Identificación biológica de repeticiones en secuencias de ADN del genoma humano

Marcos A. Foglino
mfoglino@dc.uba.ar

Directores
Martín Urtasun murtasun@dc.uba.ar
Verónica Becher vbecher@dc.uba.ar

Tesis de licenciatura

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Argentina

October 26, 2009

Resumen

En [5] Paul Vitányi definió la Medida de Similitud Universal (Universal Similarity Metric) basada en la complejidad de largo de programa definida por Chaitin-Kolmogorov [16][8]. Esta métrica aplicada a dos secuencias de caracteres, permite medir su similitud en función de la cantidad de información que tienen en común. Dado que esta noción no es computable, el mismo Vitányi propuso su aproximación utilizando compresores estándares de texto, a la cual denominó Distancia de Compresión Normalizada (Normalized Compression Distance, NCD).

Esta noción de distancia o similitud, puede ser aplicada al dominio de la biología molecular, debido a la utilización de cadenas de caracteres como medio de representación. En particular, el ADN es representado como una secuencia de caracteres correspondientes al alfabeto A, G, C y T. El estudio de las repeticiones de secuencias dentro del ADN, es algo de relevante importancia hoy en día.

Con el objetivo de aplicar los conceptos relacionados con la NCD a grandes volúmenes de información de naturaleza biológica, el grupo KAPOW desarrolló e implementó varios algoritmos, dentro de los cuales se destaca la herramienta *findpat*[3], la cual es capaz de identificar repeticiones entre secuencias de gran tamaño.

El principal aporte del presente trabajo, consistió en la identificación biológica de las repeticiones detectadas por *findpat* al aplicarse a cromosomas enteros del hombre. Dicha identificación, permitió no sólo confirmar el buen funcionamiento de la herramienta, sino también despertar el interés del estudio particular de las salidas generadas por el programa, aplicado a diversas fuentes de datos (genes, transposones, etc.). A partir de la identificación lograda, se realizó un análisis comparativo entre las repeticiones encontradas entre los cromosomas de la especie humana y algunas bases de datos de secuencias conocidas, obteniendo conclusiones que abren las puertas para futuros estudios.

Abstract

On [5] Paul Vitányi defined the Universal Similarity Metric, based on the program-size complexity by Chaitin-Kolmogorov [16][8]. This metric applied to two sequences of characters, allows to measure their similarity in terms of the amount of information they have in common. Since this notion is not computable, Vitányi proposed his approximation using standard text compressors. He called it the Normalized Compression Distance (NCD).

This notion of distance or similarity can be applied to the domain of molecular biology, because of the using of strings as a mean of representation. In particular, DNA is represented as a sequence of characters belonging to the alphabet A, G, C and T. Investigating the repeats within DNA is something extremely important nowadays.

Aiming to implement the concepts related to the NCD for large volumes of biological information, the Kapow group developed and implemented several algorithms. Among them, *findpat*[3] is the one that highlights the most. It identifies repeats between large sequences.

The main contribution of this work was the identification of biological repeats identified by *findpat* when applied to whole human chromosomes. Not only did this identification allowed to prove the proper functionality of the tool, it also aroused interest over the output generated by the program when applied to various sources of data (genes, transposons, etc.). From the successful identification, we made an analysis of the repeats found among the different chromosomes of the human species, comparing them against some databases of known sequences, drawing conclusions that open the door for future studies.

Contents

1 Objetivo **6**

2 Introducción **6**
2.1 Conceptos básicos de la biología molecular 6
2.1.1 Elementos transponibles o transposones 8
2.2 Motivación 9
2.3 *BLAST* 11
2.3.1 NCBI *BLAST* 11
2.4 *findpat* 12
2.4.1 ¿Qué es un patrón? 12
2.4.2 Entrada 12
2.4.3 Salida 13
2.5 Breve reseña sobre otros algoritmos 14
2.5.1 RepeatMasker 14
2.5.2 Tandem Repeat Finder (TRF) 14
2.5.3 DUST 14

3 La primer identificación **16**
3.1 Búsquedas en *BLAST* 16
3.2 El transposón M80343 17

4 Patrones y Repeticiones **18**
4.1 Sobre los patrones analizados 19
4.2 Sobre las repeticiones analizadas 19
4.3 Algoritmo 23
4.3.1 Descripción 23
4.3.2 Entrada 23
4.3.3 Salida 24
4.3.4 Pseudocódigo 26
4.4 Comparaciones realizadas 29
4.4.1 Resultados obtenidos 30
4.4.2 Tiempos de ejecución 33
4.5 El patrón más “famoso” 34
4.6 Posición de los patrones respecto a las *repeticiones* 35

5 Conclusiones y Preguntas Abiertas **36**

A Tablas de resultados **40**

Referencias **43**

Agradecimientos

A:
Verónica, Martín,
KAPOW, Hernan, Javier Herrero, la gente que hizo trabajos prácticos conmigo, mi mamá, mi hermano, mi tía Ana, mi abuela, mi gata, mi novia, Seba y Mariano, hal9000, multivac, grep, find, cut, wc, excel, access.

1 Objetivo

El grupo KAPOW desarrolló el algoritmo *findpat*[3], el cual fue aplicado al genoma humano, produciendo así un conjunto de repeticiones, las cuales, en principio, no sabemos si son coherentes con lo conocido hoy en día en materia de repeticiones en el ADN. La meta de este trabajo es poder validar si las repeticiones o patrones sobre secuencias de ADN que el algoritmo *findpat* reporta, tienen un interés desde el punto de vista biológico.

2 Introducción

2.1 Conceptos básicos de la biología molecular

A continuación se abordarán los conceptos básicos sobre biología molecular necesarios para el desarrollo y la comprensión de este trabajo. Fuentes: [2][19][20].

El ácido desoxirribonucleico (**ADN**) es un ácido nucleico que contiene la información genética usada en el desarrollo y el funcionamiento de la mayoría de los seres vivos. La función principal de las moléculas de ADN es la de ser portador y transmisor de la información genética entre las sucesivas generaciones de organismos vivos.

Desde el punto de vista químico, el ADN está compuesto de nucleótidos o bases: la adenina, la timina, la citosina y la guanina. Estas bases son generalmente referenciadas por sus iniciales: A, T, C y G. Estructuralmente, el ADN está formado por dos hebras (strands) o cadenas de bases entrelazadas que conforman una doble hélice de tal manera que las bases de tipo A se aparean solo con las bases T y las bases C lo hacen solamente con las bases G. Debido a esta propiedad de apareamiento, una hebra de la molécula puede ser determinada unívocamente examinando la otra hebra y viceversa.

Las **proteínas** son la "maquinaria" de la célula. Son macromoléculas codificadas por los genes que llevan a cabo la mayoría de las actividades biológicas de las células. Algunas tienen una función estructural, dando forma a las células. Otras cumplen un papel funcional, como las proteínas contráctiles del músculo o las enzimas, que hacen posibles las reacciones químicas que tienen lugar en el organismo. Desde el punto de vista químico están formadas por cadenas lineales de aminoácidos.

Las proteínas se construyen a partir de secuencias de ADN y es por esto que la información que contiene el ADN es esencial para el organismo.

Las proteínas son sintetizadas por las regiones codificantes del ADN (**genes**). Las regiones codificantes son caracterizadas por tripletes de bases (tres bases contiguas), por ende hay 64 (4^3) tripletas posibles. Estas tripletas reciben el nombre de codones. En el proceso de "traducción" de ADN a proteínas, cada codón se mapea con un único aminoácido. Este mapeo recibe el nombre de código genético. Dado que hay solo 20 aminoácidos y 64 codones, hay aminoácidos que son mapeados por más de un codón.

Los **genes** consisten de una región codificante y una región regulatoria. La región codificante es la parte del gen que codifica una proteína. La región regulatoria es la porción del

ADN que contribuye al control de expresión del gen en respuesta a señales del medio celular.

Una cadena de ADN presenta dos tipos de regiones: ADN codificante y ADN no codificante. El ADN codificante está constituido por los genes que son las unidades de información hereditaria. El ADN no codificante es llamado así, ya que este no codifica proteínas, razón por la cual también se lo denomina ADN basura. Sin embargo, en los últimos años se ha descubierto que ciertas regiones cumplen importantes funciones biológicas.

En el nucleo de cada celula, una molecula de ADN es empaquetada en estructuras similares a hilos llamados **cromosomas**. Cada cromosoma esta hecho de ADN enrollado en proteinas que soportan su esctructura[1].

Todo el conjunto de cromosomas que esta dentro de la celula recible el nombre de **genoma**.

En la Tabla 1 pueden verse cada uno de los cromosomas que conforman el genoma humano, y sus respectivos tamaños [10].

Cromosoma	Tamaño (Mbp)
1	249
2	237
3	192
4	183
5	174
6	165
7	153
8	135
9	132
10	132
11	132
12	123
13	108
14	105
15	99
16	84
17	81
18	75
19	69
20	63
21	54
22	57
X	141
Y	60

Tabla 1: Tamaño de los cromosomas humanos
Los cromosomas que conforman el genoma humano, con su respectivo tamaño en millones de bases o caracteres.

Un **transposón** es una secuencia de ADN que puede moverse o copiarse desde un lugar a otro dentro del genoma, este proceso es llamado transposición. En el proceso pueden causar una mutación y cambiar la cantidad de ADN en el genoma. Debido a esta propiedad de copiarse o moverse, antiguamente eran conocidos como "genes saltarines". La mayor parte del ADN esta formada por ellos. Actualmente se les atribuye a estos elementos un rol importante en la evolución del genoma. La genética comparativa a puesto de manifiesto que diferentes linajes de vertebrados contienen, de manera cualitativa y cuantitativa, diferentes poblaciones de elementos de ADN (transposones y retrotransposones). Los transposones son herramientas

[1]Genetics Home Referene, http://ghr.nlm.nih.gov/handbook/basics/chromosome.

muy poderosas en el ambiente de los genetistas, y el descubrimiento de un nuevo transposón en un organismo dado puede ayudar mucho en los estudios genéticos de tal organismo.

En este trabajo usamos mucho el concepto de repeticiones y transposones, por lo cual detallamos un poco mas estos aspectos en la sección 2.1.1.

El **dogma central de la biología molecular**, dice que la información genética está almacenada en el ADN, es transcripta a ARN mensajero (ARNm) y luego traducida a proteínas. Si se considera que algunas proteínas regulan la transcripción, se ve que los factores de ésta proveen una retroalimentación , en la que algunos genes regulan la expresión de otros genes. Este esquema puede verse en la Figura 1.

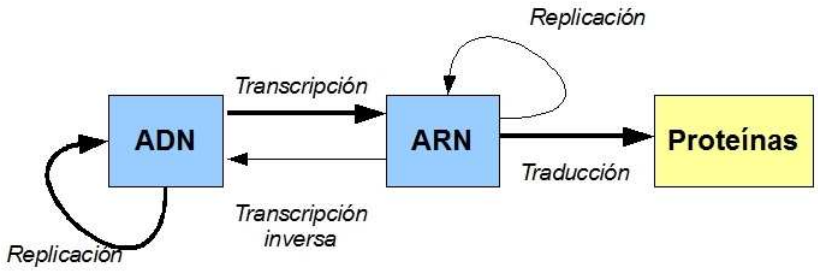

Figura 1: Dogma de la biologia molecular
Esquema del dogma central de la biología molecular, que muestra el flujo de la información genética de las células. Los genes se perpetúan como secuencias de ácidos nucleicos, pero funcionan mediante su traducción a proteínas. La replicación es la responsable de la herencias de la información genética, mientras que la transcripción y la traducción hacen posible la conversión de ADN en ARN y de ARN a proteínas respectivamente.

2.1.1 Elementos transponibles o transposones

Estos elementos comprenden una gran porción del genoma de las eucariotas como secuencias repetidas. Se cree que los elementos transponibles (Trasposable elements, TEs) juegan un rol importante en la evolución de estos genomas. Los *TEs* son secuencias de ADN que pueden moverse desde una posición de un cromosoma a otra mas rápido que lo que lleva a un cromosoma replicarse. La presencia de estos elementos puede ser demostrada usando programas que detectan regiones de baja complejidad (*low-complexity regions*) en secuencias (ver sección 2.5).

Los *TEs* de las eucariotas pueden dividirse en dos clases principales en cuanto a la similaridad de las secuencias y el mecanismo de transposición:

- Clase I, esta basada en un mecanismo de transcripción mediante ARN. Existen tres subclases principales de estos.

