

Universidad Autónoma de Querétaro  
Facultad de Ingeniería  
Maestría en Ciencias en Inteligencia Artificial

# Algoritmos para la Predicción de Estructuras Secundarias de ARN

## TESIS

Que como parte de los requisitos para obtener el Grado de  
Maestro en Ciencias en Inteligencia Artificial

### Presenta:

Ing. Pedro Ayala Elizarráraz

### Dirigida por:

Dr. Arturo González Gutiérrez

### Codirigida por:

Dr. Rolando Tenoch Bárcenas Luna

## SINODALES

Dr. Arturo González Gutiérrez

Presidente

Dr. Rolando Tenoch Bárcenas Luna

Secretario

Dr. Saúl Tovar Arriaga

Vocal

Dr. Fausto Arellano Carbajal

Suplente

M. C. Fidel González Gutiérrez

Suplente

Centro Universitario, Querétaro, Qro.

Febrero 2021

México

Dirección General de Bibliotecas de la UAQ

*A mi esposa Mónica.*

---

# Agradecimientos

---

Se agradece al Consejo Nacional de Ciencia y Tecnología (CONACYT) por proveer los fondos necesarios para realizar esta investigación a través del Programa de Becas Nacionales. Así como a la Universidad Autónoma de Querétaro por la Beca Institucional. También, agradezco profundamente a mis sinodales por sus asesorías y apoyo en el desarrollo de la tesis. Quiero agradecer al Dr. Arturo González Gutiérrez a quien considero mi mentor académico por haberme motivado y asesorado en la realización de mis estudios de maestría.

Dirección General de Bibliotecas de la UAQ

---

# Resumen

---

El comportamiento de una molécula de ARN depende directamente de sus estructuras secundarias y terciarias. Sin embargo, se ha demostrado que la predicción de estructuras secundarias con pseudonudos arbitrarios es un problema NP-completo. En esta tesis se presenta el problema de la predicción de estructuras secundarias de ARN a través del estudio de tres algoritmos basados en la técnica de programación dinámica. Los algoritmos de Nussinov y Zuker predicen estructuras secundarias sin pseudonudos, mientras que el algoritmo de Akutsu lo hace con pseudonudos simples. También presentamos una metaheurística que utiliza un algoritmo genético para producir subestructuras cuasi-óptimas que a su vez permiten la predicción de estructuras secundarias con pseudonudos simples. También presentamos un análisis experimental de los cuatro algoritmos implementados utilizando instancias públicas de estructuras de ARN provistas por las bases de datos RNA STRAND y PseudoBase ++.

Finalmente, se presenta un prototipo de sistema con el cual se puede obtener la estructura secundaria para una secuencia de ARN provista por el usuario. El prototipo cuenta con la implementación de los cuatro algoritmos estudiados en la presente tesis.

(**Palabras clave:** Estructura Secundaria de ARN, Mínima Energía Libre, Pseudonudo, Programación Dinámica, Algoritmo Genético).

---

# Summary

---

The behaviour of an RNA molecule is directly linked to its secondary and tertiary structures. However, it has been proved that the secondary structure prediction problem with arbitrary pseudoknots is an NP-complete problem. In this theses, we present the RNA secondary structure prediction problem through the study of three algorithms based on the dynamic programming technique. The Nussinov and Zuker algorithms predict secondary structures without pseudoknots, whereas Akutsu algorithm does it with simple pseudoknots. We also present a metaheuristic that uses a genetic algorithm to produce quasi-optimal substructures which in turn allow the prediction of secondary structures with simple pseudoknots. We also present an experimental analysis of the four algorithms using public instances of RNA structures provided by RNA STRAND and PseudoBase ++ databases.

Finally, a prototype system is presented in which the secondary structure for an RNA sequence provided by the user can be obtained. The prototype has the implementation of the four algorithms studied in this thesis.

(**Keywords:** RNA Secondary Structure, Minimum Free Energy, Pseudoknot, Dynamic Programming, Genetic Algorithm)

---

# Índice general

---

<b>Agradecimientos</b>	<b>III</b>
<b>Resumen</b>	<b>IV</b>
<b>Summary</b>	<b>V</b>
<b>Índice</b>	<b>VI</b>
<b>Índice de Figuras</b>	<b>IX</b>
<b>Índice de Tablas</b>	<b>XII</b>
<b>Índice de Algoritmos</b>	<b>XIII</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Justificación . . . . .	1
1.2. Hipótesis . . . . .	2
1.3. Objetivos . . . . .	2
1.3.1. Objetivo General . . . . .	2
1.3.2. Objetivos Específicos . . . . .	2
1.4. Marco Teórico . . . . .	3
1.4.1. ARN . . . . .	3
1.4.2. Estructura Secundaria . . . . .	4
1.4.2.1. Pseudonudos . . . . .	8
1.4.2.2. Mínima energía libre . . . . .	10
1.4.3. Notación Dot-Bracket . . . . .	11
1.4.4. Forna: Visualización de estructuras secundarias . . . . .	11
1.5. Esquema de la Tesis . . . . .	12
<b>2. Modelo Termodinámico del Vecino Más Cercano</b>	<b>14</b>
2.1. Hélices . . . . .	15
2.2. Bucles . . . . .	17
2.2.1. Bucle en horquilla . . . . .	19

2.2.2.	Protuberancia . . . . .	23
2.2.3.	Bucle interno . . . . .	25
2.2.4.	Bucle múltiple . . . . .	30
<b>3.</b>	<b>Metodología</b>	<b>33</b>
3.1.	Algoritmos Exhaustivos . . . . .	33
3.1.1.	Algoritmo de Nussinov . . . . .	33
3.1.2.	Algoritmo de Zuker . . . . .	45
3.1.3.	Algoritmo de Akutsu . . . . .	51
3.1.4.	Algoritmo de McCaskill . . . . .	54
3.2.	Algoritmos Metaheurísticos . . . . .	56
3.2.1.	Algoritmos Genéticos . . . . .	56
3.2.1.1.	Terminología de AGs . . . . .	56
3.2.1.2.	Operadores Genéticos . . . . .	57
3.2.2.	Algoritmos GAknot, GA-GRASP y RNAknot . . . . .	58
3.2.3.	Algoritmo Genético Propuesto . . . . .	61
3.2.3.1.	Determinar posibles hélices . . . . .	61
3.2.3.2.	Generación de la población . . . . .	64
3.2.3.3.	Iteración . . . . .	66
3.2.3.4.	Pseudocódigo . . . . .	70
<b>4.</b>	<b>Resultados y Discusión</b>	<b>72</b>
4.1.	Métricas de precisión . . . . .	72
4.2.	Resultados Experimentales . . . . .	74
<b>5.</b>	<b>Prototipo</b>	<b>85</b>
5.1.	Predicción de una sola secuencia de ARN . . . . .	85
5.2.	Benchmark de los algoritmos . . . . .	87
<b>6.</b>	<b>Conclusiones</b>	<b>94</b>
	<b>Referencias Bibliográficas</b>	<b>96</b>

<b>Apéndice</b>	<b>100</b>
A. Configuraciones del Algoritmo Genético . . . . .	100
B. Artículo XIV Coloquio de Posgrado . . . . .	102

Dirección General de Bibliotecas de la UAQ



---

# Índice de Figuras

---

1.1. Estructura química de los ribonucleótidos base que componen el ARN.	3
1.2. Estructura primaria de una secuencia de ARN. . . . .	4
1.3. Representación de las posibles subestructuras que una cadena de ARN puede adoptar . . . . .	6
1.4. Estructura secundaria correspondiente al Ejemplo 1.1. . . . .	7
1.5. Representación de la subestructura de un pseudonudo. . . . .	8
1.6. Representación planar de un pseudonudo. . . . .	9
1.7. Pseudonudo de tipo H-type. . . . .	9
1.8. Estructura secundaria libre de pseudonudos. . . . .	11
1.9. Estructura secundaria libre de pseudonudos en formato Dot-Bracket.	11
1.10. Visualización de una estructura secundaria usando Forna. . . . .	12
2.1. Representación de los pares base consecutivos de una hélice. . . . .	15
2.2. Estructura secundaria formada por hélices de pares complementarios tipo Watson-Crick. . . . .	16
2.3. Ajustes de energía terminales para el par de cierre (G, C). . . . .	20
2.4. Estructura secundaria formada por una horquilla. . . . .	20
2.5. Estructura secundaria de una protuberancia. . . . .	23
2.6. Parámetros de energía para bucle interno $1 \times 1$ con cierre (G, C) y (G, C). . . . .	25
2.7. Bucle interno simétrico de tamaño $1 \times 1$ . . . . .	26
2.8. Parámetros de energía para bucle interno $1 \times 2$ con cierre (U, A) y (C, G). . . . .	26
2.9. Bucle interno asimétrico de tamaño $1 \times 2$ . . . . .	26
2.10. Parámetros de energía para bucle interno $2 \times 2$ con cierre (C, G) y (C, G). . . . .	27
2.11. Bucle interno simétrico de tamaño $2 \times 2$ . . . . .	28
2.12. Bucle múltiple de 3 ramificaciones y 6 bases libres. . . . .	30
2.13. Base libre adyacente a una hélice. . . . .	31
2.14. Pares base de cierre del bucle múltiple . . . . .	31

3.1. Los cuatro posibles casos para la función recursiva de Nussinov. . . . .	35
3.2. Cálculo del valor $i, j$ en la matriz del algoritmo Nussinov. . . . .	35
3.3. Nussinov Paso 1. Inicialización de la matriz. . . . .	38
3.4. Nussinov Paso 2A. Cálculo de la segunda diagonal . . . . .	38
3.5. Nussinov Paso 2B. Cálculo de la tercera diagonal . . . . .	39
3.6. Nussinov Paso 2C. Cálculo de la cuarta diagonal . . . . .	39
3.7. Nussinov Paso 2D. Cálculo de la quinta diagonal . . . . .	40
3.8. Nussinov Paso 2E. Cálculo de la sexta diagonal . . . . .	40
3.9. Nussinov Paso 2F. Cálculo de la séptima diagonal . . . . .	41
3.10. Nussinov Paso 2G. Cálculo de la octava diagonal . . . . .	41
3.11. Nussinov Paso 2H. Cálculo de la novena diagonal . . . . .	42
3.12. Nussinov Paso 3. Ruta de la puntuación máxima de la estructura secundaria. . . . .	43
3.13. Estructura secundaria para $S_r = \{(0, 8), (1, 7), (3, 6)\}$ . . . . .	44
3.14. Estructura secundaria para $S_r = \{(0, 8), (1, 7), (2, 6)\}$ . . . . .	44
3.15. Matriz $W$ procesada por el algoritmo de Zuker. . . . .	49
3.16. Matriz $V$ procesada por el algoritmo de Zuker. . . . .	50
3.17. Matriz $V_{ds}$ procesada por el algoritmo de Zuker. . . . .	50
3.18. Estructura secundaria y notación Dot-Bracket para $r =$ AAACAUGAGGAUUACCCAUGU. . . . .	52
3.19. Representación de las ecuaciones $S_L, S_M$ y $S_R$ . . . . .	53
3.20. Diagrama de flujo del algoritmo genético básico. . . . .	59
3.21. Proceso de creación de la población inicial de un algoritmo genético. . . . .	61
3.22. Matriz de probabilidad obtenida mediante el algoritmo de McCaskill. . . . .	63
3.23. Proceso de cruzamiento. . . . .	68
4.1. Secuencia de ARN, estructura secundaria de referencia y estructura predicha mediante el algoritmo de Zuker. . . . .	73
4.2. Sensibilidad de los algoritmos para instancias sin pseudonodos. . . . .	77
4.3. Valor Predictivo Positivo de los algoritmos para instancias sin pseudonodos. . . . .	77

4.4. Valor-F de los algoritmos para instancias sin pseudonudos. . . . .	78
4.5. Coeficiente de Correlación de Matthews de los algoritmos para instancias sin pseudonudos. . . . .	79
4.6. Tiempo de ejecución de los algoritmos para instancias sin pseudonudos.	80
4.7. Sensibilidad de los algoritmos para instancias con pseudonudos. . . .	81
4.8. Valor Predictivo Positivo de los algoritmos para instancias con pseudonudos. . . . .	82
4.9. Valor-F de los algoritmos para instancias con pseudonudos. . . . .	83
4.10. Coeficiente de Correlación de Matthews de los algoritmos para instancias con pseudonudos. . . . .	83
4.11. Tiempo de ejecución de los algoritmos para instancias con pseudonudos.	84
5.1. Ejemplo de archivo en formato FASTA. . . . .	85
5.2. Pantalla para la predicción de una sola secuencia. . . . .	86
5.3. Selección del algoritmo a ejecutar. . . . .	86
5.4. Pantalla de resultado de la predicción de la estructura secundaria. . .	87
5.5. Ejemplo de archivo válido para la aplicación. . . . .	87
5.6. Pantalla de inicio de la opción benchmark. . . . .	88
5.7. Cuadro de dialogo para la selección de carpeta. . . . .	88
5.8. Pantalla de benchmark con el listado de las instancias a probar. . . .	89
5.9. Ventana de información de la secuencia. . . . .	90
5.10. Prototipo: algoritmos en ejecución. . . . .	90
5.11. Pantalla de resultados para una instancia . . . . .	91
5.12. Opción <i>Benchmark Report</i> . . . . .	92
5.13. Pantalla de reporte de las métricas de precisión . . . . .	92
5.14. Pantalla de reporte con los tiempos de ejecución de los algoritmos. . .	93

---

# Índice de Tablas

---

2.1. Energía de las hélices entre cada par base. . . . .	15
2.2. Energías de desestabilización por tamaño del bucle. . . . .	18
2.3. Energías para horquillas especiales de tamaño 4. . . . .	19
2.4. Energías para horquillas especiales . . . . .	21
2.5. Parámetros para horquillas especiales . . . . .	23
3.1. Lista de hélices obtenidas a partir de la matriz de probabilidades. . .	64
3.2. Creación del individuo {4, 2, 5, 9, 11}. . . . .	65
3.3. Caracterización de los algoritmos para modelación de estructuras secundarias. . . . .	71
4.1. Secuencias de ARN sin pseudonudos en la estructura de referencia. .	76
4.2. Secuencias de ARN con pseudonudos en la estructura de referencia. .	81
1. Configuración del algoritmo genético para instancias sin pseudonudos.	100
2. Configuración del algoritmo genético para instancias con pseudonudos.	101

---

# Índice de Algoritmos

---

2.1. Energía de la hélice . . . . .	17
2.2. Energía del bucle en horquilla . . . . .	22
2.3. Energía de la protuberancia . . . . .	24
2.4. Energía del bucle interno . . . . .	29
3.1. Nussinov . . . . .	36
3.2. Nussinov Rastreo . . . . .	37
3.3. Zuker . . . . .	47
3.4. Zuker Rastreo . . . . .	48
3.5. McCaskill . . . . .	55
3.6. Hélices por probabilidad . . . . .	62
3.7. Evaluación . . . . .	66
3.8. Cruza . . . . .	68
3.9. Mutación . . . . .	69
3.10. Algoritmo Genético . . . . .	70

Dirección General de Bibliotecas de la UAQ

# Introducción

*Sólo podemos ver poco del futuro, pero lo suficiente para darnos cuenta de que hay mucho que hacer.*

—ALAN MATHISON TURING (1912 - 1954)

## 1.1. Justificación

Debido a que la funcionalidad de una molécula de ARN depende en gran medida de sus estructuras secundarias y terciarias, es importante su estudio y caracterización (Fatmi, Bekri, y Benhlima, 2018). La caracterización y modelación de la estructura terciaria representa un mayor desafío científico y tecnológico que la estructura secundaria. Por ello, los esfuerzos se han enfocado en gran medida en la predicción de estructuras secundarias, ya que es experimentalmente accesible y contiene información para determinar la relación entre estructura y funcionalidad (Churkin, Weinbrand, y Barash, 2015).

Existen dos técnicas de predicción de estructuras secundarias, las que se obtienen mediante experimentación y modelación computacional. Las técnicas de experimentación más utilizadas para determinar las estructuras de ARN son: *Rayos X*, *Cristalografía* y la *Resonancia Magnética Nuclear* (Fatmi et al., 2018) (Zhao, Ni, y Wang, 2009).

A través de la modelación computacional es posible reducir el tiempo y costo de la predicción de estructuras secundarias para una molécula (Fatmi et al., 2018). Sin embargo, no se está exento de ciertas limitantes. Por ejemplo, en la predicción de estructuras con pseudonudos se incrementa la complejidad tanto del tiempo y del espacio computacional requerido para realizar la tarea. Es sabido que la predicción de estructuras con pseudonudos es un problema **NP-Completo** (Osman, Abdullah, y AbdulRashid, 2010) (Liu, Zhu, Cui, y Liu, 2013), de ahí la importancia de algoritmos

basados en heurísticas que nos permitan resolver el problema de manera óptima en tiempo no mayor a  $O(n^4)$ , donde  $n$  representa la longitud de la secuencia de ARN.

## 1.2. Hipótesis

La implementación de un modelo basado en técnicas de programación avanzada y algoritmos genéticos permite que la predicción de estructuras secundarias de ARN se realice de manera eficiente en tiempo polinomial.

## 1.3. Objetivos

### 1.3.1. Objetivo General

Diseñar e implementar un modelo computacional para la predicción de estructuras secundarias de secuencias de ARN, mediante técnicas de optimización, con la finalidad de obtener la estructura que presente la mayor estabilidad en términos de energía libre entre sus enlaces moleculares.

Si bien existen factores estructurales tanto químicos como físicos que pueden ser incluidos dentro del modelo de predicción de estructuras, en este trabajo nos centraremos únicamente en los factores termodinámicos de energía libre incluidos en el Modelo Termodinámico del Vecino Más Cercano, que sirven como criterio de optimización para los algoritmos implementados.

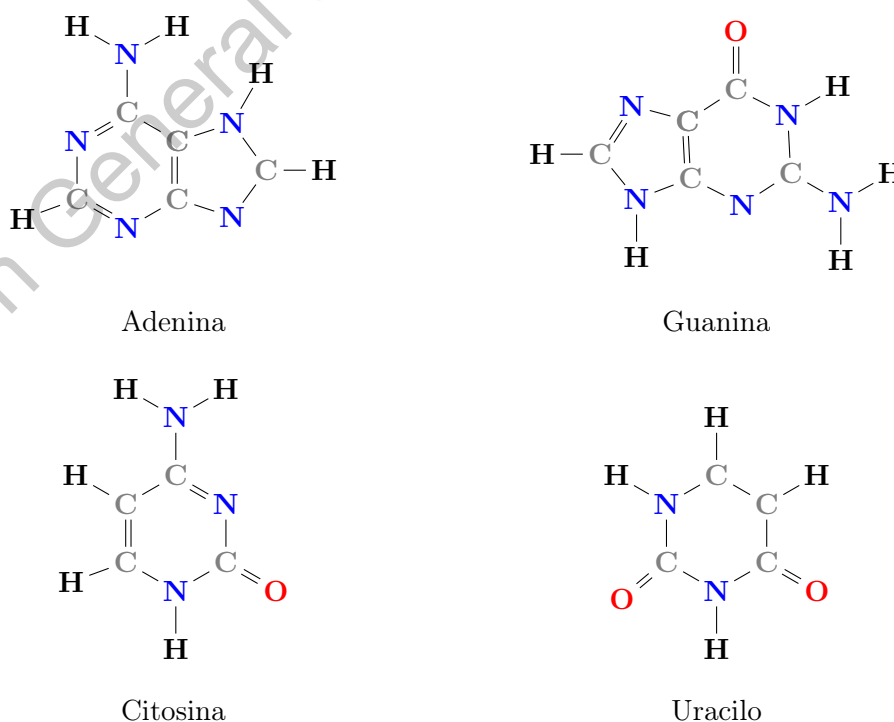
### 1.3.2. Objetivos Específicos

- Modelar la energía libre entre moléculas usando técnicas de programación dinámica.
- Construir los pseudonudos y subestructuras durante el proceso de plegamiento de las secuencias usando heurísticas eficientes.
- Evaluar experimentalmente y determinar órdenes de complejidad en tiempo de las técnicas empleadas.
- Validar el modelo biocomputacional empleando datos de bases de datos reales.

## 1.4. Marco Teórico

### 1.4.1. ARN

El ácido ribonucleico (*ARN*) es una molécula que consiste en una cadena de nucleótidos con una estructura similar a la del ácido desoxirribonucleico (*ADN*). Las principales diferencias del ARN con respecto al ADN es que el ARN consiste en una hélice sencilla, mientras que el ADN está formado por una doble hélice. Adicionalmente, la base nitrogenada *Timina* (T) en el ADN es sustituida por el *Uracilo* (U) en el ARN, mientras que las bases de la *Adenina* (A), *Citosina* (C) y *Guanina* (G) se conservan (Figura 1.1). En comparación con el ADN, las moléculas del ARN son menos estables y exhiben una mayor variabilidad en su estructura tridimensional (Polanski y Kimmel, 2007). El ARN adquiere una gran relevancia, ya que está involucrado en varios procesos biológicos, que incluyen: la codificación y decodificación de información genética, regulación de la expresión génica, detección y comunicación de respuestas a señales celulares y catalización de las reacciones biológicas (Punetha et al., 2018).



**Figura 1.1:** Estructura química de los ribonucleótidos base que componen el ARN.



A la serie de nucleótidos a lo largo de una molécula de ARN se le conoce como *estructura primaria* (Figura 1.2). La serie de nucleótidos o secuencia de ARN tiene una dirección, con los extremos etiquetados como 5' y 3'.

r: 5' ... GUGACUGGAGUUCAGACGUGUGCUC ... 3'

**Figura 1.2:** Estructura primaria de una secuencia de ARN.

Computacionalmente, denotamos a la estructura primaria de una secuencia de ARN como un *string* o *cadena* producida por el lenguaje del alfabeto  $\Sigma_{ARN} = \{A, C, G, U\}$ . En la Definición 1.1, se enuncia formalmente este concepto.

**Definición 1.1 (Secuencia de ARN).** Sea  $r = r_1 \dots r_n$  una cadena y  $\Sigma_{ARN} = \{A, C, G, U\}$  el conjunto que representa el alfabeto de los nucleótidos que conforman el ARN. Si  $r$  es una cadena conforme al alfabeto  $\Sigma_{ARN}$ , entonces,  $r$  es una secuencia de ARN.

#### 1.4.2. Estructura Secundaria

La estructura secundaria de una secuencia de ARN consiste en el plegamiento de la cadena de nucleótidos consigo misma, mediante enlaces de hidrógeno entre sus bases y en diferentes posiciones sobre la misma cadena, en donde cada nucleótido sólo puede formar parte de un par base (Böckenhauer y Bongartz, 2007) (Punetha et al., 2018).

Computacionalmente representamos a la estructura secundaria como el conjunto  $S_r$  de pares de índices, que corresponden con las posiciones de los pares base sobre la secuencia  $r$ ,

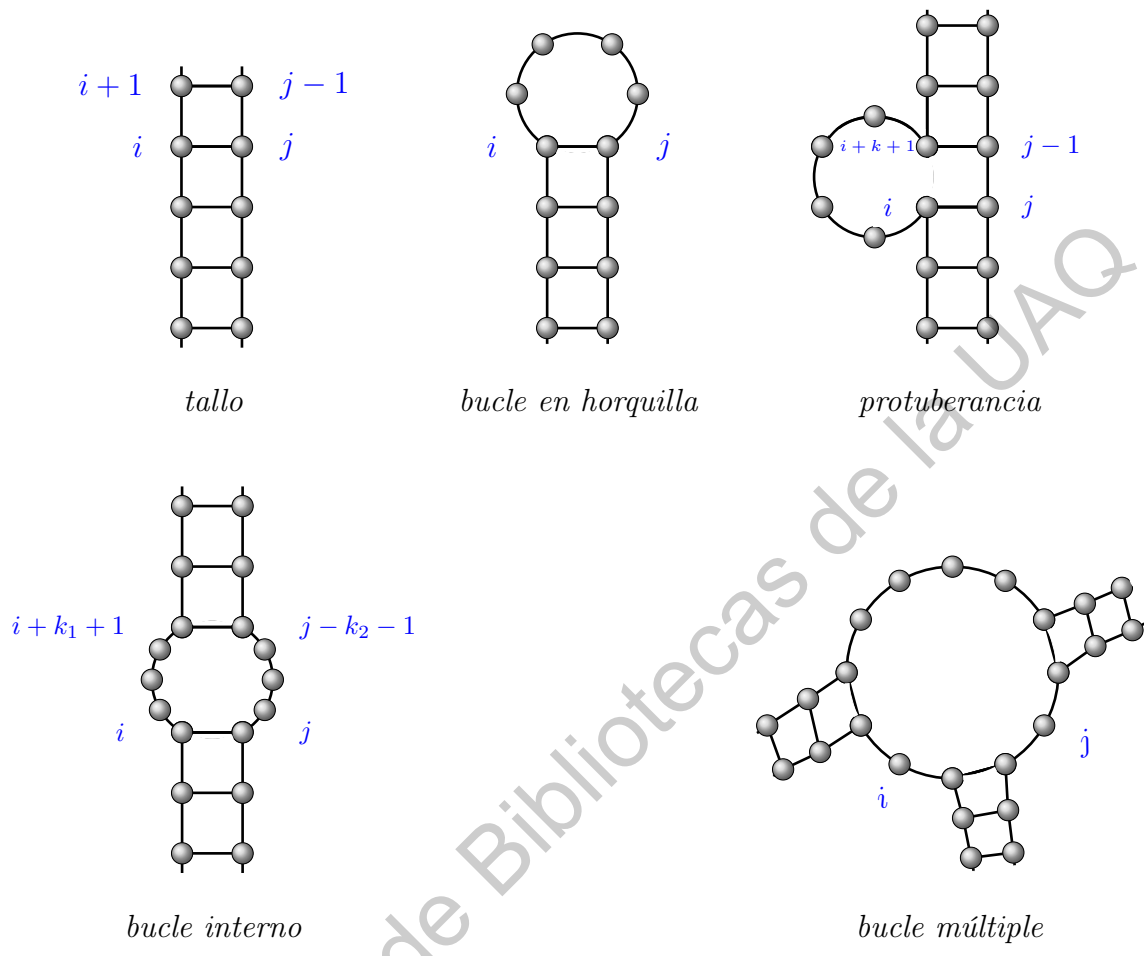
$$S_r \subseteq \{(i, j) \mid 1 \leq i < j \leq n\}$$

donde  $n$  es la longitud de la secuencia,  $i$  y  $j$  son posiciones en la cadena que representan bases complementarias entre sí. Además,  $S_r$  debe satisfacer las siguientes restricciones (Böckenhauer y Bongartz, 2007) (Liu et al., 2013):

- I. Cada índice  $k \in \{1, \dots, n\}$  ocurre a lo más una vez en el emparejamiento de  $S_r$ .
- II. Para cada par  $(i, j)$  de  $S_r$ , el par de bases  $(i, j)$  es un par de tipo *Watson-Crick* si  $(r_i, r_j) \in \{(A, U), (U, A), (C, G), (G, C)\}$ , o es un par tipo *Wobble* si  $(r_i, r_j) \in \{(G, U), (U, G)\}$ .
- III. Para cada par  $(i, j)$  de  $S_r$ ,  $j - i \geq 4$  (donde 4 es el tamaño mínimo de una región de bases no complementarias, llamada bucle en horquilla).

Los elementos secundarios (Figura 1.3) de una estructura de ARN se pueden clasificar en cinco tipos básicos: hélices, bucles en horquilla, protuberancias, bucles internos y bucles múltiples.

- *tallo/hélice*: es una región de bases alineadas con su base complemento.
- *bucle en horquilla*: es una subestructura en donde las regiones cercanas de bases complementarias se emparejan, y se separan por una región no emparejada que permite que la secuencia se doble para formar un tallo.
- *protuberancia*: es una estructura en donde hay una región no emparejada en un solo lado de la hélice.
- *bucle interno*: es una estructura similar a la protuberancia, pero hay regiones no emparejadas en ambos lados de la hélice.
- *bucle múltiple*: región en donde se unen múltiples hélices.

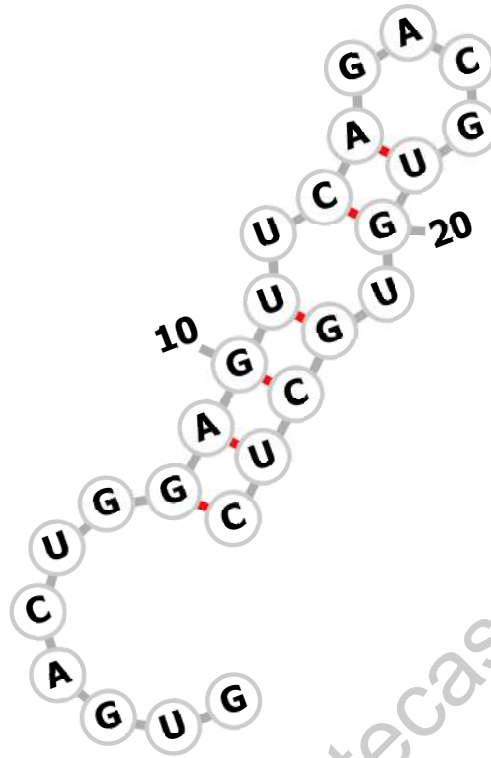


**Figura 1.3:** Representación de las posibles subestructuras que una cadena de ARN puede adoptar. Las bases se muestran como puntos y las líneas representan las conexiones entre las bases. Figura adaptada de (Böckenhauer y Bongartz, 2007).

Tomando en cuenta las reglas de apareamiento de bases y las subestructuras permitidas, considere el siguiente ejemplo:

**Ejemplo 1.1.** Sea  $r = GUGACUGGAGUUCAGACGUGUGCUC$  una cadena de ARN. La estructura secundaria está representada en la Figura 1.4.

$r: 5' \dots GUGACUGGAGUUCAGACGUGUGCUC \dots 3'$



**Figura 1.4:** Estructura secundaria correspondiente al Ejemplo 1.1.

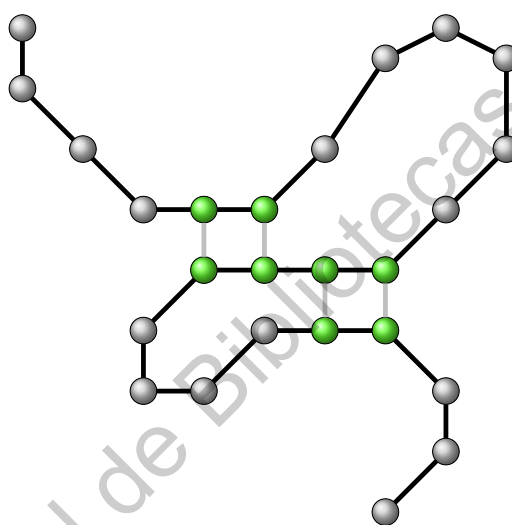
En la estructura de la Figura 1.4, se observan tres diferentes subestructuras: un bucle en horquilla (formado por las bases: G,A,C,G), un tallo (formado por los pares base: G-C, A-U, G-C, U-G, C-G, A-U) y una protuberancia (formada por el par base: U-U); además de una región de bases sin apareamiento (formada por las bases: G,U,G,A,C,U,G).

Formalmente, en la Definición 1.2 se enuncia el problema de la predicción de estructuras de ARN.

**Definición 1.2 (Predicción de la estructura secundaria).** *Dada una secuencia  $r$  conforme al alfabeto  $\sum_{ARN}$ , realizar el plegamiento de la secuencia utilizando las restricciones anteriormente listadas, generando las subestructuras permitidas (Böckenhauer y Bongartz, 2007) (Reidys, 2011) (Lorenz, Wolfinger, Tanzer, y Hofacker, 2016), para obtener la estructura secundaria  $S_r$  con la mínima energía libre o la estructura que maximice el número de pares base.*

### 1.4.2.1. Pseudonudos

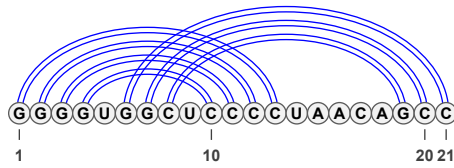
Un *pseudonudo* es una subestructura de ARN caracterizada por el emparejamiento de bases complementarias entre nucleótidos en un bucle con residuos complementarios fuera del mismo bucle (ver Figura 1.5). Algunos pseudonudos desempeñan un papel en el marco ribosomal (*ribosomal frameshifting*), mientras que otros son esenciales para la topología tridimensional y la función de muchos ARN estructurados (Aigner, Dreßen, y Steger, 2012).



**Figura 1.5:** Representación de la subestructura de un pseudonudo.

**Definición 1.3 (Pseudonudo).** Para todos los pares base de una estructura secundaria, si existen dos pares base  $(i, j)$  y  $(k, l)$  que no son continuos ( $i < j < k < l$ ), ni anidados ( $i < k < l < j$ ), pero se intersectan ( $i < k < j < l$ ); entonces la estructura secundaria contiene un pseudonudo (Tong, Cheung, Lee, y Leung, 2013).

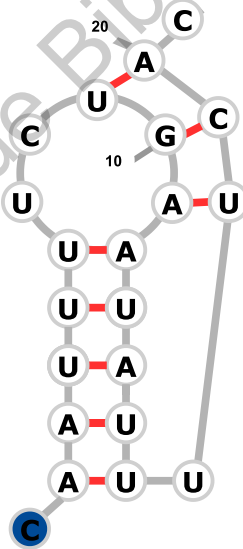
La intersección de los pares base  $(i, j)$  y  $(k, l)$  que obedece a la restricción ( $i < k < j < l$ ) se muestra en la Figura 1.6. Si consideramos el par base  $(i, j)$  con las posiciones  $(1, 13)$  y al par base  $(k, l)$  con las posiciones  $(6, 21)$ , claramente se observa esta intersección o superposición de pares base ( $1 < 6 < 13 < 21$ ).



**Figura 1.6:** Representación planar de un pseudonudo.

Existen diferentes topologías o tipos de pseudonudos. La más simple es en la que existen dos pares bases que se intersectan, aunque determinarlos no es tan sencillo. La base de datos especializada *PseudoBase++* (Taufer et al., 2009) contiene 6 tipos de pseudonudos clasificados. Por otro lado, en el trabajo realizado por (Sharma, Singh, y Chand, 2015), se tienen 11 clasificaciones de pseudonudos.

El tipo más común de pseudonudo es el *H-type* (Figura 1.7), el cual consiste en dos hélices y dos ciclos y la intersección de pares base.



**Figura 1.7:** Pseudonudo de tipo *H-type*.

#### 1.4.2.2. Mínima energía libre

Desde una perspectiva biofísica, la forma natural de enfocar la predicción de la estructura es buscar la estructura fundamental, es decir, la estructura con la mínima energía libre (*MFE*, por sus siglas en inglés). El problema de predicción de acuerdo a la MFE consiste en encontrar la estructura que posea la energía libre mínima entre todas las estructuras posibles (Aigner et al., 2012). Sin embargo, a medida que aumenta la longitud de la secuencia, el número de posibles estructuras secundarias que una molécula de ARN en particular puede adoptar crece exponencialmente, y por lo tanto es generalmente inviable enumerar todos los posibles resultados (Lorenz et al., 2016).

La estabilidad de una estructura secundaria se mide por el cambio de energía libre  $\Delta G$ . Este cambio se aproxima como la suma de las contribuciones individuales de cada subestructura que conforma la estructura secundaria (Huson, 2006). A continuación, se listan los cambios de energía que pueden ocurrir durante la construcción de la estructura secundaria.

- $\Delta G = 0$  indica equilibrio en la estructura.
- $\Delta G > 0$  indica desestabilización en la estructura.
- $\Delta G < 0$  indica que la estructura es estable.

Las reglas de cambio de energía se dividen por tipo de estructura: hélices, energías desestabilizadoras por tamaño de bucle (horquilla, protuberancia, interno), bucles internos generales, bucles internos simétricos, bucles internos asimétricos, extremos no-pareados y bucles múltiples.

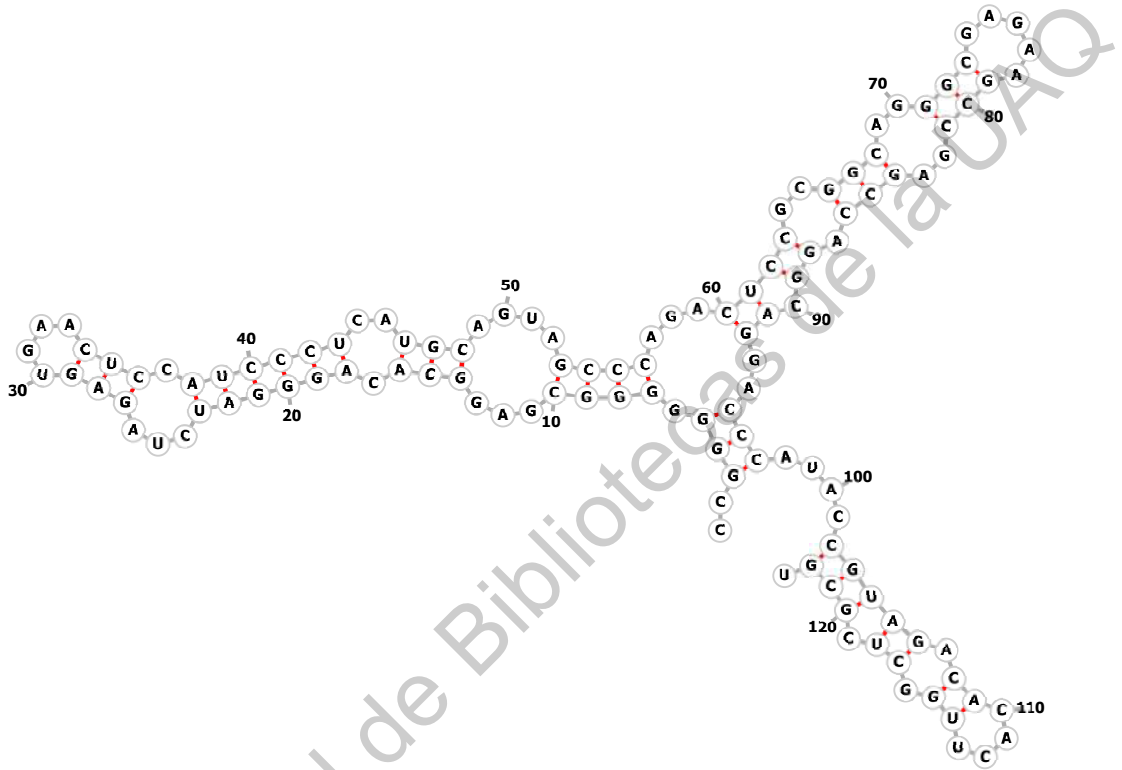
Respecto al criterio de optimización utilizado por los algoritmos del estado del arte estudiados en el presente trabajo, así como para el algoritmo genético propuesto es el de minimización de la energía libre mediante la implementación de un modelo termodinámico. En el Capítulo 2, se presenta el modelo termodinámico utilizado para la minimización de la energía libre en la predicción de estructuras secundarias del ARN.





ser accesada desde <http://nibiru.tbi.univie.ac.at/forna/forna.html>.

