Artículo original

Metodología para el minado in silico de loci polimórficos en microsatélites

Methodology for in silico mining of microsatellite polymorphic loci

Carlos M. Martínez Ortiz1*
Alejandro Rivero Bandínez1

1 Departamento de Bioquímica, Universidad de Ciencias Médicas, ICPB "Victoria de Girón", La Habana, Cuba

*Autor para la correspondencia: cmmo@infomed.sld.cu

RESUMEN

Los polimorfismos con número variable de repeticiones en tándem (VNTR), constituyen marcadores genéticos utilizados en áreas de la genómica como estudios evolutivos, epidemiológicos y de genética poblacional. Los bancos de secuencias genómicas y las herramientas computacionales como BLAST permiten el minado de estos marcadores sin utilizar métodos experimentales, extendiéndolo a organismos no modelos de importancia médica o económica. Debido a la baja complejidad de estas secuencias y el número de candidatos que se presentan al inspeccionar un genoma cuando el procedimiento es escalado, surgen dificultades para procesar el volumen de datos generado y detectar por inspección visual los polimorfismos en los marcadores candidatos.

Se presentan una metodología y varios software que permiten la identificación y extracción rápida y fiable de *loci* polimórficos de SSRs. El procesamiento se hace por la concatenación de los programas MIDAS, BLAST, y el *script* PSSR-Extractor. Las entradas son rutas de directorios donde se encuentren múltiples archivos de secuencia en formato FASTA o GBFF y las salidas son los SSRs, códigos de acceso al GenBank, posiciones en el genoma, número de repeticiones y el grado de polimorfismo expresado como rango de variación, frecuencia alélica, cantidad de alelos y contenido de información polimórfica (PIC). Un *script* opcional, SSRMerge, permite la identificación de *loci* únicos (no redundantes) a nivel de especie, de género o en general del conjunto las secuencias que se desee procesar.

Se procesaron 23 genomas completos (RefSeq del NCBI) pertenecientes a diversos aislamientos de *Mycobacterium tuberculosis*. Se detectaron 4433 SSRs extrayéndose 414 *loci* no redundantes dentro de la especie. Realizado el minado de polimorfismos en las

salidas del servidor BLAST para estos SSRs se reportan medidas que reflejan las variaciones que presentan estos *loci*.

Palabras Clave: SSR; VNTR; marcador molecular; minería de datos; algoritmo

ABSTRACT

Polymorphisms with variable number of tandem repeats (VNTR), are genetic markers used in areas of genomics as evolutionary, epidemiological and population genetics studies. The growth of genomic sequences in data banks and the development of computational tools for bioinformatics allow the mining of these markers without the need to use experimental methods, extending the analysis to non-model organisms of medical or economic importance. Due to the low complexity of these sequences and the high number of candidates presented when inspecting one or several genomes in a scaled manner, difficulties arise in processing the volume of data that is generated and the detection of polymorphisms by visual inspection in candidate markers.
A methodology and its algorithmic specificities are described, implemented in a software pipeline, which allow the fast and reliable identification of polymorphic SSRs loci. The global processing is done by the concatenation of the programs MIDAS, BLAST and the PSSR-Extractor script. The inputs are directory paths where multiple sequence files are found in FASTA or GBFF format and the outputs are the SSRs, access codes to the databases, positions in the genome, number of repetitions and the degree of polymorphism expressed as range of variation, allelic frequency, allele number and polymorphic information content (PIC). An optional script, SSRMerge, allows the identification of unique (non-redundant) loci in the set of processed genome sequences with taxonomically closed relationship.
Twenty three complete genomes (RefSeq from NCBI) belonging to various isolates of Mycobacterium tuberculosis were processed, 4433 SSRs were detected and from them 414 non-redundant loci were extracted within the species. The polymorphisms for these SSRs were mined in the BLAST server outputs and different measures are reported that reflect loci variations.