  Las repeticiones terminales largas (Long Terminal Repeats retrotransposons, LTRs retrotransposons), que están relacionados por su estructura genética con los retrovirus. Estos se propagan por medio de transcripciones de ARN.

  Las repeticiones nucleares interdispersas cortas (Short Interspersed Nuclear Elements, SINEs).

Las repeticiones nucleares interdispersas largas (Long Interspersed Nuclear Elements, LINEs).

- Clase II, son las que usan un mecanismo de transcripción basado en ADN. Se mueven dentro del ADN por un mecanismo de "cortar y pegar".

Resumiendo, podemos decir que las repeticiones conforman mas del **45%** del genoma humano.

Elemento	Tamaño(bp)	Cantidad de Copias	Porcentaje del genoma
SINEs	100-300	1.500.000	13%
LINEs	6000-8000	850.000	21%
LTRs	15.000-110.000	450.000	8%
Transposones de ADN fosiles	80-3000	300.000	3%

Tabla 2: Composición del genoma humano

2.2 Motivación

El presente trabajo surge en el contexto de investigación del grupo KAPOW[2]. Este grupo de la *Facultad de Ciencias Exactas y Naturales* de la *Universidad de Buenos Aires*, tiene como tema principal de investigación la resolución de problemas sobre secuencias de caracteres[3].

En el marco de la biología molecular, como se dijo antes, las secuencias de caracteres suelen ser utilizadas como medio de representación. En particular, el ADN es representado mediante una secuencia de caracteres, con un alfabeto conformado por cuatro letras: A, C, G y T. Este hecho, permite que la resolución de problemas sobre cadenas de caracteres desde el punto de vista de la computación, tenga un interés particular desde el punto de vista de la biología molecular. Unos de los principales objetivos del grupo KAPOW es abordar algunos de estos problemas, en el contexto de un dominio biológico, manejando grandes volúmenes de datos.

Las cuestiones abordadas principalmente fueron:

- Conocer la similitud entre dos secuencias de ADN.
- Identificar las regiones comunes entre dos secuencias de ADN.
- Identificar repeticiones dentro de una misma secuencia.

El punto de partida para la concreción de estos objetivos fue el uso de la *Normalized Compression Distance* definida en [9]. Esta noción, brinda la posibilidad de estimar la distancia entre secuencias de caracteres, utilizando algoritmos de compresión. La NCD es una aproximación a un concepto teórico denominado *Similitud Universal* [5] que define la similitud entre dos secuencias en función de su *complejidad algorítmica* [8]. Existen varios trabajos sobre la

[2]http://kapow.dc.uba.ar

[3]En el presente trabajo, se utilizarán los términos *palabra*, *string*, *cadena de caracteres* y *secuencia de caracteres* de manera indistinta.

utilización de la NCD (o variantes de la misma) en el campo de la biología. Sin embargo la gran mayoría, tarde o temprano, se encuentra frente a una serie de problemas con los cuales también tuvimos que lidiar:

- La NCD puede interpretarse como una función que recibe como argumento dos secuencias y devuelve un número real entre 0 y 1. Dicho número representa la similitud entre las secuencias. Para dos secuencias idénticas, el valor correspondiente será uno. Para dos secuencias totalmente distintas, cero. No obstante, no brinda información alguna sobre por qué dos secuencias son similares: no permite exhibir, detallar o listar la información en común, aquellas subsecuencias de caracteres que comparten ambas secuencias.

- Es posible aproximar la NCD utilizando compresores estándares (GZIP, BZIP2, etc.). Sin embargo, por lo general estos compresores encuentran rápidamente limitaciones frente a grandes volúmenes de datos, ya que por cuestiones implementativas los mismos no cumplen las propiedades necesarias para definir una nocion de distancia. Por ejemplo algunos de estos compresores utilizan un sistema de ventana deslizante, y al ser el tamaño de la misma muy chico, hace que estos compresores no cumplan las propiedades de *distancia* para secuencias muy grandes[6]. Cabe destacar, que las secuencias de caracteres a considerar poseen una longitud muy grande, llegando hasta 1GB gigabyte en algunos casos.

- Existen muchos y distintos algoritmos de compresión, cada uno de los cuales posee su propio método o estrategia para lograr la compresión de información. No todos estos métodos son aplicables a la aproximación de la complejidad algorítmica. Por lo tanto, no cualquier compresor puede ser utilizado para estimar la NCD de dos cadenas.

Habiendo descartado el uso de algún compresor estándar y con el objetivo concreto de comparar los genomas completos de cinco especies: humano, chimpancé, perro, ratón y rata, el grupo KAPOW tomó las siguientes decisiones:

- Proponer una noción de similitud entre secuencias basada en la complejidad algorítmica.
- Adaptar algún algoritmo de compresión existente o diseñar uno nuevo, e implementarlo con el fin aproximar la noción de complejidad algorítmica, permitiendo al mismo tiempo trabajar con secuencias de caracteres de longitudes en el orden de los gigabytes.
- Diseñar e implementar un algoritmo que permita identificar y extraer la información en común entre dos secuencias de considerable tamaño. Esto último oculta una gran dificultad asociada: es necesario definir que se entiende como información en común.

Finalmente, se desarrollaron una serie de herramientas de las cuales destacaremos a *findpat*, la cual permite encontrar repeticiones o patrones entre secuencias de caracteres[4]. En la sección 2.4 encontraremos una descripción de la misma.

[4] *findpat* considera un alfabeto reducido compuesto por sólo cuatro caracteres: A, C, G y T.

Teniendo la salida de *findpat*, necesitamos ver sí estos patrones (repeticiones) tienen un interés desde el punto de vista biológico.

A partir de varios gigabytes (mas de 200GB) de secuencias de ADN (correspondientes a los distintos cromosomas de los distintos genomas), y luego de muchísimas horas de cómputo, utilizando *findpat* es posible obtener varios (del orden de los millones) patrones de distintas longitudes (que pueden ir desde algunos bytes hasta varios miles). ¿Cómo determinar la utilidad de estas secuencias encontradas?, ¿Cómo corroborar que estas secuencias tengan alguna interpretación biológica relevante?. Si la solución es buscar cada patrón en alguna de las bases de datos de secuencias conocidas, ¿Por cuál empezar?, ¿Cuál base de datos se debe utilizar?, ¿Cuál es el mejor criterio de búsqueda?. En síntesis, la principal pregunta que debíamos responder era la siguiente: entre esos varios gigabytes de información generada por *findpat* a partir de genomas conocidos, ¿existe alguna secuencia biológicamente reconocida o etiquetada?. De haberla, podría inducir futuros análisis y/o experimentos. Hasta tanto no tengamos estas respuestas, la real utilidad de *findpat* para este tipo de trabajos de investigación no podrá ser comprobada.

2.3 *BLAST*

Como las bases de datos de secuencias de ADN y de aminoácidos continúan creciendo en tamaño, también crece su utilidad para el análisis de los de genes y proteínas, debido a las posibilidades de encontrar tales coincidencias. Es por esto que contar con un algoritmo para buscar secuencias de ADN en estas bases de datos es de mucha utilidad.

Este algoritmo propuesto en [1], permite a un investigador comparar una secuencia de ADN contra una, o un conjunto de bases de datos, e identificar secuencias dentro de estas con las cuales se asemeja.

El mismo introduce varios refinamientos a la búsqueda en bases de datos, que mejoran el tiempo de búsqueda. *BLAST* hace énfasis en la velocidad por sobre la sensibilidad. Esto es fundamental para que el algoritmo sea practico a la hora de buscar en las bases de datos gigantes que están disponibles hoy día.

BLAST no esta basado en un algoritmo que garantiza el alineamiento óptimo, sino que usa una *heurística* que en la practica funciona bien la mayoría de las veces, pero podría fallar con algunas secuencias poco relacionadas entre sí. *BLAST* es alrededor de 50 veces más rápido que otros algoritmos que garantizan el alineamiento local de secuencias óptimo, usando programación dinámica.

2.3.1 NCBI *BLAST*

Existen varias implementaciones de la idea que propone *BLAST*. En este trabajo utilizamos el *BLAST* implementado por el *National Center for Biotechnology Information (NCBI)*. Una de las ventajas de ésta implementación, es que puede comparar secuencias contra la base de datos *GenBank*.

GenBank es una base de datos de secuencias genéticas del *National Institute of Health*,

que incluye una colección anotada[5] de la gran mayoría de las secuencias de ADN disponibles públicamente.

2.4 *findpat*

Es un algoritmo que encuentra repeticiones (patrones) maximales con matching exacto dentro de secuencias de ADN, y sin límite en la longitud de las repeticiones que este arroja. Esto lo hace de una manera conocida como *ab initio*, ya que no necesita de alguna semilla (secuencia de referencia) previa para su funcionamiento. Solamente necesita como entrada una secuencia o dos (en el caso de buscar patrones entre dos secuencias), y la longitud mínima deseada para las repeticiones resultantes. Este algoritmo puede recibir secuencias genéticas de hasta 500 Megabytes, es así que brinda la posibilidad de analizar cromosomas enteros (un cromosoma, o parejas de cromosomas). Por una cuestión de simplicidad y mas que nada por cuestiones implementativas del algoritmo, se buscan repeticiones con correspondencia exacta y que cumplen con la definición de patrón que abordaremos en la siguiente sección. Con correspondencia exacta queremos decir que no se permite inserción, modificación, o eliminación de caracteres para la búsqueda de los patrones.

2.4.1 ¿Qué es un patrón?

findpat trabaja con secuencias maximales, con matching exacto. Para *findpat*, un **patrón** es una secuencia de ADN que aparece al menos 2 veces, y cada una de sus extensiones (ya sea hacia la izquierda, derecha o en ambas direcciones) ocurre menos veces. En este sentido, se dice que las secuencias son maximales.

Por ejemplo, sea la secuencia:

$$ABEABCCABCE$$

$$ABE\boldsymbol{ABC}C\boldsymbol{ABC}E$$

$$\boldsymbol{AB}E\boldsymbol{AB}CC\boldsymbol{AB}CE$$

Los patrones de al menos dos caracteres son ABC y AB. BC no es patrón ya que es subcadena de ABC pero no aparece más veces que ABC.

2.4.2 Entrada

findpat recibe como entrada:

- Una sola secuencia, en cuyo caso un patrón será reportado cuando aparezca al menos dos veces dentro de la misma. O dos secuencias, en cuyo caso un patrón será reportado cuando aparezca al menos una vez en cada una de las secuencias.
- La longitud mínima de los patrones a reportar.

[5] Una secuencia anotada, es aquella que tiene asociada algún tipo de información, y se conoce algo al respecto de la misma.

Ejemplos:

`./findpat secuencia1 20`
(busca secuencias de longitud mayor a 20 caracteres, repetidas dentro de *secuencia1*)

`./findpat secuencia1 secuencia2 40`
(busca secuencias en común entre *secuencia1* y *secuencia2* de longitud mayor a 40)

2.4.3 Salida

Como resultado de una corrida de *findpat* obtenemos un archivo de texto, que contiene los patrones encontrados y alguna información sobre cada uno de ellos. Explicaremos la estructura del archivo de salida con un ejemplo. Supongamos que ejecutamos *findpat* para obtener los patrones en común entre el cromosoma 1 y el cromosoma 2 del hombre. En la Figura 2 podemos ver como son los primeros registros de esta salida.

En dicha figura se muestran los primeros 10 patrones en común entre *cromosoma1* y el *cromosoma2* del hombre. La información de cada patrón se muestra en dos líneas de texto. La primera contiene el patrón (secuencia) propiamente dicho, la cantidad de apariciones del mismo (precedida por el símbolo #), y la longitud de este (encerrada entre paréntesis). La segunda, es una lista con las posiciones de cada una de las apariciones del patrón en cuestión, relativas al cromosoma pertinente. En nuestro caso cuando una posición se encuentra precedida por el símbolo "<", entonces quiere decir que la posición de esa aparición es relativa al *cromosoma1*. En cambio si el símbolo que la precede es ">" dicha aparición es en el *cromosoma2*.