La Figura 1.10, es la representación gráfica de la estructura secundaria dada en la Figura 1.8.



**Figura 1.10:** Visualización de una estructura secundaria usando Forna.

## 1.5. Esquema de la Tesis

En esta tesis se presenta la investigación realizada sobre el problema de la predicción de estructuras secundarias de ARN, primero se presenta una visión general del problema planteado, así como una introducción a los conceptos biológicos y de minimización de energía libre involucrados en el plegamiento del ARN. Después, en el Capítulo 2 se especifica a detalle el modelo termodinámico utilizado para la minimización de energía libre. Posteriormente, en el Capítulo 3 se presentan los algoritmos para la predicción de estructuras secundarias de ARN. Finalmente, el Capítulo 4 se divide en dos secciones; en la primera sección se detallan las métricas de *Sensibilidad* y *Valor Predictivo Positivo* utilizadas para evaluar la eficiencia de

los algoritmos en términos de precisión de la estructura secundaria predicha con respecto a la estructura secundaria de referencia. Cabe destacar que las estructuras de referencia de las instancias utilizadas para medir la eficiencia de los algoritmos fueron obtenidas mediante técnicas experimentales. En la segunda parte del capítulo se discuten los resultados obtenidos.

Dirección General de Bibliotecas de la UAQ

# Modelo Termodinámico del Vecino

## Más Cercano

*La cosa más difícil es dormir por la noche sabiendo que hay tantas cosas urgentes que necesitan ser hechas. Existe una brecha enorme entre lo que sabemos que es posible con las máquinas de hoy en día y lo que hemos sido capaces de terminar.*

—DONALD E. KNUTH(1938 - )

La base de datos del vecino más cercano (Nearest Neighbor Database, NNDB) (Turner y Mathews, 2009), es un modelo termodinámico que asume que la estabilidad de un par base específico depende directamente de las bases vecinas. Esta base de datos proporciona los parámetros de energía que permiten la predicción de una estructura secundaria estable. La base de datos puede ser descargada desde la dirección web <https://rna.urmc.rochester.edu/NNDB/index.html>.

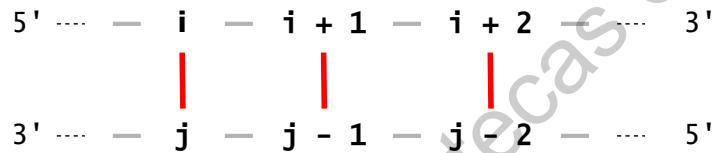
Como se mencionó anteriormente (Sub-sección 1.4.2.2) las reglas de cambio de energía se dividen por tipo de subestructura. En las siguientes secciones, se detalla como se aplican estas reglas y parámetros para determinar la energía libre de cada subestructura. Por otro lado, la Ecuación 2.1 se determina la energía libre de una estructura secundaria.

$$\Delta G = \sum \Delta G_{\text{Hélice}} + \sum \Delta G_{\text{B. Horquilla}} + \sum \Delta G_{\text{B. Interno}} + \sum \Delta G_{\text{Protuberancia}} + \sum \Delta G_{\text{B. Múltiple}} \quad (2.1)$$

## 2.1. Hélices

Una hélice es una subestructura de ARN, formada por al menos dos pares base consecutivos. Para cada par base, se tiene el par de cierre  $(i, j)$ , en el cual  $i$  y  $j$  son bases complementarias, y su par consecutivo  $(i + 1, j - 1)$ ; que también son bases complementarias.

Otra forma de ver esta subestructura, es considerándola con un arreglo o pila de pares base consecutivos. En la Figura 2.1, se muestra gráficamente esta idea.



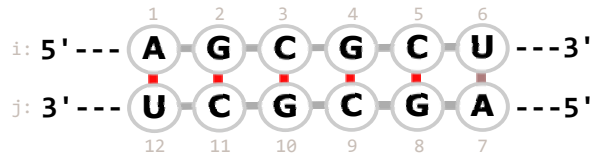
**Figura 2.1:** Representación de los pares base consecutivos de una hélice. Adaptada de (Bastolla et al., 2007)

En la Tabla 2.1, se presenta la energía libre  $\Delta G$  entre cada par base que conforma una hélice dentro de la estructura secundaria. Para las bases que no son complementarias; es decir, aquellas que no están representadas en la tabla, la energía se asume como infinita.

**Tabla 2.1:** Energía de las hélices entre cada par base. Esta tabla se construyó a partir de la información obtenida de la base de datos *Nearest Neighbor Database* (Turner y Mathews, 2009) y basándose en el proceso descrito en (Zuker et al., 1999).

	AU	CG	GC	UA	GU	UG
AU	-0.9	-2.2	-2.1	-1.1	-0.6	-1.4
CG	-2.1	-3.3	-2.4	-2.1	-1.4	-2.1
GC	-2.4	-3.4	-3.3	-2.2	-1.5	-2.5
UA	-1.3	-2.4	-2.1	-0.9	-1.0	-1.3
GU	-1.3	-2.5	-2.1	-1.4	-0.5	+1.3
UG	-1.0	-1.5	-1.4	-0.6	+0.3	-0.5

A continuación, considerando la estructura secundaria de la Figura 2.2, se ejemplifica el uso de la Tabla 2.1 para determinar la energía libre de la estructura.



**Figura 2.2:** Estructura secundaria formada por hélices de pares complementarios tipo Watson-Crick.

Los pares  $(i, j)$  que conforman la estructura (con orden de lectura de 5' a 3') son:

1. AU (1 – 12) seguido por GC (2 – 11)
2. GC (2 – 11) seguido por CG (3 – 10)
3. CG (3 – 10) seguido por GC (4 – 9)
4. GC (4 – 9) seguido por CG (5 – 8)
5. CG (5 – 8) seguido por UA (6 – 7)

Teniendo identificados el par de cierre  $(i, j)$  y el par complementario  $(i + 1, j - 1)$ , estos están representados en la Tabla 2.1 por las filas y columnas, respectivamente. Por lo tanto, la energía libre  $\Delta G$  de la estructura es la siguiente:

$$\Delta G = \Delta G_{(AU \text{ seguido por } GC)} + \Delta G_{(GC \text{ seguido por } CG)} + \Delta G_{(CG \text{ seguido por } GC)} \\ + \Delta G_{(GC \text{ seguido por } CG)} + \Delta G_{(CG \text{ seguido por } UA)}$$

$$\Delta G = -2.1 \text{ kcal/mol} - 3.4 \text{ kcal/mol} - 2.4 \text{ kcal/mol} - 3.4 \text{ kcal/mol} - 2.1 \text{ kcal/mol} \\ = -13.4 \text{ kcal/mol}$$

El Algoritmo 2.1, es el método utilizado para determinar la energía libre de una hélice de la estructura secundaria. *StackEnergyP*, *StackEnergyIntMismatch* y *TerminalMismatch* son estructuras de datos (específicamente, diccionarios) que contienen los parámetros de energía para cada par apilado con cierre  $(i, j)$ . *StackEnergyP* y *StackEnergyIntMismatch* contienen los valores mostrados en la Tabla 2.1, para los pares complementarios de tipo Watson-Crick y Wobble, respectivamente. Mientras que *TerminalMismatch* son parámetros de energía para los pares base adyacentes a los extremos de las hélices.

---

**Algoritmo 2.1:** Energía de la hélice

---

```
1 Entrada: Secuencia  $S$ , par base de cierre  $(i, j)$  y par de complemento  $(i', j')$ 
2 Salida: Energía libre  $\Delta G$ 
3 StackEnergy $(S, i, j, i', j')$ 
4    $\Delta G = \infty$ 
5    $n = \text{Length}(S)$ 
6    $\text{stack\_pair} = \text{Concat}(S_i, S_{i'}) + "/" + \text{Concat}(S_j, S_{j'})$ 
7   if  $i \geq 0$  and  $j \leq n - 1$  then
8     if  $\text{stack\_pair} \in \text{StackEnergyP}$  then
9        $\Delta G = \text{StackEnergyP}[\text{stack\_pair}]$ 
10    else if  $\text{stack\_pair} \in \text{StackEnergyIntMismatch}$  then
11       $\Delta G = \text{StackEnergyIntMismatch}[\text{stack\_pair}]$ 
12    end
13    return  $\Delta G$ 
14  end
15  if  $i == 0$  and  $j == n - 1$  then
16    if  $\text{stack\_pair} \in \text{StackEnergyP}$  then
17       $\Delta G = \text{StackEnergyP}[\text{stack\_pair}]$ 
18    else if  $\text{stack\_pair} \in \text{TerminalMismatch}$  then
19       $\Delta G = \text{TerminalMismatch}[\text{stack\_pair}]$ 
20    end
21    return  $\Delta G$ 
22  end
23  return  $\Delta G$ 
24 end
```

---

## 2.2. Bucles

Dentro del modelo termodinámico, existen penalizaciones de energía. Estas penalizaciones son cambios de energía positiva, que están asociadas a los bucles en horquilla, internos y protuberancias, dependiendo de su longitud. La longitud de estos bucles, se mide por la cantidad de bases libres entre los pares base de cierre (Andronescu, 2003). Estos valores, también son conocidos como *parámetros de iniciación*.

En la Tabla 2.2, se muestran estos parámetros para longitudes de 1 hasta 30. Para los bucles internos de tamaño 1 a 3, la energía se considera como infinita; lo mismo que para bucles en horquilla de tamaño 1 y 2. Sin embargo, para bucles de longitud mayor a 30 es posible extrapolar los valores. La extrapolación de energía se obtiene mediante la Ecuación 2.2 (Turner y Mathews, 2009).

$$\Delta G_{(n>30)} = \Delta G_{\text{Iniciación}}(30) + 1.75 \times R \times T \times \ln(n/30) \quad (2.2)$$

Donde,  $n$  es el tamaño del bucle deseado,  $\Delta G_{\text{Iniciación}}$  es la energía del bucle (interno, protuberancia, horquilla) de longitud 30,  $R$  es la constante del gas ( $1.9872 \times 10^{-3}$ ) y  $T$  es la temperatura (310.15 K).

**Tabla 2.2:** Energías de desestabilización por tamaño del bucle (Turner y Mathews, 2009).

Tamaño	Interno	Protuberancia	Horquilla
1	.	3.8	.
2	.	2.8	.
3	.	3.2	5.4
4	1.1	3.6	5.6
5	2.0	4.0	5.7
6	2.0	4.4	5.4
7	2.1	4.6	6.0
8	2.3	4.7	5.5
9	2.4	4.8	6.4
10	2.5	4.9	6.5
11	2.6	5.0	6.6
12	2.7	5.1	6.7
13	2.8	5.2	6.8
14	2.9	5.3	6.9
15	2.9	5.4	6.9
16	3.0	5.4	7.0
17	3.1	5.5	7.1
18	3.1	5.5	7.1
19	3.2	5.6	7.2
20	3.3	5.7	7.2
21	3.3	5.7	7.3
22	3.4	5.8	7.3
23	3.4	5.8	7.4
24	3.5	5.8	7.4
25	3.5	5.9	7.5
26	3.5	5.9	7.5
27	3.6	6.0	7.5
28	3.6	6.0	7.6
29	3.7	6.0	7.6
30	3.7	6.1	7.7

En las Subsecciones 2.2.1, 2.2.2 y 2.2.3, se detalla como determinar la energía libre para cada tipo de bucle, a partir del parámetro de inicialización.

### 2.2.1. Bucle en horquilla

El modelo de NNDB, contempla reglas para determinar la energía del bucle en horquilla dependiendo de su tamaño: bucles de tamaño 4 o más, bucles de tamaño 3 y bucles especiales (para los cuales, la energía se determino por experimentación).

#### Horquilla de 4 o más nucleótidos

Las horquillas de tamaño 4 se consideran especiales, ya que se cree que son particularmente estables. Por ello, el modelo termodinámico incluye bonificaciones en función del par base de cierre para estos bucles especiales (Andronescu, 2003) (Turner y Mathews, 2009). En la Tabla 2.3, se muestran estas horquillas y su correspondiente energía dentro del modelo NNDB.

**Tabla 2.3:** Energías para horquillas especiales de tamaño 4, la primer y última base representan el par base de cierre.

Secuencia	Energía kcal/mol	Secuencia	Energía kcal/mol	Secuencia	Energía kcal/mol
GGGGAC	-3.0	CUACGG	-2.5	GGGAAC	-1.5
GGUGAC	-3.0	GGCAAC	-2.5	UGAAAA	-1.5
CGAAAG	-3.0	CGCGAG	-2.5	AGCAAU	-1.5
GGAGAC	-3.0	UGAGAG	-2.5	AGUAAU	-1.5
CGCAAG	-3.0	CGAGAG	-2.0	CGGGAG	-1.5
GGAAAC	-3.0	AGAAAU	-2.0	AGUGAU	-1.5
CGGAAG	-3.0	CGUAAG	-2.0	GGCGAC	-1.5
CUUCGG	-3.0	CUAACG	-2.0	GGGAGC	-1.5
CGUGAG	-3.0	UGAAAG	-2.0	GUGAAC	-1.5
CGAAGG	-2.5	GGAAGC	-1.5	UGGAAA	-1.5

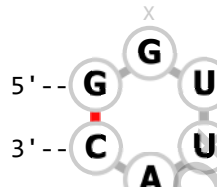
Adicionalmente, para horquillas de longitud mayor o igual a 4 se agrega una bonificación por los desajustes terminales o pares base adyacentes a los extremos de la hélice. En la Figura 2.3, se muestran los ajustes de energía terminales, cuando el par base de cierre es (G, C).



		(Y)		
(X)	A	C	G	U
		5' ==> 3'		
		GX		
		CY		
		3' <== 5'		
A	-1.1	-1.5	-1.3	-2.1
C	-1.1	-0.7	-2.4	-0.5
G	-2.4	-2.9	-1.4	-1.2
U	-1.9	-1.0	-2.2	-1.5

**Figura 2.3:** Ajustes de energía terminales para el par de cierre (G, C).

En la Figura 2.4, se muestra un bucle en horquilla de tamaño 4 cerrado por el par base (G, C).



**Figura 2.4:** Estructura secundaria formada por una horquilla.

Mediante el parámetro de iniciación (Tabla 2.2) y el parámetro por ajuste terminal (Figura 2.3), la energía de la estructura mostrada en la figura anterior, se calcula como:

$$\begin{aligned} \Delta G_{Horquilla} &= \Delta G_{Iniciación}(4) + \Delta G_{Ajuste\ terminal}(GC-G_XA_Y) \\ &= 5.6\ kcal/mol - 2.4\ kcal/mol = 3.2\ kcal/mol \end{aligned}$$

### Horquilla de 3 nucleótidos

Para bucles en horquilla de 3 nucleótidos, el cambio de energía libre se estima usando la Ecuación 2.3 (Turner y Mathews, 2009).

$$\Delta G_{Horquilla}(3) = \Delta G_{Iniciación}(3) + \Delta G_{Penalización}(C_3) \quad (2.3)$$

Donde,  $C_3$  es igual a 1.4 kcal/mol.

## Horquillas especiales

Existen bucles en horquilla de longitud de 3, 4 y 6 nucleótidos (nt.) que tienen desajustes por el modelo de energía libre, a los cuales se les asigna una penalización basada en datos experimentales. En la Tabla 2.4, se muestra la secuencia, longitud y energía de las horquillas especiales contempladas en el modelo NNDB.

**Tabla 2.4:** Energías para horquillas especiales, la primer y última base representan el par base de cierre.

Secuencia	Longitud nt.	Energía kcal/mol	Secuencia	Longitud nt.	Energía kcal/mol
CAACG	3	6.8	CCGCGG	4	3.6
GUUAC	3	6.9	CCUCGG	4	2.5
CUACGG	4	2.8	CUAAGG	4	3.6
CUCCGG	4	2.7	CUCAGG	4	3.7
CUUCGG	4	3.7	CUUAGG	4	3.5
CUUUGG	4	3.7	CUGCGG	4	2.8
CCAAGG	4	3.3	CAACGG	4	5.5
CCCAGG	4	3.4	ACAGUGCU	6	2.9
CCGAGG	4	3.5	ACAGUGAU	6	3.6
CCUAGG	4	3.7	ACAGUGUU	6	1.8
CCACGG	4	3.7	ACAGUACU	6	2.8

## Algoritmo General

A continuación, el Algoritmo 2.2 define la forma en que se determina la energía libre para cualquier bucle en horquilla. *Hairpin3*, *Hairpin4* y *TerminalMismatchHairpin* son estructuras de datos que contienen los parámetros de energía para cada par base apilado con cierre en  $(i, j)$ . *Hairpin3* contiene las horquillas de longitud 3 mostradas en la Tabla 2.4; *Hairpin4* son las horquillas de la Tabla 2.3; y *TerminalMismatchHairpin* almacena los parámetros de energía por ajuste terminal. Los parámetros completos que conforman los ajustes terminales para las horquillas se encuentran en el archivo `tstackh.dat`<sup>1</sup>.

<sup>1</sup><https://github.com/payalae116/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/tstackh.dat>

---

**Algoritmo 2.2:** Energía del bucle en horquilla

---

1 **Entrada:** Secuencia  $S$ , par base de cierre  $(i, j)$ , const. gas  $R$ , temperatura  $T$   
2 **Salida:** Energía libre  $\Delta G$

3 **HairpinEnergy** $(S, i, j, R, T)$

```
4    $\Delta G = \infty$ 
5    $l = j - i - 1$ 
6    $hairpin = S_{i+1 \dots j-1}$ 
7    $closing\_pair = Concat(S_i, S_{i+1}) + "/" + Concat(S_j, S_{j-1})$ 
8   if  $l < 3$  then
9     | return  $\Delta G$ 
10  end
11  if  $l \leq 30$  then
12    |  $\Delta G = \Delta G_{Initiation}(l)$ 
13  else
14    |  $\Delta G = \Delta G_{Initiation}(30) + 1.75 \times R \times T \times \ln(l/30)$ 
15  end
16  if  $l == 3$  then
17    |  $\Delta G += Non\_GC\_Penalty$ 
18    | if  $hairpin \in Hairpin3$  then
19      |  $\Delta G += Hairpin3[hairpin]$ 
20    | end
21  else
22    | if  $closing\_pair \in TerminalMismatchHairpin$  then
23      |  $\Delta G += TerminalMismatchHairpin[closing\_pair]$ 
24    | end
25    | if  $l == 4$  and  $hairpin \in Hairpin4$  then
26      |  $\Delta G += Hairpin4[hairpin]$ 
27    | end
28  end
29  if  $S_{i-2} == S_{i-1} == S_i == G$  and  $S_j == U$  then
30    |  $\Delta G += Bonus\_GGG$ 
31  else if  $S_{i+1} \dots S_{j-1} == C$  then
32    | if  $l == 3$  then
33      |  $\Delta G += C\_Hairpin3$ 
34    | else
35      |  $\Delta G += C\_Hairpin\_Intercept + C\_Hairpin\_Slope \times l$ 
36    | end
37  end
38  return  $\Delta G$ 
39 end
```

---

En la Tabla 2.5, se muestra el resto de los parámetros de energía utilizados por el algoritmo. Estos parámetros, se aplican para horquillas clasificadas como especiales de acuerdo con las bases libres que la forman y la longitud.

**Tabla 2.5:** Parámetros para horquillas especiales.

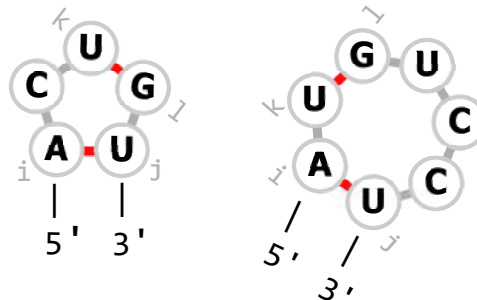
Parámetro	Energía kcal/mol
Bonus_GGG	-2.2
C_Hairpin3	1.4
C_Hairpin_Intercept	1.6
C_Hairpin_Slope	0.30

### 2.2.2. Protuberancia

Una protuberancia es un tipo de bucle interno, que ocurre cuando la cadena de ARN se fuerza a formar un bucle corto, debido a que una o más bases no encuentran su base complementaria durante el plegamiento (“RNA bulges as architectural and recognition motifs”, 2000). Es decir, son regiones de bases no apareadas que ocurren en un solo lado de la estructura.

Los bucles internos, tienen dos pares base de cierre  $(i, j)$  y  $(k, l)$  para  $i < k < j < l$ . En el caso de las protuberancias, los pares de cierre son vecinos sin un solo nucleótido en el medio. Esto es si  $j = l + 1$  o  $k = i + 1$ ; lo cual forma una protuberancia del lado izquierdo o derecho, respectivamente.

En la Figura 2.5, se muestran dos protuberancias por la izquierda (de longitud 1) y derecha (de longitud 3), respectivamente.



**Figura 2.5:** Estructura secundaria de una protuberancia.

Para determinar la energía de un protuberancia, se consideran dos posibles casos:

### Longitud de 1

$$\Delta G_{Protuberancia}(n = 1) = \Delta G_{Iniciación}(n) + \Delta G_{Hélice ((i,j) \text{ seguido por } (k,l))}$$

### Longitud mayor a 1

$$\Delta G_{Protuberancia}(n > 1) = \Delta G_{Iniciación}(n) + \Delta G_{Penalización(GU)}$$

La penalización por GU, se asigna cuando alguno o ambos pares base de cierre, forman el par base GU. En el Algoritmo 2.3, se muestra la forma de determinar la energía libre de una protuberancia.

---

#### Algoritmo 2.3: Energía de la protuberancia

---

```

1 Entrada: Secuencia  $S$ , los pares base de cierre  $(i, j)$  y  $(k, l)$ , constante del
   gas  $R$ , temperatura  $T$ 
2 Salida: Energía libre  $\Delta G$ 
3 BulgeEnergy ( $S, i, j, k, l$ )
4    $\Delta G = \infty$ 
5    $l = \max(k - i - 1, j - l - 1)$ 
6    $closing\_left = \text{Concat}(S_i, S_j)$ 
7    $closing\_right = \text{Concat}(S_k, S_l)$ 
8   if  $l \leq 0$  then
9     | return  $\Delta G$ 
10  end
11  if  $l \leq 30$  then
12    |  $\Delta G = \Delta G_{Iniciación}(l)$ 
13  else
14    |  $\Delta G = \Delta G_{Iniciación}(30) + 1.75 \times R \times T \times \ln(l/30)$ 
15  end
16  if  $l == 1$  then
17    |  $\Delta G += \text{StackEnergy}(S, i, j, k, l)$ 
18  else
19    |  $\Delta G += \text{UG\_Penalty}(closing\_left) + \text{UG\_Penalty}(closing\_right)$ 
20  end
21  return  $\Delta G$ 
22 end

```

---

### 2.2.3. Bucle interno

Los bucles internos son regiones que se forman cuando las bases que no coinciden (no complementarias), se encuentran una frente a la otra y están unidas en ambos lados de la hélice por los pares de cierre  $(i, j)$  y  $(k, l)$ .

Para bucles internos de tamaño  $1 \times 1$ ,  $1 \times 2$  y  $2 \times 2$ ; el modelo NNDB cuanta con los parámetros de energía ya tabulados, valores que se determinaron experimentalmente.

#### Bucle Interno $1 \times 1$

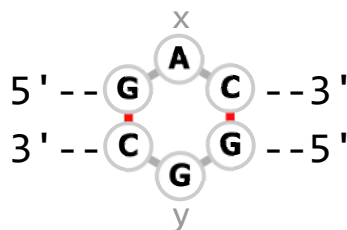
Este bucle interno es simétrico (tiene una base libre en cada lado de la estructura). El modelo NNDB contiene 576 parámetros de energía para este bucle en particular. En la Figura 2.6, se muestran los parámetros de energía cuando los pares de cierre son  $(G, C)$  y  $(G, C)$ , para  $(i, j)$  y  $(k, l)$  respectivamente, y con orden de lectura de  $5'$  a  $3'$ . El resto de los parámetros de energía se encuentran en el archivo `int11.dat`<sup>2</sup>.

		(Y)		
	A	C	G	U
(X)		5' ==> 3'		
		x		
		G	C	
		C	G	
		y		
		3' <== 5'		
	0.8	0.4	0.4	0.4
	0.4	0.4	0.4	0.4
	0.4	0.4	-2.1	0.4
	0.4	0.4	0.4	-0.7

**Figura 2.6:** Parámetros de energía para bucle interno  $1 \times 1$  con cierre  $(G, C)$  y  $(G, C)$ .

En la Figura 2.7, se muestra un bucle interno simétrico de tamaño  $1 \times 1$ , cuya energía es fácil de ver en los parámetros de la figura anterior; dicha energía es de  $0.4 \text{ kcal/mol}$ .

<sup>2</sup><https://github.com/payalae116/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/int11.dat>



**Figura 2.7:** Bucle interno simétrico de tamaño  $1 \times 1$ .

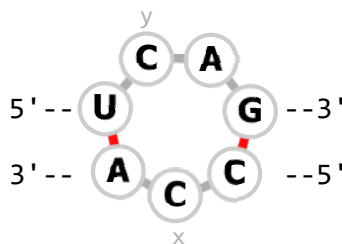
### Bucle Interno $1 \times 2$

Es bucle asimétrico de tamaño 3, tiene una base libre en un lado de la estructura y dos en el otro lado. Al igual que el bucle  $1 \times 1$ , los parámetros de energía ya están tabulados dentro del modelo. En la Figura 2.8 se muestran los parámetros de energía con pares de cierre (U, A) y (C, G). En el archivo int21.dat<sup>3</sup> se encuentra el resto de los parámetros de energía.

(X)	A	C	(Y) G	U
			5' ==> 3'	
			x	
			U G	
			A C	
			yA	
			3' <== 5'	
	3.2	3.0	2.4	4.8
	3.1	3.0	4.8	3.0
	2.5	4.8	1.6	4.8
	4.8	4.8	4.8	4.8

**Figura 2.8:** Parámetros de energía para bucle interno  $1 \times 2$  con cierre (U, A) y (C, G).

La Figura 2.9, muestra un bucle interno asimétrico de tamaño  $1 \times 2$  para el cual la energía es de  $3.0 \text{ kcal/mol}$  de acuerdo a los parámetros de la Figura 2.8.



**Figura 2.9:** Bucle interno asimétrico de tamaño  $1 \times 2$ .

<sup>3</sup><https://github.com/payalae116/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/int21.dat>

## Bucle Interno $2 \times 2$

Este es otro de los casos especiales de bucles internos, es un bucle simétrico de tamaño 4 (dos bases libres en cada lado). Al igual que los bucles anteriores, su energía está tabulada dentro del modelo. En la Figura 2.10, se muestran los parámetros de energía cuando los pares de cierre son (C, G) y (C, G). El resto de los parámetros de energía para bucles internos  $2 \times 2$ , se encuentran en el archivo int22.dat<sup>4</sup>.

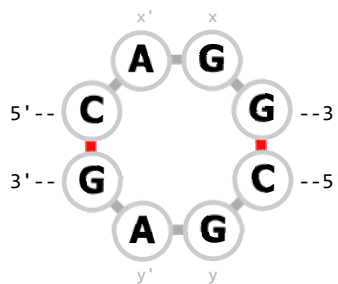
A	A	A	A	C	C	C	C	G	G	G	G	U	U	U	U
A	C	G	U	A	C	G	U	A	C	G	U	A	C	G	U
							5'	→	3'						
						C	∨	∨	G						
						G	∧	∧	C						
							3'	←	5'						
1.3	1.2	0.3	2.0	1.6	2.1	2.0	1.9	0.3	2.0	1.0	-0.4	2.0	1.9	1.1	1.4
1.6	1.5	0.6	2.0	2.0	1.8	2.0	1.7	0.6	2.0	1.4	-1.1	2.0	1.7	0.4	0.8
0.3	0.2	-0.7	2.0	0.6	1.5	2.0	1.3	-0.7	2.0	0.0	-0.6	2.0	1.3	0.9	1.3
2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
1.2	1.1	0.2	2.0	1.5	1.4	2.0	1.2	0.2	2.0	0.9	-1.5	2.0	1.2	0.0	0.3
2.1	1.4	1.5	2.0	1.8	1.7	2.0	1.5	1.5	2.0	1.8	-0.2	2.0	1.5	1.3	0.6
2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
1.9	1.2	1.3	2.0	1.7	1.5	2.0	1.4	1.3	2.0	1.7	-0.4	2.0	1.4	1.1	0.5
0.3	0.2	-0.7	2.0	0.6	1.5	2.0	1.3	-0.7	2.0	0.0	-0.6	2.0	1.3	0.9	1.3
2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
1.0	0.9	0.0	2.0	1.4	1.8	2.0	1.7	0.0	2.0	0.8	-0.7	2.0	1.7	0.9	1.2
1.1	0.0	0.9	2.0	0.4	1.3	2.0	1.1	0.9	2.0	0.9	-2.6	2.0	1.1	-1.1	1.1
2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
1.9	1.2	1.3	2.0	1.7	1.5	2.0	1.4	1.3	2.0	1.7	-0.4	2.0	1.4	1.1	0.5
-0.4	-1.5	-0.6	2.0	-1.1	-0.2	2.0	-0.4	-0.6	2.0	-0.7	-4.2	2.0	-0.4	-2.6	-0.5
1.4	0.3	1.3	2.0	0.8	0.6	2.0	0.5	1.3	2.0	1.2	-0.5	2.0	0.5	1.1	-0.4

**Figura 2.10:** Parámetros de energía para bucle interno  $2 \times 2$  con cierre (C, G) y (C, G).

La Figura 2.11, muestra un bucle interno simétrico de tamaño  $2 \times 2$ , de acuerdo con los parámetros de la Figura 2.10, su energía es de  $1.0 \text{ kcal/mol}$ . El par  $x, y$  corresponde con la columna GG y  $x', y'$  con la fila AA.

<sup>4</sup><https://github.com/payalae116/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/int22.dat>





**Figura 2.11:** Bucle interno simétrico de tamaño  $2 \times 2$ .

### Bucles Internos Generales

La energía libre para bucles internos generales, es decir, de longitud mayor a 4 o de tamaño 4 asimétrico; se determina de la siguiente forma:

$$\Delta G_{Interno}(n) = \Delta G_{Iniciación}(n) + \Delta G_{Asimetría} \times |n_1 - n_2| + \Delta G_{Ajuste\ terminal}(i, j) + \Delta G_{Ajuste\ terminal}(k, l)$$

El Algoritmo 2.4, muestra la forma de determinar la energía libre para cualquier bucle interno. *Internal\_2*, *Internal\_3* y *Internal\_4* son funciones que determinan la energía libre para bucles de tamaño  $1 \times 1$ ,  $1 \times 2$  y  $2 \times 2$  respectivamente, y de acuerdo a las reglas establecidas previamente. Por otro lado, *Internal\_n* es una función que pondera la energía libre para los ajustes terminales entre el par base de cierre y las bases libres vecinas que conforman el bucle; los parámetros de energía para estos ajustes terminales se encuentran en el archivo *tstacki.dat*<sup>5</sup>.  $\Delta G_{Initiation}$  corresponde a la penalización de energía por el tamaño del bucle. Finalmente, *InternalLoopAsymmetric* es la función que penaliza los bucles asimétricos, utilizando la siguiente función (Andronescu, 2003):

$$InternalLoopAsymmetric(l_1, l_2) = \min \left\{ \begin{array}{l} Asym - Max \\ |l_1 - l_2| \times Asym - Par[\min(2, \min(l_1, l_2)) - 1] \end{array} \right.$$

Donde, *Asym - Max* es igual a 3.0 kcal/mol y *Asym - Par* es un arreglo con los valores de energía [0.5, 0.5, 0.5, 0.5].

<sup>5</sup><https://github.com/payalael16/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/tstacki.dat>

---

**Algoritmo 2.4:** Energía del bucle interno

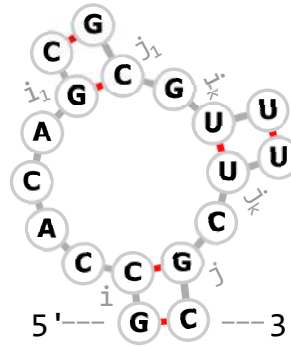
---

```
1 Entrada: Secuencia  $S$ , los pares base de cierre  $(i, j)$  y  $(k, l)$ , constante del
   gas  $R$ , temperatura  $T$ 
2 Salida: Energía libre  $\Delta G$ 
3 InternalLoopEnergy ( $S, i, j, k, l, R, T$ )
4    $\Delta G = \infty$ 
5    $interior\_left = k - i - 1$ 
6    $interior\_right = j - l - 1$ 
7    $interior\_len = interior\_left + interior\_right$ 
8   if  $interior\_left < 1$  or  $interior\_right < 1$  then
9     | return  $\Delta G$ 
10  end
11  if  $l \leq 0$  then
12    | return  $\Delta G$ 
13  end
14  if  $interior\_left == 1$  and  $interior\_right == 1$  then
15    | return  $Internal\_2(i, j, k, l, S)$ 
16  else if  $interior\_left == 1$  and  $interior\_right == 2$  then
17    | return  $Internal\_3(i, j, k, l, S)$ 
18  else if  $interior\_left == 2$  and  $interior\_right == 1$  then
19    | return  $Internal\_3(l, k, j, i, S)$ 
20  else if  $interior\_left == 2$  and  $interior\_right == 2$  then
21    | return  $Internal\_4(i, j, k, l, S)$ 
22  else
23    | if  $interior\_len \leq 30$  then
24      |  $\Delta G = \Delta G_{Initiation}(interior\_len)$ 
25    | else
26      |  $\Delta G = \Delta G_{Initiation}(30) + 1.75 \times R \times T \times \ln(interior\_len/30)$ 
27    | end
28    |  $\Delta G += Internal\_n(i, j, S)$ 
29    |  $\Delta G += Internal\_n(k, l, S)$ 
30    |  $\Delta G += InternalLoopAsymmetric(interior\_left, interior\_right)$ 
31  end
32  return  $\Delta G$ 
33 end
```

---

## 2.2.4. Bucle múltiple

Los bucles múltiples (también, llamados uniones) son bucles de los cuales se desprenden  $k$  número de ramificaciones o hélices (dos o más, Figura 2.12). Con cada ramificación, se tienen los pares de cierre  $(i, j)$ ,  $(i_1, j_1)$ ,  $\dots$ ,  $(i_k, j_k)$ .



**Figura 2.12:** Bucle múltiple de 3 ramificaciones y 6 bases libres. Fragmento de la secuencia CRW\_00548, obtenida de *RNA STRAND* (Andronescu et al., 2008).

La función para determinar la energía libre de los bucles múltiples es la siguiente (Andronescu, 2003) (Kravchenko, 2009):

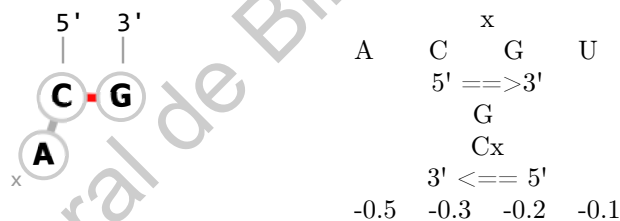
$$\begin{aligned}
 M(S, i, j, i_1, i_2, \dots, i_k, j_k) = & \\
 & Multi_a + \\
 & Multi_b \times (k + 1) + \\
 & Multi_c \times \left[ (i_i - i - 1) + \sum_{h=1}^{k-1} (i_{h+1} - j_h - 1) + (j - j_k - 1) \right] + \\
 & Penalización\_No\_GC(s_i, s_j) + \sum_{h=1}^k Penalización\_No\_GC(s_{i_h}, s_{j_h}) + \\
 & D(S, j, i, i_1, j_1) + \sum_{h=1}^{k-1} D(S, i_h, j_h, i_{h+1}, j_{h+1}) + D(S, i_k, j_k, j, i)
 \end{aligned}$$

Donde,  $Multi_a$  es una penalización por iniciar un bucle múltiple (3.40 kcal/mol),  $Multi_b$  es la penalización por iniciar una nueva hélice (0.4 kcal/mol) y  $Multi_c$  es la penalización por cada base libre localizada entre las hélices que se desprenden del bucle (0 kcal/mol).  $Penalización\_No\_GC$ , es la penalización para las hélices que cierran con par base diferente de  $\{(C, G), (G, C)\}$  (0.5 kcal/mol).

