Ibaraki, N. Katoh & H. Ono (Eds.) *Proc. 14th Annual Int. Symp. on algorithms and computation (ISSAC)*, Springer-Verlag Berlin Heidelberg: 352–363.

[7] Geem, Z.W.; Kim, J.H.; Loganathan, G.V. (2001) "A new heuristic optimization algoritm: harmony search". *Simulation* **76**(2): 60–68.

[8] Grimaldi, R.P. (1998) *Matemticas Discretas y Combinatorias. Una Introduccin con Aplicaciones*, Tercera Edición. Pearson Printice Hall, México.

[9] Gusfield, D. (1993) "Efficient methods for multiple sequence alignment with guaranteed error bounds", *Bulletin of Mathematical Biology* **55**(1): 141–154.

[10] Ingram G.; Zhang, T. (2009) "Overview of applications and developments in the harmony search algorithm", in: Z.W. Geem (Ed.) *Music-Inspired Harmony Search Algorithm. Theory and Applications*, Studies in Computational Intelligence 191, Springer, Berlin: 15–37.

[11] Lee, R.C.T.; Tseng, S.S.; Chang, R.C.; Tsai, Y.T. (2007) *Introduccin al Diseo y Anlisis de Algoritmos. Un Enfoque Estratgico*. Mc Graw-Hill, México.

[12] Ma, B.; Wang, L.; Li, M. (2007) "Near optimal multiple alignment within a band in polynomial time", *Journal of Computer and System Sciences* **73**(6): 997–1011.

[13] Manthey, B.(2003) "Non-approximability of weighted multiple sequence alignment", *Theoretical Computer Science* **296**(1): 179–192.

[14] Mora-Gutiérrez, R.A. (2009) *Desarrollo de un Procedimiento para Solucionar el Problema de Alineamiento Múltiple de Secuencias*. Tesis de Maestría en Ingeniería, Universidad Nacional Autónoma de México.

[15] Needleman, S.B.; Wunsch, C.D. (1970) "A general method applicable to the search for similarities in the amino acid sequence of two proteins", *Journal of Molecular Biology* **48**(3): 443–453.

[16] Notredame, C. (2000) "T-coffee: a novel method for fast and accurate multiple sequence alignment", *Journal of Molecular Biology* **302**(1): 205–217.

[17] Omar, M.F.; Salam, R.A.; Abdullah, R.; Rashid, N.A. (2005) "Multiple sequence alignment using optimization algorithms", *International Journal of Computational Intelligence* **1**(2): 81–89.

[18] Rech, D.H.; Pilatti, R.(2004) *Align-UMA ferramenta para alinhamento múltiplo de seqüencias de DNA e proteínas*. Tesis en Ciencias de la Computación, Universidade Federal de Santa Catarina, Brazil.

[19] Thompson, J.D.; Plewnlak, F.; Poch, O. (1999) "BALiBASE: A benchmark aligment database for the evaluatios of multiple alignmet programs", *Bioinformatics* **15**(1): 87–88.

[20] Wang, Y.; Li, K.B. (2004) "An adaptive and iterative algorithm for refining multiple sequence alignment", *omputational Biology and Chemistry* **28**(2): 141–148.

[21] Lee, Z.J.; Su, S.F.; Chuang, C.C.; Liu, K.H. (2008) "Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment". *Applied Soft Computing* **8**(1): 55–78.