Para mayor detalle sobre este algoritmo, referirse a [3].

```
TGCTGTCTTTTTGTTTGTCTGTGCCCTGCCCCCAGAGGTGGAGCCTACAGAGGCAGGCAGGCCTCCTTGAGCTGTGGTGGGCTCCACCCAGTTCGAGCTT #68 (100)
  <168330379 >122825776 <246123686 >192846841 <69119492 <82374631 <105119837 >86741317 <34809707 >153577972 >176166957 >185997849 <179108016 <68977821
TAATGGTAAAGGGATCAATTCAACAAGAAGAGCTAACTATCCTAAATATATATGCACCCAATACAGGAGCACCCAGATTCATAAAGCAAGTCCTGAGTGA #55 (100)
  <25707935 >57501616 >58408377 >115480678 <156470654 <218105807 <45420921 <108835583 <40605375 <116398462 <49497829 <73432417 >76987394 >77302253
TTTCATCCATGTCCCTACAAAGGACATGAACTCATCATTTTTTATGGCTGCATAGTATTCCATGGTGTATATGTGCCACATTTTCTTAATCCAGTCTATC #227 (100)
  >179673042 >16921033 >160556518 >79186626 >81431928 >205779483 >153816528 >159345073 >228993247 <229673595 >66005125 >210171714 <215930752 <86183167
TAAAACCATAAAAACCCTAGAAGAAAACCTAGGCATTACCATTCAGGACATAGGCATGGGCAAGGACTTCATGTCTAAAACACCAAAAGCAATGGCAACA #116 (100)
  <88450630 >158920868 <88585008 <187995573 <111684381 >227642709 <34072719 <80647392 <157194049 <34335480 <175874574 >182790270 >207578770 >134025665
GGCGCGGTGGCTCACGCCTGTAATCCCAGCACTTTGGGAGGCCGAGGCGGGTGGATCATGAGGTCAGGAGATCGAGACCATCCTGGCTAACAAGGTGAAA #92 (100)
  >130235140 >165237177 >77430679 >203359237 <243816531 >65195204 <238683635 >55875785 >84638004 >202429914 >203100476 <54554623 <24831013 >66029735
TCACAATAGCAAAGACTTGGAACCAACCCAAATGTCCATCAGTGATAGACTGGATTAAGAAAATGTGGCACATATACACCATGGAATACTATGCAGCCAT #11 (100)
  >46304610 <93135371 >67105242 <146440450 <99098079 <57807568 <190068854 >223742319 <118782879 >102090105 >98060250
TAATGGTAAAGGGATCAATTCAACAAGAAGAGCTAACTATCCTAAATATATATGCACCCAATACAGGAGCACCCAGATTCATAAAGCAAGTCCTTAGAGA #19 (100)
  >9027363 >118055694 >53243778 <168702040 <50175512 <177075672 <90407388 >68678009 >63470448 <160686983 <91537342 >223159733 >191046456 <218129067
TGGTGTATAAGAATGCTTGTGATTTTTGTACATTGATTTTGTATCCTGAGACTTTGCTGAAGTTGCTTATCAGCTTAAGGAGATTTTGGGCTGAGACGAT #47 (100)
  >108785327 >72347703 >84451709 <197259797 >34651964 >205341779 <81179319 >41632466 >214142013 <105116204 >185522079 <215932266 >134685042 <178936785

⋮
```

Figura 2: Encabezado del archivo patrones_Homo1Homo2_MayoresA100.txt

Aquí puede observarse el formato de la salida de *findpat*. Se muestran 10 patrones. Para cada patrón (secuencia) se detalla la cantidad de apariciones del mismo (por ejemplo: #68) y entre paréntesis la longitud del patrón, en este caso todos tienen una longitud 100 de caracteres. Además cada patrón posee una línea adicional donde se indican la posiciones de cada una de sus apariciones dentro de los cromosomas originales. Se indica con "<" si la aparición es en el primer cromosoma de la pareja, y con ">" si la aparición ocurre en el segundo cromosoma. Por ejemplo para el primer patrón sabemos que aparece en las posiciones 168330379, 246123686, 69119492, etc. del cromosoma1, y en las posiciones 122825776, 192846841, 86741317, etc. del cromosoma2.

2.5 Breve reseña sobre otros algoritmos

Aquí hablaremos brevemente de algunos algoritmos mencionados en este trabajo.

2.5.1 RepeatMasker

RepeatMasker[22] es la herramienta predominante, basada en librerías[6], usada en la detección de repeticiones, y llego a ser el estándar *de facto* para este fin, por sobre otros métodos.

RepeatMasker fue diseñada para descubrir repeticiones y enmascararlas (i.e. removerlas) para prevenir complicaciones en el ensamblado de secuencias y en la caracterización de genes [21].

2.5.2 Tandem Repeat Finder (TRF)

A *tandem repeat* es un patrón dentro de una secuencia de ADN, el cual tiene varias apariciones, una al lado de otra, dentro de esa secuencia.

Por ejemplo, en la secuencia $ACTG\boldsymbol{GTAGTAGTA}GCAGC$, GTA es un *tandem repeat* ya que aparece tres veces de manera contigua.

TRF[4] es un programa para localizar y mostrar *tandem repeats*.

Para usar el mismo, el usuario ingresa una secuencia en formato *FASTA*[7], sin necesidad de ingresar otros parámetros. La salida consiste de dos archivos, un archivo con la tabla de repeticiones y un archivo con los alineamientos. La tabla contiene información acerca de cada repetición, incluyendo su locación, tamaño, numero de copias, etc. Seleccionando los índices de cada locación, se despliega una nueva pagina web mostrando los alineamientos de cada copia del *tandem repeat* contra la secuencia original.

2.5.3 DUST

Dentro de una secuencia de ADN, es posible encontrar repeticiones de secuencias de un mismo caracter.

Las ocurrencias de tales repeticiones dificulta el alineamiento de secuencias. En *BLAST*, que realiza alineamiento de secuencias, esto puede provocar falsos positivos o falsos negativos, reportando de esta manera resultados espurios.

Un problema similar ocurre con regiones en las cuales se encuentran secuencias repetidas de muy pocos caracteres, llamadas regiones de baja complejidad (low-complexity regions).

Es por esto que se hace necesario un programa que elimine estas regiones antes de usar un programa como *BLAST*.

[6] Necesita una librería de secuencias como punto de partida, además de la secuencia a analizar.

[7] Es formato basado en texto, para representar secuencias de nucleótidos, contiene un header con una descripción de la secuencia y la secuencia propiamente dicha. http://www.ebi.ac.uk/help/formats.html#fasta.

DUST encuentra este tipo de regiones dentro de una secuencia de nucleótidos y las filtra. De manera que *BLAST* pueda hacer un mejor trabajo a la hora de comparar secuencias contra una base de datos. Es un algoritmo heurístico, que se puede usar en conjunto con *BLAST*o por separado. Para más información ver [2](pag.386), [20](pag.64), y WindowMasker supplementary information[8].

[8] `ftp://ftp.ncbi.nlm.nih.gov/pub/agarwala/windowmasker/windowmasker_suppl.pdf`

3 La primer identificación

Teniendo a nuestra disposición los patrones sobre el genoma humano encontrados por *findpat*, comienza nuestro proceso de identificación biológica de los mismos. En esta primera etapa lo hacemos de una manera manual. La primer identificación (transposón M80343) a priori fue muy afortunada, pues fue resultado de un proceso manual de ir probando uno a uno, los patrones a identificar.

3.1 Búsquedas en *BLAST*

Como mencionamos anteriormente la versión web de *BLAST*, es una herramienta que permite buscar secuencias biológicas en las bases de datos de `NCBI`[9].

Existe una versión de *BLAST* que puede ser ejecutada desde la línea de comandos del sistema operativo, la cual envía la secuencia consultada vía internet, es procesada por *BLAST* y luego retorna el resultado al usuario, en el formato elegido (txt, html, etc).

Nosotros no usaremos esta versión, ya que preferimos usar la versión web[10], debido a que esta permite la navegabilidad de los resultados, y provee una manera clara de visualizar los mismos.

La búsqueda en *BLAST* no es una tarea sencilla, ya que las mismas son realizadas a través de una pagina web, lo cual implica la participación activa de una persona con dicho servicio. Realizar una búsqueda en *BLAST* requiere determinar la base de datos en la cual buscar (de las disponibles en `NCBI`) y los parámetros de configuración a utilizar. En general, éstas decisiones dependen de lo que se esté buscando y en dónde se lo esté buscando. Cuestiones que en nuestro caso eran desconocidas. En este punto, fue fundamental para nosotros la interacción con biólogos para que nos brindasen una introducción que nos permitiese entender el uso de la herramienta.

Si tenemos en cuenta que una búsqueda de este tipo puede demorar como mínimo 30 segundos, y que el volumen de repeticiones a analizar supera los 200GB de información, plantear una búsqueda exhaustiva es una tarea casi impracticable.

Queremos recalcar que de haber usado el *BLAST* por línea de comandos, no hubiéramos mejorado los tiempos de identificación de patrones, ya que el mayor costo de la consulta radica en el tiempo que toma hacer la consulta vía internet, y no contar con las bases de datos de manera local.

En particular la tarea de encontrar una secuencia, entre los millones de secuencias disponibles, que tenga un interés biológico, parece una utopía. A priori, es como buscar una aguja en un pajar. Como veremos en la siguiente sección, la fortuna nos ayudó a encontrar una *aguja*.

[9] National Center for Biotechnology Information.

[10] http://blast.ncbi.nlm.nih.gov/Blast.cgi, opción 'nucleotide blast'.

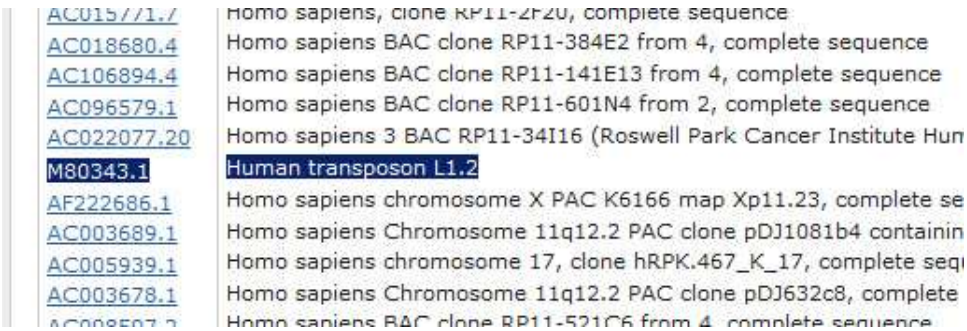

Figura 3: Primer patrón identificado

Esta es una captura de pantalla del resultado de buscar un patrón a través de la pagina web de *BLAST*. En particular este patrón pudo ser identificado contra el transposón M80343, que se ve resaltado en la imagen, ya que este patrón es subsecuencia del transposón.

3.2 El transposón M80343

Desde el punto de vista de la intuición o el sentido común, los primeros resultados de *findpat* reflejaban cierta coherencia y un comportamiento cercano al esperado. Restaba determinar si las repeticiones (patrones) identificadas por el programa tenían algún interés desde el punto de vista biológico.

Como mencionamos antes, la única alternativa para averiguarlo era poder encontrar información sobre estos patrones en alguna de las bases de datos de ADN conocidas. A tal fin se comenzó a buscar manualmente mediante *BLAST*, lo cual llevaba demasiado tiempo por búsqueda y era muy difícil de sistematizar. A pesar de ello fuimos muy afortunados, ya que pudimos encontrar que uno de los tantos patrones hallados se correspondía con un transposón ya conocido, el **M80343** (ver Figura 3).

Un patrón de 781 caracteres, resultó ser parte del transposón M80343. El mismo está catalogado por `NCBI` mas precisamente como *"Human transposon L1.2."* (version M80343.1 gi:339773).

Este hallazgo fue la primer confirmación de que *findpat* es una herramienta confiable para la búsqueda de patrones dentro de cadenas de nucleótidos.

4 Patrones y Repeticiones

La primer identificación obtenida, nos dio una idea sobre qué clase de repeticiones era posible encontrar entre los patrones determinados por *findpat*.

Esto nos incitó a seguir buscando. Sin embargo, la búsquedas mediante la versión web de *BLAST* eran muy demandantes en cuanto a tiempo. Pero como veremos mas adelante, el hecho de contar con una base de datos de *repeticiones* ya conocidas (ver sección 4.2), de manera local, nos motivó para construir un algoritmo que permite identificar nuestros patrones contra esta base de datos de una manera mucho mas eficiente.