Finalmente,  $D$  es una función que determina la energía libre de las bases colgantes (bases libres) que son vecinas directas de una hélice.

$$D(S, i_1, j_1, i_2, j_2) = \begin{cases} D_{3'}(s_{j_1}, s_{i_1}, s_{j_1+1}) + D_{5'}(s_{j_2}, s_{i_2}, s_{i_2-1}) & \forall i_1 + 1 < i_2 - 1 \\ \min(D_{3'}(s_{j_1}, s_{i_1}, s_{j_1+1}), D_{5'}(s_{j_2}, s_{i_2}, s_{i_2-1})) & \forall i_1 + 1 = i_2 - 1 \\ 0 & \forall i_1 + 1 > i_2 - 1 \end{cases}$$

$D_{5'}$  y  $D_{3'}$ , son funciones que determinan la energía de las bases libres vecinas a una hélice, cuando la base libre es cercana al lado 5' y 3' de la estructura. En la Figura 2.13, se muestra un ejemplo de una base colgante en  $D_{5'}$ , para esta estructura se observa que la contribución de energía por parte de la base libre es de -0.5 kcal/mol. Los parámetros de energía para todas las bases colgantes, se encuentran en el archivo dangle.dat<sup>6</sup>.



**Figura 2.13:** Base libre adyacente a una hélice (izquierda) y tabla de energía (derecha) para el par de cierre (C, G).

Retomando el bucle múltiple de la Figura 2.12, y desarrollando la función  $M$  establecida anteriormente, la energía libre para el bucle es de 3.1 kcal/mol. Primero, es necesario identificar los pares base de cierre de cada ramificación (Figura 2.14).

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
G	C	C	A	C	A	G	C	G	C	G	U	U	U	U	C	G	C
(	(	.	.	.	.	(	(	)	)	.	(	(	)	)	.	)	)

**Figura 2.14:** Pares base de cierre del bucle múltiple. Los pares base son: (1, 16), (6, 9) y (11, 14).

<sup>6</sup><https://github.com/payalael16/RNASecStrucPredict/blob/main/NNDBEnergyParser/nndb/dangle.dat>

Luego, sustituimos los valores conocidos, y desarrollamos las sumatorias:

$$\begin{aligned}
M(S, 1, 16, 6, 9, 11, 14) = & \\
& Multi_a^{\nearrow 3.4} + \\
& Multi_b^{\nearrow 0.4} \times (k^{\nearrow 2} + 1) + \\
& Multi_c^{\nearrow 0} \times \left[ (6 - 1 - 1) + \sum_{h=1}^1 (11 - 9 - 1) + (16 - 14 - 1) \right]^{\nearrow 6} + \\
& Penalización\_No\_GC(C_{s_1}, G_{s_{16}})^{\nearrow 0} + \sum_{h=1}^k \left( Penalización\_No\_GC(s_{ih}, s_{jh}) + \right. \\
& \left. D_3^{\nearrow (D_3, (9,6,10) + D_5, (14,11,10))}^{-1.7} + D_3^{\nearrow (D_3, (14,11,15) + D_5, (1,16,15))}^{-0.3} \right. \\
& \left. D(S, 16, 1, 6, 9)^{\nearrow 0} + \sum_{h=1}^1 D(S, 6, 9, 11, 14) + D(S, 11, 14, 16, 1) \right)
\end{aligned}$$

Finalmente obtenemos:

$$M(S, 1, 16, 6, 9, 11, 14) = 3.4 + 0.4 \times 3 + 0 \times 6 + 0 + 0.5 + 0 - 1.7 - 0.3 = 3.1$$

# Metodología

*Si no trabajas en problemas importantes, es probable que no hagas un trabajo importante.*

—RICHARD HAMMING (1915 - 1998)

## 3.1. Algoritmos Exhaustivos

### 3.1.1. Algoritmo de Nussinov

El algoritmo de Nussinov (Nussinov y Jacobson, 1980) es un algoritmo de programación dinámica que calcula el máximo número de pares base en el plegamiento de una molécula de ARN. En la actualidad al tratarse de un algoritmo clásico de predicción de estructuras secundarias, sirve como base para algoritmos más sofisticados.

Este algoritmo está diseñado para evaluar la contribución individual de cada par base a la estructura secundaria de una cadena. Para maximizar el número de emparejamientos utiliza un conjunto de reglas de plegado. La estabilidad de los pares G-C se considera igual a los pares A-U y G-U. Las contribuciones debidas al apilamiento (stacking) se ignoran, al igual que los efectos de desestabilización de los bucles. Bajo estas condiciones, el problema de plegar una secuencia de nucleótidos en una estructura con la mínima energía libre se convierte en un problema más simple de encontrar una estructura con el número máximo de pares de bases; bajo la hipótesis de que entre más emparejamientos existan entre bases de la cadena de ARN, menor será la energía libre de la molécula (Nussinov y Jacobson, 1980).

Los cálculos del algoritmo se realizan de manera recursiva. Si se considera una secuencia de  $n$  nucleótidos  $S_1 \dots S_n$ , para obtener la estructura con el número máximo de pares base, el algoritmo agrega un 1 si  $S_i$  y  $S_j$  son pares base complementarios y 0 en caso contrario (Función 3.2). El puntaje óptimo  $S(i, j)$  de una subsecuencia desde la posición  $i$  a la posición  $j$ , puede ser definida recursivamente mediante la puntuación de subsecuencias más pequeñas (Función 3.1)<sup>1</sup>. Partiendo de esta idea, sólo hay cuatro posibles formas de obtener la mejor estructura para  $i, j$  desde subsecuencias más pequeñas.

$$M(i, j) = \max \left\{ \begin{array}{l} M(i + 1, j) \\ M(i, j - 1) \\ M(i + 1, j - 1) + \delta(r_i, r_j) \\ \max_{i < k < j} \{ M(i, k) + M(k + 1, j) \} \end{array} \right. \quad (3.1)$$

$$\delta(r_i, r_j) = \begin{cases} 1 & \text{si } (i, j) = (A, U), (C, G), (G, U) \\ 0 & \text{en otro caso} \end{cases} \quad (3.2)$$

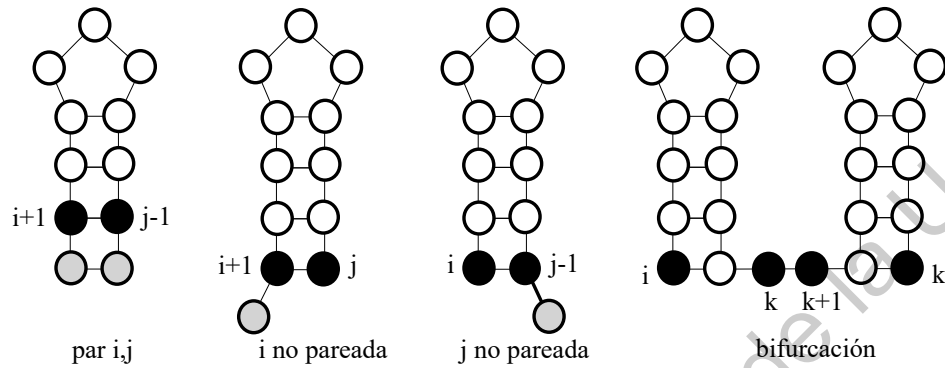
Las cuatro formas posibles de construir una estructura de pares de bases de acuerdo a la función de Nussivov son las siguientes (Sperschneider, 2008) (Zhao y Sahni, 2016):

- 1) Agregar  $i$  sin aparear a la mejor estructura  $i + 1, j$ .
- 2) Agregar  $j$  sin aparear a la mejor estructura  $i, j - 1$ .
- 3) Agregar el par  $i, j$  sobre la mejor estructura  $i + 1, j - 1$ .
- 4) Combinar las dos estructuras óptimas  $i, k$  y  $k + 1, j$ .

---

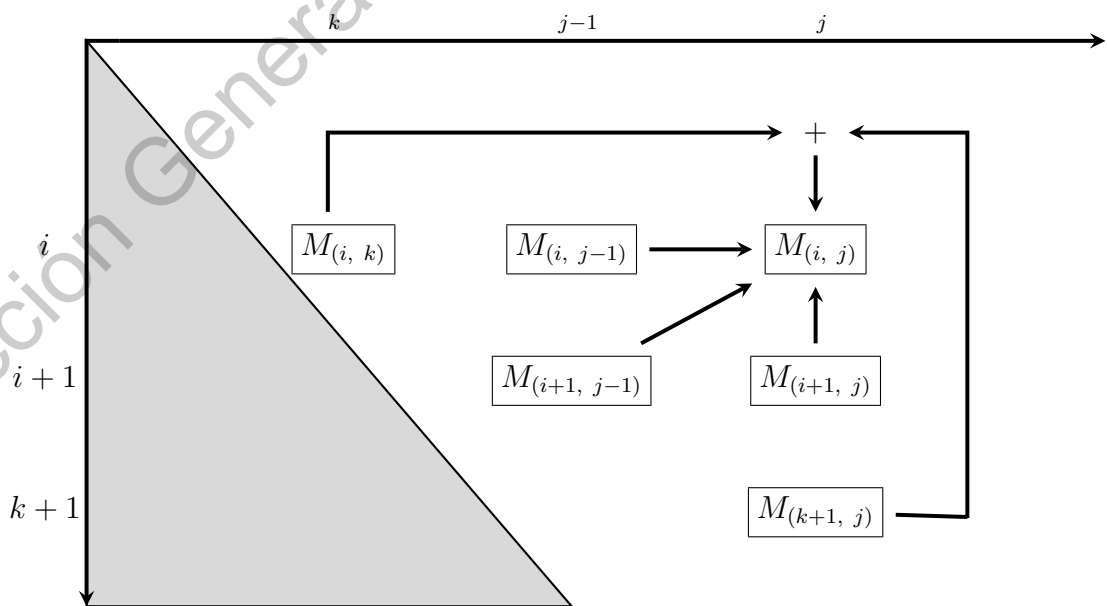
<sup>1</sup>Función recursiva de Nussivov (Böckenhauer y Bongartz, 2007).

Los diferentes casos de recurrencia en el algoritmo de Nussinov se muestran en la Figura 3.1:



**Figura 3.1:** Los cuatro posibles casos para la función recursiva de Nussinov (Durbin et al., 1998).

La matriz será llenada a lo largo de las diagonales y la solución será recuperada a través de un paso de rastreo. En la Figura 3.2, se visualiza cómo se calcula recursivamente una entrada de la matriz, una forma práctica de mejorar la eficiencia del algoritmo, es calculando solo la matriz triangular superior. Esta ilustración es una representación gráfica de la Función 3.1.



**Figura 3.2:** Cálculo del valor  $i, j$  en la matriz del algoritmo Nussinov (Sperschneider, 2008).



A continuación, se muestra el algoritmo de Nussinov (Algoritmo 3.1) adaptado de (Böckenhauer y Bongartz, 2007). El cual requiere  $O(n^2)$  pasos para calcular la matriz triangular superior, para una matriz de  $n \times n$  para los casos (1), (2) y (3). Adicionalmente, deben calcularse los  $n$  posibles valores del parámetro  $k$  en el caso (4); esto requiere  $O(n)$ , por lo tanto, el tiempo de ejecución del algoritmo es  $O(n^3)$ .

---

**Algoritmo 3.1:** Nussinov

---

```

1 Entrada: Una cadena  $r = \{r_0, \dots, r_{n-1}\}$  donde  $r_i \in \Sigma_{ARN}$  y un entero  $l$ 
   que representa el tamaño mínimo del bucle.
2 Salida: La estructura secundaria  $S_r$  en formato DotBracket.
3 Nussinov ( $r, l$ )
4    $n = \text{Length}(r)$ 
5    $\text{SecStruct} = \emptyset$ 
6   for  $k = 0$  to  $l$  do
7     for  $i = 0$  to  $n - k$  do
8        $j = i + k$ 
9        $M(i, j) = 0$ 
10    end
11  end
12  for  $k = l$  to  $n$  do
13    for  $i = 0$  to  $n - k$  do
14       $j = i + k$ 
15       $M(i, j) = \max \left\{ \begin{array}{l} M(i + 1, j) \\ M(i, j - 1) \\ M(i + 1, j - 1) + \delta(r_i, r_j) \\ \max_{i < k < j} \{ M(i, k) + M(k + 1, j) \} \end{array} \right.$ 
16    end
17  end
18  Call NussinovTraceback( $0, n - 1, M, \text{SecStruct}$ )
19   $\text{dot\_bracket} = \{ '!', \dots, n \}$ 
20  foreach  $s \in \text{SecStruct}$  do
21     $\text{dot\_bracket}[\min(s)] = "("$ 
22     $\text{dot\_bracket}[\max(s)] = ")"$ 
23  end
24  return  $\text{dot\_bracket}$ 
25 end

```

---

El proceso para obtener la solución se muestra en el Algoritmo 3.2 (Huson, 2006) (Vialette, s.f.). Básicamente, consiste en encontrar un camino que conduzca a la puntuación máxima, así la ruta resultante definirá la estructura secundaria óptima en términos del máximo número de pares base. Este procedimiento es recursivo y exhibe una complejidad  $O(n)$  en tiempo de ejecución.

---

**Algoritmo 3.2:** Nussinov Rastreo

---

```

1 NussinovTraceback ( $i, j, M, SecStruct$ )
2   if  $i < j$  then
3     if  $M(i, j) == M(i + 1, j)$  then
4        $NussinovTraceback(i + 1, j, M, SecStruct)$ 
5     else if  $M(i, j) == M(i, j - 1)$  then
6        $NussinovTraceback(i, j - 1, M, SecStruct)$ 
7     else if  $M(i, j) == M(i + 1, j - 1) + \delta(i, j)$  then
8        $SecStruct = SecStruct \cup \{(i, j)\}$ 
9        $NussinovTraceback(i + 1, j - 1, M, SecStruct)$ 
10    else
11      for  $k = i + 1$  to  $j - 1$  do
12        if  $M(i, j) == M(i, k) + M(k + 1, j)$  then
13           $NussinovTraceback(i, k, M, SecStruct)$ 
14           $NussinovTraceback(k + 1, j, M, SecStruct)$ 
15          break
16        end
17      end
18    end
19  end
20  return  $SecStruct$ 
21 end

```

---

A continuación, se muestra un ejemplo paso a paso que detalla el funcionamiento del algoritmo de Nussinov.

**Ejemplo 3.1.** *Mediante el algoritmo de Nussinov, determine la estructura secundaria para la cadena  $r = AACUCCGUUA$ . Considere una longitud mínima del bucle  $l = 2$ .*

**Paso 1.** Se inicializa la matriz  $m \times m$  (donde  $m$  representa la longitud de la cadena  $r$ ). Como  $l = 2$  los valores de las primeras dos diagonales se asignan en 0 (consideramos las diagonales iniciando con índice 0).

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0								
A <sub>1</sub>		0	0							
C <sub>2</sub>			0	0						
U <sub>3</sub>				0	0					
C <sub>4</sub>					0	0				
C <sub>5</sub>						0	0			
G <sub>6</sub>							0	0		
U <sub>7</sub>								0	0	
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.3:** Inicialización de la matriz de Nussinov para  $r = \text{AACUCCGUUA}$ .

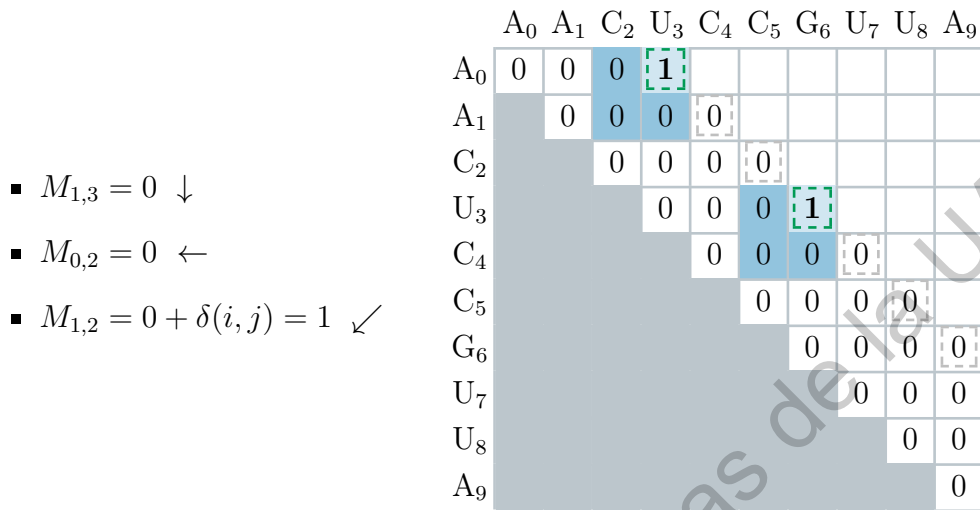
**Paso 2A.** Cálculo de la segunda diagonal (Figura 3.4).

- $M_{1,2} = 0 \downarrow$
- $M_{0,1} = 0 \leftarrow$
- $M_{1,1} = 0 + \delta(i, j) = 0 \checkmark$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0							
A <sub>1</sub>		0	0	0						
C <sub>2</sub>			0	0	0					
U <sub>3</sub>				0	0	0				
C <sub>4</sub>					0	0	0			
C <sub>5</sub>						0	0	0		
G <sub>6</sub>							0	0	0	
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

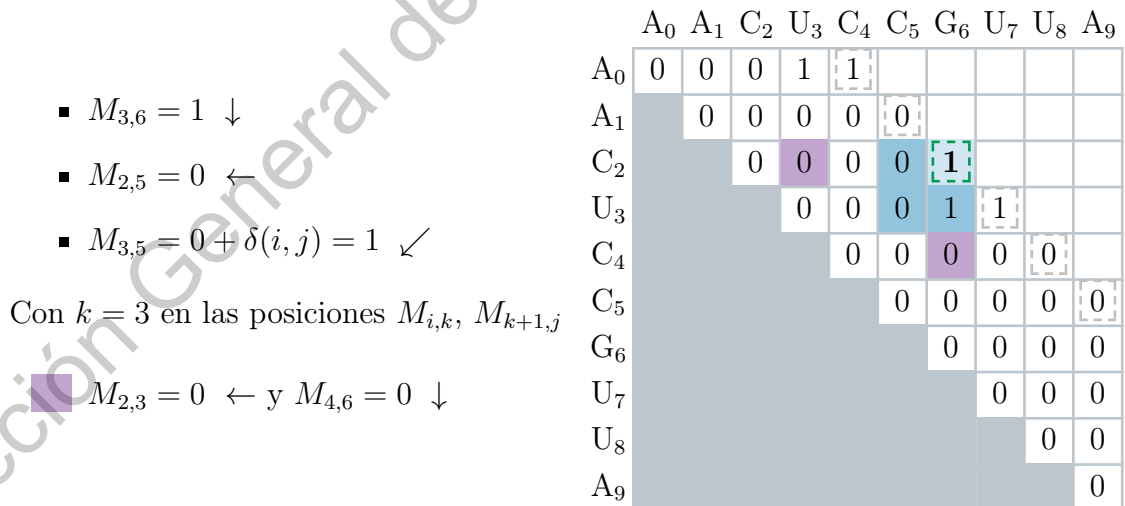
**Figura 3.4:** Cálculo de la segunda diagonal. Para determinar el valor de la posición  $M_{0,2}$  se obtiene el valor máximo de las posiciones  $M_{1,2}$ ,  $M_{0,1}$  y  $M_{1,1}$ .

**Paso 2B.** Cálculo de la tercera diagonal (Figura 3.5).



**Figura 3.5:** Cálculo de la tercera diagonal. Para determinar el valor de la posición  $M_{0,3}$  se obtiene el valor máximo de las posiciones  $M_{1,3}$ ,  $M_{0,2}$  y  $M_{1,2}$ .

**Paso 2C.** Cálculo de la cuarta diagonal (Figura 3.6).



**Figura 3.6:** Cálculo de la cuarta diagonal. Para determinar el valor de la posición  $M_{2,6}$  se obtiene el valor máximo de las posiciones  $M_{3,6}$ ,  $M_{2,5}$ ,  $M_{3,5}$  y el máximo de:  $\{(M_{2,3}, M_{4,6})\}$ .

**Paso 2D.** Cálculo de la quinta diagonal (Figura 3.7).

- $M_{1,5} = 0 \downarrow$
- $M_{0,4} = 1 \leftarrow$
- $M_{1,4} = 0 + \delta(i, j) = 0 \checkmark$

Con  $k = \{1, 2\}$  en las posiciones  $M_{i,k}$ ,  
 $M_{k+1,j}$

- $M_{0,1} = 0 \leftarrow$  y  $M_{2,5} = 0 \downarrow$
- $M_{0,2} = 0 \leftarrow$  y  $M_{3,5} = 0 \downarrow$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1				
A <sub>1</sub>		0	0	0	0	0	1			
C <sub>2</sub>			0	0	0	0	1	1		
U <sub>3</sub>				0	0	0	1	1	1	
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.7:** Cálculo de la quinta diagonal. Para determinar el valor de la posición  $M_{0,5}$  se obtiene el valor máximo de las posiciones  $M_{1,5}$ ,  $M_{0,4}$ ,  $M_{1,4}$  y el máximo de:  $\{(M_{0,1}, M_{2,5}), (M_{0,2}, M_{3,5})\}$ .

**Paso 2E.** Cálculo de la sexta diagonal (Figura 3.8).

- $M_{2,7} = 1 \downarrow$
- $M_{1,6} = 1 \leftarrow$
- $M_{2,6} = 1 + \delta(i, j) = 2 \checkmark$

Con  $k = \{2, 3, 4\}$  en las posiciones  $M_{i,k}$ ,  
 $M_{k+1,j}$

- $M_{1,2} = 0 \leftarrow$  y  $M_{3,7} = 1 \downarrow$
- $M_{1,3} = 0 \leftarrow$  y  $M_{4,7} = 0 \downarrow$
- $M_{1,4} = 0 \leftarrow$  y  $M_{5,7} = 0 \downarrow$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1	1			
A <sub>1</sub>		0	0	0	0	0	1	2		
C <sub>2</sub>			0	0	0	0	1	1	1	
U <sub>3</sub>				0	0	0	1	1	1	1
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.8:** Cálculo de la sexta diagonal. Para determinar el valor de la posición  $M_{1,7}$  se obtiene el valor máximo de las posiciones  $M_{2,7}$ ,  $M_{1,6}$ ,  $M_{2,6}$  y el máximo de:  $\{(M_{1,2}, M_{3,7}), (M_{1,3}, M_{4,7}), (M_{1,4}, M_{5,7})\}$ .

**Paso 2F.** Cálculo de la séptima diagonal (Figura 3.9).

- $M_{(2,8)} = 1 \downarrow$
- $M_{(1,7)} = 2 \leftarrow$
- $M_{(2,7)} = 1 + \delta(i, j) = 2 \swarrow$

Con  $k = \{2, 3, 4, 5\}$  en las posiciones  $M_{i,k}$ ,  
 $M_{k+1,j}$

- $M_{(1,2)} = 0 \leftarrow$  y  $M_{(3,8)} = 1 \downarrow$
- $M_{(1,3)} = 0 \leftarrow$  y  $M_{(4,8)} = 0 \downarrow$
- $M_{(1,4)} = 0 \leftarrow$  y  $M_{(5,8)} = 0 \downarrow$
- $M_{(1,5)} = 0 \leftarrow$  y  $M_{(6,8)} = 0 \downarrow$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1	1	2		
A <sub>1</sub>		0	0	0	0	0	1	2	2	
C <sub>2</sub>			0	0	0	0	1	1	1	1
U <sub>3</sub>				0	0	0	1	1	1	1
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.9:** Cálculo de la séptima diagonal. Para determinar el valor de la posición  $M_{1,8}$  se obtiene el valor máximo de las posiciones  $M_{2,8}$ ,  $M_{1,7}$ ,  $M_{2,7}$  y el máximo de:  $\{(M_{1,2}, M_{3,8}), (M_{1,3}, M_{4,8}), (M_{1,4}, M_{5,8}), (M_{1,5}, M_{6,8})\}$ .

**Paso 2G.** Cálculo de la octava diagonal (Figura 3.10).

- $M_{(1,8)} = 2 \downarrow$
- $M_{(0,7)} = 2 \leftarrow$
- $M_{(1,7)} = 2 + \delta(i, j) = 3 \swarrow$

Con  $k = \{1, 2, 3, 4, 5\}$  en las posiciones  $M_{i,k}$ ,  
 $M_{k+1,j}$

- $M_{(0,1)} = 0 \leftarrow$  y  $M_{(2,8)} = 1 \downarrow$
- $M_{(0,2)} = 0 \leftarrow$  y  $M_{(3,8)} = 1 \downarrow$
- $M_{(0,3)} = 1 \leftarrow$  y  $M_{(4,8)} = 0 \downarrow$
- $M_{(0,4)} = 1 \leftarrow$  y  $M_{(5,8)} = 0 \downarrow$
- $M_{(0,5)} = 1 \leftarrow$  y  $M_{(6,8)} = 0 \downarrow$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1	1	2	3	
A <sub>1</sub>		0	0	0	0	0	1	2	2	2
C <sub>2</sub>			0	0	0	0	1	1	1	1
U <sub>3</sub>				0	0	0	1	1	1	1
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.10:** Cálculo de la octava diagonal. Para determinar el valor de la posición  $M_{0,8}$  se obtiene el valor máximo de las posiciones  $M_{1,8}$ ,  $M_{0,7}$ ,  $M_{1,7}$  y el máximo de:  $\{(M_{0,1}, M_{2,8}), (M_{0,2}, M_{3,8}), (M_{0,4}, M_{5,8}), (M_{0,5}, M_{6,8})\}$ .

**Paso 2H.** Cálculo de la novena diagonal (Figura 3.11).

- $M_{(1,9)} = 2 \downarrow$
- $M_{(0,8)} = 3 \leftarrow$
- $M_{(1,8)} = 2 + \delta(i, j) = 2 \swarrow$

Con  $k = \{1, 2, 3, 4, 5, 6\}$  en las posiciones  $M_{i,k}, M_{k+1,j}$

- $M_{(0,1)} = 0 \leftarrow$  y  $M_{(2,9)} = 1 \downarrow$
- $M_{(0,2)} = 0 \leftarrow$  y  $M_{(3,9)} = 1 \downarrow$
- $M_{(0,3)} = 1 \leftarrow$  y  $M_{(4,9)} = 0 \downarrow$
- $M_{(0,4)} = 1 \leftarrow$  y  $M_{(5,9)} = 0 \downarrow$
- $M_{(0,5)} = 1 \leftarrow$  y  $M_{(6,9)} = 0 \downarrow$
- $M_{(0,6)} = 1 \leftarrow$  y  $M_{(7,9)} = 0 \downarrow$

	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1	1	2	3	3
A <sub>1</sub>		0	0	0	0	0	1	2	2	2
C <sub>2</sub>			0	0	0	0	1	1	1	1
U <sub>3</sub>				0	0	0	1	1	1	1
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

**Figura 3.11:** Cálculo de la novena diagonal. Para determinar el valor de la posición  $M_{0,9}$  se obtiene el valor máximo de las posiciones  $M_{1,9}, M_{0,8}, M_{1,8}$  y el máximo de:  $\{(M_{0,1}, M_{2,9}), (M_{0,2}, M_{3,9}), (M_{0,3}, M_{4,9}), (M_{0,4}, M_{5,9}), (M_{0,5}, M_{6,9}), (M_{0,6}, M_{7,9})\}$ .

**Paso 3.** Obtención del plegamiento.

La estructura resultante  $S_r$  se obtiene mediante el Algoritmo 3.2, este inicia en la parte superior derecha  $M_{0,n-1}$  la cual corresponde con el número máximo de pares base plegados y continúa hasta que el valor de la posición  $i$  es menor al valor de  $j$  (Figura 3.12).

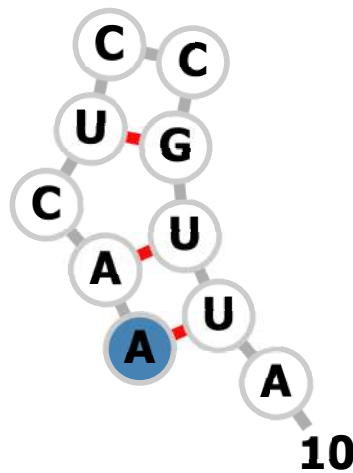
	A <sub>0</sub>	A <sub>1</sub>	C <sub>2</sub>	U <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	G <sub>6</sub>	U <sub>7</sub>	U <sub>8</sub>	A <sub>9</sub>
A <sub>0</sub>	0	0	0	1	1	1	1	2	3	3
A <sub>1</sub>		0	0	0	0	0	1	2	2	2
C <sub>2</sub>			0	0	0	0	1	1	1	1
U <sub>3</sub>				0	0	0	1	1	1	1
C <sub>4</sub>					0	0	0	0	0	0
C <sub>5</sub>						0	0	0	0	0
G <sub>6</sub>							0	0	0	0
U <sub>7</sub>								0	0	0
U <sub>8</sub>									0	0
A <sub>9</sub>										0

1. Inicia con  $i = 0$ ,  $j = 9$  se cumple la segunda condición  $M_{i,j} == M_{i,j-1}$ , por lo tanto los nuevos valores de entrada son:  $i = 0$ ,  $j = 8$ .
2. Entra  $i = 0$ ,  $j = 8$  se cumple la tercera condición  $M_{i,j} == M_{i+1,j-1} + \delta(i, j)$ , por lo que se agrega el par  $(0, 8)$  a  $S_r$  y los nuevos valores de entrada son:  $i = 1$ ,  $j = 7$ .
3. Entra  $i = 1$ ,  $j = 7$  se cumple nuevamente la tercera condición  $M_{i,j} == M_{i+1,j-1} + \delta(i, j)$ , se agrega el par  $(1, 7)$  a  $S_r$  y se asignan los nuevos valores de entrada:  $i = 2$ ,  $j = 6$ .
4. Entra  $i = 2$ ,  $j = 6$  se cumple la primera condición  $M_{i,j} == M_{i+1,j}$ , por lo que se asignan los nuevos valores de entrada:  $i = 3$ ,  $j = 6$ .
5. Entra  $i = 3$ ,  $j = 6$  se cumple la tercera condición  $M_{i,j} == M_{i+1,j-1} + \delta(i, j)$  por lo que se agrega el par  $(3, 6)$  a  $S_r$  y se asignan los nuevos valores de entrada:  $i = 4$ ,  $j = 5$ .
6. Entra  $i = 4$ ,  $j = 5$  se cumple la primera condición  $M_{i,j} == M_{i+1,j}$ , por lo que se asignan los nuevos valores de entrada:  $i = 5$ ,  $j = 5$ .
7. Entra  $i = 5$ ,  $j = 5$  no se cumple con la condición inicial  $i < j$ , por lo cual termina el algoritmo.

**Figura 3.12:** Ruta de la puntuación máxima de la estructura secundaria y llamadas recursivas del Algoritmo 3.2.

A partir del resultado mostrado en la Figura 3.12, se obtiene que la estructura secundaria  $S_r$  esta formada por los pares de índices  $\{(0, 8), (1, 7), (3, 6)\}$ . Estos índices representan los pares base complementarios A-U, A-U, U-G obtenidos durante el plegamiento, la Figura 3.13 representa la estructura correspondiente.

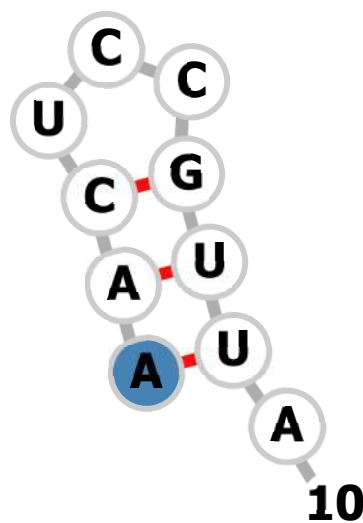




**Figura 3.13:** Estructura secundaria para  $S_r = \{(0, 8), (1, 7), (3, 6)\}$ .

Si en el ejemplo anterior se da prioridad a los pares Watson-Crick sobre los tipo Wobble, o estos últimos no se consideran. El algoritmo obtiene una segunda solución óptima, que estaría formada por los pares de índices  $\{(0, 8), (1, 7), (2, 6)\}$  sobre la matriz de la Figura 3.12.

La segunda solución (Figura 3.14), al igual que la primera tiene un máximo de tres pares base complementarios. La diferencia topológica entre ambas estructuras es que en la primera se forma una protuberancia entre los pares base A-U y U-G. Mientras que en la segunda se forma una hélice desde el par base A-U hasta el par C-G.



**Figura 3.14:** Estructura secundaria para  $S_r = \{(0, 8), (1, 7), (2, 6)\}$ .

### 3.1.2. Algoritmo de Zuker

El algoritmo de Zuker (Zuker y Stiegler, 1981), se basa en el algoritmo de Nussinov. La idea principal de este es que cada par base agregado a la estructura secundaria reduce la energía libre, mientras que subestructuras como bucles y protuberancias la desestabilizan (es decir, incrementan la energía); de esta manera el algoritmo obtiene una solución óptima.

El modelo de contribución de energía utilizado por el algoritmo, es el modelo de Turner y Mathews, visto en el Capítulo 2.

La idea matemática del algoritmo es calcular las posibles energías para cada subsecuencia  $S_{i,j}$  dada. La Ecuación 3.4, representa la mínima energía libre de todas las posibles estructuras formadas de la subsecuencia  $S_{i,j}$ , denotada por  $W(i, j)$ . Por otro lado, la Ecuación 3.5, es la mínima energía libre de la subsecuencia  $S_{i,j}$  donde  $S_i$  y  $S_j$  son pares base complementarios entre si, denotada por  $V(i, j)$ . En caso de que  $S_i$  y  $S_j$  no sean complementarios,  $V(i, j) = \infty$ . Para subsecuencias  $S_{i,j}$ , donde  $j - i = 4$ , la energía  $W(i, j) = \infty$  (Ecuación 3.3).

$$W(i, j) = \infty, \forall i, j \text{ con } j - 4 < i < j \leq n \quad (3.3)$$

$$W(i, j) = \min \begin{cases} W(i + 1, j) \\ W(i, j - 1) \\ V(i, j) \\ W(i, j) = \min_{i < k < j} \{ W(i, k) + W(k + 1, j) \} \end{cases} \quad (3.4)$$

$$V(i, j) = \min \begin{cases} eh(i, j) \\ es(i, j) + V(i + 1, j - 1) \\ VBI(i, j) \\ VM(i, j) \end{cases} \quad (3.5)$$

Donde,  $VBI$  es la mínima energía entre una protuberancia y un bucle interno:

$$VBI(i, j) = \min_{i < k < l < j} \left\{ eb(i, j, k, l) + V(k, l), ei(i, j, k, l) + V(k, l) \right\}$$

y  $VM$ , representa la energía por la formación de un bucle múltiple:

$$VM(i, j) = \min_{i < k < j-1} \left\{ W(i+1, k) + W(k+1, j-1) \right\}$$

A continuación, se enlistan las contribuciones energéticas de  $V$  y  $VBI$ :

- $eh(i, j)$ : es la contribución energía dada por la formación de un bucle en horquilla acotado por el par base  $(i, j)$ , ver Subsección 2.2.1.
- $es(i, j)$ : es la energía por la formación de una hélice con el par base  $(i, j)$ , ver Sección 2.1.
- $eb(i, j, k, l)$ : energía por la formación de una protuberancia acotada por los pares base  $(i, j)$  y  $(k, l)$ , para  $i < k < l < j$ , ver Subsección 2.2.2.
- $ei(i, j)$ : es la contribución de energía, debida a la formación de un bucle interno acotada por los pares base  $(i, j)$  y  $(k, l)$ , para  $i < k < l < j$ , ver Subsección 2.2.3.

El algoritmo de Zuker tiene un orden de complejidad en tiempo  $O(n^4)$  y espacio  $O(n^2)$  (Zuker y Stiegler, 1981), pero al restringir el tamaño máximo de un bucle interno o protuberancia a 30 bases libres, el orden de complejidad en tiempo de ejecución reduce a  $O(n^3)$  (Huson, 2006).

A continuación, en el Algoritmo 3.3 se muestra el proceso del algoritmo de Zuker, se puede observar que las matrices  $W$  y  $V$  se llenan simultáneamente, de forma similar al del algoritmo de Nussinov. En el Algoritmo 3.4, se muestra el proceso para obtener la estructura secundaria, a partir de los valores energéticos de la matriz  $V$ . M. Andronescu en (Andronescu, 2003), agrega una estructura de datos  $V_{ds}$  asociada a la matriz  $V$ , que incluye los valores de energía de  $V$  y la información

de la estructura: tipo (hélice, bucle en horquilla, bucle interno o bucle múltiple), pares base de cierre y longitud.