Keywords: SSR; VNTR; molecular marker; data mining; algorithm

Introducción

Los microsatélites, o repeticiones de secuencia simple (SSR, por sus siglas en inglés), son pequeños motivos de ADN (entre 1 y 6 nucleótidos) repetidos en tándem, presentes en todos los genomas de organismos procariotas y eucariotas [1]. Estas secuencias se encuentran formando tractos que pueden ir desde unas pocas copias hasta cientos de ellas. El mecanismo molecular que explica estas secuencias es el llamado deslizamiento de la replicación, siendo el propio mecanismo el causante de la variabilidad observada en el número de copias [2]. Los microsatélites presentan altos niveles de polimorfismo, que se traduce en el número variable de repetidos en tándem (VNTR, siglas en inglés) [3], con tasas de mutación entre 10-2 y 10-5 por locus por generación, contrastando con las de otros valiosos marcadores, por ejemplo los polimorfismos de simple nucleótido (SNP) que presentan una tasa de mutación alrededor de 10-9. La variación en las tasas de mutación de los VNTR produce igualmente un amplio rango de diversidad alélica haciendo a estos marcadores muy valiosos para determinar el grado de relación biológica entre poblaciones de una misma especie.

Cuando se escogen los microsatélites como marcadores para determinado estudio, los investigadores tienen dos opciones para su detección y caracterización: generar datos de secuencia o hacer minería en repositorios de secuencias ya sean públicos o privados. La primera opción requiere la preparación de librerías genómicas, plataformas de secuenciación y software para la detección posterior. La segunda opción hace los dos primeros pasos de la primera opción innecesarios, eliminando sus elevados costos, quedando solo la etapa asistida por software [4].

El minado in silico de SSRs polimórficos comprende dos etapas: 1ra detectar los SSR, para la cual se han desarrollado una amplia gama de aplicaciones, que, a pesar de tener el mismo fin, emplean criterios estadísticos y computacionales diversos los cuales influyen en los resultados que se obtienen [5,6]; 2da determinar si estos SSR presentan polimorfismos en el número de copias, para lo cual es necesario hacer una comparación de sus secuencias contra repositorios de secuencias de especies relacionadas o de la misma especie. Las aplicaciones ideales para este fin son las de tipo BLAST (Basic Local Alignment Search Tool, http://blast.ncbi.nlm.nih.gov/) [7]. Esta segunda etapa, debido al propio objetivo del BLAST de buscar mediante alineamiento local subsecuencias homólogas y a la naturaleza propia de los SSR, que son regiones de muy baja complejidad, conlleva un post-procesamiento de las salidas que presenta serias dificultades cuando queremos hacer esta minería a gran escala. Este post-procesamiento se hace normalmente editando manualmente los alineamientos e inspeccionándolos visualmente cuando son pocos los SSR candidatos a explorar, pero cuando se trata de cientos o miles de candidatos, requiere necesariamente una formalización y automatización del proceso. En artículos revisados sobre el tema, la metodología para estos fines no está esclarecida, presentándose en algunos casos de manera poco explícita y en otros omitiéndose completamente. El hecho de que aparezcan pocas referencias sobre esta segunda etapa se debe a que el procedimiento habitual es pasar directamente al genotipado funcional de forma experimental, para lo

cual se requiere un extenso y costoso sistema de detección que incluye PCR, electroforesis y en ocasiones secuenciación.

En el caso de los microsatélites, la determinación del polimorfismo por variación en el número de copia, ya sea experimentalmente o in silico, se soporta en la especificidad de los flancos en cada locus. Estas secuencias flancos 5´ y 3´, que normalmente se reportan por los software que detectan SSR con tamaños alrededor de 20pb, se asumen conservadas y únicas en los genomas de cada especie, permitiendo la ubicación no ambigua del locus y convirtiéndose en las candidatas a secuencias cebadoras para la técnica de PCR.

Cuando hacemos una búsqueda extensiva con BLAST (blastn, para nucleótidos) y las secuencias de consulta (query) son SSR con sus respectivos flancos 5´ y 3´, se nos presentan varias complicaciones debido justamente a la baja complejidad de dichas secuencias y a la posibilidad de que los flancos no estén debidamente conservados. Debemos recordar que el BLAST es un sistema diseñado para detectar secuencias homólogas, y precisamente incluye filtros para de alguna forma eliminar las elevadas puntuaciones (score) que producen estas regiones repetidas. Este sistema no tiene un diseño específico para detectar secuencias homólogas no redundantes separadas por una región redundante y así captar el locus completo y poder comparar las variaciones en el número de copias.