Nuestro objetivo es estimar, que tipos de repeticiones fueron encontradas por *findpat* y cuál es la proporción de sus apariciones.

En esta sección veremos que patrones y que *repeticiones* utilizamos en este trabajo, los experimentos realizados para validar *findpat*, los resultados que se desprenden de estos, y el algoritmo utilizado para realizar dichos experimentos de validación.

El algoritmo, que explicaremos luego, esta basado en una búsqueda binaria y básicamente lo que hace es buscar encajes de intervalos. Este concepto es a lo que nosotros nos referimos en este trabajo como "**solapamiento**" (ver sección 4.3).

4.1 Sobre los patrones analizados

Habiendo ejecutado *findpat* sobre todas las parejas posibles entre los distintos cromosomas del genoma humano, se tomó un muestreo eligiendo sólo algunas de las combinaciones posibles. Las parejas seleccionadas fueron:

$ParejasA$: *Homo1–Homo<i> con cromosomas* $i = 2, ..., 22, X, Y$

$ParejasB$: *Homo<i>–Homo<i+1>, con cromosomas* $i = 3, 5, 7, 9, 11, 13, 15, 17, 19, 21$ *y HomoX–HomoY*

Los casos de $ParejasA$ fueron así elegidos ya que Homo1 es uno de los cromosomas más grandes y por lo tanto con mayor probabilidad de tener más cantidad de patrones que puedan solapar con *repeticiones*. Por lo tanto se tomaron las parejas que tienen el cromosoma 1 como primera componente, contra todos los demás cromosomas en la segunda componente.

Los casos de $ParejasB$ fueron elegidos de manera de cubrir todos los cromosomas, pero sin repetir ninguno en las parejas.

La lista de patrones analizados puede verse en la Figura 4a. Cada uno de estos archivos es el resultado del algoritmo $findpat$, aplicado a la pareja correspondiente, luego de ser procesado con el objetivo de descartar los patrones de longitud menor a 100 caracteres (reduciendo así, de manera notable, el espacio de búsqueda).

La decisión de descartar los patrones de longitud menor a 100 caracteres, fue tomada con el objetivo de reducir el espacio de búsqueda. Al quedarnos con las secuencias más grandes esperamos tener mayor probabilidad de que un patrón se superponga con alguna de las *repeticiones* contra las cuales haremos la comparación. Por otro lado es más probable encontrar secuencias de menor longitud dentro del genoma humano, lo cual a priori, parece ser menos interesante, justamente por el hecho de que es más probable encontrar repeticiones de poca longitud.

A los efectos del objetivo de este trabajo, si logramos identificar biológicamente las secuencias mayores a 100 caracteres, esto ya nos da una pauta de que *findpat* es una herramienta útil para encontrar repeticiones dentro de secuencias de gran longitud (del orden de los gigabytes).

4.2 Sobre las repeticiones analizadas

Los patrones encontrados por *findpat* fueron comparados contra una base de datos de *repeticiones*, previamente conocidas, que se encuentran en el genoma humano.

Las *repeticiones* analizadas en este trabajo son secuencias de ADN las cuales aparecen repetidas en diferentes sectores del genoma humano. Comparamos nuestros patrones contra estas repeticiones, ya que estas fueron identificadas anteriormente por la comunidad científica, y están clasificadas según la familia a la que pertenece cada una de ellas. La mayoría de estas *repeticiones* son transposones.

patrones_Homo1Homo2_MayoresA100.txt	patrones_Homo1Homo3_MayoresA100.txt
patrones_Homo1Homo4_MayoresA100.txt	patrones_Homo1Homo5_MayoresA100.txt
patrones_Homo1Homo6_MayoresA100.txt	patrones_Homo1Homo7_MayoresA100.txt
patrones_Homo1Homo8_MayoresA100.txt	patrones_Homo1Homo9_MayoresA100.txt
patrones_Homo1Homo10_MayoresA100.txt	patrones_Homo1Homo11_MayoresA100.txt
patrones_Homo1Homo12_MayoresA100.txt	patrones_Homo1Homo13_MayoresA100.txt
patrones_Homo1Homo14_MayoresA100.txt	patrones_Homo1Homo15_MayoresA100.txt
patrones_Homo1Homo16_MayoresA100.txt	patrones_Homo1Homo17_MayoresA100.txt
patrones_Homo1Homo18_MayoresA100.txt	patrones_Homo1Homo19_MayoresA100.txt
patrones_Homo1Homo20_MayoresA100.txt	patrones_Homo1Homo21_MayoresA100.txt
patrones_Homo1Homo22_MayoresA100.txt	patrones_Homo1HomoX_MayoresA100.txt
patrones_Homo1HomoY_MayoresA100.txt	
patrones_Homo3Homo4_MayoresA100.txt	patrones_Homo5Homo6_MayoresA100.txt
patrones_Homo7Homo8_MayoresA100.txt	patrones_Homo9Homo10_MayoresA100.txt
patrones_Homo11Homo12_MayoresA100.txt	patrones_Homo13Homo14_MayoresA100.txt
patrones_Homo15Homo16_MayoresA100.txt	patrones_Homo17Homo18_MayoresA100.txt
patrones_Homo19Homo20_MayoresA100.txt	patrones_Homo21Homo22_MayoresA100.txt
patrones_HomoXHomoY_MayoresA100.txt	

(a) patrones

human_repeats.txt	*human_other_repeats*.txt

(b) *repeticiones*

Figura 4: Fuentes de datos usadas en los experimentos realizados.
Estos son todos los archivos involucrados en los análisis realizados en el presente trabajo.
(a) Por un lado tenemos los patrones resultantes de *findpat*. Donde por ejemplo *patrones_Homo3Homo4_MayoresA100.txt* contiene los patrones entre el cromosoma 3 y el cromosoma 4 del humano, de longitud >100 caracteres. De manera similar para los otros archivos.
(b) Por otro lado tenemos los archivos con las *repeticiones*, las cuales son conocidas actualmente, y contra las cuales comparamos nuestros patrones.

En las figuras 5 y 6 pueden verse las familias de *repeticiones* que fueron analizadas, así como la relación de estas familias con su tipo de repetición.

Esta base de datos esta formada por todas las repeticiones registradas en *Ensembl*[11], ampliada por secuencias que fueron detectadas por programas como *DUST*, *TRF* y *RepeatMasker*. La misma fue recopilada por Javier Herrero en `EMBL-EBI UK`[12] y puede ser descargada de `ftp://ftp.ebi.ac.uk/pub/databases/ensembl/jherrero/.repeats/`.

Es importante destacar que dicha base de datos está ordenada (ascendentemente) por el campo *START*, es decir, por la posición donde comienza la *repetición*. Esta es la característica fundamental que nos permitió utilizar un algoritmo basado en búsqueda binaria para realizar la identificación de patrones. De no contar con esta propiedad, la identificación hubiera sido muy costosa, y casi impracticable en cuanto a tiempo de procesamiento.

[11] Ensembl es un proyecto en común entre el EMBL-EBI y el *Wellcome Trust Sanger Institute*, para desarrollar un software el cual produzca y mantenga automáticamente anotaciones sobre genomas de eucariotas. http://www.ensembl.org.
[12] European Bioinformatics Institute.

Otra cuestión a mencionar es que las *repeticiones* que en su columna *STRAND* tienen el valor -1, fueron descartadas de ante mano. El valor -1 en este campo, significa que la secuencia aparece en orden inverso, por lo tanto nosotros sólo estamos teniendo en cuenta las secuencias que aparecen en el orden normal dentro del cromosoma (de izquierda a derecha).

La estructura de los archivos de *repeticiones*, posee las siguientes columnas:

- *CHR*: cromosoma.
- *START*: posición de inicio.
- *END*: posición de fin.
- *STRAND*: determina si aparece en orden inverso o no.
- *CLASS*: familia a la que pertenece la repetición.
- *NAME*: nombre específico dentro de la familia.

En las figuras 7 y 8 pueden verse los encabezados de estos archivos.

Clase	Cantidad
SINE/Alu	580730
LINE/L1	446303
SINE/MIR	298070
LINE/L2	202601
LTR/MaLR	163789
DNA/MER1_type	106677
LTR/ERV1	86497
LTR/ERVL	64265
DNA/MER2_type	40890
LINE/CR1	30280
DNA/Tip100	13389
DNA/AcHobo	9439
LINE/RTE	8112
DNA/Mariner	7989
DNA	6739
LTR/ERVK	5088
DNA/Tc2	3890
DNA/hAT	1716
DNA/PiggyBac	1079
DNA/MuDR	1015
DNA/MER1_type?	943
LTR/ERV	295
DNA/Merlin	17

(a) Clases de *human_repeats*

Clase	Cantidad
dust	2935091
trf	1392275
Other	2426
snRNA	2139
Satellite	1779
Satellite/centr	1113
tRNA	882
rRNA	882
scRNA	667
srpRNA	477
RNA	350
Satellite/telo	195
Satellite/acro	15

(b) Clases de *human_other_repeats*

Figura 5: Clases de *repeticiones*.
Clases a las que pertenecen las *repeticiones* utilizadas en los experimentos. Para cada clase se indica la cantidad de elementos que existen en cada una de ellas.

Clase	Tipo de repetición
dust	Dust
LTR/ERV	LTRs
LTR/ERV1	LTRs
LTR/ERVK	LTRs
LTR/ERVL	LTRs
LTR/MaLR	LTRs
Other	Other repeats
RNA	RNA repeats
rRNA	RNA repeats
scRNA	RNA repeats
snRNA	RNA repeats
srpRNA	RNA repeats
tRNA	RNA repeats
Satellite	Satellite repeats
Satellite/acro	Satellite repeats
Satellite/centr	Satellite repeats
Satellite/telo	Satellite repeats
trf	Tandem repeats
LINE/CR1	Type I Transposons/LINE
LINE/L1	Type I Transposons/LINE
LINE/L2	Type I Transposons/LINE
LINE/RTE	Type I Transposons/LINE
SINE/Alu	Type I Transposons/SINE
SINE/MIR	Type I Transposons/SINE
DNA	Type II Transposons
DNA/AcHobo	Type II Transposons
DNA/hAT	Type II Transposons
DNA/Mariner	Type II Transposons
DNA/MER1_type	Type II Transposons
DNA/MER1_type?	Type II Transposons
DNA/MER2_type	Type II Transposons
DNA/Merlin	Type II Transposons
DNA/MuDR	Type II Transposons
DNA/PiggyBac	Type II Transposons
DNA/Tc2	Type II Transposons
DNA/Tip100	Type II Transposons

Figura 6: Relación entre las familias de repeticiones y los tipos de las mismas.
Aquí puede verse que tipo de repetición se corresponde con cada una de las familias de repeticiones analizadas. Se puede ver que la mayoría de las mismas son transposones.
Notar que los tipos de repeticiones aquí enumerados tienen directa relación con lo visto en la sección 2.1.1.

4.3 Algoritmo

Aquí se detalla el algoritmo utilizado en los experimentos o pruebas realizadas en la sección 4.4, para identificar biológicamente los patrones.

4.3.1 Descripción

Este algoritmo (*PatronesVsTransposones*[13]) tiene como objetivo básico, determinar los patrones que se solapan con alguna *repetición*, de la lista de *repeticiones*(ver Figura 4b) dada. Tanto los patrones como las *repeticiones* están delimitados por una posición de comienzo y una de final[14], siendo cada una de ellas relativa al cromosoma al que pertenece.

Dados un patrón $P : [start, end]$ y un transposón $T : [start, end]$, decimos que P y T se solapan cuando:

$$CantidadDeLetrasSolapadas(P, T) > 0$$

donde:

$$CantidadDeLetrasSolapadas(P, T) = \min(P.end, T.end) - \max(P.start, T.start) + 1$$

Es decir que el algoritmo básicamente busca encajes de intervalos. Para lo cual usa una adaptación de la búsqueda binaria.

Resaltamos que es un problema de encaje de intervalos, ya que cada aparición de un patrón y cada *repetición* pueden ser expresadas como intervalos dentro de un cromosoma particular, por lo tanto ver si un patrón y una *repetición* coinciden en varios de sus caracteres, se reduce a ver si sus intervalos se superponen (ver ejemplo en la sección 4.3.4).

4.3.2 Entrada

Patrones: Una de las entradas del algoritmo es un archivo con el formato de salida de la herramienta *findpat*. Los archivos utilizados pueden verse en la Figura 4a. Parte del contenido de uno de estos archivos puede verse en la Figura 2.