---

**Algoritmo 3.3:** Zuker

---

```

1 Entrada: Una cadena  $r = \{r_0, \dots, r_{n-1}\}$  donde  $r_i \in \Sigma_{ARN}$  y un entero  $l$ 
   que representa el tamaño mínimo del bucle.
2 Salida: La estructura secundaria  $S_r$  en formato DotBracket.
3 Zuker ( $r, l$ )
4    $n = \text{Length}(r)$ 
5    $\text{SecStruct} = \emptyset$ 
6   Inicializar :  $W(i, j) = V(i, j) = \infty, \forall i, j$  con  $j - 4 < i < j \leq n$ 
7   for  $k = l$  to  $n$  do
8     for  $i = 0$  to  $n - k$  do
9        $j = i + k$ 
10      
$$W(i, j) = \min \begin{cases} W(i + 1, j) \\ W(i, j - 1) \\ V(i, j) \\ W(i, j) = \min_{i < k < j} \{W(i, k) + W(k + 1, j)\} \end{cases}$$

11     end
12   end
13   Call ZukerTraceback( $0, n - 1, V_{ds}, \text{SecStruct}$ )
14    $\text{dot\_bracket} = \{'.', \dots, n\}$ 
15   foreach  $s \in \text{SecStruct}$  do
16      $\text{dot\_bracket}[\text{min}(s)] = "("$ 
17      $\text{dot\_bracket}[\text{max}(s)] = ")"$ 
18   end
19   return  $\text{dot\_bracket}$ 
20 end

```

---

---

**Algoritmo 3.4:** Zuker Rastreo
 

---

```

1 ZukerTraceback (i, j, Vds, SecStruct)
2   if i < j then
3     if Vds(i, j).type == HAIRPIN then
4       | SecStruct = SecStruct ∪ {(i, j)}
5     else if Vds(i, j).type == STACK then
6       | SecStruct = SecStruct ∪ {(i, j)}
7       | ZukerTraceback(i + 1, j - 1, Vds, SecStruct)
8     else if Vds(i, j).type == INTERNAL or Vds(i, j).type == BULGE
9       then
10      | SecStruct = SecStruct ∪ {(i, j)}
10      | ZukerTraceback(Vds(i, j).k, Vds(i, j).l, Vds, SecStruct)
11     else if Vds(i, j).type = MULTIPLE then
12       | SecStruct = SecStruct ∪ {(i, j)}
13       | for k = i + 1 to j - 1 do
14         | if M(i, j) == M(i, k) + M(k + 1, j) then
15           | ZukerTraceback(i, k, Vds, SecStruct)
16           | ZukerTraceback(k + 1, j, Vds, SecStruct)
17           | break
18         | end
19       | end
20     end
21   end
22   return SecStruct
23 end

```

---

En la Figura 3.15, se muestra el resultado de la matriz  $W$ , para la secuencia de ARN  $r = \text{AAACAUGAGGAUUACCCAUGU}$ . Nótese, que algunos valores están resaltados; estos valores se utilizan para ejemplificar como se determina el valor  $(i, j)$ , que en este caso es la posición  $(0, 20)$ . Siguiendo la Ecuación 3.4:

$$W(i+1, j) = W(1, 20) = -4.4$$

$$W(i, j-1) = W(0, 19) = -1.1$$

$$V(i, j) = V(0, 20) = 0.2$$

$$W(i, j) = \min_{i < k < j} \{W(i, k) + W(k+1, j)\}$$

Con  $k = \{1, 2, 3, 4, 5, 15, 16, 17, 18\} : W(i, k) = \infty$ . Mientras, que para  $k = \{6, \dots, 14\}$ :

■  $W_{(0,6)} + W_{(7,20)} = 5.4$

■  $W_{(0,7)} + W_{(8,20)} = 7.1$

- $W_{(0,8)} + W_{(9,20)} = 9.8$
- $W_{(0,9)} + W_{(10,20)} = 7.2$
- $W_{(0,10)} + W_{(11,20)} = 7$
- $W_{(0,11)} + W_{(12,20)} = 7$
- $W_{(0,12)} + W_{(13,20)} = 7$
- $W_{(0,13)} + W_{(14,20)} = 8.1$
- $W_{(0,14)} + W_{(15,20)} = 9$

De esta forma, el valor para  $M(0, 20)$  es -4.4

	A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	C <sub>3</sub>	A <sub>4</sub>	U <sub>5</sub>	G <sub>6</sub>	A <sub>7</sub>	G <sub>8</sub>	G <sub>9</sub>	A <sub>10</sub>	U <sub>11</sub>	U <sub>12</sub>	A <sub>13</sub>	C <sub>14</sub>	C <sub>15</sub>	C <sub>16</sub>	A <sub>17</sub>	U <sub>18</sub>	G <sub>19</sub>	U <sub>20</sub>	
A <sub>0</sub>	∞	∞	∞	∞	∞	∞	5.8	5.8	5.8	3.2	3	3	3	3	3	1.6	-0.4	-0.4	-0.5	-1.1	-4.4	
A <sub>1</sub>		∞	∞	∞	∞	∞	6.7	6.7	6.7	3.2	3	3	3	3	3	1.6	-0.4	-0.4	-0.5	-1.1	-4.4	
A <sub>2</sub>			∞	∞	∞	∞	∞	∞	∞	3.2	3	3	3	3	3	1.6	-0.4	-0.4	-0.5	-1.1	-4.4	
C <sub>3</sub>				∞	∞	∞	∞	∞	∞	4.3	4.1	4.1	4.1	4.1	3.9	1.6	-0.4	-0.4	-0.5	-1.1	-3.3	
A <sub>4</sub>					∞	∞	∞	∞	∞	6	4.8	4.5	4.5	3.9	1.6	-0.4	-0.4	-0.5	-1	-1		
U <sub>5</sub>						∞	∞	∞	∞	6.8	5.6	5.6	5.4	4.7	1.6	-0.4	-0.4	-0.4	-0.4	-0.4		
G <sub>6</sub>							∞	∞	∞	∞	∞	6.2	5.5	5.5	1.6	-0.4	-0.4	-0.4	-0.4	-0.4		
A <sub>7</sub>								∞	∞	∞	∞	∞	6.9	6.1	6.1	1.6	-0.4	-0.4	-0.4	-0.4	-0.4	
G <sub>8</sub>									∞	∞	∞	∞	∞	6.8	6.8	2.9	1.3	1.3	1.3	1.3	1.3	
G <sub>9</sub>										∞	∞	∞	∞	∞	∞	4.6	4.6	4.6	4.6	4.6	4	
A <sub>10</sub>											∞	∞	∞	∞	∞	∞	∞	∞	8.2	5.9	4	
U <sub>11</sub>												∞	∞	∞	∞	∞	∞	∞	∞	8.2	5.9	4
U <sub>12</sub>													∞	∞	∞	∞	∞	∞	∞	8.8	5.9	4
A <sub>13</sub>														∞	∞	∞	∞	∞	∞	∞	6	4
C <sub>14</sub>															∞	∞	∞	∞	∞	∞	∞	5.14
C <sub>15</sub>																∞	∞	∞	∞	∞	∞	6
C <sub>16</sub>																	∞	∞	∞	∞	∞	∞
A <sub>17</sub>																		∞	∞	∞	∞	∞
U <sub>18</sub>																			∞	∞	∞	∞
G <sub>19</sub>																				∞	∞	∞
U <sub>20</sub>																					∞	∞

**Figura 3.15:** Matriz  $W$  procesada por el algoritmo de Zuker.

La estructura secundaria  $S_r$ , se obtiene a partir de la matrices  $V$  y  $V_{ds}$  (Figuras 3.16 y 3.17), mediante el uso del Algoritmo 3.4. A diferencia del algoritmo de Nussinov, en el cual el *rastreo* de la estructura se inicia en la posición  $(0, n - 1)$ ; con Zuker se inicia en la posición en donde se encuentre la menor energía, en este ejemplo es la posición  $(2, 20)$  de la matriz  $V$ . Este valor  $V_{(2,20)} = -5.8$ , corresponde con la energía libre en kcal/mol, de la estructura secundaria resultante.

	A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	C <sub>3</sub>	A <sub>4</sub>	U <sub>5</sub>	G <sub>6</sub>	A <sub>7</sub>	G <sub>8</sub>	G <sub>9</sub>	A <sub>10</sub>	U <sub>11</sub>	U <sub>12</sub>	A <sub>13</sub>	C <sub>14</sub>	C <sub>15</sub>	C <sub>16</sub>	A <sub>17</sub>	U <sub>18</sub>	G <sub>19</sub>	U <sub>20</sub>		
A <sub>0</sub>		∞	∞	∞	∞	5.3	∞	∞	∞	∞	∞	4.2	4.4	∞	∞	∞	∞	∞	∞	4.2	∞	0.2	
A <sub>1</sub>			∞	∞	∞	6.2	∞	∞	∞	∞	∞	∞	5.3	4.6	∞	∞	∞	∞	∞	3.8	∞	2.2	
A <sub>2</sub>				∞	∞	∞	∞	∞	∞	∞	∞	∞	5.5	6.2	∞	∞	∞	∞	∞	3.4	∞	-5.8	
C <sub>3</sub>					∞	∞	∞	∞	4.1	4.2	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	-3.6	
A <sub>4</sub>						∞	∞	∞	∞	∞	∞	∞	∞	3.8	4.8	∞	∞	∞	∞	∞	-1.5	∞	2.8
U <sub>5</sub>							∞	∞	∞	∞	6.2	4.9	∞	∞	3.9	∞	∞	∞	∞	-0.4	∞	4.2	
G <sub>6</sub>								∞	∞	∞	∞	∞	5.8	4.9	∞	4.8	8.1	1.7	∞	4.1	∞	4.3	
A <sub>7</sub>									∞	∞	∞	∞	6.2	5.8	∞	∞	∞	∞	∞	∞	5	∞	4.3
G <sub>8</sub>										∞	∞	∞	∞	6.2	∞	3.2	1.2	3.9	∞	5.5	∞	4.1	
G <sub>9</sub>											∞	∞	∞	∞	∞	4.5	7.2	7.3	∞	5.8	∞	4	
A <sub>10</sub>												∞	∞	∞	∞	∞	∞	∞	∞	5.6	∞	3.6	
U <sub>11</sub>													∞	∞	∞	∞	∞	∞	8.6	∞	5.1	∞	
U <sub>12</sub>														∞	∞	∞	∞	∞	∞	8.1	∞	4.5	
A <sub>13</sub>															∞	∞	∞	∞	∞	5.5	∞	1.7	
C <sub>14</sub>																∞	∞	∞	∞	∞	4.8	∞	
C <sub>15</sub>																	∞	∞	∞	∞	5.7	∞	
C <sub>16</sub>																		∞	∞	∞	∞	∞	
A <sub>17</sub>																			∞	∞	∞	∞	
U <sub>18</sub>																				∞	∞	∞	
G <sub>19</sub>																					∞	∞	
U <sub>20</sub>																						∞	

Figura 3.16: Matriz  $V$  procesada por el algoritmo de Zuker.

	A <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	C <sub>3</sub>	A <sub>4</sub>	U <sub>5</sub>	G <sub>6</sub>	A <sub>7</sub>	G <sub>8</sub>	G <sub>9</sub>	A <sub>10</sub>	U <sub>11</sub>	U <sub>12</sub>	A <sub>13</sub>	C <sub>14</sub>	C <sub>15</sub>	C <sub>16</sub>	A <sub>17</sub>	U <sub>18</sub>	G <sub>19</sub>	U <sub>20</sub>		
A <sub>0</sub>						H						I	S							B		B	
A <sub>1</sub>						H						I	S								B		I
A <sub>2</sub>						H						H	H								B		S
C <sub>3</sub>						H		H	H													S	
A <sub>4</sub>												S	H								S		S
U <sub>5</sub>							H	H	H	H			S							S		I	
G <sub>6</sub>												H	S		H	H	B			I		H	
A <sub>7</sub>												H	H								B		H
G <sub>8</sub>												H	H		H	S	S			H		H	
G <sub>9</sub>												H	H		H	H	H			H		H	
U <sub>11</sub>													H							H		H	
U <sub>12</sub>																				H		S	
A <sub>13</sub>																					H		H
C <sub>14</sub>																						H	
C <sub>15</sub>																						H	
C <sub>16</sub>																						H	
A <sub>17</sub>																							H
U <sub>18</sub>																							
G <sub>19</sub>																							
U <sub>20</sub>																							

Figura 3.17: Matriz  $V_{ds}$  procesada por el algoritmo de Zuker. Las hélices están representadas por el carácter S, las horquillas por H, las protuberancias por B y finalmente los bucles internos están representados por el carácter I.

A partir, de la posición inicial (2, 20), las llamadas recursivas del algoritmo son las siguientes:

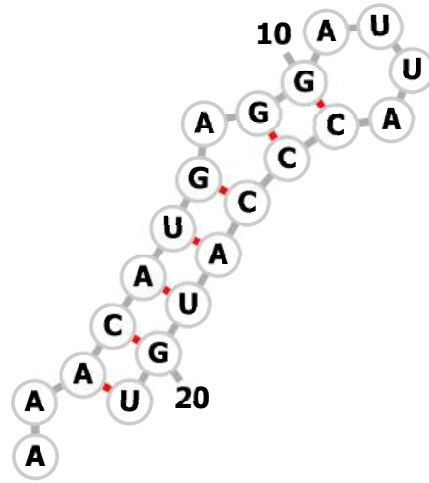
1.  $V_{ds}(2, 20) == S$ 
  - $S_r \cup \{(2, 20)\}$
  - $\text{ZukerTraceback}(3, 19, V_{ds}, S_r)$
2.  $V_{ds}(3, 19) == S$ 
  - $S_r \cup \{(3, 19)\}$
  - $\text{ZukerTraceback}(4, 18, V_{ds}, S_r)$
3.  $V_{ds}(4, 18) == S$ 
  - $S_r \cup \{(4, 18)\}$
  - $\text{ZukerTraceback}(5, 17, V_{ds}, S_r)$
4.  $V_{ds}(5, 17) == S$ 
  - $S_r \cup \{(5, 17)\}$
  - $\text{ZukerTraceback}(6, 16, V_{ds}, S_r)$
5.  $V_{ds}(6, 16) == B$ 
  - $S_r \cup \{(6, 16)\}$
  - $\text{ZukerTraceback}(8, 15, V_{ds}, S_r)$
6.  $V_{ds}(8, 15) == S$ 
  - $S_r \cup \{(8, 15)\}$
  - $\text{ZukerTraceback}(9, 14, V_{ds}, S_r)$
7.  $V_{ds}(9, 14) == H$ 
  - $S_r \cup \{(9, 14)\}$
  - Termina el algoritmo

De la ejecución del algoritmo 3.4, se obtiene  $S_r = \{(2, 20), (3, 19), (4, 18), (5, 17), (6, 16), (8, 15), (9, 14)\}$ . En la Figura 3.18, se muestra la estructura secundaria obtenida a partir de  $S_r$ .

### 3.1.3. Algoritmo de Akutsu

El algoritmo de Akutsu maximiza el número de pares base e incluye la predicción de pseudonudos simples con un tiempo de ejecución  $O(n^4)$  y espacio  $O(n^3)$  (Akutsu, 2000). En (Kravchenko, 2009) se propone una modificación al algoritmo de Akutsu que incluye el modelo termodinámico de minimización de energía libre y reduce la complejidad del espacio de  $O(n^3)$  a  $O(n^2)$ , manteniendo el tiempo de ejecución en  $O(n^4)$ .





**Figura 3.18:** Estructura secundaria y notación Dot-Bracket para  $r = AAACAUGAGGAUUACCCAUGU$ .

El algoritmo básico incluye 3 matrices tridimensionales:  $S_L$ ,  $S_R$  y  $S_M$ , y una matriz bidimensional  $S$ :

- $S_L(i, j, k)$  es el número máximo de pares de bases en un pseudonudo para  $S[i_0..k_0]$  dado que  $(i, j)$  es un par de bases:

$$S_L(i, j, k) = \delta(r_i, r_j) + \max \begin{cases} S_L(i-1, j+1, k) \\ S_M(i-1, j+1, k) \\ S_R(i-1, j+1, k) \end{cases} \quad (3.6)$$

$$\delta(r_i, r_j) := \begin{cases} 1 & \text{Si } (r_i, r_j) \text{ es un par base} \\ 0 & \text{en otro caso} \end{cases}$$

- $S_R$  es el número máximo de pares de bases debajo en un pseudonudo para  $S[i_0..k_0]$  dado que  $(j, k)$  es un par de bases:

$$S_R(i, j, k) = \delta(r_i, r_j) + \max \begin{cases} S_L(i, j+1, k-1) \\ S_M(i, j+1, k-1) \\ S_R(i, j+1, k-1) \end{cases} \quad (3.7)$$

- $S_M$  es el número máximo de pares de bases debajo del triplete  $(i, j, k)$  en un

pseudonudo para  $S[i_0..k_0]$  dado que  $(i, j)$  y  $(j, k)$  no son base pares:

$$S_M(i, j, k) = \max \begin{cases} S_M(i-1, j, k), S_M(i, j+1, k), S_M(i, j, k-1) \\ S_L(i-1, j, k), S_L(i, j+1, k) \\ S_R(i, j+1, k), S_R(i, j, k-1) \end{cases} \quad (3.8)$$

- $S(i_0, k_0)$  representa la puntuación óptima para un pseudonudo en la región  $(i_0, k_0)$ :

$$S(i_0, k_0) = \max_{i_0 \leq i < j \leq k \leq k_0} \{ S_L(i, j, k), S_M(i, j, k), S_R(i, j, k) \} \quad (3.9)$$

Para calcular los valores de las matrices  $S_L(i, j, k)$ ,  $S_R(i, j, k)$  y  $S_M(i, j, k)$ , se requieren algunas restricciones iniciales (Wong, 2004):

$$S_R(i_0 - 1, j, j + 1) = \delta(S_j, S_{j+1}) \quad \forall j \quad (1)$$

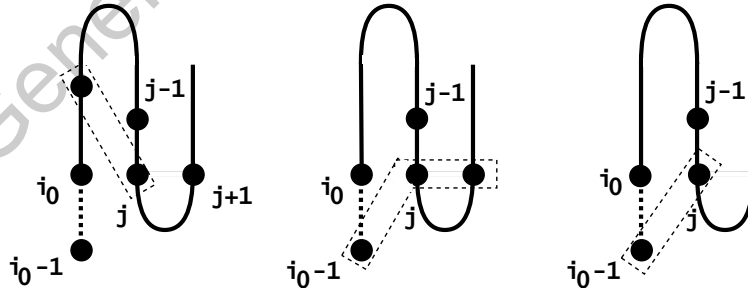
$$S_R(i_0 - 1, j, j) = 0 \quad \forall j \quad (2)$$

$$S_L(i, j, j) = \delta(S_i, S_j) \quad \forall i_0 \leq i < j \quad (3)$$

$$S_L(i_0 - 1, j, k) = 0 \quad \forall k = j + 1, k = j \quad (4)$$

$$S_M(i_0 - 1, j, k) = 0 \quad \forall k = j + 1, k = j \quad (5)$$

En la Figura 3.19, se muestran estos casos base. El caso (3) corresponde a la imagen izquierda, el caso (1) a la imagen central y el caso (2) a la imagen derecha.



**Figura 3.19:** Representación de las ecuaciones  $S_L$ ,  $S_M$  y  $S_R$ .

En este trabajo de tesis, se hace uso de la implementación del código propuesto en (Kravchenko, 2009), para el algoritmo de Akutsu.

### 3.1.4. Algoritmo de McCaskill

Al algoritmo de McCaskill (McCaskill, 1990), convierte la energía libre  $\Delta G$  en probabilidades utilizando la ecuación de Gibbs-Boltzmann sumando las probabilidades de todas las posibles estructuras para una secuencia, en lugar de solo elegir estructura con la energía mínima. Esta suma de las probabilidades de cada una de las estructuras que contienen el par base  $(i, j)$ , dividida entre la suma de todas las estructuras, se interpreta como una estimación de confianza del par base  $(i, j)$  (Durbin et al., 1998).

El algoritmo hace uso de tres matrices de tamaño  $n \times n$  (donde  $n$  es la longitud de la secuencia de ARN):  $Q$ ,  $Q^{bp}$  y  $P$ . Donde,  $Q$  es la función de partición de la subsecuencia  $S_{i,\dots,j}$  (Ecuación 3.10),  $Q^{bp}$  la función de partición de la subsecuencia cuando  $S_i$  y  $S_j$  forman un par base complementario (Ecuación 3.11), finalmente,  $P$  es la suma de probabilidades para el par  $(i, j)$  (Ecuación 3.12).

$$Q_{i,j} = Q_{i,j-1} + \sum_{i \leq k < (j-l)} Q_{i,k-1} \times Q_{k,j}^{bp} \quad (3.10)$$

Donde,  $l$  es la longitud mínima del bucle.

$$Q_{k,j}^{bp} = \begin{cases} Q_{i+1,j-1} \times \exp(-E_{bp}/RT) & \text{Si } (s_i, s_j) \text{ es un par base} \\ 0 & \text{en otro caso} \end{cases} \quad (3.11)$$

Donde,  $E_{bp}$  es la energía del par base, en nuestra implementación del algoritmo, hacemos uso de los parámetros de la Tabla 2.1.  $R = 1.9872 \times 10^{-3}$  es la constante del gas y  $T = 310.15 \text{ K}$  es la temperatura.

$$P_{i,j} = \frac{Q_{0,i-1} \times Q_{i,j}^{bp} \times Q_{j+1,n}}{Q_{0,n}} + \sum_{p < i, j < q} P_{p,q} \times \frac{\exp(-E_{bp}/RT) \times Q_{p+1,i-1} \times Q_{i,j}^{bp} \times Q_{j+1,q-1}}{Q_{p,q}^{bp}} \quad (3.12)$$

A continuación, se muestra el algoritmo de McCaskill (Zhao y Sahni, 2020):

---

**Algoritmo 3.5:** McCaskill

---

```
1 Entrada: Una cadena  $r = \{r_0, \dots, r_{n-1}\}$  donde  $r_i \in \Sigma_{ARN}$  y un entero  $l$  que
   representa el tamaño mínimo del bucle, constante del gas  $R$ , temperatura  $T$ .
2 Salida: La matriz de probabilidades  $P$ .
3 McCaskill( $r, l, R, T$ )
4   for  $i = n - 1$  to  $0$  do
5     for  $j = i + 1$  to  $n$  do
6        $Q(i, j) = Q(i, j - 1)$ 
7       for  $k = i$  to  $j - l$  do
8          $Q^{bp}(k, j) = Q(k + 1, j - 1) \times \exp(-E_{bp}/RT) * \delta(k, j)$ 
9          $Q(i, j) += Q(i)(k - l) \times Q^{bp}(k, j)$ 
10      end
11    end
12  end
13 end
14 for  $i = 0$  to  $n$  do
15   for  $j = i + 1$  to  $n$  do
16     if  $i \geq j - l$  or  $\delta(i, j) == 0$  then
17       continue
18     end
19      $rs = Q^{bp}(i, j)$ 
20     if  $i > 1$  then
21        $rs *= Q(0, i - 1)$ 
22     end
23     if  $j + 1 < n$  then
24        $rs *= Q(j + 1, n)$ 
25     end
26      $rs /= Q(0, n)$ 
27     for  $p = 0$  to  $i$  do
28       for  $q = j + 1$  to  $n$  do
29          $s = P(p, q) + Q(i, j)$ 
30         if  $p + 1 < i$  then
31            $s *= Q(p + 1, i - 1)$ 
32         end
33         if  $j + 1 < q$  then
34            $s *= Q(j + 1, q - 1)$ 
35         end
36          $s *= \exp(-E_{bp}/RT)$ 
37          $s /= Q^{bp}(p, q)$ 
38          $rs += s$ 
39       end
40     end
41      $P(i, j) = rs$ 
42   end
43 end
```

---

## 3.2. Algoritmos Metaheurísticos

### 3.2.1. Algoritmos Genéticos

El desarrollo de los algoritmos genéticos (AGs, en singular AG) se remonta a la década de 1960, sin embargo, fueron popularizados en los 90s por Holland. Los AGs, se han aplicado principalmente en dos áreas de investigación: optimización en la que se utiliza la metáfora en la que el problema representa entorno y las soluciones viables se consideran individuos de ese entorno; y en el estudio de la adaptación en sistemas complejos, en donde se simula la evolución de una población y su adaptación en el tiempo (Sivanandam y Deepa, 2008) (Du y Swamy, 2016).

Para que los AG encuentren la mejor solución, es necesario realizar ciertas operaciones sobre estos individuos, así como la correcta interpretación de la terminología de AGs en el diseño del algoritmo.

#### 3.2.1.1. Terminología de AGs

##### Población

Una población es un conjunto finito de individuos a evaluar (Definición 3.1). Dos aspectos importantes sobre las poblaciones que se consideran dentro de los AGs son: el tamaño de la población y la generación de la población inicial.

**Definición 3.1 (Población).** *Un conjunto de individuos  $P = \{x_1, x_2, \dots, x_n\}$ , donde  $x_i$  es  $i$ -ésimo individuo y  $n$  es el tamaño de la población.*

El tamaño de la población depende de la complejidad del problema a resolver. La población inicial debería ser tan grande como sea posible, a fin de explorar todo el espacio de búsqueda. En la mayoría de los casos, esta población inicial se crea de manera aleatoria, aunque hay problemas en los que es posible crearla mediante alguna heurística o preprocesamiento (Sivanandam y Deepa, 2008).

## **Individuo**

Un individuo es una solución viable del problema que se intenta resolver. Cada individuo se representa mediante un cromosoma. Un cromosoma, es un conjunto de parámetros (genes) que definen una solución al problema, en donde, cada gen codifica un parámetro del problema.

## **Genotipo y Fenotipo**

Un genotipo representa una solución codificada, es decir, el cromosoma de un individuo. Mientras, que el fenotipo es el conjunto de características de un individuo (aptitud y genotipo). En nuestra aplicación de los GAs a la predicción de estructuras secundarias de ARN, el genotipo son las hélices que construyen un individuo, y el fenotipo es la estructura secundaria y su energía libre (aptitud).

## **Aptitud**

La aptitud de un individuo, es el valor de una función objetivo (en nuestro caso, corresponde con la minimización de energía libre para la predicción de una estructura secundaria). Para calcular la aptitud, primero se debe decodificar el cromosoma y evaluarlo en la función objetivo.

## **Codificación**

La codificación es el proceso, por el cual se representan los genes de un individuo. Puede realizarse utilizando bits, números, cadena o cualquier otro objeto. En nuestro caso de estudio, la codificación se hace mediante números enteros, que sirven como identificador de cada hélice.

### **3.2.1.2. Operadores Genéticos**

#### **Selección**

La selección es el proceso de elegir a dos individuos de la población inicial para cruzarlos y generar descendientes que pasen a la siguiente generación (los individuos seleccionados se conocen como parental). El propósito de la selección es enfatizar a los individuos más aptos de la población con la esperanza de que sus descendientes tengan una mayor aptitud (Sivanandam y Deepa, 2008).

## Cruza

La cruza es el proceso en el cual dados dos individuos, estos se recombinan para producir descendencia. A partir, de esta descendencia la población se puede enriquecer con individuos de mejor aptitud.

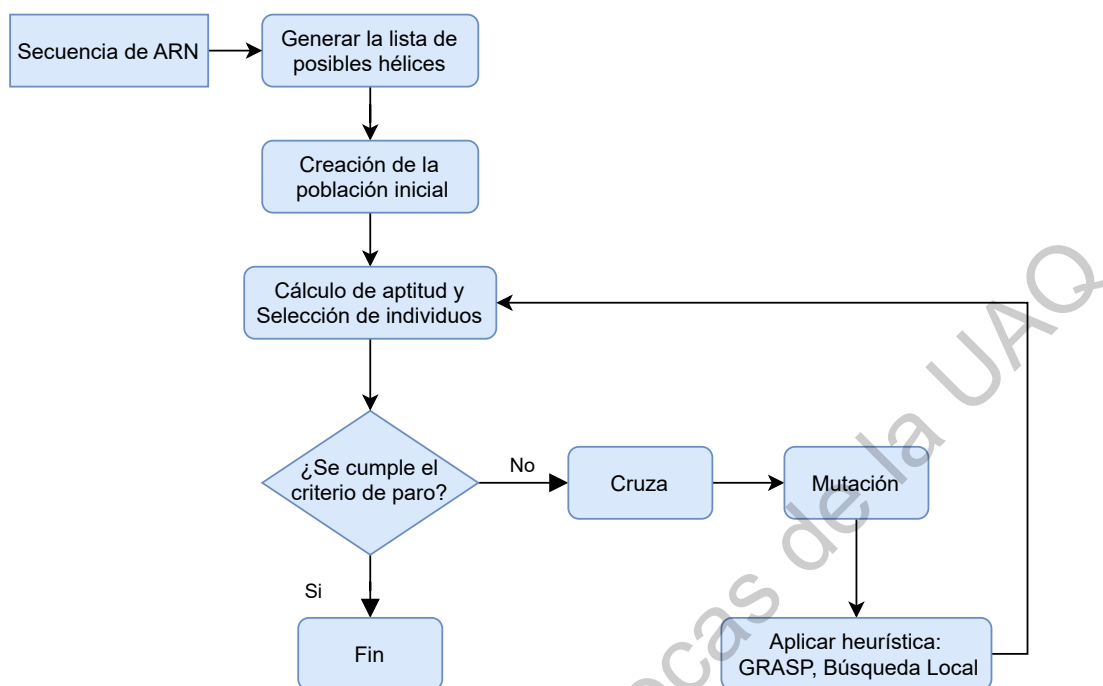
## Mutación

La mutación evita que el algoritmo quede atrapado en un mínimo local, recuperando genes perdidos. Este operador, requiere de un solo parental para producir un descendiente, normalmente se selecciona una posición aleatoria de un cromosoma aleatorio y se reemplaza el gen correspondiente por otro (Du y Swamy, 2016).

### 3.2.2. Algoritmos **GAknot**, **GA-GRASP** y **RNAknot**

En los últimos años se han propuesto una variedad de algoritmos genéticos, como **GAknot** (Tong et al., 2013), el cual genera un conjunto de hélices mediante un enfoque similar al del algoritmo Nussinov. Después, a partir de este conjunto de hélices se genera la población inicial, en donde cada individuo se crea a través de las permutaciones de las hélices previamente encontradas, los genes de cada individuo están representados por cada una de las hélices. Finalmente, se aplican los operadores genéticos de selección, cruza y mutación para obtener la solución.

El algoritmo propuesto en (Fatmi, Chentoufi, Bekri, Benhlima, y Sabbane, 2017) (**GA-GRASP**) combina un algoritmo genético y la metaheurística GRASP (Greedy Randomized Adaptive Search Procedure), en combinación del modelo termodinámico de energía libre. El algoritmo **RNAknot** (El Fatmi, Bekri, y Benhlima, 2019) es una mejora del algoritmo **GA-GRASP** que incluye la predicción de todos los tipos de subestructuras básicas (Figura 1.3, Subsección 1.4.2) y dos tipos de pseudonudos: simple (*H-type*) y *Kissing-Hairpin*, con un orden de complejidad  $O(n^2) + O(k \times nbs)$ , donde  $k$  es el número de iteraciones y  $nbs$  es el número de individuos en la población. En la Figura 3.20, se muestra el diagrama de flujo básico de estos AGs mencionados.



**Figura 3.20:** Diagrama de flujo del algoritmo genético básico.

Los algoritmos GAKnot, GA-GRASP y RNAknot siguen una idea similar en el proceso de creación los individuos que conforman la población inicial. En la Figura 3.21, se ilustra este proceso:

- **Construcción de la matriz:** se crea una matriz triangular  $M$  de tamaño  $n \times n$ , donde  $n$  es la longitud de la cadena de ARN ( $S$ ). Posteriormente, se procede al llenado de la parte inferior; se asigna un 1 en  $M_{(i,j)}$  si  $S_i$  y  $S_j$  son pares base complementarios y 0 en otro caso.
- **Lista de hélices:** se hace un barrido de la matriz  $M$  para identificar diagonales con 1's consecutivos (la cantidad de 1's consecutivos debe ser mayor o igual 3). Una vez identificadas las diagonales, estas se representan mediante una triada de enteros (*inicio*, *fin*, *longitud*). Adicionalmente, se tiene la restricción de la longitud mínima del bucle en horquilla ( $l_h$ ) que podría formarse en la estructura (se considera como longitud mínima de 3, ver Tabla 2.2). El valor de  $l_h$ , lo obtenemos de la siguiente forma:

$$bp_i = inicio + (longitud - 1)$$

$$bp_j = fin - (longitud - 1)$$

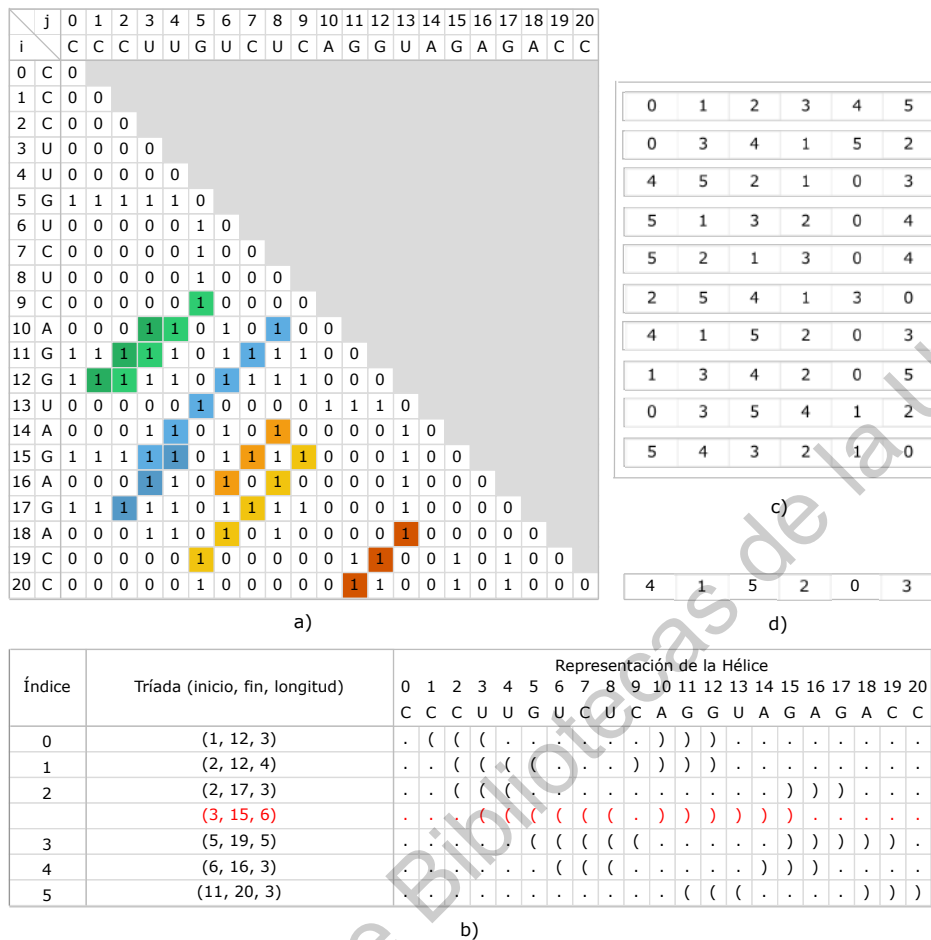
$$l_h = bp_j - (bp_i + 1)$$



Con esta información, se representa la estructura secundaria de la hélice en formato Dot-Bracket. Para ello, se crea una cadena de longitud  $n$  ( $S_r$ , conformada inicialmente por '.'), en donde los símbolos '.' son remplazados por '(' y ')' en las posiciones de las bases complementarias, es decir, desde la posición de inicio de la triada (izquierda a derecha) hasta  $l$  elementos (*longitud*) y desde la posición final (derecha a izquierda) menos  $l$  elementos. En la figura, se observa que la triada que no tiene un índice, no cumple la restricción de  $l_h \geq 3$ .

- **Individuo:** para la creación de un individuo, primero se define el tamaño del cromosoma. Una vez definido, cada gen que conforma el cromosoma es el identificador una hélice, de la lista encontrada previamente. La estructura secundaria de cada gen, se lee de izquierda a derecha, así, si dos hélices se sobreponen (es decir, existen pares base comunes en ambas hélices), la hélice de la izquierda conserva todos los pares base, y la hélice derecha solo conserva aquellos pares base que no se sobreponen.
- **Población:** la población inicial, se crea a partir de las permutaciones de los identificadores de cada hélice. Al tratarse de permutaciones, el número de posibles individuos con cromosoma de longitud  $n$ , donde  $n$  es la cantidad de hélices encontradas, sería  $n!$ . Para evitar el problema combinatorio, se crean cierto número de permutaciones (tamaño de la población) aleatorias entre el identificador mínimo y máximo de las hélices.

Por ejemplo, si se tienen los identificadores de hélice  $\{1, 2, \dots, 10\}$ , para cada individuo se generan números aleatorios diferentes en ese rango (tantos como la longitud del cromosoma), en donde cada número sería el identificador para una hélice. Para prevenir la generación de individuos con baja aptitud, una heurística simple, es dar prioridad a las hélices con menor energía libre.



**Figura 3.21:** Proceso de creación de la población inicial de un algoritmo genético: a) construcción de la matriz, b) lista de hélices, c) población, d) individuo. Figura adaptada de (Shahidul Islam y Rafiqul Islam, 2020).

### 3.2.3. Algoritmo Genético Propuesto

En la presente investigación, proponemos la implementación de un Algoritmo Genético para la predicción de estructuras secundarias de ARN utilizando el modelo termodinámico de energía NNDB e incorporando ideas de otros algoritmos del estado del arte. Las etapas principales del algoritmo son: determinar posibles hélices, generación de la población e iteración.

#### 3.2.3.1. Determinar posibles hélices

En esta etapa determinamos las hélices que podrían formar parte de la estructura secundaria, utilizando una heurística simple; la cual consiste en elegir las hélices con base en la probabilidad de ocurrencia de los pares base  $(i, j)$  en la secuencia de ARN. De

igual manera que en los algoritmos antes mencionados, utilizamos la restricciones sobre la cantidad mínima de pares base  $l$  y la longitud mínima del bucle  $l_h$ .