En las Figuras 1A-1D se observan las salidas BLAST para un SSR con dichas características y los distintos tipos de alineamientos que producen los hallazgos (hits) para diferentes entradas en la base de datos escogida. Se trata de un SSR extraído de un genoma de vibrio cholerae enfrentado a una base de datos de secuencias nucleotídicas de esta especie bacteriana. El SSR query se muestra con flacos de 20 bp en letras mayúsculas y la región repetida, que es la región que debe variar en longitud en caso de ser un locus polimórfico, en letras minúsculas, con motivo de repetición aacaga. En la Figura 1A se observa como el blastn encuentra una secuencia que es idéntica al query, produciendo un 100% de identidad y un e-value de 1e-44. En este caso no se descarta que la secuencia encontrada (ID: AP018677.1) sea la misma de la cual se extrajo el SSR aunque no necesariamente tiene que ser así. En Fig. 1B se observa un caso ideal donde se encuentra una entrada en la base de datos (ID: CP026647.1) con un locus que presenta variación en el número de repeticiones, en este caso es una supresión de (aacaga)2. En la Fig. 1C se observan dos hits para una misma secuencia (ID: CP010812.1), que representan dos entradas en el fichero de salida del BLAST (hit-table) donde no se pudo cubrir el locus completamente por el sistema y se presentan dos alineamientos que se solapan en determinada región. Este representa un caso donde es complicado detectar, visualmente o por otro tipo de método, la variación en el número de copias. Por último, en Fig. 1D se observa una entrada para una secuencia (ID: CP028892.1) donde el alineamiento se truncó, no llegando hasta el otro flanco y no reportando ningún otro hit para esa misma secuencia que contuviera el flanco derecho.

Estos son algunos ejemplos donde se observan las complicaciones que pudiera presentar, para un marcador, el interpretar y detectar el polimorfismo cuando se hace computacionalmente. Cuando esto es escalado a cientos de marcadores es totalmente

imposible hacerlo por simple inspección visual de los alineamientos, incluso editando estos para resolver las entradas truncadas.

La metodología que presentamos describe las etapas y las bases algóritmicas para la detección computacional de polimorfismos de SSRs. El procedimiento general se hace por la concatenación de software que van desde la detección de los SSR, el procesamiento de los mismos por el sistema BLAST y la interpretación de las salidas del BLAST para la detección de los marcadores polimórficos.

En la siguiente sección (Métodos) se describe en detalle la secuencia de pasos que sigue esta metodología y los softwares empleados con la explicación de sus especificidades. En la sección Resultados, se expone y analiza la salida correspondiente a la detección de polimorfismos de SSRs en genomas de Mycobacterium tuberculosis. También se describen los parámetros de entrada, formatos de entrada y salida y los valores reportados.