Repeticiones: Por otro lado el algoritmo recibe como entrada un archivo con las repeticiones a comparar contra los patrones. En este caso contamos con dos archivos de *repeticiones*, cuya estructura es igual pero difieren en su contenido. En las figuras 7 y 8 pueden verse los encabezados de estos archivos.

[13]El nombre hace alusión a transposones, ya que cuando se creo el mismo, se pensaba que la base de datos que ahora llamamos *repeticiones* eran todos transposones. Ahora sabemos que en dicha base no necesariamente todas las repeticiones son transposones.

[14]En el caso de los patrones, la posición de final se obtiene a partir de la posición de comienzo y la longitud del mismo.

El algoritmo utiliza uno de ellos por vez. En la corrida del mismo, debe elegirse contra cual de ellos se quiere comparar los patrones.

CHR	START	END	STRAND	CLASS	NAME
1	1367	1538	-1	LINE/L1	L1MC
1	1541	1643	-1	DNA/MER1_type	MER5B
1	5128	5218	-1	SINE/MIR	MIR
1	8770	8912	1	LINE/L2	L2
1	9811	10590	1	LINE/CR1	L3
1	11812	12013	1	LTR/MaLR	MLT1K
1	12983	13234	-1	SINE/MIR	MIR
1	13688	13869	1	LINE/L2	L2
1	13951	14113	1	SINE/MIR	MIR
1	14127	14311	1	LINE/L2	L2
1	15755	15835	1	SINE/MIR	MIRm
1	16215	16278	1	SINE/MIR	MIRm
1	16654	16925	1	SINE/Alu	AluSp
1	17132	17381	1	DNA/MER1_type	MER33

⋮

Figura 7: Encabezado del archivo *human_repeats.txt*
Aquí pueden observarse las primeras líneas de este archivo. La primera línea contiene los nombres de las columnas. Cada uno de los registros indica una repetición en el cromosoma (CHR) indicado, la cual se encuentra dentro de ese cromosoma en el intervalo [START, END], si la repetición se encuentra al derecho o al revés dentro del cromosoma, la familia y nombre a la que pertenece la *repetición*. Este archivo contiene mayormente transposones.

CHR	START	END	STRAND	CLASS	NAME
1	1	468	0	dust	dust
1	1	468	0	trf	trf
1	464	1310	-1	Satellite/telo	TAR1
1	621	860	0	trf	trf
1	686	728	0	trf	trf
1	872	889	0	dust	dust
1	1030	1311	0	trf	trf
1	6537	6607	0	dust	dust
1	9169	9306	0	trf	trf
1	18275	18283	0	dust	dust
1	18452	18466	0	dust	dust
1	20051	20126	0	dust	dust
1	20718	20829	0	dust	dust
1	20718	20826	0	trf	trf

⋮

Figura 8: Encabezado del archivo *human_other_repeats.txt*
Extracto del archivo *human_other_repeats.txt*. Este posee la misma estructura que el visto en la Figura 7. Difiere en las *repeticiones* que este contiene. Aquí la mayoría de las repeticiones, fueron encontradas por programas como TRF y DUST.

4.3.3 Salida

La salida del algoritmo es un archivo de texto que contiene las *repeticiones* que se solapan con patrones, con la respectiva información del patrón, las *repeticiones* afectadas y la cantidad de caracteres en la que se solapan (ver ejemplo en Figura 9).

La estructura del archivo de salida es la siguiente:

- LineaPatron: el número de línea que tiene el patrón en el archivo de patrones.
- Chr: cromosoma afectado.
- Apariciones: cantidad de apariciones del patrón.
- Longitud: longitud del patrón.
- StartPatron: posición de inicio del patrón.
- EndPatron: posición de fin del patrón.
- IdTransp: numero de linea del transposón en el archivo de transposones.
- StartTransp: posición de inicio del transposón.
- EndTransp: posición de fin del transposón.
- Longitud: longitud del transposón.
- Superposicion: cantidad de letras superpuestas entre el patrón y el transposón.
- OffSetPatron: Una tupla $[a, b]$ donde a es el offset del inicio del patrón respecto al inicio del transposón. Y de manera similar b, pero para el final del patrón y final del transposón.
- Clase: clase del transposón.
- Nombre: nombre del transposón.

Además el algoritmo brinda las siguientes estadísticas sobre lo procesado:

- *#solapados*: cantidad de patrones que se solapan con *repeticiones*.
- *#patrones*: cantidad total de patrones analizados.
- *porcentaje*: (*#solapados*/*#patrones*), es decir el porcentaje de patrones que se solapan con *repeticiones*.

Estos datos se ven reflejados en las tablas y gráficos de resultados de las corridas del algoritmo.

LineaPatron	Chr	Apariciones	Longitud	StartPatron	EndPatron	IdTransp	StartTransp	EndTransp	Longitud	Superposición	OffSetPatron	Clase	Nombre
3	1	55	100	25707936	25708035	45965	25705664	25711797	100	100	[2272,-3762]	LINE/L1	L1PA2
5	1	227	100	102749469	102749568	174023	102747792	102750756	100	100	[1677,-1188]	LINE/L1	L1M2
7	1	116	100	88450631	88450730	153688	88450619	88451762	100	100	[12,-1032]	LINE/L1	L1PA3
9	2	92	100	130235141	130235240	529704	130235136	130235455	100	100	[5,-215]	SINE/Alu	AluYb8
11	2	11	100	46304611	46304710	417668	46304578	46305045	100	100	[33,-335]	LINE/L1	L1PA5
13	2	19	100	9027364	9027463	361243	9025595	9031111	100	100	[1769,-3648]	LINE/L1	L1PA5
17	2	61	100	201877960	201878059	619302	201872914	201879066	100	100	[5046,-1007]	LINE/L1	L1PA5
21	1	33	100	169313929	169314028	239761	169313897	169315393	100	100	[32,-1365]	LINE/L1	L1PA3
25	1	145	100	92494932	92495031	159525	92491362	92497051	100	100	[3570,-2020]	LINE/L1	L1PA7

⋮

Figura 9: Encabezado del archivo *pvst_repeats_Homo1Homo2.txt*.
Es el encabezado de la salida de una corrida del algoritmo *PatronesVsTransposones* en la que se comparan los patrones entre los cromosomas humanos 1 y 2, contra la base de datos de repeticiones *human_repeats*.

4.3.4 Pseudocódigo

Para cada patrón disponemos de su longitud y la lista de las posiciones de cada una de sus apariciones.

El algoritmo recorre todos los patrones. Por cada uno de ellos recorre cada una de sus apariciones hasta que encuentra una que sea un éxito (según el criterio utilizado), es decir que esa aparición del patrón se solape con alguna *repetición* del archivo de *repeticiones*, y además que el cromosoma de esa aparición del patrón se corresponda con el cromosoma al cual pertenece la *repetición* (entonces, si por ejemplo un patrón pertenece al cromosoma 1, nos fijaremos si este se solapa con *repeticiones* que pertenecen a dicho cromosoma). Para saber si un patrón se solapó con una *repetición*, nos basamos fuertemente en la propiedad que posee la lista de *repeticiones*, que es que la misma este ordenada por el campo *START*. Gracias a esta propiedad podemos usar una variación de la búsqueda binaria para saber si un patrón se solapa con una *repetición*. El pseudocódigo puede verse en las figuras 10 y 11.

Ejemplo:

Patrón de Homo3–Homo4

```
GGGGTTTCACCGTGTTAGCCAGGATGGTCTCGATCTCCTGACCTTGTGATCTGCCCGCCTCGGCCTCCCAAAGTGCTGGGATTACAGGCGTGAGCCACCG #3 (100)
>1000 <4000 >6000
.
.
.
```

Repeticiones

```
CHR  START  END    STRAND  CLASS     NAME
3    3819   4126   1       SINE/Alu  AluY
4    4127   4766   1       LTR/ERV1  MER31-int
.
.
.
```

En el ejemplo podemos ver que el patrón tiene 3 apariciones (una en el cromosoma 3, y dos en el cromosoma 4).

En este ejemplo tenemos un éxito para este patrón, ya que la segunda aparición (<4000) pertenece al intervalo [4000, 4099][15] (longitud 100), y este se solapa con el intervalo del primer transposón [3819, 4126], y tanto el patrón como el transposón corresponden al cromosoma 3.

En este caso:

$$CantidadDeLetrasSolapadas(P,T) = \min(P.end, T.end) - \max(P.start, T.start) + 1 =$$

$$\min(4099, 4126) - \max(4000, 3819) + 1 = 100$$

Este resultado es coherente con el hecho de que el patrón esta totalmente incluido en el transposón. Este no siempre es el caso, pueden darse otros escenarios de solapamiento, como veremos en la sección 4.6.

[15] Corrigiendo los índices según se numeran las *repeticiones*, el intervalo sería [4001, 4100].

```
/// Encuentra todos los transposones de los cromosomas chrA y chrB, que se solapan con los patrones entre chrA y chrB
/// y los reporta en un archivo
void PatronesVsTransposones(chrA, chrB, archivoPatronesAB.txt, archivoTransposones.txt)
{
    List transposonesA, transposonesB;
    List patrones;

    //carga en transposonesA y en transposonesB los transposones correspondientes
    LeerTransposones(chrA, chrB, transposonesA, transposonesB, "archivoTransposones.txt");
    //carga en patrones los patrones entre A y B
    LeerPatrones(patrones, "archivoPatronesAB.txt");

    patronesSolapados = 0;
    foreach (patron in patrones)
    {
        hits = 0;
        foreach (posicion in patron.Posiciones && hits<=0)
        {
            // determina los limites del patron en cuestion
            startPatron = posición;
            endPatron = startPatron + patron.longitud -1;

            // corrección, ya que los transposones comienzan a contar desde 1
            startPatron++;
            endPatron++;

            // determina de que cromosoma es la posición  (chrA o chrB)
            chr = DeterminarCromosoma(posición, chrA, chrB);
            if(chr == chrA)
                hits = Solapados(chrA, chrB, startPatron, endPatron, patron, transposonesA);
            if(chr == chrB)
                hits = Solapados(chrA, chrB, startPatron, endPatron, patron, transposonesB);
        }
        if(hits > 0) patronesSolapados++;

        EscribirEnArchivo(patronesSolapados, hits, transposones.cantidad, transposonesA.cantidad, transposonesB.cantidad );
    }
}
```

Figura 10: Algoritmo principal

```
/// Devuelve la cantidad de transposones que se solapan con un patron dado.
/// Y escribe cada uno de ellos en un archivo.
int Solapados(int startPatron, int endPatron, patron, List transposones)
{
    int desde = 0;
    int hasta = transposones.Cantidad - 1;
    return SolapadosDesdeHasta(startPatron, endPatron, patron, transposones, desde, hasta);
}

/// Devuelve la cantidad de transposones que se solapan con un patron dado en el rango estipulado.
/// Y escribe cada uno de ellos en un archivo.
int SolapadosDesdeHasta(int startPatron, int endPatron, patron, List transposones, int desde, int hasta)
{
    // no hay solapados
    hits = 0;

    while (desde <= hasta)
    {
        medio = (desde + hasta) / 2;

        transpMedio = transposones[medio];

        superposicion = CantidadDeLetrasSolapadas(startPatron, endPatron, transpMedio.Start, transpMedio.End);

        if (superposicion >= patron.longitud)
        {
            hits++;
            // busca solapados en la mitad inferior
            hits+= SolapadosDesdeHasta(startPatron, endPatron, patron, transposones, desde, medio - 1);
            // escribe la información encontrada en un archivo.
            EscribirResultado(startPatron, endPatron, patron, transpMedio, superposición);
            // busca solapados en la mitad superior
            hits+= SolapadosDesdeHasta(startPatron, endPatron, patron, transposones, medio + 1, hasta);

            // devuelve la cantidad de transposones que se solapan con el patron y termina.
            return hits;
        }

        // buscar en la mitad inferior
        if (startPatron <= transpMedio.Start) { hasta = medio - 1; }

        // buscar en la mitad superior
        if (startPatron > transpMedio.Start) { desde = medio + 1; }
    }

    return hits;
}
```

Figura 11: Algoritmo: subfunciones

4.4 Comparaciones realizadas

Para cada uno de los patrones a identificar, se debe verificar si éste se encuentra o no en la base de datos de repeticiones utilizada. Responder a ésta pregunta, requiere definir de ante-mano cuál será el criterio para determinar si se encuentra o no. En el presente trabajo, se determinaron dos criterios:

- **Solapamiento>=20**. El patrón se solapa con una *repetición* en al menos 20 caracteres.