Para determinar la probabilidad de cada par base, utilizamos las funciones de partición de equilibrio del algoritmo de McCaskill (Algoritmo 3.5). Después de obtener la matriz de probabilidades, buscamos las diagonales con valores consecutivos (diferentes de cero), empezando desde la parte superior derecha hasta la parte inferior izquierda. Si en la diagonal contiene al menos  $l$  valores consecutivos, se guarda el arreglo de enteros representativo de la hélice  $(i, j, l, p)$ , donde  $i, j$  son las posiciones de inicio de la diagonal,  $l$  la cantidad de valores consecutivos y  $p$  es la probabilidad (Algoritmo 3.6).

---

**Algoritmo 3.6:** Hélices por probabilidad

---

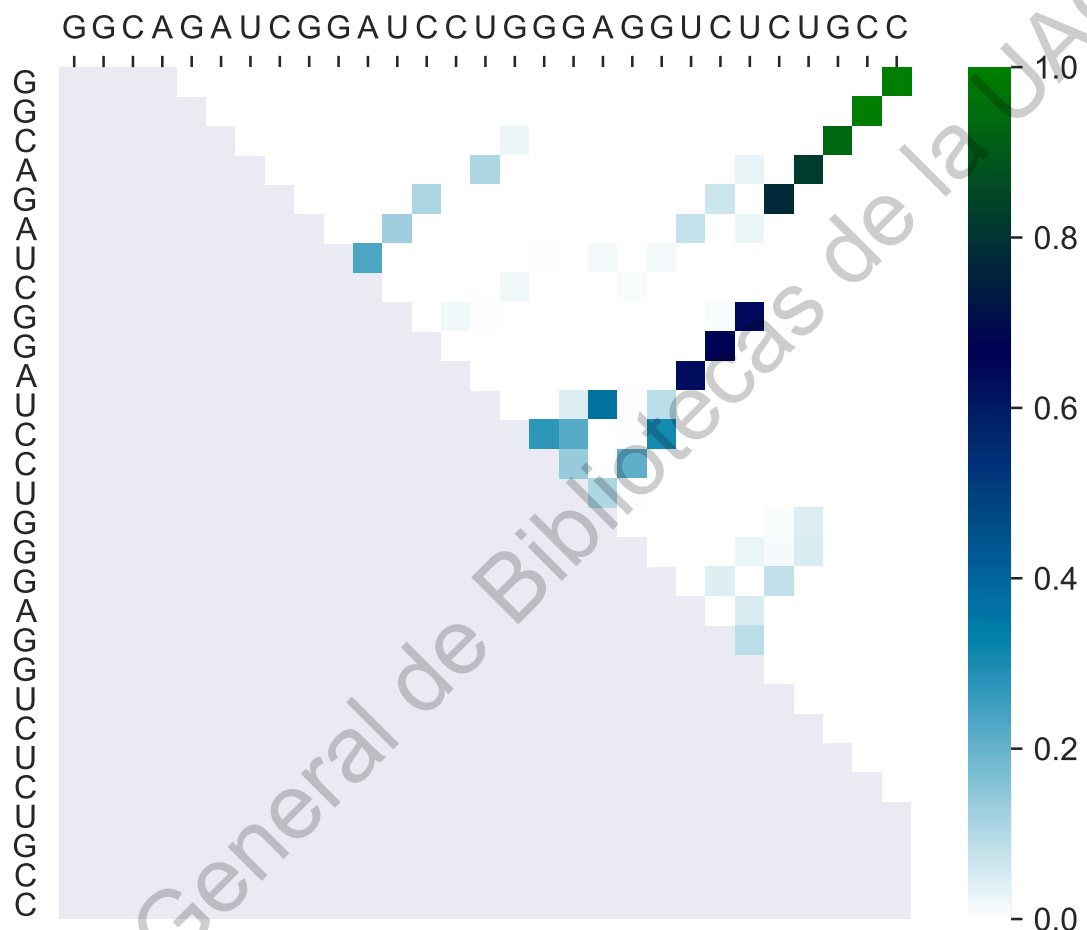
```

1 Entrada: Una secuencia de ARN  $r$ , la matriz de probabilidades  $m$ , el tamaño
  mínimo del bucle  $l_h$  y el número mínimo de pares base consecutivos  $l$ .
2 Salida: Lista de hélices  $S_L$ .
3 StemsByProbability( $r, m, l_h, l$ )
4    $n = \text{Length}(r)$ 
5    $S_L = \emptyset$ 
6    $mp = m$ 
7   for  $i = 0$  to  $n$  do
8     for  $j = n - 1$  to  $0$  do
9       if  $mp[i][j] > 0$  and  $mp[i + 1][j - 1] > 0$  then
10         $consecutive = 0$ 
11         $k = 0$ 
12        while True do
13          if  $mp[i + k][j - k] > 0$  then
14             $count + = 1$ 
15             $mp[i + k][j - k] = -1$ 
16          else
17            break
18          end
19           $k = k + 1$ 
20        end
21         $loopsize = \text{len}(\text{range}(i + consecutive, j - (consecutive - 1)))$ 
22        if  $consecutive \geq l$  and  $loopsize \geq l_h$  then
23           $p = m[i][j]$ 
24           $S_L := S_L \cup \{(i, j, l, p)\}$ 
25        end
26      end
27    end
28  end
29 end

```

---

Dada la secuencia  $r = \text{GGCAGAUCGGAUCCUGGGAGGUCUCUGCC}$ , el tamaño mínimo de la hélice  $l = 3$  y con  $l_h = 3$  como longitud mínima del bucle. Mediante el algoritmo de McCaskill (Algoritmo 3.5), se obtiene la matriz de probabilidades mostrada en la Figura 3.22.



**Figura 3.22:** Matriz de probabilidad obtenida mediante el algoritmo de McCaskill.

Procesando la matriz de probabilidades, la lista de hélices ( $L_S$ ) obtenidas se muestra en la Tabla 3.1. Para secuencias de gran tamaño, se pueden producir una gran cantidad de hélices; para lidiar con esto, en nuestra implementación agregamos la restricción de que la hélice debe tener una probabilidad mínima. Así, por ejemplo, si aplicamos el criterio de probabilidad mínima ( $p \geq 0.1$ ) a lista mostrada en la Tabla 3.1, solo se conservarían las hélices:  $\{4, 2, 5, 9\}$ .

**Tabla 3.1:** Lista de hélices obtenidas a partir de la matriz de probabilidades. Los valores de la triada son (inicio, fin, longitud).

Índice	Triada	Representación de la hélice																												Probabilidad
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	
1	(3, 23, 5)	. . .	(( ( ( ( . . . . . ) ) ) ) . . . . .	0.035																										
2	(8, 23, 5)	. . . . .	(( ( ( ( . . . . . ) ) ) ) . . . . .	0.639																										
3	(15, 24, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.011																										
4	(0, 28, 6)	(( ( ( ( ( . . . . . ) ) ) ) ) . . . . .	0.995																											
5	(12, 20, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.305																										
6	(15, 23, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.003																										
7	(7, 26, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.001																										
8	(6, 16, 4)	. . . . .	(( ( ( . . . . . ) ) ) . . . . .	0.007																										
9	(4, 12, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.113																										
10	(16, 25, 3)	. . . . .	(( ( . . . . . ) ) . . . . .	0.051																										
11	(2, 15, 3)	. .	(( ( . . . . . ) ) . . . . .	0.027																										

### 3.2.3.2. Generación de la población

Para la construcción de la población, primero determinamos la longitud del cromosoma ( $l_c$ ) de los individuos. Para ello, definimos la heurística siguiente: si el número de hélices ( $n_h$ ) encontradas en la etapa anterior es menor o igual a 5, entonces,  $l_c = n_h - 2$ , en otro caso,  $l_c = n_h/2$ .

Una vez determinada la longitud del cromosoma, generamos  $l_c$  números aleatorios (aleatoriedad guiada por la probabilidad de la hélice) en el rango de  $[1, n_h]$ , donde cada número generado es el identificador de una hélice en  $L_S$ , y que además, representa un gen del cromosoma. Adicionalmente, el individuo guarda dos listas (listas que serán de utilidad en la aplicación de los operadores genéticos): la primera, con los identificadores de las hélices complemento (es decir, aquellas que no están en el cromosoma) y la segunda, con el subconjunto de identificadores del cromosoma de aquellos genes que no contribuyen a la formación de la estructura secundaria.

Una vez definido el cromosoma, comenzamos la construcción de la estructura secundaria. Para ello, creamos un arreglo de tamaño igual a la longitud de la secuencia de

ARN y lo poblamos con el símbolo de bases libres '!'. Después, para cada gen, partiendo de los valores que definen a la hélice  $(i, j, l)$ , colocamos consecutivamente  $l$  símbolos '(' desde  $i$  en orden creciente, y  $l$  número consecutivos de ')' desde  $j$  en orden decreciente. Al realizar este proceso, es altamente probable que ocurran sobre-posiciones en las hélices (es decir, que existan pares base comunes entre las hélices). Para lidiar con esto, seguimos el método propuesto por Md. Shahidul (Shahidul Islam y Rafiqul Islam, 2020): si dos hélices se sobreponen, la última hélice agregada es truncada en la última posición validada  $l'$ , si  $l' \geq 3$  se reanuda el proceso con  $(i, j, l')$ , en caso contrario la hélice (gen) se descarta de la construcción de la estructura secundaria. En nuestra idea, el gen descartado se agrega a la lista de genes que no contribuyen a la formación de la estructura.

Por ejemplo, con las hélices de la Tabla 3.1 y con  $l_c = 5$  como longitud del cromosoma. Un posible individuo estaría definido por el cromosoma  $\{4, 2, 5, 9, 11\}$ . En la Tabla 3.2, se muestra el proceso de creación de la estructura secundaria de este individuo. El primer gen en agregarse a la estructura es el 4 generando la estructura temporal  $S_{r_t}$ . Enseguida, se agrega el gen 2 con el cual se observa una sobre-posición en un par base con respecto a  $S_{r_t}$ , este par base se omite y se obtiene la nueva estructura  $S_{r_t}$ . Con el gen 5, hay una sobre-posición en dos pares base, mientras que con los genes 9 y 11 se sobreponen todos los pares base, por lo cual los genes son descartados de la construcción de la estructura. Finalmente, se obtiene la estructura secundaria final  $S_r$  del individuo, a partir de ella y mediante la Ecuación 2.1 determinamos su aptitud (energía libre). La lista complemento de genes es  $\{1, 3, 6, 7, 8, 10\}$ , mientras que la lista de los genes que no contribuyen a la construcción estaría conformada por los genes  $\{5, 9, 11\}$ .

**Tabla 3.2:** Creación del individuo  $\{4, 2, 5, 9, 11\}$ .

Índice	Representación de la estructura secundaria																												Energía kcal/mol
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	
	G G C A G A U C G G A U C C U G G A G G U C U C U G C C																												
$S_{r_t}$	. . . . .																												
4	( ( ( ( ( ( . . . . . ) ) ) ) ) ) )																												-0.1
$S_{r_t}$	( ( ( ( ( ( . . . . . ) ) ) ) ) ) )																												-0.1
2	. . . . . ( ( ( ( ( . . . . . ) ) ) ) ) . . . . .																												-2.9
$S_{r_t}$	( ( ( ( ( ( . . . ( ( ( . . . . . ) ) ) ) ) ) ) ) ) )																												-10.3
5	. . . . . ( ( ( . . . ) ) ) . . . . .																												0.1
9	. . . . ( ( ( . . . ) ) ) . . . . .																												1.7
11	. . ( ( ( . . . . . ) ) ) . . . . .																												2.4
$S_r$	( ( ( ( ( ( . . . ( ( ( . . . . . ) ) ) ) ) ) ) ) ) )																												-10.3

Para la creación de la población inicial (*InitPop*), el proceso de creación de un individuo, se repite tantas veces como se especifique el tamaño de la población ( $\alpha$ ).

### 3.2.3.3. Iteración

#### Evaluación

Durante la evaluación de la población (Algoritmo 3.7), determinamos la probabilidad que cada individuo tiene de ser seleccionado. Primero, creamos una lista normalizada con los valores de aptitud de cada individuo. Después, obtenemos la probabilidad de cada individuo mediante la función de densidad de probabilidad exponencial. Adicionalmente, obtenemos la mejor aptitud, y el promedio de aptitud de la población.

---

#### Algoritmo 3.7: Evaluación

---

```

1 Entrada: Población de individuos  $p$ .
2 Salida: Lista de probabilidades de cada individuo  $P_L$ , mejor aptitud  $BestScore$  y
   aptitud promedio  $AvgScore$ .
3 Evaluation( $p$ )
   | // Lista de aptitud
4   |  $scores = \emptyset$ 
5   | foreach  $individual \in p$  do
6   | |  $scores = scores \cup \{individual.Energy\}$ 
7   | end
   | // Normalización
8   | for  $i = 0$  to  $Length(scores)$  do
9   | |  $scores[i] = \frac{scores[i] - \min(scores)}{\max(scores) - \min(scores)}$ 
10  | end
   | // Distribución de probabilidad
11  |  $dist_p = \emptyset$ 
12  | for  $i = 0$  to  $Length(scores)$  do
13  | |  $dist_p[i] = e^{-score[i]}$ 
14  | end
15  |  $dist_p = \frac{dist}{sum(dist)}$ 
16  | return  $dist_p, \min(score), avg(scores)$ 
17 end

```

---

El criterio de terminación del AG, esta dado por el número de generaciones ( $\delta$ ). Sin embargo, con el valor de mejor aptitud y aptitud promedio, podemos hacer un paro anticipado, si estos valores son iguales después de ciertas generaciones consecutivas.

## Selección

El método de selección de nuestro algoritmo es por elitismo. Es decir, seleccionamos los  $ne$  mejores individuos de la población con base a la energía libre, y los guardamos en una nueva lista, llamada  $Pop_1$ . El valor de  $ne$ , esta dado por el tamaño de la población multiplicado por la tasa de elitismo ( $\beta$ ),  $ne = \alpha \times \beta$ . Por otro lado, la selección de individuos para el operador de Cruza se realiza de manera pseudoaleatoria, con base a la probabilidad del individuo (probabilidad obtenida en la Evaluación).

## Cruza

En este operador, seleccionamos dos individuos  $p_1$  y  $p_2$  de la población inicial y los cruzamos para obtener dos nuevos descendientes ( $c_1$  y  $c_2$ ), que son agregados en una nueva lista llamada  $Pop_2$ . La cantidad de cruzamientos, esta determinada por el número  $nc = (\alpha - ne)/2$ .

En nuestro algoritmo utilizamos un método de cruza multipunto similar al propuesto en (Shahidul Islam y Rafiqul Islam, 2020). En el método de Shahidul, se seleccionan dos números enteros aleatorios en el rango de 0 a  $l_c$  (tamaño del cromosoma). Estos números sirven como pivote para particionar el cromosoma en tres segmentos, los segmentos pares de intercambian entre  $p_1$  y  $p_2$ , mientras que los impares se conservan; generando así los cromosomas de los nuevos descendientes. Bajo este enfoque, cabe la posibilidad de que en un cromosoma se dupliquen genes si en el parental existe algún gen en común.

Debido, al hecho de que un gen duplicado dentro de un cromosoma no contribuiría a la construcción de la estructura secundaria. En nuestro algoritmo (Algoritmo 3.8) hemos agregado dos validaciones que impidan que un gen se duplique: 1) agregar el gen original del parental en la posición del gen que se duplicaría y 2) agregar el gen complementario del descendiente con respecto a su parental en caso de que se duplique el último gen. En la Figura 3.23, se muestra un ejemplo de cruzamiento usando el método original y el modificado. En el proceso original (figura izquierda), se observa que ambos descendientes tienen genes duplicados. Mientras, que en el proceso modificado (figura derecha), las líneas punteadas indican la duplicidad de genes que se produciría.



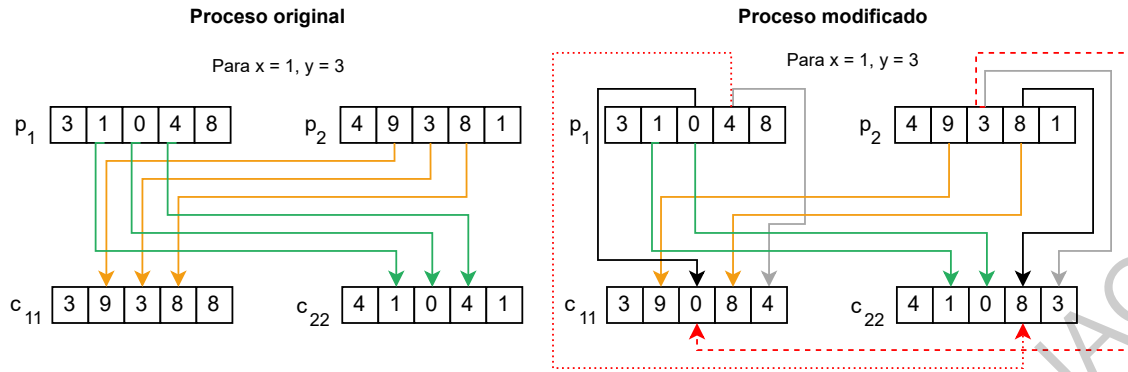


Figura 3.23: Proceso de cruzamiento.

A continuación, se muestra el pseudocódigo para el cruzamiento de dos individuos.

---

**Algoritmo 3.8:** Cruza

---

```

1 Entrada: Individuos  $p_1$  y  $p_2$ .
2 Salida: Descendientes  $c_1$  y  $c_2$ 
3 Crossover( $p_1, p_2$ )
4    $x, y = \text{random}(0, \text{Length}(p_1))$ 
5   for  $i = 0$  to  $\text{Length}(p_1)$  do
6     if  $i < x$  or  $i > y$  then
7       if  $p_1.\text{Gene}(i).\text{Index}$  not in  $c_1.\text{ChromosomeList}$  then
8          $c_1[i] = p_1[i]$ 
9       else
10         $c_1[i] = \bar{c}_1.\text{Chromosome}$ 
11      end
12      if  $p_2.\text{Gene}(i).\text{Index}$  not in  $c_2.\text{ChromosomeList}$  then
13         $c_2[i] = p_2[i]$ 
14      else
15         $c_2[i] = \bar{c}_2.\text{Chromosome}$ 
16      end
17    else if  $x \leq i$  and  $i \leq y$  then
18      if  $p_2.\text{Gene}(i).\text{Index}$  not in  $c_1.\text{ChromosomeList}$  then
19         $c_1[i] = p_2[i]$ 
20      else
21         $c_1[i] = p_1[i]$ 
22      end
23      if  $p_1.\text{Gene}(i).\text{Index}$  not in  $c_2.\text{ChromosomeList}$  then
24         $c_2[i] = p_1[i]$ 
25      else
26         $c_2[i] = p_2[i]$ 
27      end
28    end
29  return  $c_1, c_2$ 
30 end

```

---

## Mutación

Para el operador de mutación, elegimos aleatoriamente un parental ( $p_1$ ) de  $Pop_2$ . De este individuo, seleccionamos aleatoriamente dos genes, dando prioridad de selección aquellos que no contribuyen a la construcción de la estructura secundaria, y que al ser reemplazados por genes de la lista complemento, podrían mejorar al individuo descendiente. Los genes de la lista complemento, son elegidos con base a su probabilidad. En el Algoritmo 3.9, se muestra este proceso.

---

### Algoritmo 3.9: Mutación

---

```
1 Entrada: Un parental  $p_1$ .
2 Salida: El descendiente  $c_1$ 
3 Mutation( $p_1$ )
4    $c_1 = p_1$ 
   // Seleccionar identificadores de 2 genes
5   if  $Length(c_1.NoEffectStems) \geq 2$  then
6     |  $x\_index, y\_index = random(c_1.NoEffectStems)$ 
7   else
8     |  $x\_index, y\_index = random(c_1.ChromosomeList)$ 
9   end
   // Identificar posición de los genes
10  for  $j = 0$  to  $Length(c_1.Chromosome)$  do
11    | if  $c_1.Chromosome[i].Index == x\_index$  then
12      |  $x = i$ 
13    | else if  $c_1.Chromosome[i].Index == y\_index$  then
14      |  $y = i$ 
15    | end
16  end
   // Intercambiar los genes
17   $g_1, g_2 = random(c_1.ChromosomePool, Prob)$ 
18   $m11.Chromosome[x] = g_1$ 
19   $m11.Chromosome[y] = g_2$ 
20  return  $c_1$ 
21 end
```

---

Una vez creado el nuevo descendiente ( $c_1$ ), si su aptitud es mejor que la del parental, reemplazamos  $c_1$  por  $p_1$  en  $Pop_2$ . El operador de mutación, se repite  $nc$  veces en cada generación del AG.

### 3.2.3.4. Pseudocódigo

A continuación, se muestra el algoritmo de nuestra propuesta.

---

**Algoritmo 3.10:** Algoritmo Genético

---

```
1 Entrada: Una secuencia de ARN  $r$ , tamaño de la población  $\alpha$ , tasa de elitismo  $\beta$ 
   y el número de iteraciones  $\delta$ .
2 Salida: Individuo  $X$  con la mínima energía libre.
3 GeneticAlgorithm( $r, m, l_h, l$ )
   // INICIALIZACIÓN
   // Obtener las hélices
4    $S_L = \mathbf{StemsByProbability}(r, m, l_h, l)$ 
   // Generar la población inicial
5    $InitPop = \text{Generar } \alpha \text{ individuos}$ 
   // Evaluar la población
6    $BestScore, AvgScore, Prob = \mathbf{Evaluation}(InitPop)$ 
   // ITERACIÓN
7   for  $i = 0$  to  $\delta$  do
   | // Selección ( $ne =$  número de elitismo)
   |  $ne = \alpha \times \beta$ 
   |  $Pop_1 = \text{Seleccionar las mejores } ne \text{ soluciones de } InitPop$ 
   | // Cruza ( $nc =$  número de cruzamientos)
   |  $nc = (\alpha - ne)/2$ 
   | for  $j = 0$  to  $nc$  do
   | | // Seleccionar dos individuos aleatorios
   | |  $x, y = \text{random}(0, \alpha, Prob)$ 
   | |  $c_1, c_2 = \mathbf{Crossover}(InitPop[x], InitPop[y])$ 
   | |  $Pop_2 := Pop_2 \cup \{c_1, c_2\}$ 
   | end
   | // Mutación
   | for  $j = 0$  to  $nc$  do
   | | // Seleccionar aleatoriamente un individuo de  $Pop_2$ 
   | |  $x = \text{random}(0, nc)$ 
   | |  $c_1 = \mathbf{Mutation}(Pop_2[x])$ 
   | | if  $c_1.Energy < Pop_2[x].Energy$  then
   | | |  $Pop_2[x] = c_1$ 
   | | end
   | end
   |  $InitPop = Pop_1 + Pop_2$ 
   |  $BestScore, AvgScore, Prob = \mathbf{Evaluation}(InitPop)$ 
   | if  $BestScore == AvgScore$  then
   | | break
   | end
8 end
9 return El mejor individuo  $X \in InitPop$ 
10 end
```

---

El orden de complejidad aproximado del algoritmo es de  $O(n^3) + O(k * p)$ , donde  $n$  es la longitud de la secuencia de entrada,  $k$  el número de iteraciones y  $p$  el tamaño de la población.

A continuación, en la Tabla 3.3 se muestra un resumen de las características de los algoritmos implementados en la presente tesis.

**Tabla 3.3:** Caracterización de los algoritmos para modelación de estructuras secundarias.

Algoritmo	Complejidad	Criterio de optimización	Predicción de pseudonodos
Nussinov	$O(n^3)$	Maximización de Pares Base	N
Zuker	$O(n^3)$	Minimización de Energía Libre	N
Akutsu	$O(n^4)$	Minimización de Energía Libre	S
Genético	$O(n^3) + O(k * p)$	Minimización de Energía Libre	S

# Resultados y Discusión

*Las máquinas deben trabajar. La gente debe pensar.*

—RICHARD HAMMING (1915 - 1998)

## 4.1. Métricas de precisión

Para evaluar la precisión de la predicción de los métodos implementados, se siguió la técnica descrita en (Zhao, Wang, Zeng, y Xiao, 2018), en donde se utilizan los conceptos de *sensibilidad* y *valor predictivo positivo*.

La *sensibilidad* ( $S$ ), representa el porcentaje de pares base presentes en la estructura de referencia que ocurren en la estructura secundaria predicha (Ecuación 4.1).

$$S = \frac{VP}{VP + FN} \quad (4.1)$$

El *valor predictivo positivo* ( $VPP$ ), describe que porcentaje de pares base predichos ocurren en la estructura secundaria de referencia (Ecuación 4.2).

$$VPP = \frac{VP}{VP + FP} \quad (4.2)$$

Donde,  $VP$  es el número de pares base predichos correctamente,  $FN$  el número de pares base en la estructura de referencia que no están en la estructura predicha y  $FP$  es el número de pares base predichos que no están en la estructura de referencia.

Si  $S$  es igual a 1 significa que la estructura predicha contiene todos los pares base correctos, es decir, contiene los mismos pares base que la estructura de referencia, aunque también puede contener pares base falsos. Si  $VPP$  es igual 1, la estructura predicha sólo contiene los pares base correctos. Por ende, si ambos valores son igual a 1, significa que la predicción de la estructura secundaria fue perfecta (Hamada, Kiryu, Sato, Mituyama, y Asai, 2008)



A continuación, en el Ejemplo 4.1 se muestran las medidas de evaluación, aplicadas a la instancia PDB\_00136.

**Ejemplo 4.1.** *Con los pares base de la estructura de referencia y la estructura predicha, mostrados en la Figura 4.1. Calcular la S, VPP, Valor-F y MCC de la estructura predicha.*

*Con los pares base en ambas estructuras, se determina que  $VP = 19$ ,  $FN = 7$  y  $FP = 5$ . Sustituyendo estos valores en las Ecuaciones (4.1, 4.2 y 4.3), tenemos que:*

$$S = \frac{VP}{VP + FN} = \frac{19}{19 + 7} = 0.7308$$

$$VPP = \frac{VP}{VP + FP} = \frac{19}{19 + 5} = 0.7917$$

$$\text{Valor-F} = \frac{2 \times S \times VPP}{S + VPP} = \frac{2 \times 0.7308 \times 0.7917}{0.7308 + 0.7917} = 0.76$$

*Para MCC, primero determinamos VN sustituyendo los valores previamente conocidos:*

$$VN = \frac{n \times (n - 1)}{2} - VP - FN - FP = \frac{70 \times (70 - 1)}{2} - 19 - 7 - 5 = 2384$$

*Finalmente, sustituimos los valores en la Ecuación 4.4:*

$$\begin{aligned} MCC &= \frac{VP \times VN - FP \times FN}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}} \\ &= \frac{19 \times 2384 - 5 \times 7}{\sqrt{(19 + 5)(19 + 7)(2384 + 5)(2384 + 7)}} = 0.758113 \end{aligned}$$

## 4.2. Resultados Experimentales

Para validar los algoritmos de la presente tesis, utilizamos dos conjuntos de datos: uno con estructuras secundarias libres de pseudonudos, y el otro con pseudonudos de tipo simple y Kissing Hairpin; los conjuntos de datos se conforman por secuencias obtenidas de las bases de datos *RNA STRAND* (Andronescu et al., 2008) y *PseudoBase ++* (Taufer et al., 2009), respectivamente. Para cada una de las secuencias se cuenta con la estructura secundaria de referencia, la cual fue obtenida mediante experimentación (*Rayos*

*X*, *Resonancia Magnética Nuclear* (NMR)) o *Análisis de Secuencia Comparativo* (CSA), de acuerdo con las fuentes de origen.

Respecto a las base de datos, RNA STRAND es una base de datos pública que proporciona una amplia colección de estructuras secundarias de ARN de diferentes tipos de organismos. RNA STRAND contiene secuencias de otras base de datos como RCSB Protein Data Bank (Westbrook, Feng, Chen, Yang, y Berman, 2003). Para mantenerse al día ofrece herramientas para agregar estructuras secundarias de ARN descubiertas por otros investigadores a la base de datos. En la actualidad, la base de datos cuenta con 94 citas en PubMed (<https://pubmed.ncbi.nlm.nih.gov/18700982/>, Consulta: Enero 2021), de las cuales las más recientes corresponden a artículos publicados entre los años 2019 y 2021. Por su parte PseudoBase++, es una base de datos con una amplia colección de estructuras secundarias con pseudonudos desarrollada por la Universidad de Leiden. Una característica de PseudoBase++ es que vincula sus registros de estructuras secundarias, con los registros de secuencias en GenBank. Lo permite entre muchas cosas, visualizar las estructuras y obtener una mayor información sobre las secuencias.

Debido a que ambas bases datos son de acceso público y cuentan con amplios catálogos de estructuras secundarias, así como numerosas citas en trabajos de investigación. En este trabajo de tesis consideramos adecuado su uso para la extracción de los conjuntos de datos de prueba.

### **Estructuras Secundarias Libres de Pseudonudos**

En la Tabla 4.1, se muestra el conjunto de datos seleccionado para las estructuras libres de pseudonudos. Este conjunto de datos se conforma por 20 secuencias de longitud variable en el rango de 60 a 237 nucleótidos. Para cada una de las instancia, se tiene la clave (identificador en la base de datos de origen), tipo de ARN, longitud (cantidad de nucleótidos en la cadena, denotado por *nt*) y el método de validación de la estructura.



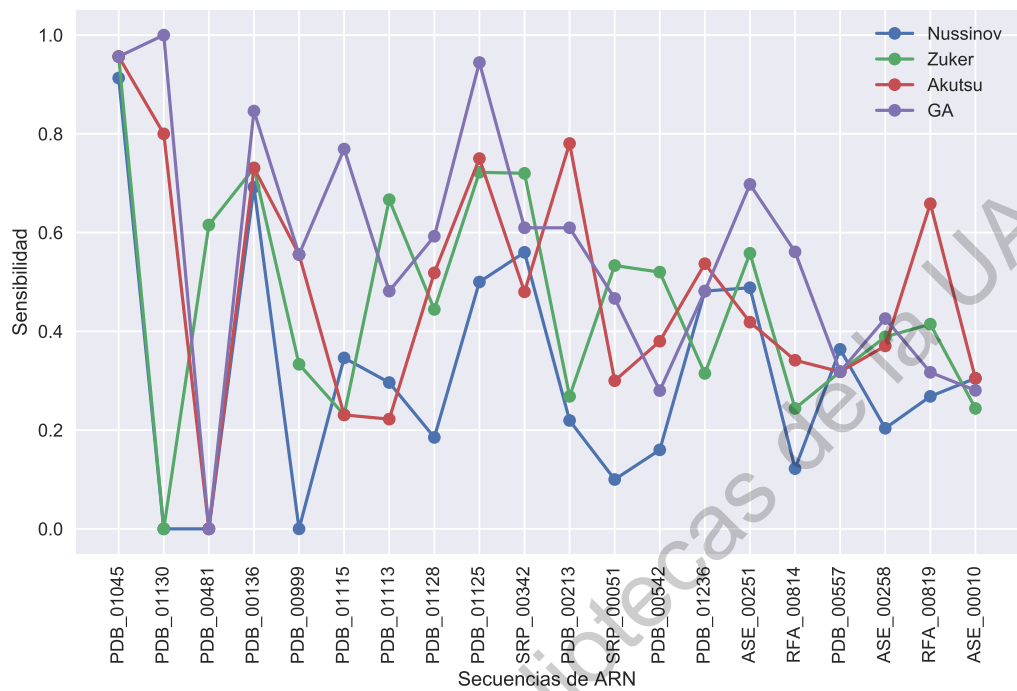
**Tabla 4.1:** Secuencias de ARN sin pseudonudos en la estructura de referencia.

Clave	Tipo	Longitud <i>nt</i>	Método de Validación
PDB_01045	ARN Sintético	50	Rayos X
PDB_01130	ARN Sintético	60	Rayos X
PDB_00481	ARN Sintético	62	Rayos X
PDB_00136	Intron Grupo II	70	Rayos X
PDB_00999	ARN de Transferencia	70	Rayos X
PDB_01115	ARN Sintético	74	Rayos X
PDB_01113	ARN Sintético	75	Rayos X
PDB_01128	ARN Sintético	75	NMR
PDB_01125	ARN Sintético	86	NMR
SRP_00342	Otro	90	CSA
PDB_00213	ARN Sintético	101	NMR
SRP_00051	ARN Sintético	105	CSA
PDB_00542	ARN Sintético	126	Rayos X
PDB_01236	ARN Sintético	153	Rayos X
ASE_00251	RNasa P	189	CSA
RFA_00814	Otro	226	CSA
PDB_00557	ARN Sintético	226	Rayos X
ASE_00258	RNasa P	232	CSA
RFA_00819	Otro	237	CSA
ASE_00010	RNasa P	316	CSA

Cada secuencia del conjunto de datos fue probada con los cuatro algoritmos expuestos en la presente tesis. El experimento consistió en realizar la ejecución de cada algoritmo 10 veces con cada una de las instancias. La idea de ejecutar cada algoritmo esa cantidad de veces es para obtener el tiempo promedio de ejecución, y dada la aleatoriedad presente en el algoritmo genético obtener las métricas promedio de cada instancia de prueba.

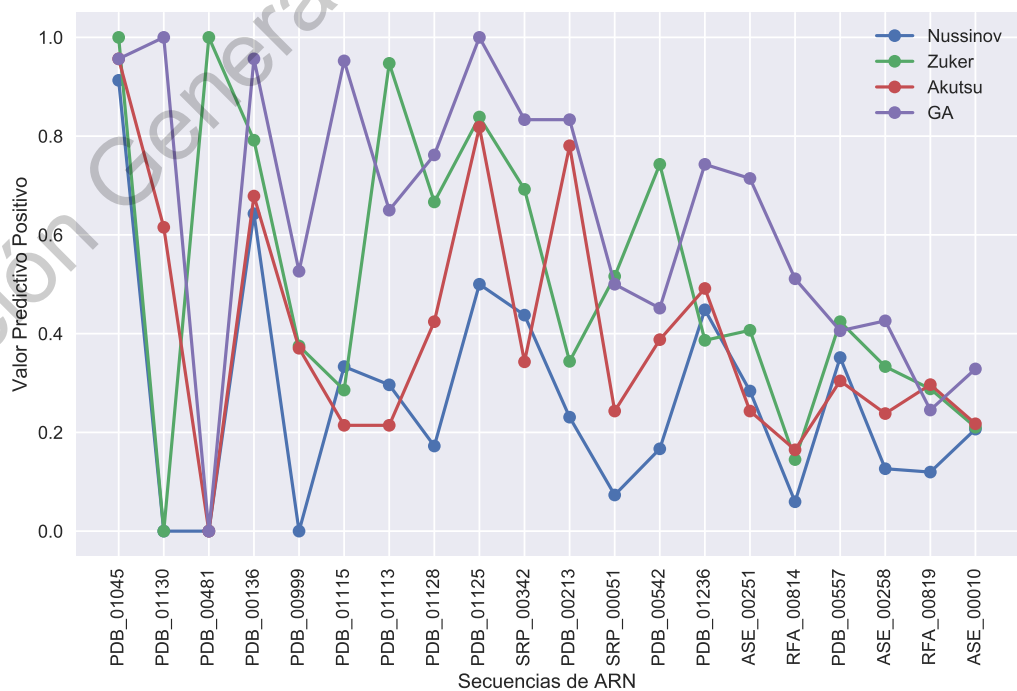
En la Figura 4.2, se muestra el valor de Sensibilidad. En la gráfica se observa que el Algoritmo Genético es el que obtiene una mayor sensibilidad, lo que significa que la estructura secundaria predicha tiene un alto grado de pares base coincidentes con la estructura de referencia, aunque también contiene pares base que no están presentes en la

estructura de referencia. Seguido por el algoritmo de Zuker, Akutsu y finalmente Nussinov.



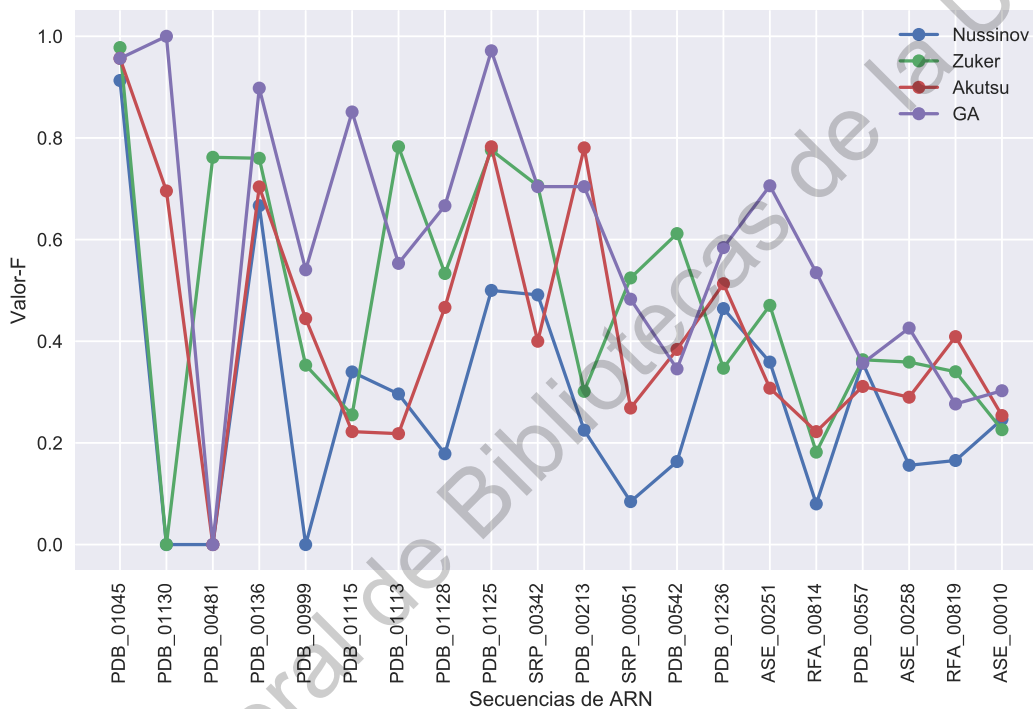
**Figura 4.2:** Sensibilidad de los algoritmos para instancias sin pseudonudos.