- **Solapamiento Exacto**. La superposición del patrón con la *repetición* es de la longitud exacta del patrón.

Cada uno de estos criterios tiene asociado su experimento, y en cada uno de ellos se compararon los patrones de *findpat* contra las dos bases de *repeticiones*. A continuación veremos los resultados obtenidos.

4.4.1 Resultados obtenidos

Solapamiento>=20. En este experimento, la cantidad de patrones solapados encontrada con *human_repeats* fue de un poco más que la mitad de la cantidad total de patrones. En cada una de las parejas la cantidad de patrones solapados es alrededor del 52%. Es decir que en promedio, el **52%** de los patrones tiene una identificación positiva contra la base de *human_repeats*. En el caso de *human_other_repeats* la cantidad de patrones solapados se acerca a un promedio de **14%**.

Como corolario podemos mencionar que la cantidad de patrones que no fueron clasificados (que no pertenecen ni a *human_repeats* ni a *human_other_repeats*), es aproximadamente del **42%** sobre el total de patrones [16].

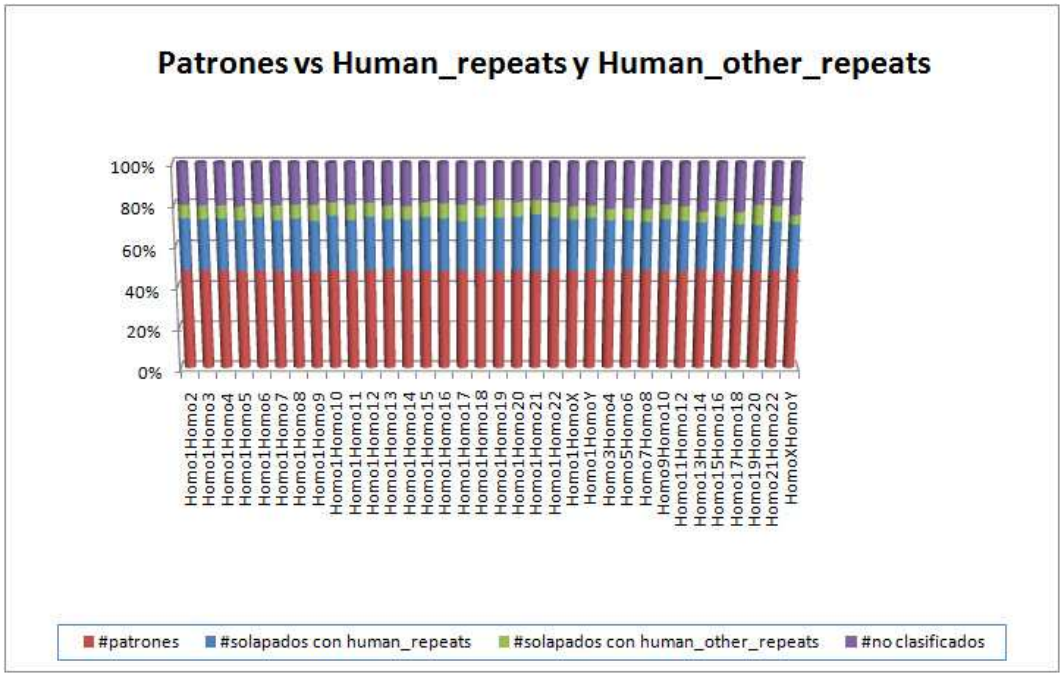

Figura 12: Patrones vs. *human_repeats* y *human_other_repeats* (solapamiento >= 20)
Este gráfico resume la información resultante de las comparaciones hechas bajo este criterio. Aproximadamente 58% de los patrones fueron biológicamente identificados, y un 42% no pudo ser clasificado. Los datos que nutren a este gráfico pueden verse en el apartado A.

[16] La razón por la cual los porcentajes de clasificados (*solapados con human_repeats*, solapados con *human_other_repeats*) y de no clasificados no suma 100% es que *solapados con human_repeats* $\cap$ *solapados con human_other_repeats* $\neq \emptyset$.

Solapamiento Exacto. En este caso, los resultados fueron muy similares al anterior. La principal diferencia fue que en este caso, la cantidad que pudo ser clasificada fue un poco menor.

La cantidad de *solapados con human_repeats* es aproximadamente de **51%**. La cantidad de *solapados con human_other_repeats* es cercana a **6%**. Y los casos no clasificados ascienden a **46%**[16].

Era esperable que la cantidad de patrones clasificados bajo el criterio exacto sea menor que bajo cualquier otro criterio, pues es un criterio más restrictivo. En *solapados con human_other_repeats* la diferencia es más notoria, ya que al haber muchas *repeticiones* de longitud pequeña (menor a 100 caracteres), las mismas quedan fuera de la clasificación debido a que la longitud mínima de los patrones analizados es mayor o igual a 100 caracteres. Naturalmente el porcentaje de patrones no clasificados es mayor respecto del primer criterio.

Es curioso que en ambos casos, el porcentaje de *solapados con human_repeats* es prácticamente el mismo.

Como hecho destacable de ambos experimentos, y apoyándonos en el resultado más fuerte (solapamiento exacto), podemos observar que aproximadamente el **54%** de los patrones fueron **biológicamente identificados**; mientras que el **46%** no pudo ser clasificado dentro del conjunto de repeticiones utilizado. Esta afirmación es verdadera para los patrones de $ParejasA$ y $ParejasB$. Queda pendiente analizar que sucede con el resto de las parejas de cromosomas de humanos.

Ambos experimentos mencionados son fundamentales a los efectos de el objetivo de este trabajo. Las conclusiones e implicaciones de los mismos serán abordadas en la sección 5.

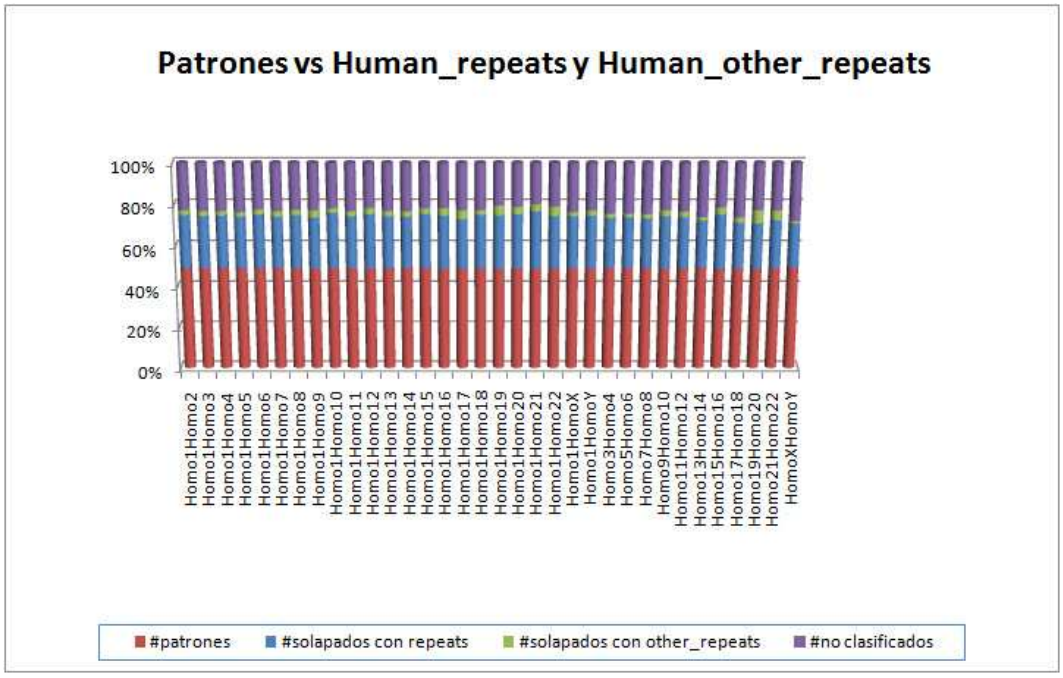

Figura 13: Patrones vs. *human_repeats* y *human_other_repeats* (solapamiento exacto)
Este gráfico resume la información resultante de las comparaciones hechas bajo este criterio.
Aproximadamente 54% de los patrones fueron biológicamente identificados, y un 46% no pudo ser clasificado.
Detalle de los datos para este gráfico en el apartado A.

4.4.2 Tiempos de ejecución

El algoritmo *PatronesVsTransposones* nos permitió tener una manera bastante rápida de analizar e identificar los patrones de *findpat*. Dicha identificación hecha con la versión web de *BLAST*, mencionada anteriormente, hubiera llevado varios meses.

Espacio de búsqueda:
Cantidad de **patrones** analizados: 4.542.157
Cantidad de ***repeticiones*** contra las cuales comparar los patrones: 6.418.104

Tiempos de ejecución[17]:
Realizar los experimentos involucrados por el criterio *Solapamiento>=20* tomo: 85 minutos aprox.
Realizar los experimentos involucrados por el criterio *SolapamientoExacto* tomo: 51 minutos aprox.

Por lo tanto realizar todos los experimentos llevo un tiempo de un poco mas que **2 horas**. Lo cual es significativamente menor que los meses que hubieran llevado, de hacerse por el camino manual.

[17]Basados en una máquina con procesador *Intel Core Duo T2250* 1,73GHz, y 2GB de RAM.

4.5 El patrón más "famoso"

De las corridas de $findpat$ se obtuvo el patrón con la mayor cantidad de apariciones (dentro del genoma humano). Este patrón[18] tiene una longitud de 100 caracteres.

Dicho patrón, esta presente en cada uno de los archivos de patrones de $ParejasA$ y $ParejasB$, salvo en *Homo21–Homo22*.

Cada archivo de patrones contiene una única entrada por patrón, por lo tanto este patrón famoso, aparece exactamente una vez en cada una de las parejas mencionadas, cada una con su respectiva lista de apariciones.

Lo que aquí analizamos, es ver si estos pueden ser identificados contra la base de *repeticiones*.

Como ya disponemos de la identificación de todos los patrones de todas las parejas contra las *repeticiones*, sólo nos resta buscar dentro de esta información si el patrón famoso de cada una de las parejas pertenece al conjunto de los patrones que pudieron ser clasificados, es decir, si este pertenece a los patrones que se solaparon con *repeticiones*.

Como resultado vimos que de las parejas de patrones que contienen al patrón más famoso, solo las parejas de $ParejasA$ fueron las que se solapan con *repeticiones*.

Resumiendo entonces, hallamos que el patrón más famoso que está en cada una de las de $ParejasA$ pudo ser identificado contra la base de datos de *repeticiones*.

Esto puede deberse a que los éxitos de esta identificación se producen en el cromosoma 1 de las *repeticiones*, por lo tanto las parejas que tienen como componente al cromosoma 1 (es el caso de $ParejasA$) son las que solapan con *repeticiones*.

[18] TTTTTATGGCTGCATAGTATTCCATGGTGTATATGTGCCACATTTTCTTAATCCAGTCTATCATTGTTGGACATTTGGGTTGGTTCCAAGTCTTTGCTAT

4.6 Posición de los patrones respecto a las *repeticiones*

Para las *repeticiones* que se solapan con cada uno de los patrones, se analizó que posición ocupa el patrón respecto de la *repetición*. Nos resulto interesante saber, si los patrones están al comienzo, al final, en el medio, u en otra posición respecto a las *repeticiones* con las que se solapan. En la Figura 14 pueden verse las posiciones que puede tomar un patrón respecto de la *repetición*.