En la Figura 4.3, se muestran los valores obtenidos para la métrica Valor Predictivo Positivo.



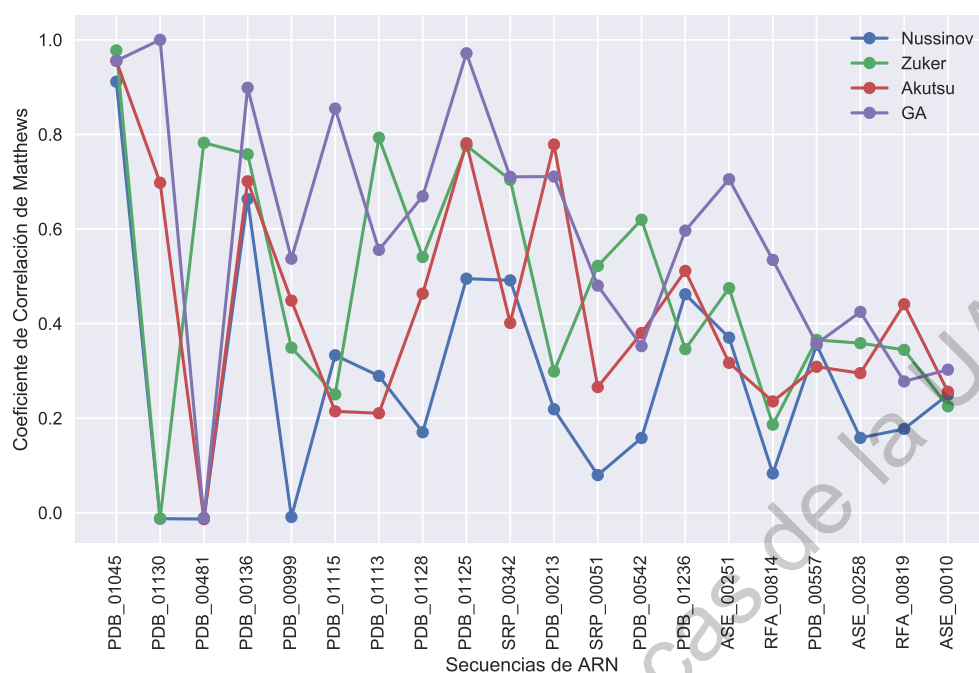
**Figura 4.3:** Valor Predictivo Positivo para instancias sin pseudonudos.

En la gráfica de la Figura 4.4, se muestran los valores obtenidos correspondientes a la métrica Valor F. Para el Algoritmo Genético el Valor F promedio es de 59.31 %, con valor mínimo de 0 % y máximo de 100 %. Para Zuker el promedio es 48.15 %, mínimo de 0 % y máximo 97.78 %. Mientras, que para Akutsu el promedio es de 43.15 %, con mínimo de 0 % y máximo 95.65 %. Finalmente, el promedio para el algoritmo de Nussinov es 28.43 %, con valor mínimo de 0 % y máximo 91.3 %.



**Figura 4.4:** Valor-F de los algoritmos para instancias sin pseudonudos.

Con respecto al MCC, en la Figura 4.5 se muestran los valores obtenidos. Para esta métrica el valor promedio del Algoritmo Genético es de 59.42 %, con valor mínimo de  $-0.01$  y máximo de 1. Para Zuker el promedio es 48.30 %, mínimo de  $-0.012$  y máximo 0.97 %. Mientras que para Akutsu el promedio es de 43.25 %, con mínimo de  $-0.012$  y máximo 0.95. Finalmente, el promedio para el algoritmo de Nussinov es 28.15 %, con valor mínimo de  $-0.013$  y máximo 0.91.



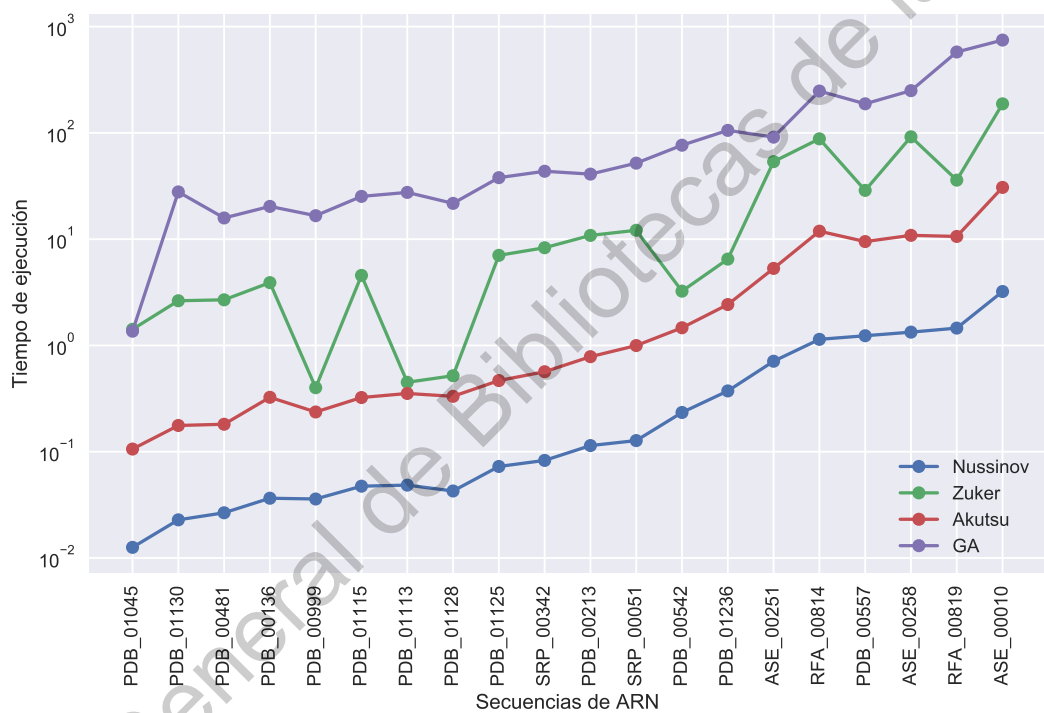
**Figura 4.5:** Coeficiente de Correlación de Matthews para instancias sin pseudonudos.

A partir de los resultados mostrados en las gráficas anteriores, se puede observar que el Algoritmo Genético es el que tiene una mejor predicción de acuerdo a los métricas planteadas y con respecto a la estructura secundaria de referencia. También, se puede observar una variabilidad de la precisión con respecto a las instancias y para algunos casos también respecto al algoritmo de predicción.

Como se menciono anteriormente, si bien existe una variabilidad con respecto a las diferentes instancias. Los resultados obtenidos por el Algoritmo Genético son mejores, esto se debe un buena parte a la combinación de soluciones factibles (población inicial) durante las diferentes etapas y en las múltiples iteraciones del algoritmo. Como se describió anteriormente, la población inicial se crea a partir de la combinación de hélices factibles encontradas probabilísticamente por el algoritmo de McCaskill, lo cual de cierto modo asegura que la estructura secundaria predicha se conforme por bares base viables.

En la Figura 4.6, se muestran los tiempos de ejecución de cada algoritmo por instancia. Se puede observar que el Algoritmo Genético es el que demanda mayor tiempo de procesamiento, seguido por el algoritmo de Zuker, Akutsu y finalmente Nussinov. Con

respecto al Algoritmo Genético, la mayor parte del consumo de tiempo esta dada por la etapa de pre-procesamiento; en la cual mediante el algoritmo de McCaskill se obtiene la probabilidad de los pares base y posteriormente en la etapa de creación de individuos en la cual se combinan las hélices encontradas para la construcción de la estructura secundaria representativa del individuo. Si bien el algoritmo de Zuker tiene un orden de complejidad menor que Akutsu, la diferencia de tiempos entre ambos puede estar relacionada al hecho de que la implementación de Akutsu se realizó en un lenguaje compilado (Java) y Zuker al igual que los otros algoritmos se implementó en un lenguaje interpretado (Python).



**Figura 4.6:** Tiempo de ejecución de los algoritmos para instancias sin pseudonudos. La gráfica de tiempos de ejecución esta en escala logarítmica debido a que existe una diferencia significativa entre los tiempos de ejecución de cada algoritmo.

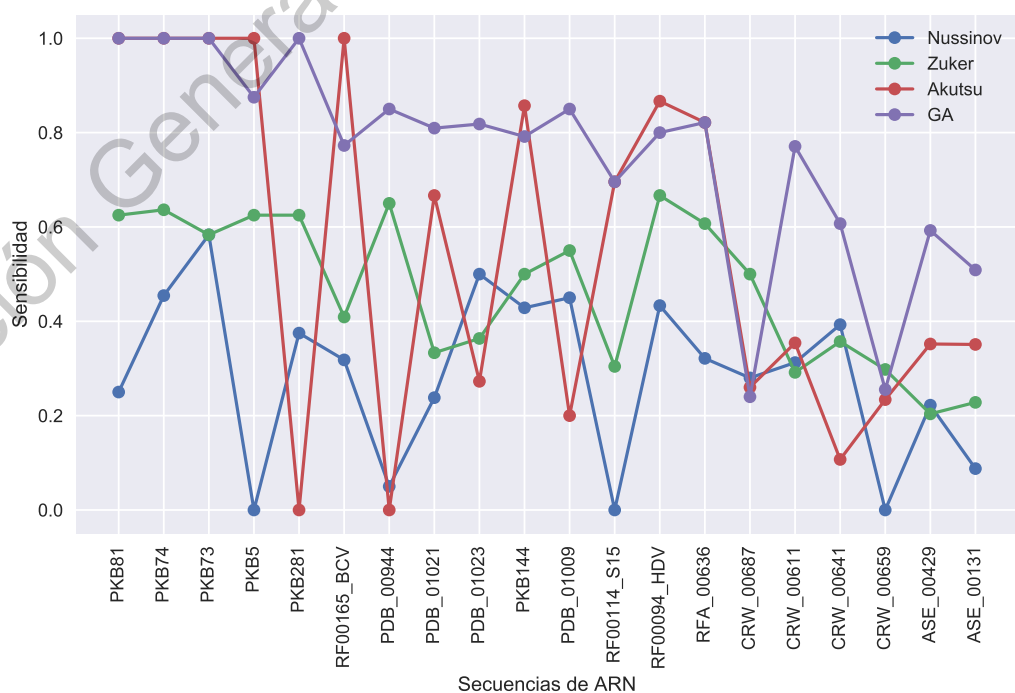
### Estructuras Secundarias con Pseudonudos

En la Tabla 4.2, se muestra el conjunto de datos para las estructuras con pseudonudos. Este conjunto de datos se conforma por 20 secuencias de longitud variable en el rango de 26 a 121 nucleótidos. Para cada instancia, se tiene la clave (identificador en la base de datos de origen), tipo de ARN, longitud (número de nucleótidos en la cadena, *nt*) y el método de validación de la estructura.

**Tabla 4.2:** Secuencias de ARN con pseudonudos en la estructura de referencia.

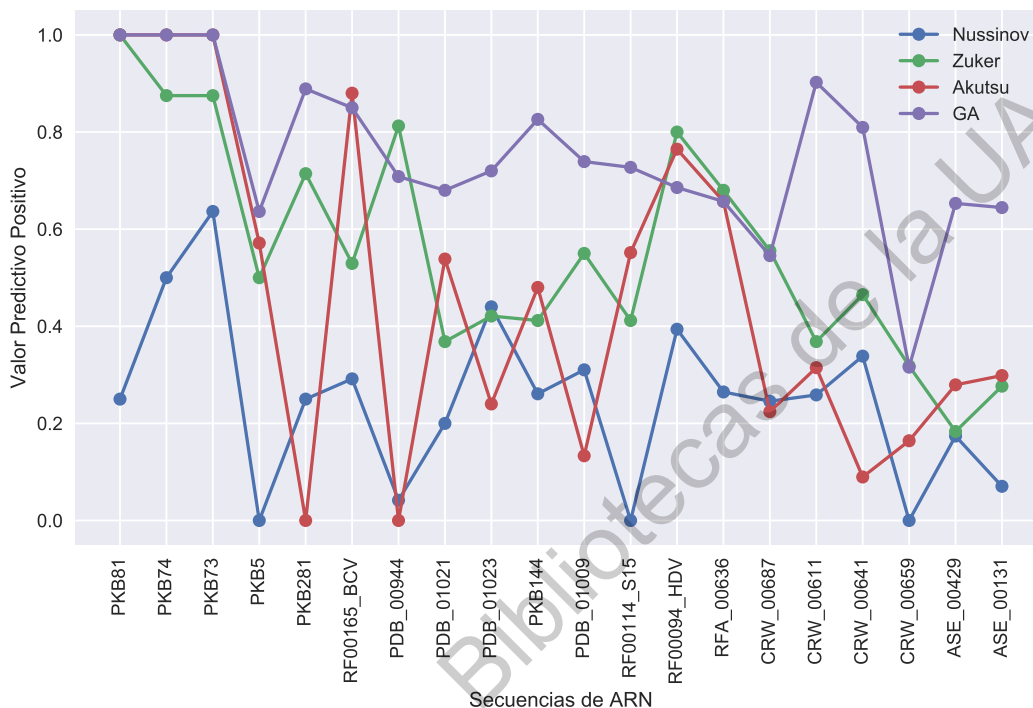
Clave	Tipo	Longitud <i>nt</i>	Método validación
PKB81	ARN Sintético	26	NMR
PKB74	ARNm	28	NMR
PKB73	ARNm	33	NMR
PKB5	ARNt viral	41	NMR
PKB281	Cambio de marco viral	43	NMR
RF00165_BCV	-	63	-
PDB_00944	ARN Sintético	65	Rayos X
PDB_01021	ARN Sintético	68	Rayos X
PDB_01023	ARN Sintético	68	Rayos X
PKB144	ARNt viral	71	CSA
PDB_01009	-	71	Rayos X
RF00114_S15	-	74	-
RF00094_HDV	-	87	-
RFA_00636	VHD	90	CSA
CRW_00687	Grupo Intron I	153	CSA
CRW_00611	Grupo Intron I	167	CSA
CRW_00641	Grupo Intron I	168	CSA
CRW_00659	Grupo Intron I	170	CSA
ASE_00429	RNasa P	189	CSA
ASE_00131	RNasa P	195	CSA

Al igual que para las instancias sin pseudonudos, cada secuencia de este conjunto fue probada por los cuatro algoritmos. Realizándose la ejecución 10 veces por cada instancia. En la Figura 4.7, se muestra el valor de la métrica Sensibilidad.



**Figura 4.7:** Sensibilidad de los algoritmos para instancias con pseudonudos.

En la Figura 4.8, se muestran los valores obtenidos para métrica Valor Predictivo Positivo. Tanto en la gráfica anterior como en la de esta figura, se observa que el Algoritmo Genético es el que tiene una mayor precisión, seguido por Akutsu, Zuker y finalmente Nussinov.



**Figura 4.8:** Valor Predictivo Positivo de los algoritmos para instancias con pseudonudos.

Con respecto a la gráfica de la métrica Valor F, en la Figura 4.9 se muestran los valores obtenidos por los algoritmos. El promedio para el Algoritmo Genético es de 74.38 %, valor mínimo de 28.24 % y máximo de 100 %. Para el algoritmo de Akutsu el promedio es de 49.73 %, valor mínimo de 0 % y máximo 100 %. Por su parte, el algoritmo de Zuker obtuvo un promedio de 50.39 %, con un valor mínimo de 25 % y máximo de 76.92 %. Finalmente, Nussinov con promedio de 26.23 %, con un mínimo de 0 % y máximo de 60.87 %.

En la Figura 4.10, se muestran los valores obtenidos por los algoritmos correspondientes al MCC. Para esta métrica, el Algoritmo Genético tiene un promedio de 74.59 %, Akutsu un promedio de 49.78 %, Zuker 50.43 % y por último Nussinov con un promedio de 25.67 %.

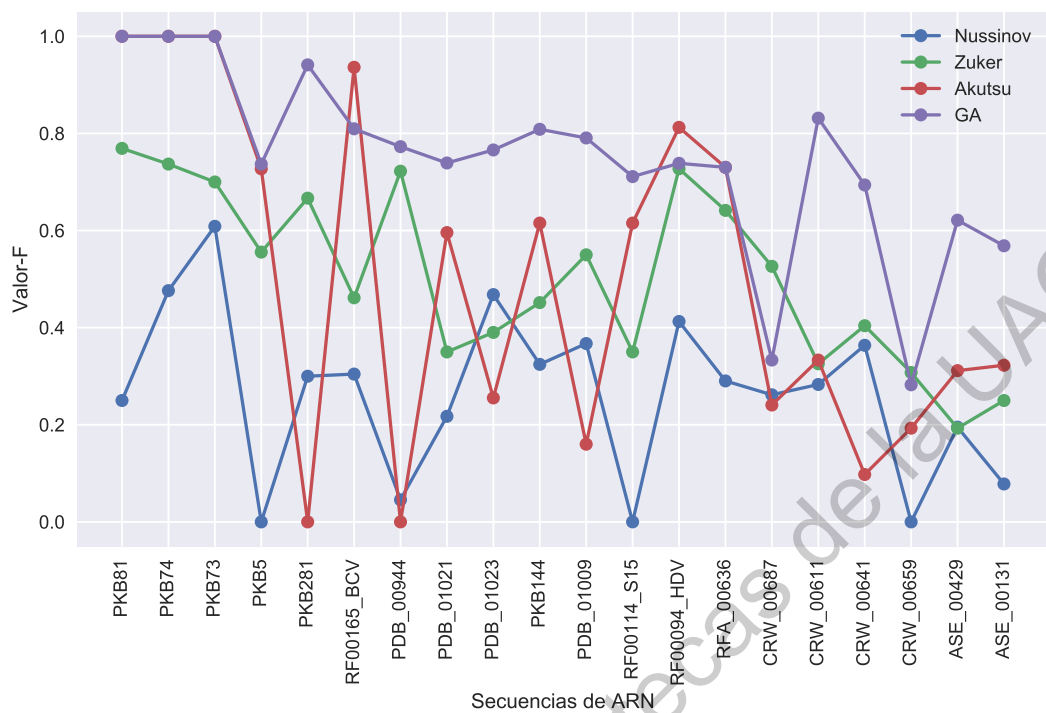


Figura 4.9: Valor-F de los algoritmos para instancias con pseudonudos.

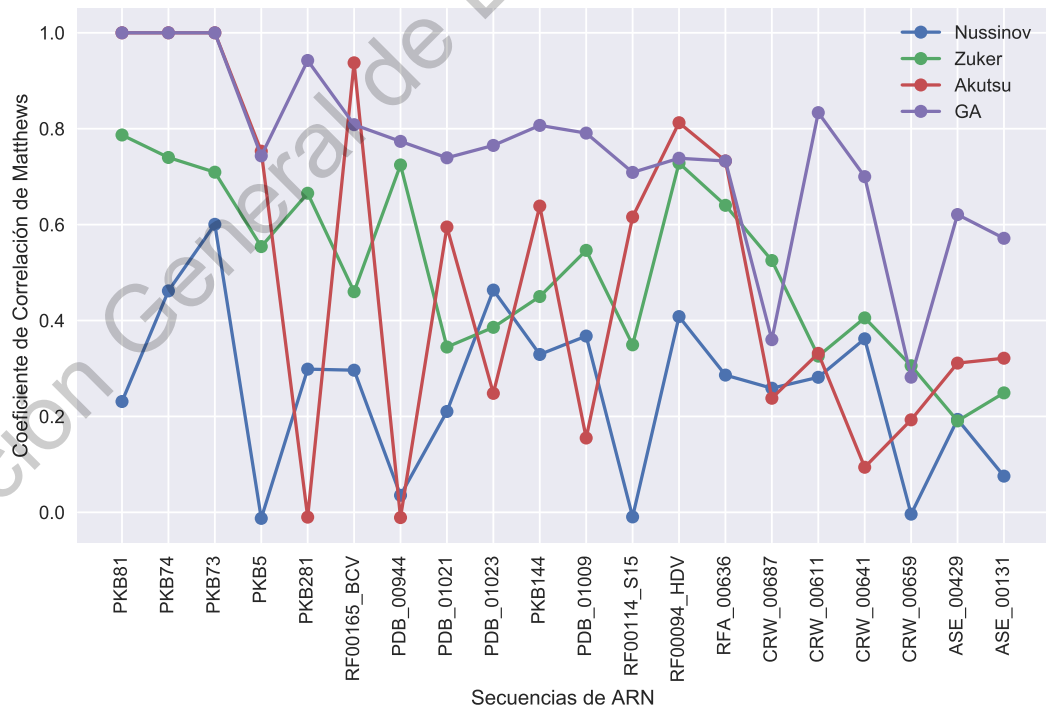
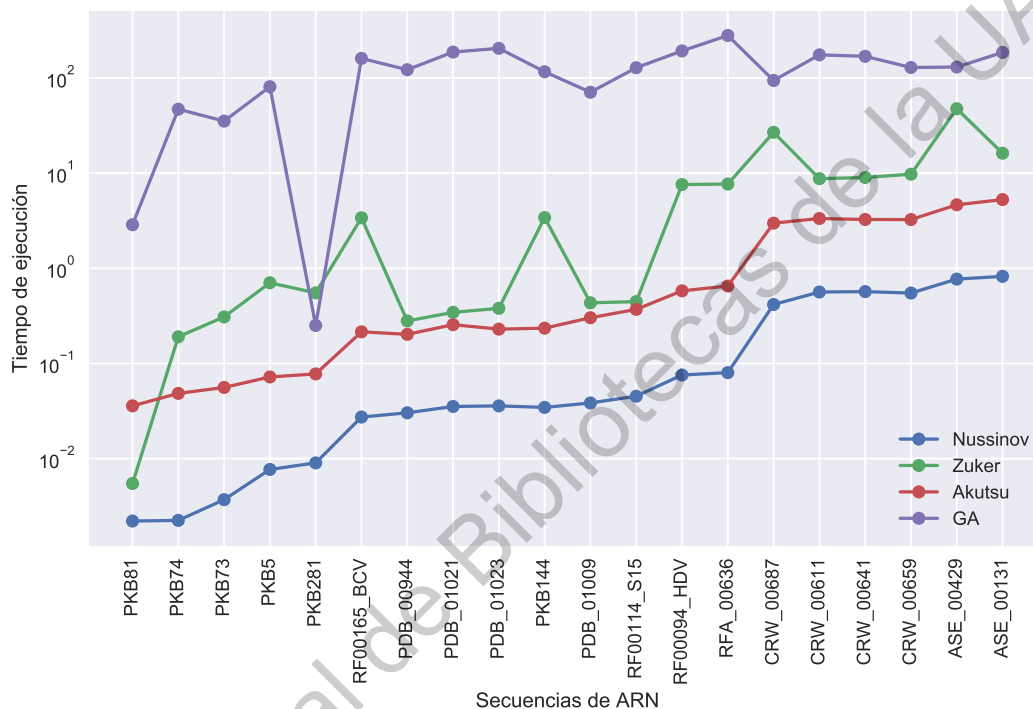


Figura 4.10: Coeficiente de Correlación de Matthews de los algoritmos para instancias con pseudonudos.



Por su parte, en la Figura 4.11, se muestran los tiempos de ejecución de cada algoritmo. En la gráfica se observa un comportamiento muy similar a la gráfica de tiempos para instancias sin pseudonudos. De igual manera, el Algoritmo Genético es el que toma mayor tiempo realizar la predicción de la estructura, seguido por Zuker, después Akutsu y finalmente Nussinov.



**Figura 4.11:** Tiempo de ejecución de los algoritmos para instancias con pseudonudos. Esta gráfica de tiempos se encuentra en escala logarítmica.

En el Apéndice A, se muestran las configuraciones utilizadas por el Algoritmo Genético.

Los algoritmos Nussinov, Zuker, y el Algoritmo Genético se implementaron construyendo un sistema en el lenguaje de programación *Python* versión 3.7.3, para el algoritmo de Akutsu se realizó la implementación en el lenguaje de programación Java 1.8.0 y se integro al prototipo mediante llamadas a procesos externos. Las pruebas se realizaron en una computadora portátil Dell G3 con procesador Intel(R) Core(TM) i5-8300H CPU@ 2.30GHz, RAM 16GB.

# Prototipo

*Si he visto más lejos es porque estoy sentado sobre los hombros de gigantes.*

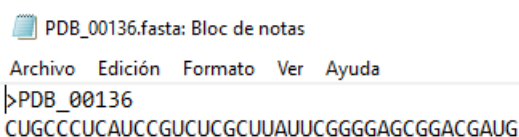
—ISAAC NEWTON (1643 - 1727)

Con base en los algoritmos implementados se creó un prototipo de sistema para la predicción de estructuras secundarias de ARN. Este prototipo implementa los algoritmos para la predicción de estructuras secundarias libres de pseudonudos: Nussinov y Zuker. Y para la predicción con pseudonudos mediante los algoritmos de Akutsu y nuestra propuesta de Algoritmo Genético. El sistema es una aplicación de escritorio escrita en el lenguaje de programación Python 3.7.3. El sistema, cuenta con dos opciones: la predicción de la estructura secundaria para una sola secuencia de ARN; y una prueba de los algoritmos para múltiples secuencias (*benchmark*). Es importante señalar que por el momento la interfaz de usuario está en idioma inglés, pensando en una futura difusión del prototipo.

Para la visualización de las estructuras secundarias, en el prototipo se implementa la herramienta Forna (Subsección 1.4.4) mediante la integración de código HTML y JavaScript.

## 5.1. Predicción de una sola secuencia de ARN

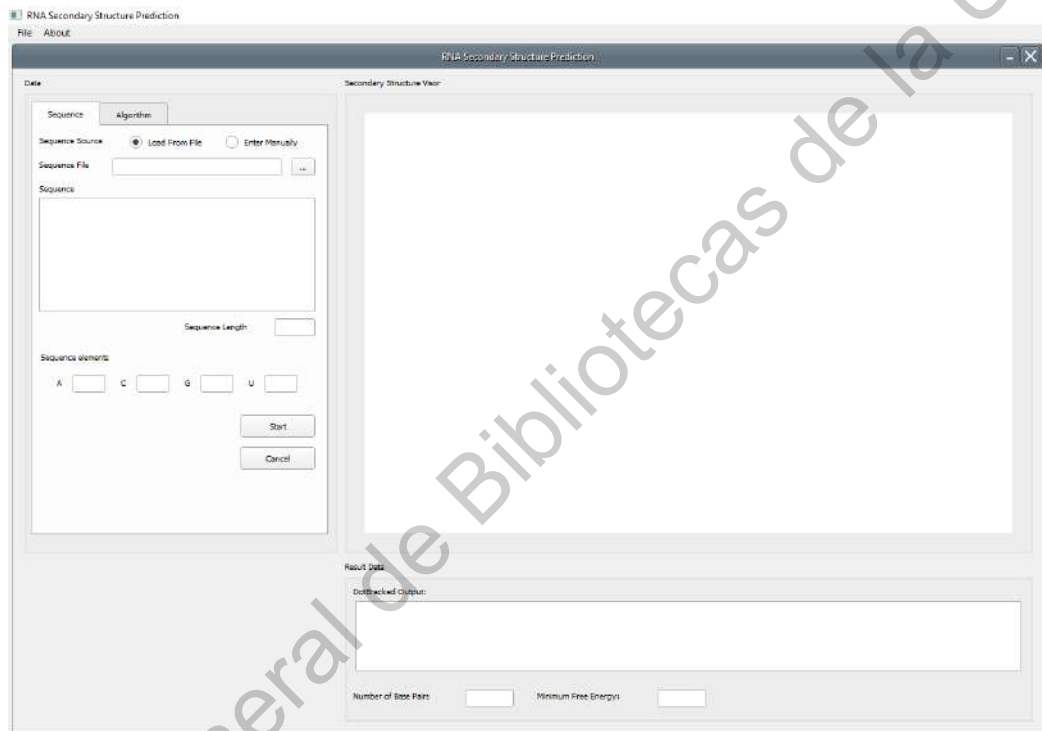
En esta opción de la aplicación se realiza la predicción de la estructura secundaria para una sola secuencia, leída de un archivo en formato *FASTA* o ingresada manualmente por el usuario. FASTA es un formato de texto para representar secuencias de nucleótidos, un archivo en formato FASTA comienza con una descripción de una sola línea, seguida de líneas de la secuencia de nucleótidos. La descripción se especifica por el símbolo mayor que (>) en la primera columna (Figura 5.1).



```
PDB_00136.fasta: Bloc de notas
Archivo Edición Formato Ver Ayuda
>PDB_00136
CUGCCCUCAUCCGUCUCGCUUAUUCGGGAGCGGACGAUG
```

**Figura 5.1:** Ejemplo de archivo en formato FASTA.

En la Figura 5.2, se muestra la pantalla para la predicción de una sola secuencia de ARN. En la pestaña *Algorithm*, podemos elegir el algoritmo con el cual queremos realizar la predicción de la estructura, esta opción se filtra por criterio de optimización: Minimización de la Energía Libre (opción *MFE*) o Maximización de Pares Base (opción *MBP*) (ver Figura 5.3). Por otro lado, en la pestaña *Sequence* se muestra la información de la secuencia ingresada, así como la opción para ejecutar el proceso de predicción.



**Figura 5.2:** Pantalla para la predicción de una sola secuencia.



**Figura 5.3:** Selección del algoritmo a ejecutar.

En esta opción, se obtiene como resultado la visualización de la estructura secundaria predicha por el algoritmo seleccionado, la estructura secundaria en formato Dot-Bracket, el número de pares base en la estructura obtenida y su correspondiente energía libre. Esta opción está pensada para los casos en los que no se tiene una estructura secundaria de referencia, por ello, no se muestran métricas de precisión.

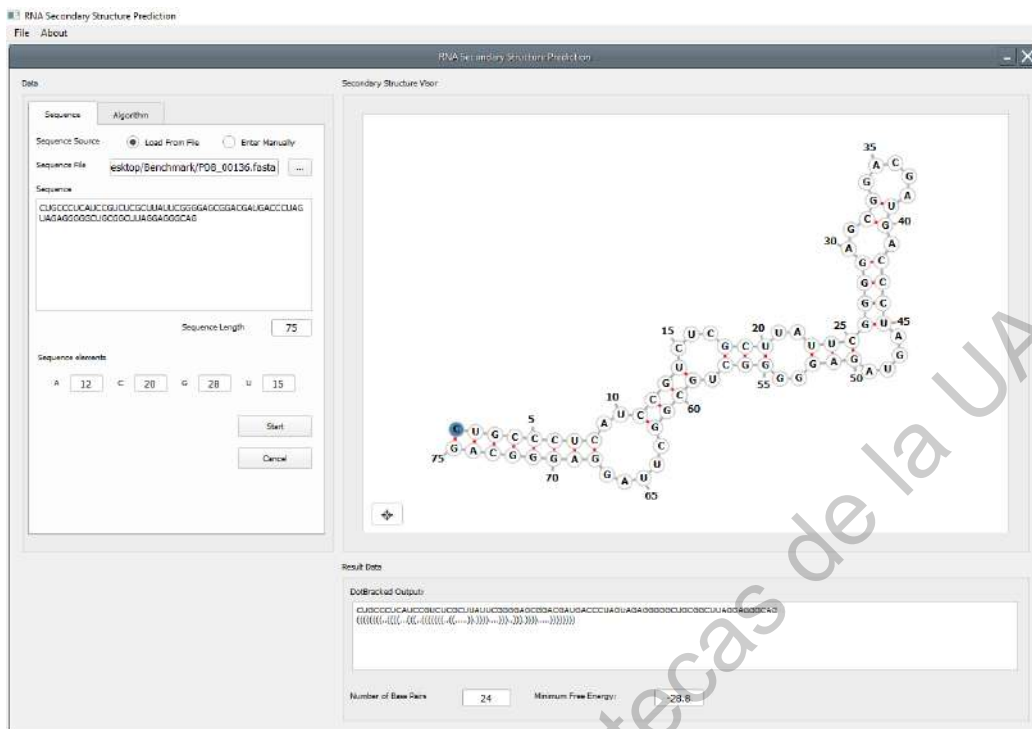


Figura 5.4: Pantalla de resultado de la predicción de la estructura secundaria.

## 5.2. Benchmark de los algoritmos

En esta opción de la aplicación, se realiza la ejecución de los cuatro algoritmos para cada una de las instancias de prueba que se hayan localizado dentro de la carpeta seleccionada por el usuario. Los archivos dentro de la carpeta, deben cumplir dos criterios para ser considerados por la aplicación. El primero es que deben tener la extensión *\*.bp* y el segundo, deben ser archivos de texto plano y solo contener tres líneas; la prima con el nombre de la secuencia, la segunda línea corresponde con la secuencia de ARN y la tercera con la estructura secundaria de referencia en formato Dot-Bracket. En la Figura 5.5, se muestra el ejemplo de un archivo válida para el prototipo.

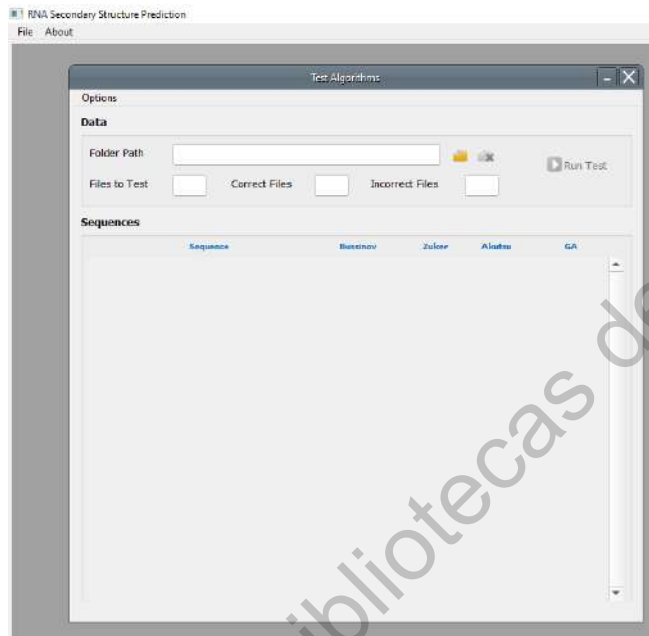
```

PDB_00988.dp: Bloc de notas
Archivo Edición Formato Ver Ayuda
#PDB_00988.dp
GCUGAAGUGCACACGGCGUGAAGUGCACACGGC
(((.(.[[[[[[.)))))(((.[.]]]]])..)))

```

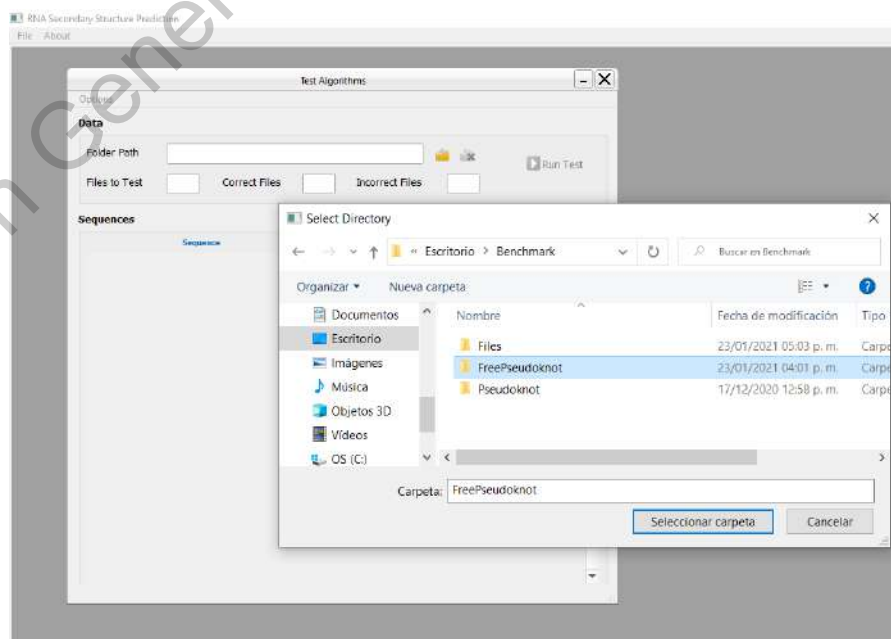
Figura 5.5: Ejemplo de archivo válido para la aplicación.

En la Figura 5.6, se muestra la pantalla principal de la opción de benchmark. Para seleccionar la carpeta que contenga las instancias de prueba, se debe hacer *click* sobre el botón *Open Folder* (botón iluminado en la figura).