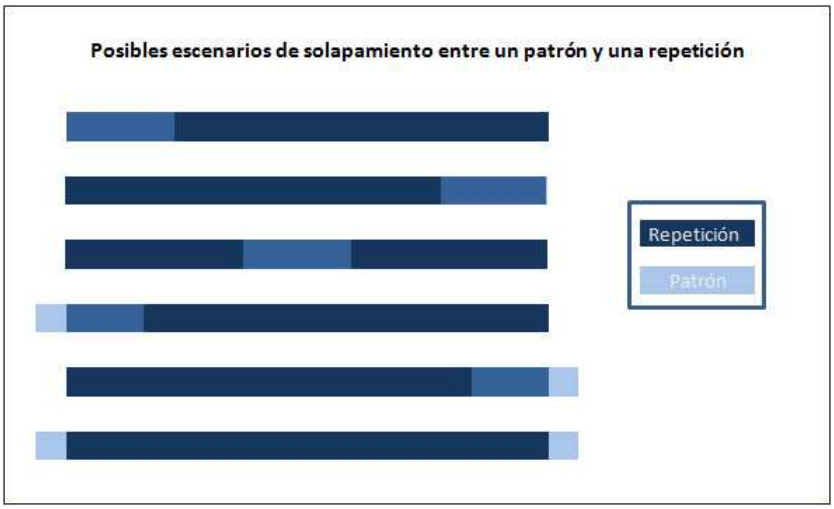

Figura 14: Los distintos casos de solapamiento entre un patrón y una *repetición* que pueden darse.

El patrón puede estar exactamente al principio de una repetición, exactamente al final, en el medio de la misma, solaparse al principio, solaparse al final, o la repetición puede estar incluida en el patrón.

5 Conclusiones y Preguntas Abiertas

Patrones vs. Repeticiones. Aproximadamente **54%** de los patrones analizados pudo ser identificado biológicamente contra la base de datos de *repeticiones* utilizada, mientras que el **46%** restante no pudo ser identificado contra las mismas. Ver Figura 15.

Dichos resultados cumplen con el objetivo fundamental de esta tesis. Los mismos implican que *findpat* es una herramienta útil para detectar repeticiones en secuencias de ADN, y además estas son de interés para la biología.

Los patrones que no pudieron ser identificados contra las repeticiones analizadas, pueden ser un buen punto de partida para futuros estudios: como por ejemplo determinar si son o no son "algo" biológicamente identificado.

Mencionamos al comienzo de este trabajo, que hallar un patrón con identificación biológica positiva parecía, a priori, una utopía. Y que haber encontrado que un patrón se correspondía (en parte) con el transposón M80343 fue un hecho bastante afortunado. Sin embargo, los porcentajes arrojados reflejan que identificar biológicamente un patrón es más probable que no hacerlo.

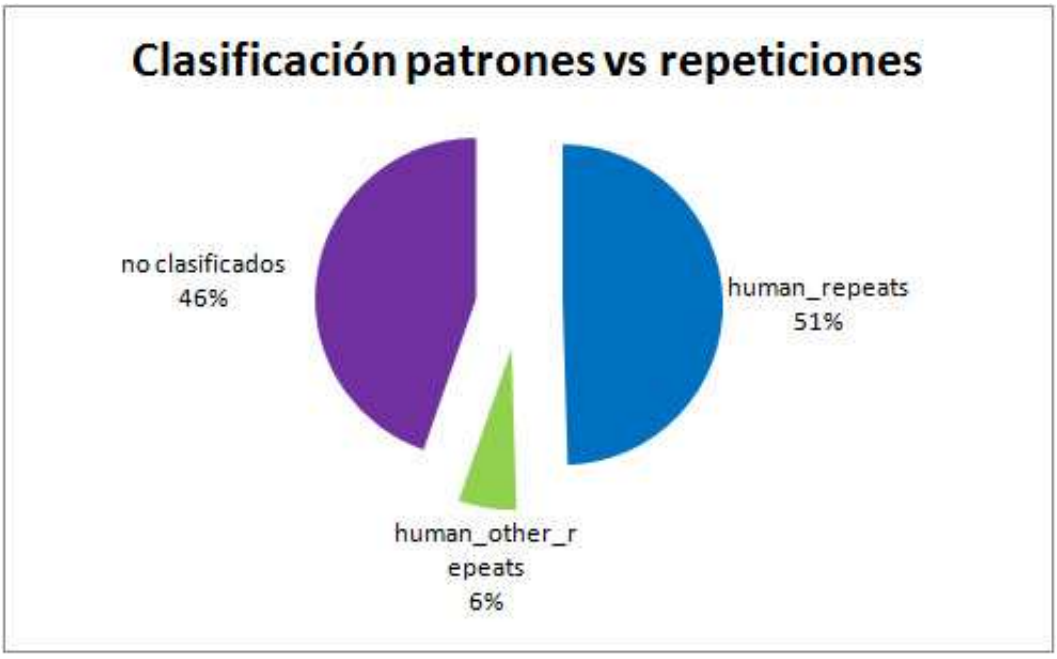

Figura 15: Patrones vs. *human_repeats* y *human_other_repeats* (solapamiento exacto)
Notar que el total no suma el 100%, debido a que entre el 51% y el 6% la intersección es no vacía, como lo mencionamos anteriormente en un pie de página.

M80343. El patrón que fue identificado contra el transposón que motivó este trabajo, pudo también ser identificado mediante el algoritmo presentado en este trabajo. El mismo es patrón en las parejas:

$$Homo1 - Homo2$$
$$Homo1 - Homo3$$
$$Homo1 - Homo4$$
$$Homo1 - Homo11$$
$$Homo1 - Homo17$$

En cada una de estas parejas, fue identificado con 775 caracteres (su longitud total es de 781 caracteres). El mismo se encuentra catalogado como clase:*LINE/L1* y nombre:*L1HS*. Esta identificación guarda coherencia con lo mencionado en la sección 3.2, ya que el mismo había sido encontrado mediante *BLAST* en las bases de datos de `NCBI`.

Patrón más famoso. Se vio que el patrón más famoso pudo ser identificado contra la base de *repeticiones*, para las parejas de $ParejasA$. Es decir que este patrón que se haya disperso en la mayoría de los cromosomas humanos, es una secuencia biológicamente identificada. ¿Pero que sucede con las parejas de $ParejasB$? ¿Quiere decir que hay posibles *repeticiones* sin identificar en estos casos?.

Clase de *repetición* más famosa. Las clases más "colisionadas" por los patrones fueron *LINE/L1*[19] y *trf*[20]. Este resultado era de esperar, ya que *LINE/L1* y *trf* son dos de las clases que mayor cantidad de elementos tienen. Como corolario podemos mencionar que según nuestros experimentos los elementos encontrados por *findpat* también son encontrados por el programa *TRF*. En la Figura 16 pueden verse de manera proporcional como fueron impactadas por patrones las diferentes clases.

clase	porcentaje
LINE/L1	0,916746351
SINE/Alu	0,073266189
LTR/ERV1	0,004802743
LTR/MaLR	0,002575982
LTR/ERVK	0,001588295
LINE/L2	0,000384526
SINE/MIR	0,000182905
LTR/ERVL	0,000178226
DNA/MER2_type	0,000128459
DNA/MER1_type	7,27366E-05
LINE/CR1	3,14767E-05
DNA/Tip100	1,10594E-05
LINE/RTE	8,08185E-06
DNA	5,95504E-06
DNA/MuDR	5,95504E-06
DNA/AcHobo	4,2536E-06
DNA/Mariner	3,40288E-06
DNA/Tc2	2,97752E-06
DNA/PiggyBac	4,2536E-07
DNA/hAT	0
DNA/MER1 type?	0
DNA/Merlin	0
LTR/ERV	0

(a) Clases de *human_repeats*

clase	porcentaje
trf	0,612928527
dust	0,339039137
Other	0,047934971
Satellite	6,35581E-05
snRNA	1,08184E-05
Satellite/acro	9,4661E-06
Satellite/centr	5,4092E-06
rRNA	2,7046E-06
scRNA	2,7046E-06
Satellite/telo	1,3523E-06
tRNA	1,3523E-06
RNA	0
srpRNA	0

(b) Clases de *human_other_repeats*

Figura 16: Clases más afectadas de *repeticiones*
Se muestra para cada clase, como fueron afectadas, según los experimentos realizados. LINE/L1 y trf son las mas "impactadas" por patrones.

[19]Long interspersed nuclear elements, esta familia representa el 20% del genoma.
[20]Repeticiones provenientes del programa *Tandem Repeats Finder*.

¿Que sucede con las parejas no analizadas? Hemos visto que aquí solo analizamos una parte ($ParejasA$ y $ParejasB$) de todas las parejas posibles (todos los cromosomas vs. todos los cromosomas). Tampoco vimos que sucede con los patrones intracromosoma, de los 24 disponibles.

Queda pendiente ver que sucede con el resto de las parejas de cromosomas de humanos. ¿se mantiene el 51% de clasificados?

Por ejemplo en una corrida con un muestreo de patrones intracromosoma, se observa que el porcentaje de clasificados es menor al visto en las corridas anteriores. (ver Tabla 3).

	human repeats			**human other repeats**			**no clasificados**		
muestra	#solapados	#patrones	porcentaje	#solapados	#patrones	porcentaje	#no clasificados	#patrones	porcentaje
Homo1	49364	103964	0,4748182	15798	103964	0,1519564	44837	103964	0,431274287
Homo2	56811	115903	0,4901599	14769	115903	0,1274255	50056	115903	0,431878381
HomoX	59099	126802	0,4660731	13898	126802	0,1096039	59800	126802	0,471601394
		promedio	**0,477017067**		promedio	**0,129661933**		**promedio**	**0,444918021**

Tabla 3: Resultado de la corrida de $PatronesVsTransposones$ sobre patrones intracromosoma (solapamiento >=20)
Resultados del algoritmo $PatronesVsTransposones$ para una corrida de los patrones intracromosoma, para los cromosomas humanos 1, 2, y X. Puede notarse que el porcentaje de patrones identificados es menor que en el caso de patrones de parejas (cromosoma-cromosoma).

Posición de los patrones respecto de las *repeticiones*. Además de determinar si el patrón y la repetición cumplen con alguno de los criterios de solapamiento, el algoritmo que utilizamos brinda información extra que permite inferir otro tipo de conclusiones. Por ejemplo, ¿en que región del transposón se produce el solapamiento?. Nuestras conclusiones al respecto son las siguientes: al 12% les sucede que, o bien el patrón comienza exactamente donde comienza el transposón, o bien el patrón termina donde lo hace el transposón. Es decir que el patrón está exactamente al comienzo del transposón o al final del mismo, en aproximadamente el 12% de los casos en promedio. Ver Figura 17.

Figura 17: Posición de los patrones respecto a las *repeticiones*

Conclusiones generales. Observamos que en las muestras analizadas, aproximadamente la mitad de los patrones fueron identificados contra la base de datos de repeticiones utilizada. Lo cual nos dice por un lado, que los patrones utilizados fueron biológicamente identificados, y por otro lado que gran parte de ellos (casi la mitad) no pudieron ser clasificados. Estos últimos podrían ser investigados como nuevas secuencias de interés. Y es posible que éstas secuencias sean nuevos transposones o genes no identificados, o secuencias de ADN con una potencial función biológica, que se desconoce actualmente, etc.

findpat no sólo sirve para encontrar transposones, sino que puede ser usado para comparar los patrones con cualquier otra base de datos[21], como por ejemplo una base de datos de genes, etc.

Es destacable el hecho de haber podido construir un método eficiente que permite identificar biológicamente los patrones. Esto fue posible, gracias a que disponíamos toda la información de manera local, y a que la base de *repeticiones* estaba ordenada por un campo clave para realizar las comparaciones.

[21] Esta debe cumplir con el formato requerido por el algoritmo

A Tablas de resultados

En ésta sección se detallan todos los resultados de los experimentos realizados (Solapamiento>=20 y Solapamiento Exacto). Estas tablas muestran los resultados de las corridas efectuadas para los experimentos mencionados. Se pueden distinguir cuatro columnas principales:

- *muestra*, pareja de cromosomas analizados.

- *human_repeats*, datos sobre las comparaciones realizadas, de patrones contra las base de datos *human_repeats*
- *human_other_repeats*, datos sobre las comparaciones realizadas, de patrones contra las base de datos *human_other_repeats*
- *#no clasificados*, los patrones que no pudieron ser identificados en ninguno de los dos casos anteriores.

Dentro de estas, encontramos las columnas:

#solapados: cantidad de patrones que contiene la muestra (*human_repeats* o *human_other_repeats*).

#patrones: cantidad de patrones analizados pertenecientes a la muestra correspondiente.

porcentaje: *#solapados/#patrones*. Osea, la proporción de patrones identificados para la muestra en cuestión.

#no clasificados: cantidad de patrones que no pudieron ser identificados contra ninguna de las dos bases de datos de *repeticiones*.

Las tablas que aquí se exponen, contienen la información en la cual están basados los gráficos de las figuras 12 y 13.

	human repeats			**human other repeats**			**no clasificados**		
muestra	#solapados	#patrones	porcentaje	#solapados	#patrones	porcentaje	#no clasificados	#patrones	porcentaje
Homo1Homo2	129745	247186	0,5248882	32753	247186	0,1325035	101363	247186	0,410067722
Homo1Homo3	122098	237356	0,5144087	31520	237356	0,1327963	98825	237356	0,416357707
Homo1Homo4	127780	245669	0,5201308	31884	245669	0,1297844	102201	245669	0,416010974
Homo1Homo5	115324	226995	0,5080464	31341	226995	0,1380691	96719	226995	0,426084275
Homo1Homo6	107999	202484	0,5333706	26165	202484	0,1292201	81713	202484	0,403552873
Homo1Homo7	93256	182296	0,5115636	26603	182296	0,145933	76065	182296	0,417260938
Homo1Homo8	99185	187052	0,5302536	25970	187052	0,1388384	76746	187052	0,410292325
Homo1Homo9	79321	153544	0,5166011	24814	153544	0,1616084	63838	153544	0,41576356
Homo1Homo10	85486	156809	0,5451601	20227	156809	0,1289913	60414	156809	0,385271254
Homo1Homo11	92931	179373	0,518088	27615	179373	0,1539529	73848	179373	0,411700758
Homo1Homo12	90020	166402	0,5409791	22997	166402	0,1382015	64956	166402	0,390355885
Homo1Homo13	67215	130519	0,5149825	17330	130519	0,1327776	54286	130519	0,415924118
Homo1Homo14	67320	131777	0,5108631	17217	131777	0,1306525	55863	131777	0,423920715
Homo1Homo15	63079	117118	0,5385935	17017	117118	0,1452979	45131	117118	0,385346403
Homo1Homo16	51959	97598	0,5323777	14658	97598	0,1501875	38801	97598	0,397559376
Homo1Homo17	43413	86943	0,4993272	14157	86943	0,1628308	35566	86943	0,409072611
Homo1Homo18	61376	115990	0,5291491	14227	115990	0,1226571	47991	115990	0,413751185
Homo1Homo19	38307	70576	0,5427766	12470	70576	0,176689	25659	70576	0,363565518
Homo1Homo20	48648	90259	0,5389823	13101	90259	0,145149	34423	90259	0,38138025
Homo1Homo21	36575	64009	0,571404	9120	64009	0,14248	23347	64009	0,364745583
Homo1Homo22	26221	49517	0,5295353	7146	49517	0,1443141	19106	49517	0,385847285
Homo1HomoX	133018	259599	0,5123979	34054	259599	0,1311792	110280	259599	0,424809032
Homo1HomoY	36886	70086	0,5262963	8180	70086	0,1167138	29398	70086	0,419456097
Homo3Homo4	126967	256763	0,494491	28989	256763	0,1129018	114901	256763	0,447498277
Homo5Homo6	102838	205599	0,5001873	24163	205599	0,1175249	90930	205599	0,442268688
Homo7Homo8	72141	146660	0,4918928	17976	146660	0,1225692	66612	146660	0,454193372
Homo9Homo10	51393	98839	0,5199668	14260	98839	0,144275	40454	98839	0,409291879
Homo11Homo12	65060	127223	0,5113855	18099	127223	0,142262	54277	127223	0,426628833
Homo13Homo14	29812	63210	0,4716342	6551	63210	0,1036387	30092	63210	0,476063914
Homo15Homo16	18458	33806	0,5459977	5199	33806	0,1537893	12740	33806	0,376856179
Homo17Homo18	12827	27996	0,4581726	3372	27996	0,1204458	13472	27996	0,481211602
Homo19Homo20	6190	13409	0,4616303	2727	13409	0,2033709	5513	13409	0,41114177
Homo21Homo22	1906	3840	0,4963542	595	3840	0,1549479	1625	3840	0,423177083
HomoXHomoY	43012	95655	0,4496576	8612	95655	0,09003188	48610	95655	0,50818044

Tabla 4: Resultado de la corrida Solapamiento>=20
Esta información es la que nutre al gráfico de la Figura 12

	human repeats			**human other repeats**			**no clasificados**		
muestra	#solapados	#patrones	porcentaje	#solapados	#patrones	porcentaje	#no clasificados	#patrones	porcentaje
Homo1Homo2	128063	247186	0,5180836	11759	247186	0,0475714	112021	247186	0,453185051
Homo1Homo3	120955	237356	0,5095932	11859	237356	0,0499629	109422	237356	0,461003724
Homo1Homo4	126078	245669	0,5132027	11168	245669	0,0454595	112770	245669	0,459032275
Homo1Homo5	113991	226995	0,5021741	9857	226995	0,0434238	107208	226995	0,472292341
Homo1Homo6	106391	202484	0,5254292	9876	202484	0,0487742	90439	202484	0,446647636
Homo1Homo7	91690	182296	0,5029732	10128	182296	0,0555579	84304	182296	0,462456664
Homo1Homo8	97714	187052	0,5223895	9619	187052	0,051424	84418	187052	0,451307658
Homo1Homo9	78271	153544	0,5097626	10926	153544	0,0711587	70390	153544	0,458435367
Homo1Homo10	84545	156809	0,5391591	7249	156809	0,0462282	67735	156809	0,431958625
Homo1Homo11	91504	179373	0,5101325	9034	179373	0,0503643	82919	179373	0,462271356
Homo1Homo12	89018	166402	0,5349575	9798	166402	0,058881	72207	166402	0,433931083
Homo1Homo13	66336	130519	0,5082479	6997	130519	0,053609	60181	130519	0,461089956
Homo1Homo14	66533	131777	0,5048909	6724	131777	0,051025	61244	131777	0,464754851
Homo1Homo15	62410	117118	0,5328814	6536	117118	0,0558069	50837	117118	0,434066497
Homo1Homo16	51250	97598	0,5251132	7166	97598	0,0734236	42626	97598	0,436750753
Homo1Homo17	42712	86943	0,4912644	7470	86943	0,0859183	39803	86943	0,457805689
Homo1Homo18	60645	115990	0,5228468	5363	115990	0,0462367	52499	115990	0,452616605
Homo1Homo19	37406	70576	0,5300102	6697	70576	0,0948906	29112	70576	0,412491499
Homo1Homo20	48060	90259	0,5324677	6175	90259	0,0684142	38134	90259	0,422495264
Homo1Homo21	36207	64009	0,5656548	4222	64009	0,0659594	25502	64009	0,398412723
Homo1Homo22	25873	49517	0,5225074	4489	49517	0,0906557	20881	49517	0,42169356
Homo1HomoX	131671	259599	0,5072092	11495	259599	0,0442798	121793	259599	0,469158202
Homo1HomoY	36269	70086	0,5174928	3578	70086	0,05105156	32032	70086	0,457038496
Homo3Homo4	125976	256763	0,4906314	10112	256763	0,03938262	125095	256763	0,487200259
Homo5Homo6	101630	205599	0,4943117	6846	205599	0,03329783	100052	205599	0,486636608
Homo7Homo8	71423	146660	0,4869971	6474	146660	0,04414292	72057	146660	0,49132006
Homo9Homo10	50938	98839	0,5153634	5717	98839	0,05784154	45018	98839	0,455467983
Homo11Homo12	64467	127223	0,5067244	6858	127223	0,05390535	59730	127223	0,46949058
Homo13Homo14	29467	63210	0,4661762	1972	63210	0,0311975	32797	63210	0,518857776
Homo15Homo16	18258	33806	0,5400816	2339	33806	0,069188	14529	33806	0,429775779
Homo17Homo18	12606	27996	0,4502786	1469	27996	0,0524717	14697	27996	0,524967853
Homo19Homo20	6083	13409	0,4536505	1718	13409	0,128122	6121	13409	0,456484451
Homo21Homo22	1863	3840	0,4851562	364	3840	0,09479167	1760	3840	0,458333333
HomoXHomoY	41407	95655	0,4328786	2102	95655	0,02197481	53440	95655	0,558674403

Tabla 5: Resultado de la corrida Solapamiento Exacto
Esta información es la que nutre al gráfico de la Figura 13

Referencias

[1] Altschul S.F.; Gish W.; Miller W.; Myers E.W.; Lipman D.J. *Basic Local Aligment Search Tool*, J. Mol. Biol. 215, 403-410, (1990).

[2] Baxevanis A.D.; Ouellette B.F.F. *Bioinformatics, A practical Guide to the Analysis of Genes and Proteins*, Wiley-Interscience, (2001).

[3] Becher V.; Deymonnaz A.; Heiber P. *Efficient computation of all perfect repeats in genomic sequences of up to half a Gigabyte, with a case study on the Human genome*, Bioinformatics, Bioinformatics 2009; doi:10.1093/bioinformatics/btp321. En prensa en (2008).

[4] Benson G. *Tandem repeats finder: a program to analyze DNA sequences*, Nucleic Acids Research, Vol. 27, No. 2, pp. 573-580. (1999).

[5] Bennett C.H.; Gacs P.; Li M.; Vitányi P.M.B.; Zurek W. *Information Distance, IEEE Transactions on Information Theory*, 44:4(1998), 1407-1423.

[6] Capparelli A.; Urtasun M. *Clasificación automática de documentos basada en la métrica de similitud universal de Vitányi*, Tesis de licenciatura, Facultad de Ciencias Exactas Y Naturales, U.B.A. (2004).

[7] Caspi A.; Pachter L. *Identification of transposable elements using multiple alignments of related genomes*, Genome Res., (16):260-270, (2006).

[8] Chaitin G. *A theory of program size formally identical to information theory*, J. Assoc. Comput. Mach., 22:329-340, (1975).

[9] Cilibrasi R.; Vitányi P.M.B. *Clustering by compression, IEEE Transactions on information theory*, vol. 51, no 4, (2005), 1523-1545.

[10] Makalowski W. *The human genome structure and organization*, NCBI, Vol. 48 Nro. 3. (2001).

[11] Edgar R.C.; Myers E.W. *PILER: identification and classification of genomic repeats*, (2005).

[12] Giordano J.; Ge Y.; Gelfand Y.; Abrusán G.; Benson G.; Warburton P.E. *Evolutionary History of Mammalian Transposons Determined by Genome-Wide Defragmentation*, PLoS Comput Biol 3(7): e137. doi:10.1371/journal.pcbi.0030137 (2007).

[13] Goodier J.L.; Kazazian H. Jr. *Retrotransposons Revisited: The Restraint and Rehabilitation of Parasites*, doi:10.1016/j.cell. (2008.09.022).

[14] Holmes I. *Transcendent Elements: Whole-Genome Transposon Screens and Open Evolutionary Questions*, Bioinformatics Group, Department of Statistics, University of Oxford, Oxford OXI 3TG, United Kingdom, (2002). CSH Press.

[15] Jurka J.; Kapitonov V.V.; Pavlicek A. *Repbase Update, a database of eukaryotic repetitive elements*, Cytogenet Genome Res, 110:462-467. (2005).

[16] Kolmogorov A.N. *Three approaches to the definition of the concept 'quantity of information', Problems in Information Transmission*, 1(1):1-7, (1965).

[17] Kurtz S.; Choudhuri J.V.; Ohlebusch E.; Schleiermacher C.; Stoye J.; Giegerich R. *REPuter: the manifold applications of repeat analysis on a genomic scale*, Nucleic Acids Res., 29(22):4633-4642, (2001).

[18] Larsson N.J.; Sadakane K. *Faster suffix sorting*, Theor. Comput. Sci., 387(3):258-272, (2007).

[19] Lesk A.M. *Introduction to Bioinformatics*, Oxford University Press. (2002).

[20] Mount D.W. *Bioinformatics, Sequence and Genome Analysis*, CSHI Press (2000).

[21] Saha S.; Bridges S.; Magbanua Z.V.; Peterson D.G. *Computational Approaches and Tools Used in Identification of Dispersed Repetitive DNA Sequences*, Springer Science+Business Media, February (2008).

[22] Smit A.F.A.; Hubley, R.; Green, P. *RepeatMasker Open-3.0*, (1996-2004).

[23] Tempel S.; Jurka M.; Jurka J. *VisualRepbase: an interface for the study of occurrences of transposable element families*, BMC Bioinformatics (2008), 9:345.