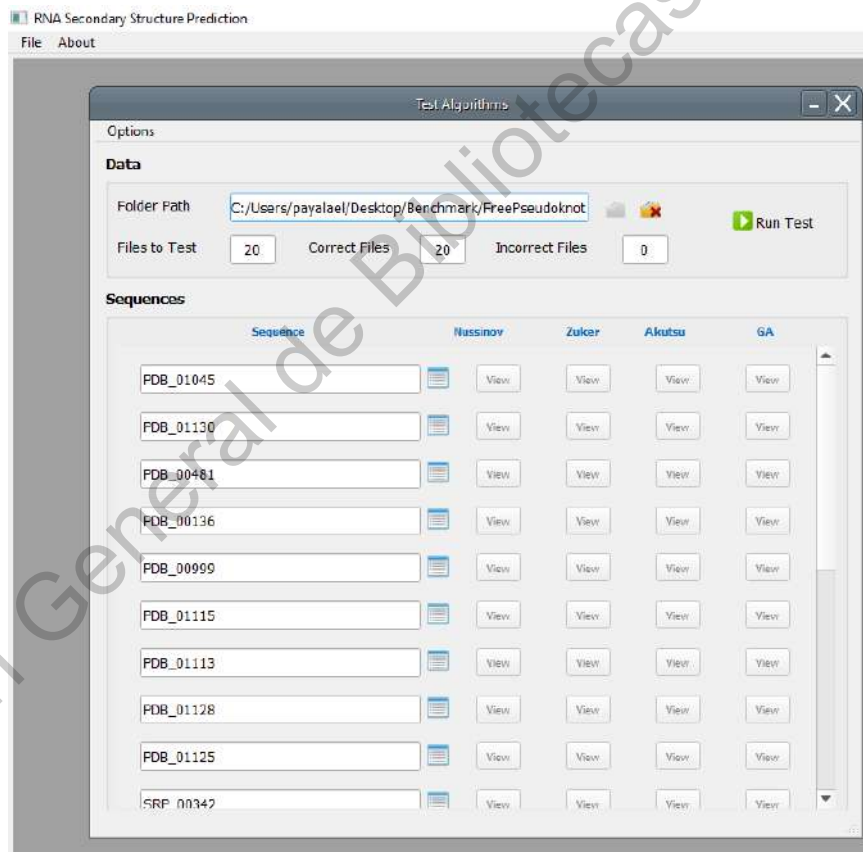
**Figura 5.6:** Pantalla de inicio de la opción benchmark.

Una vez que se ha dado *click* sobre el botón, aparece un cuadro de dialogo en el cual el usuario deberá seleccionar la carpeta deseada (Figura 5.7).



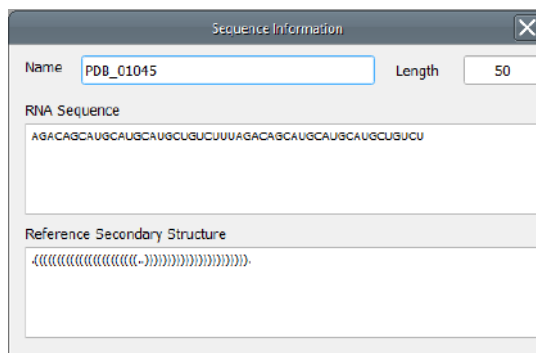
**Figura 5.7:** Cuadro de dialogo para la selección de carpeta.

Seleccionado el directorio deseado, se mostrara el detalle de los archivos encontrados en la ruta seleccionada: total de archivos, archivos correctos y archivos incorrectos (archivos que no estan en el formato correcto, o que el contenido no es una secuencia de ARN válida). Así, como el listado de las secuencias encontradas en los archivos (sección *Sequences*). Adicionalmente, se habilitarán los botones: *Remove Folder* y *Run Test* (ver Figura 5.8). El botón *Remove Folder*, remueve la selección de la carpeta anteriormente elegida, es decir, limpia la aplicación para que se seleccione una nueva ruta. Por otro lado, el botón *Run Test* inicia con la ejecución de los algoritmos. Una característica de la ejecución, es que se realiza en un segundo plano lo posibilita que se pueda interactuar con la aplicación mientras la ejecución esta en curso.



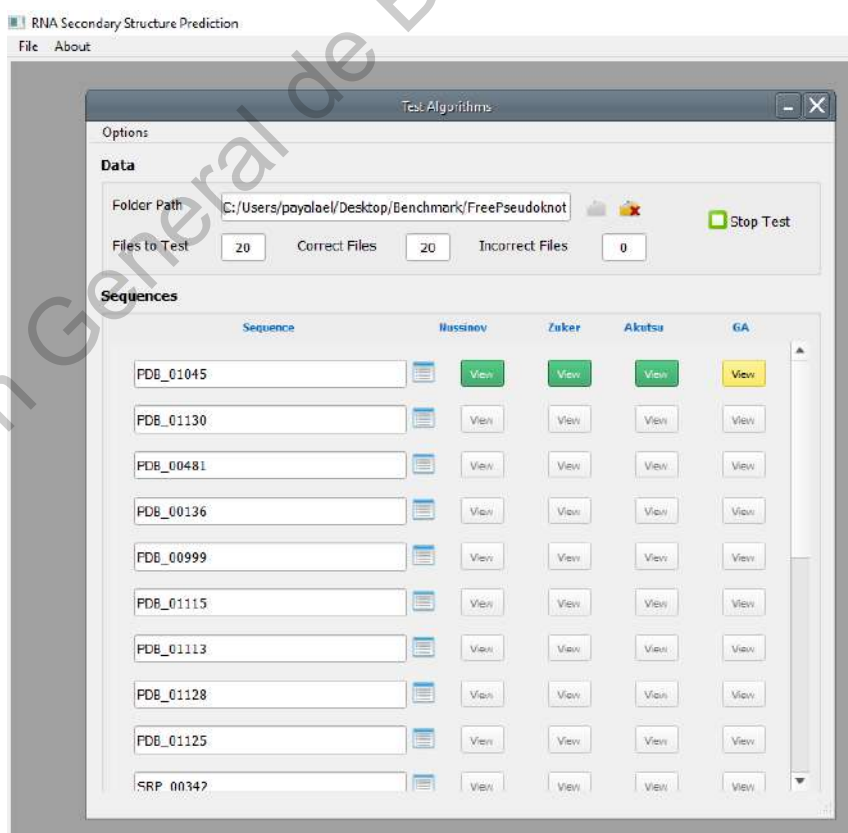
**Figura 5.8:** Pantalla de benchmark con el listado de las instancias a probar.

Es posible consultar el detalle de cada una de las instancias de prueba haciendo *click* en el botón *Sequence Information* junto al nombre de la secuencia (Figura 5.9), esta opción muestra el nombre de la secuencia, longitud, secuencia de ARN y las estructura de referencia.



**Figura 5.9:** Ventana de información de la secuencia.

Al iniciar la ejecución de los algoritmos mediante *Run Test*, el botón cambia de nombre por *Stop Test* con el cual podemos detener la ejecución. Por otro lado, observamos que para cada una de las instancias se realiza la ejecución de los cuatro algoritmos; en el detalle de las secuencias, observamos como se habilitan los botones *View*. Cuando este cambia a color Verde significa que sea terminado la ejecución del algoritmo correspondiente, mientras este en color amarillo, el algoritmo correspondiente se esta ejecutando. Este proceso, se muestra en la Figura 5.10.



**Figura 5.10:** Prototipo: algoritmos en ejecución.





F y Coeficiente de Correlación de Matthews (Figura 5.13. Las gráficas mostradas en este reporte son similares a las mostradas en la Sección 4.2) y el reporte de los tiempos de ejecución (pestaña *Execution Time*) por algoritmo para cada una de las instancias del conjunto de prueba (Figura 5.14).

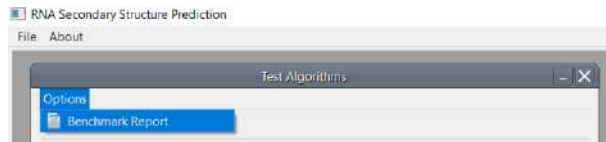


Figura 5.12: Opción *Benchmark Report*.

Benchmark								
Instance	Algorithm	TP	FN	FP	S	PPV	F-measure	MCC
PDB_01045	Nussinov	21	2	2	0.913	0.913	0.913	0.9114
	Zuker	22	1	0	0.9505	1.0	0.9778	0.9776
	Akutsu	22	1	1	0.9505	0.9505	0.9505	0.9507
	GA	22	1	1	0.9505	0.9505	0.9505	0.9507
PDB_01130	Nussinov	0	20	24	0.0	0.0	0.0	-0.0126
	Zuker	0	20	23	0.0	0.0	0.0	-0.0123
	Akutsu	16	4	10	0.0	0.6154	0.6957	0.6978
	GA	20	0	0	1.0	1.0	1.0	1.0
PDB_00481	Nussinov	0	26	24	0.0	0.0	0.0	-0.0134
	Zuker	16	10	0	0.6154	1.0	0.7619	0.7824
	Akutsu	0	26	22	0.0	0.0	0.0	-0.0128
	GA	0	26	15	0.0	0.0	0.0	-0.0106
PDB_00130	Nussinov	18	8	10	0.6923	0.6420	0.6657	0.6634
	Zuker	19	7	5	0.7308	0.7917	0.76	0.7581
	Akutsu	10	7	9	0.7308	0.6786	0.7037	0.7009
	GA	22	4	1	0.8462	0.9505	0.898	0.8906
PDB_00990	Nussinov	0	18	25	0.0	0.0	0.0	-0.0059
	Zuker	6	12	10	0.3333	0.375	0.3529	0.349
	Akutsu	10	8	17	0.5556	0.3704	0.4445	0.4487
	GA	10	8	9	0.5556	0.5263	0.5406	0.5372
	Nussinov	9	17	18	0.3487	0.3333	0.3306	0.3331

Figura 5.13: Pantalla de reporte de las métricas de precisión. Para mostrar las gráficas de las métricas basta con dar click sobre el nombre de la métrica.

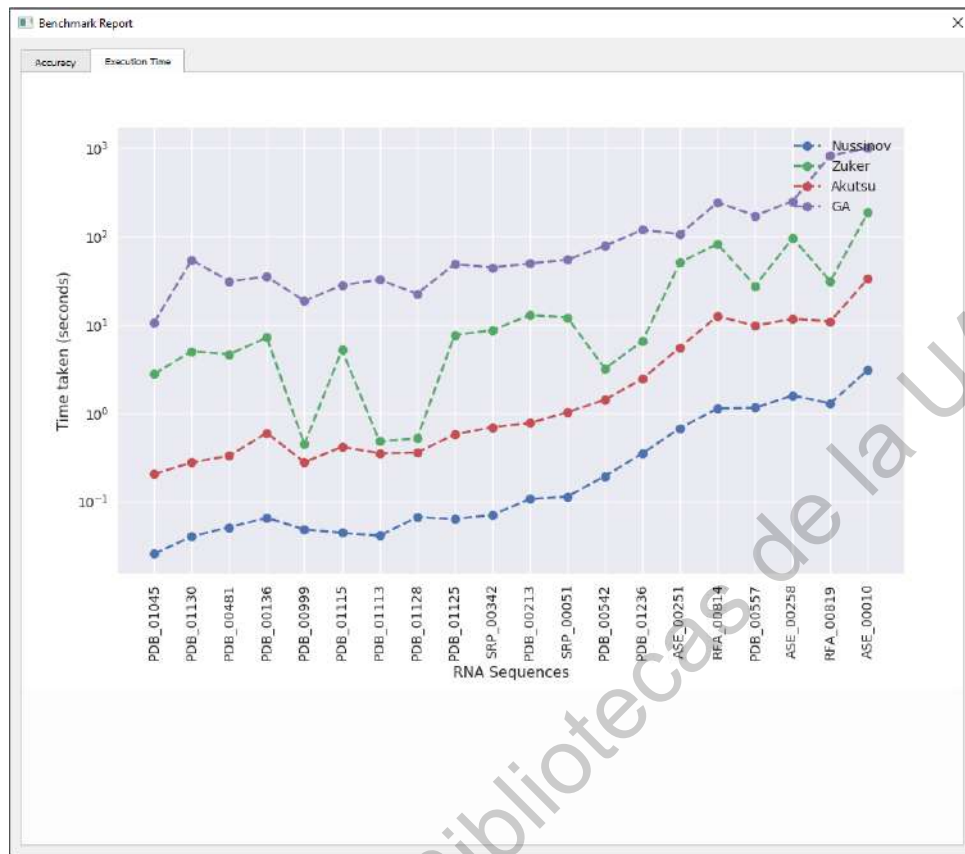


Figura 5.14: Pantalla de reporte con los tiempos de ejecución de los algoritmos.

## Conclusiones

*La ciencia es lo que entendemos lo suficientemente bien como para explicarle a una computadora; el arte es todo lo demás.*

—DONALD E. KNUTH (1938)

En este trabajo se implementaron y evaluaron los principales algoritmos basados en programación dinámica y un algoritmo genético para la predicción de estructuras secundarias. Con los valores obtenidos a través de las métricas planteadas, observamos que el algoritmo genético propuesto y los algoritmos de Akutsu y Zuker que hacen uso del modelo termodinámico para la minimización de energía libre NNDB, tienen un porcentaje de precisión mayor que el algoritmo Nussinov, el cual utiliza un enfoque de maximización de pares base. También se observa que existe una variabilidad en la precisión y sensibilidad con respecto a las diferentes instancias de los conjuntos de pruebas. Es por ello, importante la ejecución de una serie de algoritmos para que a partir de los diferentes resultados, ofrecer una mejor propuesta.

En el mismo sentido de la precisión de la estructura predicha con respecto a la estructura de referencia, el algoritmo genético es el que tiene un mejor desempeño. Sin embargo, en cuanto al tiempo de computo requerido, con el análisis experimental realizado observamos que este algoritmo demanda mayor procesamiento que los otros algoritmos. La causa principal de este incremento está dada por el uso del algoritmo de McCaskill en la etapa de creación de la población inicial, así como en la etapa de creación de los individuos ya que para cada secuencia se válida la posible formación de pseudonudos.

De acuerdo a los resultados obtenidos por el algoritmo genético, se mostró que el método para obtener las hélices mediante la probabilidad que tienen los pares base de aparecer en la estructura secundaria en la mayoría de casos es una buena técnica. Sin embargo, puede haber instancias en las que dada su naturaleza la estructura secundaria se conforme por hélices con baja probabilidad, lo que ocasionaría que el resultado obtenido por el algoritmo no sea favorable.

Como trabajo futuro se plantea el estudio y mejora del prototipo para incluir un análisis estructural de las estructuras secundarias obtenidas, así como la comparación entre estructuras secundarias dadas. Ya que en el presente trabajo nos enfocamos, solo en los algoritmos para la predicción de estructuras secundarias a partir de una secuencia de ARN dada, con ello cumpliendo los objetivos establecidos al inicio de la tesis.

Otro trabajo a futuro, es el estudio e implementación de técnicas de paralelización de algoritmos, con lo cual se ayudaría a mejorar los tiempos de ejecución de los algoritmos. Además, de la inclusión de técnicas de optimización del código Python para el procesamiento en GPU mediante PyCUDA.

Finalmente con respecto al prototipo para la predicción de estructuras secundarias, el código fuente estará disponible en el repositorio de Github (<https://github.com/payalae16/RNASecStrucPredict>), el cual podrá ser utilizado y modificado libremente.

# Referencias Bibliográficas

- Aigner, K., Dreßen, F., y Steger, G. (2012). Methods for predicting rna secondary structure. En L. Neocles y W. Eric (Eds.), *Rna 3d structure analysis and prediction* (p. 19-41). Springer Verlag.
- Akutsu, T. (2000, 08). Dynamic programming algorithms for rna secondary structure prediction with pseudoknots\* 1,\* 2. *Discrete Applied Mathematics*, 104, 45-62.
- Andronescu, M. (2003). Algorithms for predicting the secondary structure of pairs and combinatorial sets of nucleic acid strands.. Descargado de <https://open.library.ubc.ca/collections/ubctheses/831/items/1.0051269>
- Andronescu, M., Bereg, V., Hoos, H., y Condon, A. (2008, 09). Rna strand: the rna secondary structure and statistical analysis database. *BMC bioinformatics*, 9, 340.
- Bastolla, U., Porto, M., Roman, H., y Vendruscolo, M. (2007). *Structural approaches to sequence evolution*.
- Böckenhauer, H.-J., y Bongartz, D. (2007). *Algorithmic aspects of bioinformatics*. Berlin: Springer Verlag.
- Churkin, A., Weinbrand, L., y Barash, D. (2015). Free energy minimization to predict rna secondary structures and computational rna design. En E. Picardi (Ed.), *Rna bioinformatics* (pp. 3–16). New York, NY: Springer New York.
- Du, K.-L., y Swamy, M. (2016). *Search and optimization by metaheuristics : Techniques and algorithms inspired by nature*. Switzerland: Springer International Publishing Switzerland.
- Durbin, R., Eddy, S. R., Krogh, A., y Mitchison, G. J. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. United Kingdom.
- El Fatmi, A., Bekri, M. A., y Benhlima, S. (2019). Rnaknot: A new algorithm for rna secondary structure prediction based on genetic algorithm and grasp method. *Journal of Bioinformatics and Computational Biology*, 17(05), 1950031.
- Fatmi, A. E., Bekri, M. A., y Benhlima, S. (2018, April). Rna secondary structure

- prediction based on genetic algorithm and comparative approach. En *2018 4th international conference on optimization and applications (icoa)* (p. 1-7).
- Fatmi, A. E., Chentoufi, A., Bekri, M. A., Benhlime, S., y Sabbane, M. (2017, April). A heuristic algorithm for rna secondary structure based on genetic algorithm. En *2017 intelligent systems and computer vision (iscv)* (p. 1-7).
- Hamada, M., Kiryu, H., Sato, K., Mituyama, T., y Asai, K. (2008, 12). Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics*, *25*(4), 465-473.
- Huson, D. (2006, December). *Lecture notes Algorithms in Bioinformatics I*. Center for Bioinformatics Tübingen, Universität Tübingen.
- Kerpedjiev, P., Hammer, S., y Hofacker, I. L. (2015, 06). Forna (force-directed RNA): Simple and effective online RNA secondary structure diagrams. *Bioinformatics*, *31*(20), 3377-3379.
- Kravchenko, A. (2009). Predicting rna secondary structures including pseudoknots..
- Liu, Z., Zhu, D., Cui, W., y Liu, N. (2013). An approximation scheme for rna folding structure prediction including pseudoknots. En *2013 ninth international conference on computational intelligence and security* (p. 6-10).
- Lorenz, R., Bernhart, S., Höner zu Siederdisen, C., Ta, H., Flamm, C., Stadler, P., y Hofacker, I. (2011, 11). Viennarna package 2.0. *Algorithms for molecular biology : AMB*, *6*, 26.
- Lorenz, R., Wolfinger, M. T., Tanzer, A., y Hofacker, I. L. (2016). Predicting rna secondary structures from sequence and probing data. *Elsevier: Advances in RNA Structure Determination*, *103*, 86-98.
- McCaskill, J. S. (1990). The equilibrium partition function and base pair binding probabilities for rna secondary structure. *Biopolymers*, *29*(6-7), 1105-1119.
- Nussinov, R., y Jacobson, A. B. (1980). Fast algorithm for predicting the secondary structure of single-stranded rna. *Proceedings of the National Academy of Sciences of the United States of America*, *77* 11, 6309-13.
- Osman, M. N., Abdullah, R., y AbdulRashid, N. (2010, June). Rna secondary structure prediction using dynamic programming algorithm - a review and proposed work. En *2010 international symposium on information technology*

(Vol. 2, p. 551-556).

- Polanski, A., y Kimmel, M. (2007). *Bioinformatics*. Berlin: Springer Verlag.
- Punetha, A., Sarkar, P., Nimkar, S., Sharma, H., KNR, Y., y Nagaraj, S. (2018). Structural bioinformatics: Life through the 3d glasses. En A. Shanker (Ed.), *Bioinformatics: Sequences, structures, phylogeny* (p. 191-253). Springer Verlag.
- Reidys, C. (2011). *Combinatorial computational biology of rna: Pseudoknots and neutral networks*. Berlin: Springer Verlag.
- Rna bulges as architectural and recognition motifs. (2000). *Structure*, 8(3), R47 - R54.
- Shahidul Islam, M., y Rafiqul Islam, M. (2020). A hybrid framework based on genetic algorithm and simulated annealing for rna structure prediction with pseudoknots. *Journal of King Saud University - Computer and Information Sciences*.
- Sharma, D., Singh, S., y Chand, T. (2015). Rna pseudoknot: Topology and prediction. En *2015 international conference on computer and computational sciences (icccs)* (p. 244-248).
- Sivanandam, S., y Deepa, S. N. (2008). *Introduction to genetic algorithms*. New York: Springer-Verlag Berlin Heidelberg.
- Sperschneider, V. (2008). *Bioinformatics: Problem solving paradigms*. Berlin: Springer Verlag.
- Taufer, M., Licon, A., Araiza, R., Mireles, D., van Batenburg, F. H. D., Gulyaev, A. P., y Leung, M.-Y. (2009, January). Pseudobase++: an extension of pseudobase for easy searching, formatting and visualization of pseudoknots. *Nucleic acids research*, 37(Database issue), D127—35. Descargado de <https://europepmc.org/articles/PMC2686561> doi: 10.1093/nar/gkn806
- Tong, K., Cheung, K., Lee, K., y Leung, K. (2013). Gaknot: Rna secondary structures prediction with pseudoknots using genetic algorithm. En *2013 ieee symposium on computational intelligence in bioinformatics and computational biology (cibcb)* (p. 136-142).
- Turner, D. H., y Mathews, D. H. (2009). Nndb: the nearest neighbor parameter

- database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Research*, 38, D280-D282.
- Vialette, S. (s.f.). *Algorithmic Aspects of RNA Secondary Structures*. (URL: <http://igm.univ-mlv.fr/~vialette/teaching/2013-2014/MPRI/lecture.pdf>.)
- Westbrook, J., Feng, Z., Chen, L., Yang, H., y Berman, H. M. (2003, 01). The Protein Data Bank and structural genomics. *Nucleic Acids Research*, 31(1), 489-491.
- Wong, L. (2004). *The practical bioinformatician*. New Jersey: World Scientific.
- Zhao, C., y Sahni, S. (2016). Cache and energy efficient algorithms for nussinov rna folding. En *2016 ieee 6th international conference on computational advances in bio and medical sciences (iccabs)*.
- Zhao, C., y Sahni, S. (2020). Efficient computation of rna partition functions using mccaskill's algorithm. En *2020 15th conference on computer science and information systems (fedcsis)* (p. 449-452).
- Zhao, Y., Ni, Q., y Wang, Z. (2009, Oct). Computational features evaluation for rna secondary structure prediction. En *2009 2nd international conference on biomedical engineering and informatics* (p. 1-5).
- Zhao, Y., Wang, J., Zeng, C., y Xiao, Y. (2018, 06). Evaluation of rna secondary structure prediction for both base-pairing and topology. *Biophysics Reports*, 4.
- Zuker, M., Mathews, D. H., y Turner, D. H. (1999). Algorithms and thermodynamics for rna secondary structure prediction: A practical guide. En J. Barciszewski y B. F. C. Clark (Eds.), *Rna biochemistry and biotechnology* (pp. 11-43). Dordrecht: Springer Netherlands.
- Zuker, M., y Stiegler, P. (1981, 01). Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9(1), 133-148.



## A. Configuraciones del Algoritmo Genético

**Tabla 1:** Configuración del algoritmo genético para instancias sin pseudonodos.

Clave	$n_h$	$\alpha$	$l_c$	$\beta$	$n_e$	$n_c$	$\delta$
PDB_01045	1	1	1	0.4	0	0	200
PDB_01130	9	27	5	0.4	10	8	200
PDB_00481	6	18	4	0.4	7	5	200
PDB_00136	7	21	4	0.4	8	6	200
PDB_00999	6	18	4	0.4	7	5	200
PDB_01115	8	24	4	0.4	9	7	200
PDB_01113	9	27	5	0.4	10	8	200
PDB_01128	7	21	4	0.4	8	6	200
PDB_01125	12	36	5	0.4	14	11	200
SRP_00342	12	36	5	0.4	14	11	200
PDB_00213	12	36	5	0.4	14	11	200
SRP_00051	15	45	7	0.4	18	13	200
PDB_00542	17	51	8	0.4	20	15	200
PDB_01236	23	69	10	0.4	27	21	200
ASE_00251	17	51	8	0.4	20	15	200
RFA_00814	23	69	10	0.4	27	21	200
PDB_00557	22	66	10	0.4	26	20	200
ASE_00258	35	105	10	0.4	42	31	200
RFA_00819	44	132	10	0.4	52	40	200
ASE_00010	38	114	10	0.4	45	34	200

**Tabla 2:** Configuración del algoritmo genético para instancias con pseudonudos.

Clave	$n_h$	$\alpha$	$l_c$	$\beta$	$n_e$	$n_c$	$\delta$
PKB81	3	6	3	0.4	2	2	200
PKB74	7	21	4	0.4	8	6	200
PKB73	8	24	4	0.4	9	7	200
PKB5	5	15	3	0.4	6	4	200
PKB281	3	6	3	0.4	2	2	200
RF00165_BCV	33	99	10	0.4	39	30	200
PDB_00944	26	78	10	0.4	31	23	200
PDB_01021	39	117	10	0.4	46	35	200
PDB_01023	40	120	10	0.4	48	36	200
PKB144	20	60	10	0.4	24	18	200
PDB_01009	29	87	10	0.4	34	26	200
RF00114_S15	28	84	10	0.4	33	25	200
RF00094_HDV	38	114	10	0.4	45	34	200
RFA_00636	48	144	10	0.4	57	43	200
CRW_00687	17	51	8	0.4	20	15	200
CRW_00611	23	69	10	0.4	27	21	200
CRW_00641	23	69	10	0.4	27	21	200
CRW_00659	23	69	10	0.4	27	21	200
ASE_00429	21	63	10	0.4	25	19	200
ASE_00131	24	72	10	0.4	28	22	200

**B. Artículo XIV Coloquio de Posgrado**

Dirección General de Bibliotecas de la UAQ



UNIVERSIDAD  
AUTÓNOMA  
DE QUERÉTARO



FACULTAD  
DE INGENIERÍA



DIPFI  
POSGRADO  
INGENIERÍA



Se otorga la presente

# CONSTANCIA

a:

**PEDRO AYALA ELIZARRARAZ**

Por su participación en el Coloquio 14º del Posgrado de Ingeniería

**COMPARACIÓN DE MÉTODOS PARA LA PREDICCIÓN DE  
ESTRUCTURAS SECUNDARIAS DE ARN MINIMIZANDO  
LA ENERGÍA LIBRE**

Universidad Autónoma de Querétaro  
Facultad de Ingeniería C. U  
Noviembre 2020

**Dr. Manuel Toledano Ayala**  
DIRECTOR  
FACULTAD DE INGENIERÍA

# “Comparación de métodos para la predicción de estructuras secundarias de ARN minimizando la energía libre”

“Comparing methods for RNA secondary structure prediction minimizing the free energy”

Pedro, Ayala Elizarraraz<sup>1</sup>, Arturo, González Gutiérrez<sup>1,\*</sup>, Rolando Tenoch, Bárcenas Luna<sup>2</sup>

<sup>1</sup>Universidad Autónoma de Querétaro, Facultad de Ingeniería, Cerro de las Campanas S/N, Querétaro CP. 76010, México

<sup>2</sup>Universidad Autónoma de Querétaro, Facultad de Ciencias Naturales, Av. de las Ciencias S/N, Querétaro CP. 76230, México

\*Autor correspondiente. Correo electrónico: aglez@uaq.mx

## Resumen

El comportamiento de una molécula de ARN depende directamente de sus estructuras secundarias y terciarias. Sin embargo, se ha demostrado que la predicción de estructuras secundarias con pseudonodos arbitrarios es un problema NP-completo. En este artículo presentamos un estudio comparativo de tres algoritmos basados en la técnica de programación dinámica. Los algoritmos de Nussinov y Zuker predicen estructuras secundarias sin pseudonodos, mientras que el algoritmo de Akutsu lo hace con pseudonodos simples. También presentamos una metaheurística que utiliza un algoritmo genético para producir subestructuras cuasi-óptimas que a su vez permiten la predicción de estructuras secundarias con pseudonodos simples. Finalmente, presentamos un análisis experimental de los cuatro algoritmos utilizando instancias públicas de estructuras de ARN provistas por las bases de datos RNA STRAND y PseudoBase ++.

**Palabras Clave:** IAR, Estructura Secundaria de ARN, Mínima Energía Libre, Pseudonodo, Programación Dinámica, Algoritmo Genético.

## Abstract

The behaviour of an RNA molecule is directly linked to its secondary and tertiary structures. However, it has been proved that the secondary structure prediction problem with arbitrary pseudoknots is an NP-complete problem. In this paper, we present a comparative study of three algorithms based on the dynamic programming technique. The Nussinov and Zuker algorithms predict secondary structures without pseudoknots, whereas Akutsu algorithm does it with simple pseudoknots. We also present a metaheuristic that uses a genetic algorithm to produce quasi-optimal substructures which in turn allow the prediction of secondary structures with simple pseudoknots. Finally, we present an experimental analysis of the four algorithms using public instances of RNA structures provided by RNA STRAND and PseudoBase ++ databases.

**Keywords:** IAR, RNA Secondary Structure, Minimum Free Energy, Pseudoknot, Dynamic Programming, Genetic Algorithm.

## 1. Introducción

El ácido ribonucleico (ARN) es una molécula que consiste en una cadena formada por los nucleótidos: Adenina (A), Citosina (C), Guanina (G) y Uracilo (U). En comparación con el ADN, las moléculas de ARN son menos estables y muestran una mayor variabilidad en su estructura tridimensional [1]. El ARN adquiere una gran relevancia, ya que está involucrado en varios procesos biológicos, tales como la codificación y decodificación de información genética; regulación, detección y comunicación de respuestas a señales celulares; así como a la catalización de las reacciones biológicas [2].

Debido a que la funcionalidad de una molécula de ARN depende en gran medida de sus estructuras secundarias y terciarias, es importante su estudio y caracterización [3]. La caracterización y modelación de la estructura terciaria representa un mayor desafío científico y tecnológico que la estructura secundaria. Por ello, los esfuerzos se han enfocado en gran medida en la predicción de estructuras secundarias, ya que es experimentalmente accesible y contiene información para determinar la relación entre estructura y funcionalidad [4].

La estructura secundaria de una secuencia de ARN consiste en el plegamiento de la cadena de nucleótidos consigo misma mediante los enlaces de hidrógeno entre sus bases y en diferentes posiciones sobre la misma, en donde cada nucleótido sólo puede formar parte de un par base [2].

**Definición 1.1 (Secuencia de ARN)** Sea  $r = r_1 \dots r_n$  una cadena y  $\Sigma_{ARN} = \{A, C, G, U\}$  el conjunto que representa el alfabeto de los nucleótidos que conforman el ARN. Si  $r$  es una cadena conforme al alfabeto  $\Sigma_{ARN}$ , entonces,  $r$  es una secuencia de ARN.

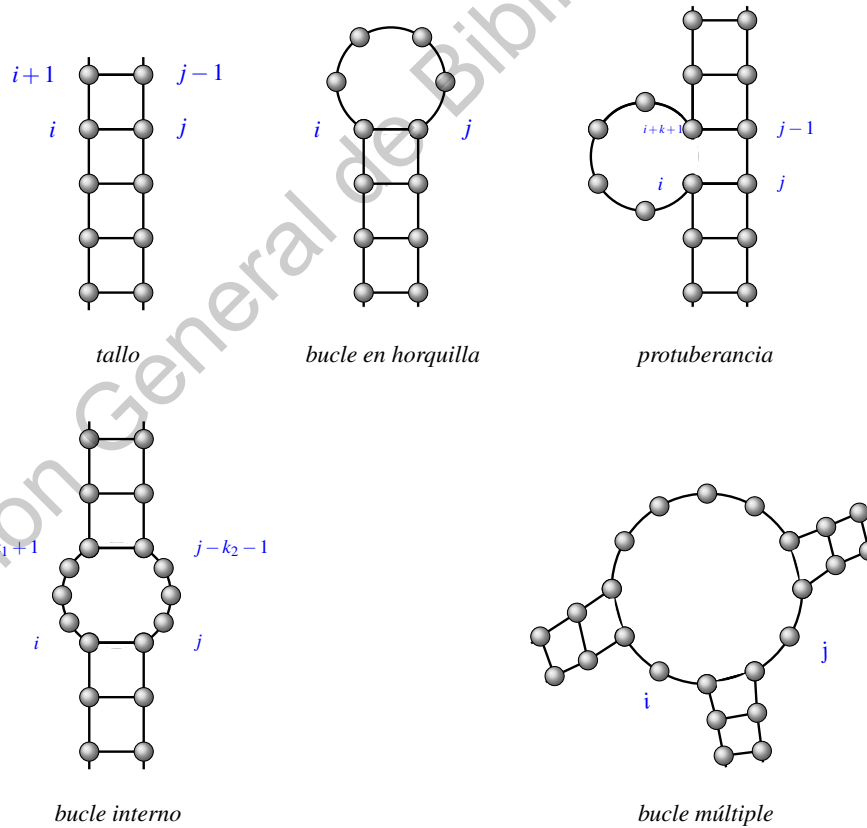
La estructura secundaria se puede representar por el conjunto  $S_r$  de pares de índices, que corresponden con las posiciones de los pares base sobre la secuencia  $r$ ,

$$S_r \subseteq \{(i, j) \mid 1 \leq i < j \leq n\}$$

donde  $n$  es la longitud de la secuencia,  $i$  y  $j$  son posiciones en la cadena que representan bases complementarias sujetas a las siguientes restricciones [5] [6]:

- I. Cada índice  $k \in \{1, \dots, n\}$  ocurre a lo más una vez en el emparejamiento de  $S_r$ .
- II. Para cada par  $(i, j)$  de  $S_r$ , el par de bases  $(i, j)$  es un par de tipo *Watson-Crick*:  $(r_i, r_j) \in \{(A, U), (U, A), (C, G), (G, C)\}$ , o es un par tipo *Wobble*:  $(r_i, r_j) \in \{(G, U), (U, G)\}$ .
- III. Para cada par  $(i, j)$  de  $S_r$ ,  $j - i \geq 4$  (donde 4 es el tamaño mínimo de una región de bases no complementarias llamada bucle en horquilla).

En la Figura 1, se muestran las subestructuras elementales que pueden contener la estructura secundaria mediante métodos de plegamiento por maximización de pares base o por minimización de energía libre entre moléculas.



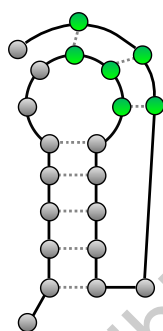
**Figura 1:** Representación de las posibles subestructuras que una cadena de ARN puede adoptar. Las bases se muestran como puntos y las líneas representan las conexiones entre las bases [5]. Los índices  $i$ ,  $j$ ,  $k$ ,  $k_1$  y  $k_2$  representan los pares base a partir de los cuales se forma la subestructura.

A continuación, se define el problema de la predicción de la estructura secundaria.

**Definición 1.2 (Predicción de la estructura secundaria)** Dada una secuencia  $r$  conforme al alfabeto  $\Sigma_{ARN}$ , realizar el plegamiento de la secuencia utilizando las restricciones anteriormente listadas, generando las subestructuras permitidas [5] [7] [8], para obtener la estructura secundaria  $S_r$  con la mínima energía libre o la estructura que maximice el número de pares base.

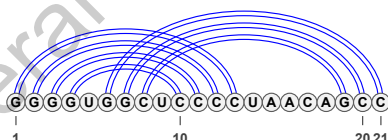
Una topología o tipo de subestructura más compleja es el *pseudonudo* (Figura 2), la cual se caracteriza por el emparejamiento de bases en un bucle con residuos no apareados de la subestructura. Enseguida se describe formalmente el pseudonudo.

**Definición 1.3 (Pseudonudo)** Para todos los pares base de una estructura secundaria, si existen dos pares base  $(i, j)$  y  $(k, l)$  que no son continuos ( $i < j < k < l$ ), ni anidados ( $i < k < l < j$ ), pero se intersectan ( $i < k < j < l$ ), entonces la estructura secundaria contiene un pseudonudo [9].



**Figura 2:** Representación esquemática de un pseudonudo. Las líneas punteadas representan las conexiones entre los pares base complementarios. Las conexiones entre pares base de los nodos resaltados conforman el pseudonudo.

La intersección de los pares base  $(i, j)$  y  $(k, l)$  que obedece a la restricción ( $i < k < j < l$ ), se muestra en la Figura 3. Si consideramos el par base  $(i, j)$  con las posiciones (1, 13) y al par base  $(k, l)$  con las posiciones (6, 21), claramente se observa esta intersección o superposición de pares base ( $1 < 6 < 13 < 21$ ).



**Figura 3:** Representación planar de un pseudonudo.

Con base en las definiciones anteriores, el problema de la predicción de estructuras secundarias con pseudonudos se define como sigue:

**Definición 1.4 (Predicción de la estructura secundaria con pseudonudos)** Dada una secuencia de ARN, la salida de la predicción con pseudonudos será la estructura secundaria que incluya pseudonudos que contienen la información de emparejamiento de bases.

## 2. Metodología y Materiales

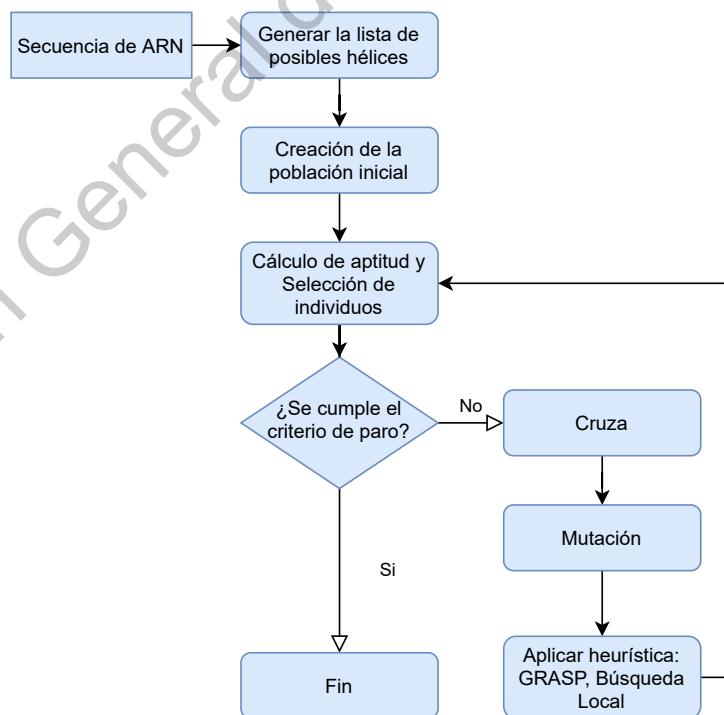
Las estructuras secundarias se determinan mediante técnicas de laboratorio. Las técnicas más utilizadas para determinar las estructuras de ARN son: los Rayos X, Cristalografía y la Resonancia Magnética Nuclear (NMR) ([3], [10]). En los últimos años ha surgido un mayor interés por la predicción de dichas estructuras mediante la modelación computacional. Con los métodos computacionales es posible reducir el tiempo y costo de la predicción de estructuras secundarias para una molécula [3]. Sin embargo, la modelación computacional no está exenta de limitantes. Un claro ejemplo de ello es la predicción de estructuras con pseudonudos, la cual incrementa la complejidad tanto en tiempo como en espacio. Se ha demostrado que la predicción de estructuras con pseudonudos utilizando el modelo termodinámico de minimización de la energía libre es un problema **NP-completo** ([11], [12]).

A lo largo de la historia se han implementado diferentes algoritmos utilizando técnicas como programación dinámica y algoritmos genéticos principalmente. Entre los algoritmos que excluyen la predicción de pseudonudos y que se basan en la técnica de programación dinámica se encuentran: **Nussinov** [13] que utiliza un enfoque de maximización de pares base; bajo la hipótesis de que entre más emparejamientos existan entre bases de la cadena de ARN, menor será la energía libre de la molécula. **Zuker** [14] que utiliza el concepto de minimización de energía libre mediante la asignación de energías termodinámicas a cada tipo de subestructura que conforman la estructura secundaria final; estas energías dependen del tamaño de las subestructuras, y la energía total de la secuencia se representa como la suma de todas las contribuciones individuales de estas subestructuras. Ambos algoritmos exhiben un orden de complejidad  $O(n^3)$ .

Por otro lado, respecto a los algoritmos que incluyen la predicción con pseudonudos destacan los propuestos por: **Rivas** [15] que propone un algoritmo con enfoque de minimización de la energía libre y con restricción en el tipo de pseudonudo con un orden de complejidad  $O(n^6)$ . **Akutsu** [11] que propone un algoritmo que maximiza el número de pares base e incluye la predicción de pseudonudos simples con un orden de complejidad  $O(n^4)$ . Del algoritmo de Akutsu en [16] se propone una modificación que incluye el modelo termodinámico de minimización de energía libre y reduce la complejidad del espacio de  $O(n^3)$  a  $O(n^2)$ , con un tiempo de ejecución de  $O(n^4)$ .

Con respecto a enfoques metaheurísticos para la predicción de estructuras secundarias, en los últimos años se han propuesto una variedad de algoritmos genéticos, como **GAknot** [9] el cual genera un conjunto de hélices mediante un enfoque similar al del algoritmo Nussinov. A partir de este conjunto de hélices se genera la población inicial, en donde cada individuo se crea a través de las combinaciones de las hélices previamente encontradas. Posteriormente, se aplican los operadores genéticos de selección, cruce y mutación para obtener la solución. El algoritmo propuesto en [17] (**GA-GRASP**) combina un algoritmo genético y la metaheurística GRASP (Greedy Randomized Adaptive Search Procedure) en combinación del modelo termodinámico de energía libre. El algoritmo **RNAknot** [18] es una mejora del algoritmo **GA-GRASP** que incluye la predicción de todos los tipos de subestructuras básicas (Figura 1) y dos tipos de pseudonudos: simple (*H-type*) y *Kissing-Hairpin*, con un orden de complejidad  $O(n^2) + O(k \times nbs)$ , donde  $k$  es el número de iteraciones y  $nbs$  es el número de individuos en la población.

En la Figura 4, se muestra el flujo básico de los algoritmos genéticos mencionados.



**Figura 4:** Diagrama de flujo del algoritmo genético.



En la Tabla 1, se presentan los enfoques de minimización de energía de cada modelo y su correspondiente algoritmo, así como sus restricciones en cuanto a si su predicción incluye o no los pseudonodos, indicando el tipo de pseudonodo en su caso.

**Tabla 1:** Caracterización de los algoritmos para modelación de estructuras secundarias.

Algoritmo	Modelo de minimización de energía	Predicción con pseudonodos	Tipo
Nussinov	Maximización de pases base	N	-
Zuker	Modelo Termodinámico	N	-
Akutsu	Modelo Termodinámico	S	Simple
RNAknot	Modelo Termodinámico	S	Simple, Kissing Hairpin

### 2.1. Parámetros de energía

La base de datos del vecino más cercano (Nearest Neighbor Database, NNDB) [19], es un modelo termodinámico que asume que la estabilidad de un par base específico depende directamente de las bases vecinas. Esta base de datos proporciona los parámetros de energía que permiten la predicción de una estructura secundaria estable.

La estabilidad de una estructura secundaria se mide por el cambio de energía libre  $\Delta G$ . Este cambio se aproxima como la suma de las contribuciones individuales de cada subestructura que conforma la estructura secundaria [20]. A continuación, se enlistan los cambios de energía que pueden ocurrir durante la construcción de la estructura secundaria:

- $\Delta G = 0$  indica equilibrio en la estructura.
- $\Delta G > 0$  indica desestabilización en la estructura.
- $\Delta G < 0$  indica que la estructura es estable.

Las reglas de cambio de energía se dividen por tipo de estructura: hélices, energías desestabilizadoras por tamaño de bucle (horquilla, protuberancia, interno), bucles internos generales (simétricos, asimétricos), extremos no-pareados y bucles múltiples. Para cubrir la contribución energética de cada tipo de subestructura, la base de datos NNDB cuenta con un total de 12883 parámetros de energía.

En la Tabla 2, se muestra la energía libre  $\Delta G$  entre cada par base que conforma una hélice dentro de la estructura secundaria. Estos parámetros de energía son estándar y se aplican en la construcción de cualquier hélice.

**Tabla 2:** Energía (*kcal/mol*) de las hélices entre cada par base.

	AU	CG	GC	UA	GU	UG
AU	-0.9	-2.2	-2.1	-1.1	-0.6	-1.4
CG	-2.1	-3.3	-2.4	-2.1	-1.4	-2.1
GC	-2.4	-3.4	-3.3	-2.2	-1.5	-2.5
UA	-1.3	-2.4	-2.1	-0.9	-1.0	-1.3
GU	-1.3	-2.5	-2.1	-1.4	-0.5	+1.3
UG	-1.0	-1.5	-1.4	-0.6	+0.3	-0.5

### 2.2. Secuencias de ARN

En la Tabla 3 y Tabla 4, se muestran las secuencias de ARN utilizadas en el desarrollo del proyecto. De cada secuencia se cuenta con la estructura secundaria obtenida mediante experimentación (*Rayos X* y/o *Resonancia Magnética Nuclear*), la cual es utilizada como la estructura de referencia para la validación de los resultados.

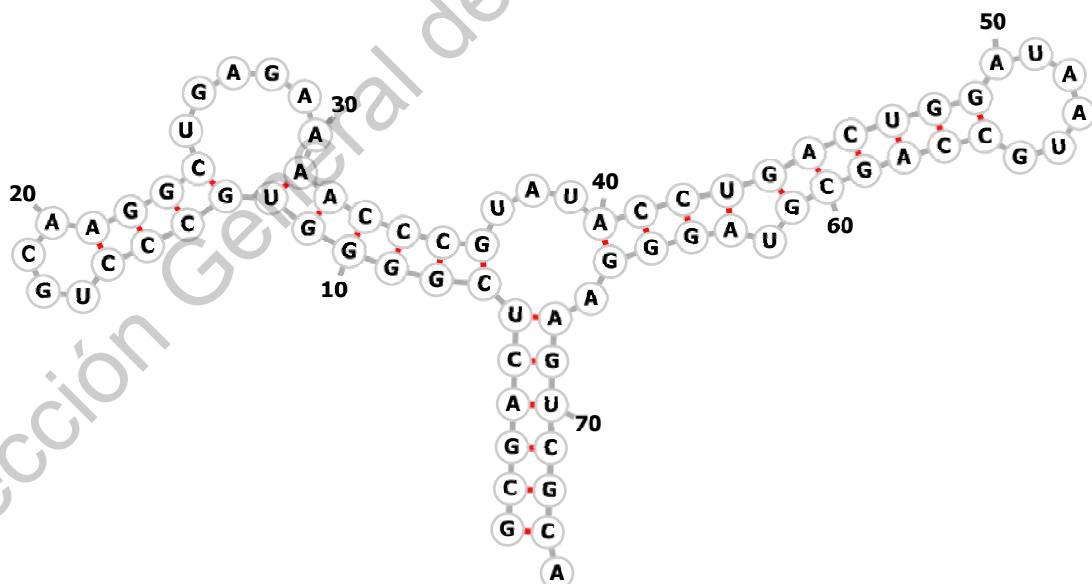
**Tabla 3:** Secuencias de ARN sin pseudonudos en la estructura de referencia. Las secuencias fueron obtenidas de la base de datos *RNA STRAND* [21].

Clave	Tipo	Longitud nt	Método validación
PDB_01130	ARN sintético	60	Rayos X
PDB_00481	ARN sintético	62	Rayos X
PDB_00136	Intron del Grupo II	70	Rayos X
PDB_01115	ARN sintético	74	Rayos X
PDB_01113	ARN sintético	75	Rayos X
PDB_01128	ARN sintético	75	NMR
PDB_01125	ARN sintético	86	NMR
PDB_00213	ARN sintético	101	NMR
PDB_00542	ARN sintético	126	Rayos X
PDB_01236	ARN sintético	153	Rayos X

**Tabla 4:** Secuencias de ARN con pseudonudos en la estructura de referencia. Las secuencias fueron obtenidas de la base de datos *PseudoBase ++* [22].

Clave	Tipo	Longitud nt	Método validación
PKB5	ARNt viral	41	NMR
PKB73	ARNm	33	NMR
PKB74	ARNm	28	NMR
PKB80	Cambio de marco Viral	49	NMR
PKB81	ARN sintético	26	NMR
PKB107	Cambio de marco Viral	52	NMR
PKB243	-	121	NMR
PKB273	Viral	48	NMR
PKB280	Cambio de marco Viral	68	NMR
PKB281	Cambio de marco Viral	43	NMR

En la Figura 5, se muestran la estructura correspondiente a la secuencia PDB\_01115 en forma esquemática y notación Dot-Bracket.



```
GCGACUCGGGGUGCCUGCAAGGCUGAGAAAACCCGUAUACCUGACUGGAUAAUGCCAGCGUAGGGAAGUCGCA  

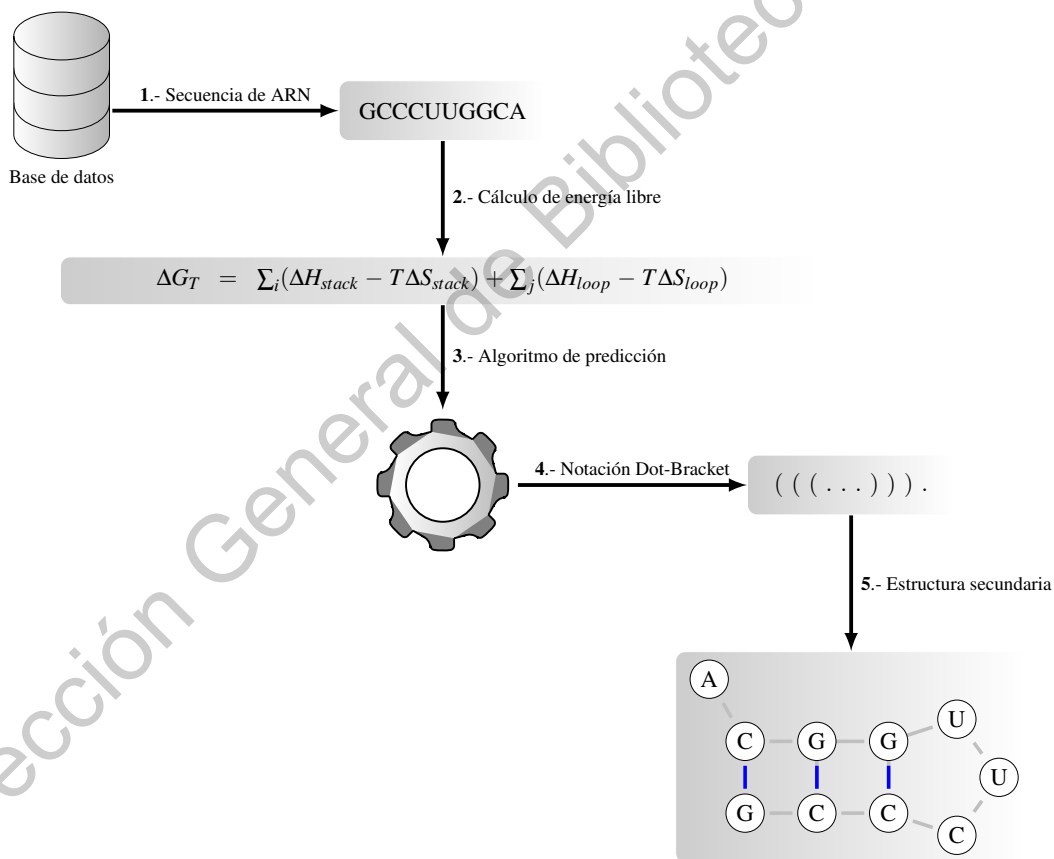
(((((((((((((((.....)))))))))))).....(((((((((((((((.....))))))))))))..)
```

**Figura 5:** Estructura secundaria correspondiente a la secuencia PDB\_01115.

### 2.3. Construcción del modelo

A continuación, se enlistan los pasos que conforman la metodología para la predicción de estructuras secundarias de ARN (Figura 6).

1. Secuencia de ARN: obtención y almacenamiento de las secuencias de ARN. Dichas secuencias son almacenadas localmente en archivos de texto plano en formato *FASTA*.
2. Cálculo de la energía libre: esta es una etapa en conjunto con el algoritmo de predicción; para ello se utiliza el modelo termodinámico del vecino más cercano (Sección 2.1), en el que se sumaría la contribución energética de cada subestructura que conforma la estructura secundaria final.
3. Algoritmo de predicción: se producen cuatro soluciones conforme a los algoritmos de Nussinov, Zuker, Akutsu y RNAknot.
4. Notación Dot-Bracket: estructura secundaria predicha en un formato específico para permitir su posterior visualización.
5. Estructura secundaria: es el resultado de la predicción (estructura secundaria), en el formato establecido y energía libre o número máximo de pares base, con la cual el prototipo desplegará el resultado.



**Figura 6:** Diagrama de flujo del proceso de la predicción de estructuras secundarias.

### 3. Resultados y discusión

#### 3.1. Métrica de precisión

Para evaluar la precisión de la predicción de los métodos, se utiliza la técnica descrita en [23], que utiliza los conceptos de *sensibilidad* y *valor predictivo positivo*.

La *sensibilidad* ( $S$ ), representa el porcentaje de pares base presentes en la estructura de referencia que ocurren en la estructura secundaria predicha, este valor es calculado con la siguiente ecuación:

$$S = \frac{VP}{VP + FN} \quad (1)$$

El *valor predictivo positivo* ( $VPP$ ), describe que porcentaje de pares base predichos ocurren en la estructura secundaria de referencia, se obtiene mediante la ecuación:

$$VPP = \frac{VP}{VP + FP} \quad (2)$$

Donde,  $VP$  es el número de pares base predichos correctamente,  $FN$  el número de pares base en la estructura de referencia que no están en la estructura predicha y  $FP$  es el número de pares base predichos que no están en la estructura de referencia.

Adicionalmente, se incluye la medida de precisión *Valor-F*, valor que varía entre 0 y 1, donde 0 indica que no hay pares base en común entre ambas estructuras y 1 indica una precisión perfecta. El cual se obtiene con la siguiente ecuación:

$$\text{Valor-F} = \frac{2 \times S \times VPP}{S + VPP} \quad (3)$$

#### 3.2. Análisis de Resultados

El conjunto de secuencias de prueba (Subsección 2.2), utilizado para evaluar el rendimiento, a través de los conceptos de sensibilidad (Ec. (1)) y de valor predictivo positivo (Ec. (2)), consta de 20 secuencias; de las cuales 10 son libres de pseudonudos y las otras 10 incluyen un pseudonudo de tipo simple en la estructura de referencia. Mediante la ejecución de cada algoritmo se obtuvo la eficiencia con respecto a las métricas definidas.

En la Tabla 5 se muestra el promedio del Valor-F (Ec. (3)) por algoritmo, agrupado por estructura con o sin pseudonudo en la estructura de referencia. La estructura obtenida de manera experimental se asume que tiene un Valor-F de 1.

**Tabla 5:** Valor-F promedio en estructuras sin y con pseudonudos.

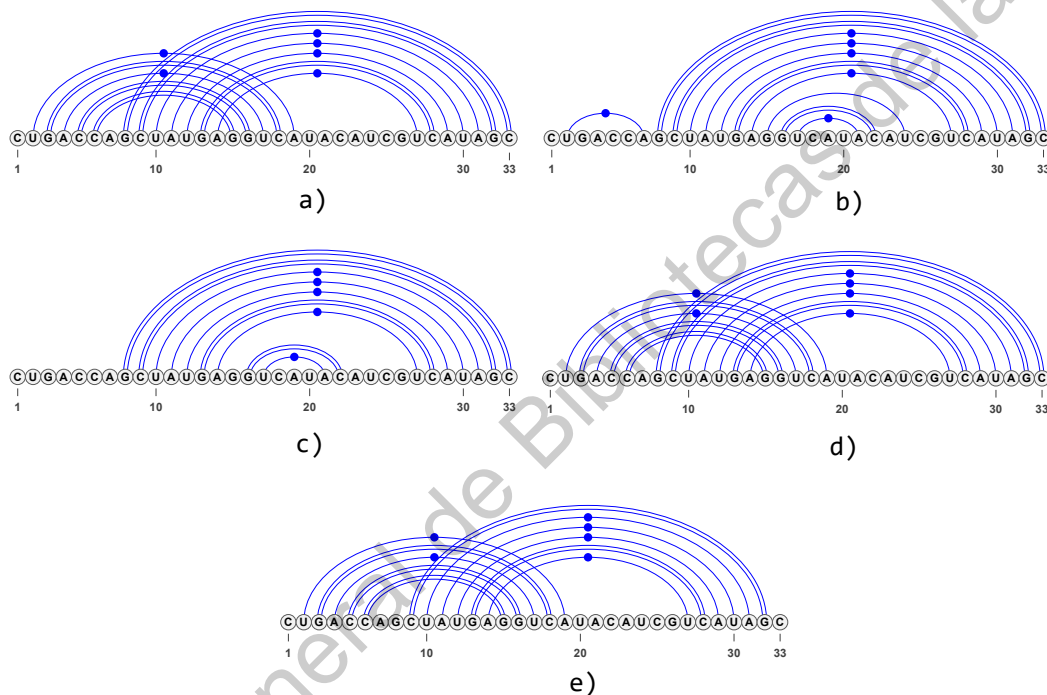
Algoritmo	Valor-F	
	Estructuras sin pseudonudos	Estructuras con pseudonudos
Nussinov	0.2655	0.2329
Zuker	0.7816	0.4817
Akutsu	0.4766	0.7349
RNAknot	0.4609	0.8039

Con respecto a las instancias sin pseudonudos, el algoritmo Zuker exhibe una mayor precisión con un 78.1 %, seguido por Akutsu con 47.6 %, luego RNAknot con 46 % y finalmente Nussinov con 26.5 %. Para Zuker es buen resultado ya que la mayoría de algoritmos de predicción logran un porcentaje de efectividad de entre 60 % y 80 % como se muestra en las comparaciones realizadas en [9] [23]. En tanto para Akutsu y RNAknot el porcentaje es superior a 45 %, lo cual es entendible debido a que la estructura de referencia



Respecto a las instancias con pseudonudos en la estructura de referencia, el algoritmo RNAknot es el que exhibe un mayor porcentaje de precisión con un 80.39%, seguido por Akutsu con 73.49%, después Zuker con 48.17% y finalmente Nussinov con un 23.29%. Como se mencionó anteriormente el porcentaje promedio de efectividad varía del 60% al 80%, por lo cual el resultado obtenido por los algoritmos RNAknot y Akutsu es favorable. Para los algoritmos Nussinov y Zuker no es sorpresa que el promedio de efectividad sea bajo, ya que estos no incluyen pseudonudos en su predicción, por lo cual en la mayoría de las instancias de prueba se encuentran pocos pares base en común entre la estructura de referencia y la estructura predicha.

Para la instancia PKB73, los cuatro algoritmos evaluados obtuvieron el Valor-F más alto. En la Figura 8, se muestra una representación planar de la estructura obtenida por cada uno de los algoritmos, con este tipo de gráfica es más simple la visualización del pseudonudo ya que se ven claramente las intersecciones entre las líneas que representan la conexión entre cada par base.



**Figura 8:** Representación planar de las estructuras secundarias correspondientes a la secuencia PKB73. La Figura a) corresponde a la estructura secundaria de referencia, b) a la predicción por Nussinov, c) predicción por Zuker, d) a la predicción obtenida por Akutsu y d) a la predicción obtenida por RNAknot. La visualización de las estructuras se realizó mediante la herramienta *Varna* [25].

En la Figura 9, se muestra la gráfica de los tiempos de ejecución de los cuatro algoritmos para cada una de las instancias de prueba. Para el caso de los algoritmos con predicción de pseudonudos Akutsu y RNAknot; se observa Akutsu tiene un mejor desempeño en tiempo de ejecución, sin embargo, como se mostró anteriormente el algoritmo RNAknot tiene un mejor desempeño en cuanto a precisión con respecto a la estructura de referencia. Lo mismo sucede para los algoritmos Nussinov y Zuker, mientras que el algoritmo Nussinov tiene un mejor desempeño en tiempo de ejecución, por mucho el algoritmo de Zuker tiene un mejor desempeño en cuanto a la precisión de la estructura secundaria predicha.

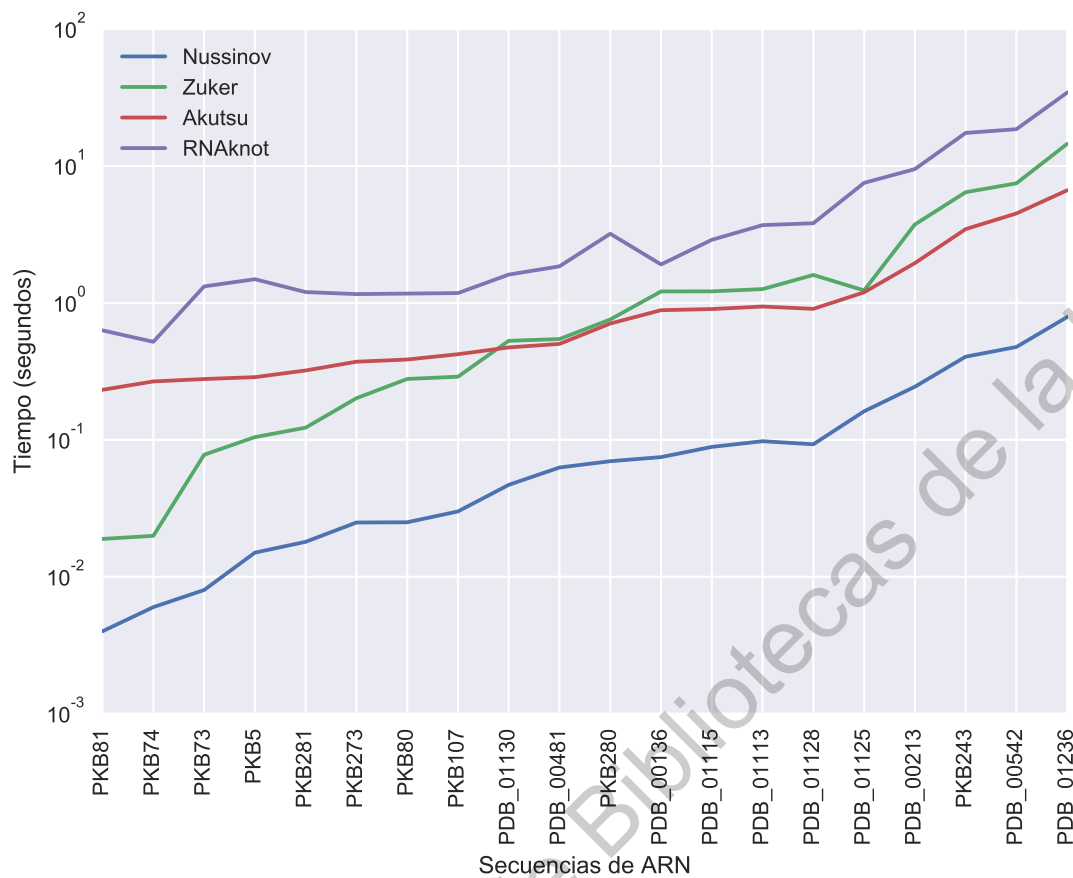


Figura 9: Tiempos de ejecución.

Los algoritmos Nussinov, Zuker y Akutsu se implementaron construyendo un sistema en el lenguaje de programación *Python* versión 3.7.3, mientras que para el algoritmo RNAknot se utilizó el software implementado en [18]. Las pruebas se realizaron en una computadora portátil Dell G3 con procesador Intel(R) Core(TM) i5-8300H CPU@ 2.30GHz, RAM 16GB.

#### 4. Conclusiones

En este artículo presentamos un análisis sobre los principales algoritmos basados en programación dinámica y algoritmos genéticos para la predicción de estructuras secundarias. Con los resultados obtenidos a través de las métricas planteadas observamos que los algoritmos RNAknot, Akutsu y Zuker que incluyen un modelo termodinámico para la minimización de energía tienen un porcentaje de precisión mayor que el algoritmo Nussinov que utiliza un enfoque de maximización de pares base. También se observa que existe una variabilidad en la precisión y sensibilidad con respecto a las diferentes instancias. Es por ello, importante en el desarrollo de un sistema robusto para la predicción de estructuras secundarias de ARN incluir varios algoritmos para que a partir de los diferentes resultados obtener una mejor propuesta.

Como trabajo futuro se planea el diseño e implementación de un algoritmo genético que incluya el modelo termodinámico de minimización de energía, el cual puede ser alimentado con poblaciones iniciales de estructuras secundarias subóptimas desde el punto de vista energético.

Con respecto a la validación de resultados, un trabajo a futuro interesante es incluir una validación desde la perspectiva topológica de la estructura que complemente los parámetros de sensibilidad y precisión.

## Referencias

- [1] A. Polanski and M. Kimmel, *Bioinformatics*. Berlin: Springer Verlag, 2007.
- [2] A. Punetha, P. Sarkar, S. Nimkar, H. Sharma, Y. KNR, and S. Nagaraj, "Structural bioinformatics: Life through the 3d glasses," in *Bioinformatics: Sequences, Structures, Phylogeny* (A. Shanker, ed.), pp. 191–253, Springer Verlag, 2018.
- [3] A. E. Fatmi, M. A. Bekri, and S. Benhlima, "Rna secondary structure prediction based on genetic algorithm and comparative approach," in *2018 4th International Conference on Optimization and Applications (ICOA)*, pp. 1–7, April 2018.
- [4] A. Churkin, L. Weinbrand, and D. Barash, *Free Energy Minimization to Predict RNA Secondary Structures and Computational RNA Design*, pp. 3–16. New York, NY: Springer New York, 2015.
- [5] H.-J. Böckenhauer and D. Bongartz, *Algorithmic Aspects of Bioinformatics*. Berlin: Springer Verlag, 2007.
- [6] Z. Liu, D. Zhu, W. Cui, and N. Liu, "An approximation scheme for rna folding structure prediction including pseudoknots," in *2013 Ninth International Conference on Computational Intelligence and Security*, pp. 6–10, Dec 2013.
- [7] C. Reidys, *Combinatorial Computational Biology of RNA: Pseudoknots and Neutral Networks*. Berlin: Springer Verlag, 2011.
- [8] R. Lorenz, M. T. Wolfinger, A. Tanzer, and I. L. Hofacker, "Predicting rna secondary structures from sequence and probing data," *Elsevier: Advances in RNA Structure Determination*, vol. 103, pp. 86–98, 2016.
- [9] K. Tong, K. Cheung, K. Lee, and K. Leung, "Gaknot: Rna secondary structures prediction with pseudoknots using genetic algorithm," in *2013 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pp. 136–142, 2013.
- [10] Y. Zhao, Q. Ni, and Z. Wang, "Computational features evaluation for rna secondary structure prediction," in *2009 2nd International Conference on Biomedical Engineering and Informatics*, pp. 1–5, Oct 2009.
- [11] T. Akutsu, "Dynamic programming algorithms for rna secondary structure prediction with pseudoknots\* 1,\* 2," *Discrete Applied Mathematics*, vol. 104, pp. 45–62, 08 2000.
- [12] S. Sheikh, R. Backofen, and Y. Ponty, "Impact of the energy model on the complexity of rna folding with pseudoknots," in *Combinatorial Pattern Matching*, pp. 321–333, Springer Berlin Heidelberg, 07 2012.
- [13] R. Nussinov and A. B. Jacobson, "Fast algorithm for predicting the secondary structure of single-stranded rna.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 77 11, pp. 6309–13, 1980.
- [14] M. Zuker and P. Stiegler, "Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information," *Nucleic Acids Research*, vol. 9, pp. 133–148, 01 1981.
- [15] E. Rivas and S. R. Eddy, "A dynamic programming algorithm for rna structure prediction including pseudoknots," *Journal of Molecular Biology*, vol. 285, no. 5, pp. 2053–2068, 1999.
- [16] A. Kravchenko, "Predicting rna secondary structures including pseudoknots," 2009.
- [17] A. E. Fatmi, A. Chentoufi, M. A. Bekri, S. Benhlima, and M. Sabbane, "A heuristic algorithm for rna secondary structure based on genetic algorithm," in *2017 Intelligent Systems and Computer Vision (ISCV)*, pp. 1–7, April 2017.
- [18] A. El Fatmi, M. A. Bekri, and S. Benhlima, "Rnaknot: A new algorithm for rna secondary structure prediction based on genetic algorithm and grasp method," *Journal of Bioinformatics and Computational Biology*, vol. 17, no. 05, p. 1950031, 2019.



- [19] D. H. Turner and D. H. Mathews, "Nndb: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure," *Nucleic Acids Research*, vol. 38, pp. D280–D282, 2009.
- [20] D. Huson, "Lecture notes Algorithms in Bioinformatics I," December 2006.
- [21] M. Andronescu, V. Bereg, H. Hoos, and A. Condon, "Rna strand: the rna secondary structure and statistical analysis database," *BMC bioinformatics*, vol. 9, p. 340, 09 2008.
- [22] M. Taufer, A. Licon, R. Araiza, D. Mireles, F. H. D. van Batenburg, A. P. Gulyaev, and M.-Y. Leung, "Pseudobase++: an extension of pseudobase for easy searching, formatting and visualization of pseudoknots," *Nucleic acids research*, vol. 37, p. D127–35, January 2009.
- [23] Y. Zhao, J. Wang, C. Zeng, and Y. Xiao, "Evaluation of rna secondary structure prediction for both base-pairing and topology," *Biophysics Reports*, vol. 4, 06 2018.
- [24] P. Kerpedjiev, S. Hammer, and I. L. Hofacker, "Forna (force-directed RNA): Simple and effective online RNA secondary structure diagrams," *Bioinformatics*, vol. 31, pp. 3377–3379, 06 2015.
- [25] K. Darty, A. Denise, and Y. Ponty, "VARNA: Interactive drawing and editing of the RNA secondary structure," *Bioinformatics*, vol. 25, pp. 1974–1975, 04 2009.

## A. Apéndice

Tabla 7: Métricas de precisión por instancia y algoritmo.

Clave	Algoritmo	VP	FN	FP	S	VPP	Valor-F	Clave	Algoritmo	VP	FN	FP	S	VPP	Valor-F
PDB_00136	Nussinov	18	8	10	0.6923	0.6428	0.6666	PKB5	Nussinov	0	8	13	0.0	0.0	0.0
	Zuker	18	8	10	0.6923	0.8571	0.7659		Zuker	5	3	6	0.6250	0.4545	0.5263
	Akutsu	19	7	9	0.7307	0.6785	0.7037		Akutsu	8	0	6	1.0	0.5714	0.7272
	RNAknot	18	8	1	0.6923	0.9473	0.80		RNAknot	4	4	4	0.50	0.50	0.50
PDB_01130	Nussinov	0	20	24	0.0	0.0	0.0	PKB73	Nussinov	7	5	4	0.5833	0.6363	0.6086
	Zuker	20	0	0	1.0	1.0	1.0		Zuker	7	5	2	0.5833	0.7777	0.6666
	Akutsu	16	4	10	0.80	0.6153	0.6956		Akutsu	12	0	0	1.0	1.0	1.0
	RNAknot	0	20	19	0.0	0.0	0.0		RNAknot	11	1	0	0.9166	1.0	0.9565
PDB_00481	Nussinov	0	26	24	0.0	0.0	0.0	PKB74	Nussinov	5	6	5	0.4545	0.50	0.4761
	Zuker	14	12	0	0.5384	1.0	0.70		Zuker	7	4	0	0.6363	1.0	0.7777
	Akutsu	0	26	22	0.0	0.0	0.0		Akutsu	11	0	0	1.0	1.0	1.0
	RNAknot	7	19	7	0.2692	0.50	0.35		RNAknot	8	3	0	0.7272	1.0	0.8421
PDB_01115	Nussinov	9	17	18	0.3461	0.3333	0.3396	PKB80	Nussinov	0	12	15	0.0	0.0	0.0
	Zuker	20	6	1	0.7692	0.9523	0.8510		Zuker	0	12	10	0.0	0.0	0.0
	Akutsu	6	20	22	0.2307	0.2142	0.2222		Akutsu	12	0	5	1.0	0.7058	0.8275
	RNAknot	10	16	10	0.3846	0.50	0.4347		RNAknot	12	0	0	1.0	1.0	1.0
PDB_01113	Nussinov	8	19	19	0.2962	0.2962	0.2962	PKB81	Nussinov	2	6	6	0.25	0.25	0.25
	Zuker	18	9	1	0.6666	0.9473	0.7826		Zuker	5	3	0	0.6250	1.0	0.7692
	Akutsu	6	21	22	0.2222	0.2142	0.2181		Akutsu	8	0	0	1.0	1.0	1.0
	RNAknot	6	21	11	0.2222	0.3529	0.2727		RNAknot	6	2	0	0.75	1.0	0.8571
PDB_01128	Nussinov	0	27	27	0.0	0.0	0.0	PKB107	Nussinov	0	12	17	0.0	0.0	0.0
	Zuker	12	15	6	0.4444	0.6666	0.5333		Zuker	0	12	11	0.0	0.0	0.0
	Akutsu	14	13	19	0.5185	0.4242	0.4666		Akutsu	12	0	5	1.0	0.7058	0.8275
	RNAknot	8	19	9	0.2962	0.4705	0.3636		RNAknot	12	0	0	1.0	1.0	1.0
PDB_01125	Nussinov	18	18	18	0.50	0.50	0.50	PKB243	Nussinov	0	31	45	0.0	0.0	0.0
	Zuker	33	3	0	0.9166	1.0	0.9565		Zuker	20	11	7	0.6451	0.7407	0.6896
	Akutsu	27	9	6	0.75	0.8181	0.7826		Akutsu	22	9	19	0.7096	0.5365	0.6111
	RNAknot	24	12	0	0.6666	1.0	0.80		RNAknot	11	20	16	0.3548	0.4074	0.3793
PDB_00213	Nussinov	9	32	30	0.2195	0.2307	0.2250	PKB273	Nussinov	5	6	11	0.4545	0.3225	0.3703
	Zuker	30	11	3	0.7317	0.9090	0.8108		Zuker	6	5	6	0.5454	0.50	0.5217
	Akutsu	32	9	9	0.7804	0.7804	0.7804		Akutsu	10	1	6	0.9090	0.6250	0.7407
	RNAknot	23	18	9	0.5609	0.7187	0.6301		RNAknot	6	5	3	0.5454	0.6666	0.60
PDB_00542	Nussinov	8	42	40	0.16	0.1666	0.1632	PKB280	Nussinov	6	8	17	0.4285	0.2608	0.3243
	Zuker	32	18	8	0.64	0.80	0.7111		Zuker	7	7	11	0.50	0.3888	0.4375
	Akutsu	19	31	30	0.38	0.3877	0.3838		Akutsu	12	2	13	0.8571	0.48	0.6153
	RNAknot	19	31	14	0.38	0.5757	0.4578		RNAknot	13	1	0	0.9285	1.0	0.9629
PDB_01236	Nussinov	26	28	32	0.4814	0.4482	0.4642	PKB281	Nussinov	3	5	9	0.3750	0.25	0.30
	Zuker	37	17	14	0.6851	0.7254	0.7047		Zuker	3	5	3	0.3750	0.50	0.4285
	Akutsu	29	25	30	0.5370	0.4915	0.5132		Akutsu	0	8	10	0.0	0.0	0.0
	RNAknot	24	30	18	0.4444	0.5714	0.50		RNAknot	8	0	1	1.0	0.8888	0.9411