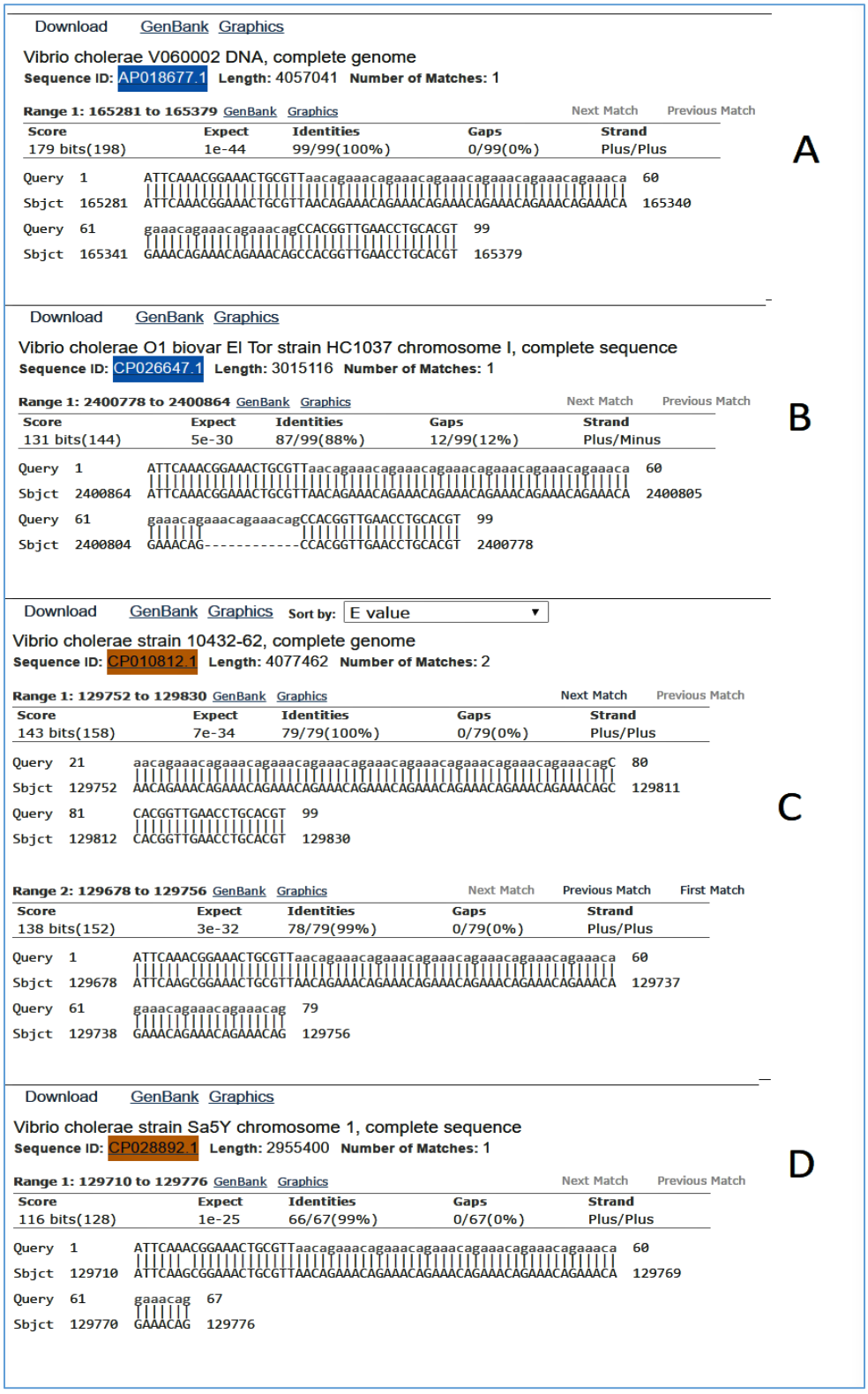


Fig. 1- Ejemplos de las salidas BLAST para un SSR donde se observan algunas de las complicaciones que afectan la identificación de un locus polimórfico. Ver en el texto para los casos A, B, C, D.

Métodos

En la Figura 2 se presenta la secuencia general de etapas para el minado in silico de SSRs polimórficos. Primeramente se hace una corrida de MIDAS [8] entrándole como parámetros un fichero de secuencia genómica que puede ser en formato FASTA o GBFF (de entradas sencillas o múltiples), la unidad mínima del repetido a detectar y los parámetros del alineamiento para match, mismatch e indel. MIDAS detecta todos los SSRs, exactos o aproximados, y genera un fichero MultiFASTA (extensión .mfaa) con ellos a los que le añade secuencias flancos de 20 bp en letras mayúsculas y la secuencia repetida en letras minúsculas. Este formato que marca la región repetida con letras minúsculas es utilizado por el BLAST como forma de enmascaramiento.

Seguidamente se corre el script SSRMerge que permite extraer el conjunto de SSR no redundantes a partir de múltiples ficheros de salida del MIDAS aplicados a múltiples genomas. Su primer parámetro es un camino en el directorio de la PC y procesará todos los ficheros con extensión .mfaa que encuentre en su interior. Esta etapa es opcional y es aplicable solo cuando estamos analizando múltiples genomas de especies relacionadas. Mientras mayor sea el vínculo taxonómico entre estos genomas, mayor será la probabilidad de encontrar locus similares conservados que se repiten en distintos genomas. El principio algorítmico de este script se basa en una comparación de los flancos de los SSR, todos contra todos, presentes en todos los ficheros que se procesen. Esta comparación se hace por alineamiento de secuencia global Nedleman-Wunsch. Cuando al comparar dos SSR y estos presentan más de un 90% de identidad se escoge uno de ellos y se desecha el otro. El resultado es un fichero MultiFASTA que contiene un conjunto de SSR no redundantes, de acuerdo a los parámetros definidos, y asumimos que estos SSRs pertenecen a loci distintos dentro del conjunto de genomas analizados.

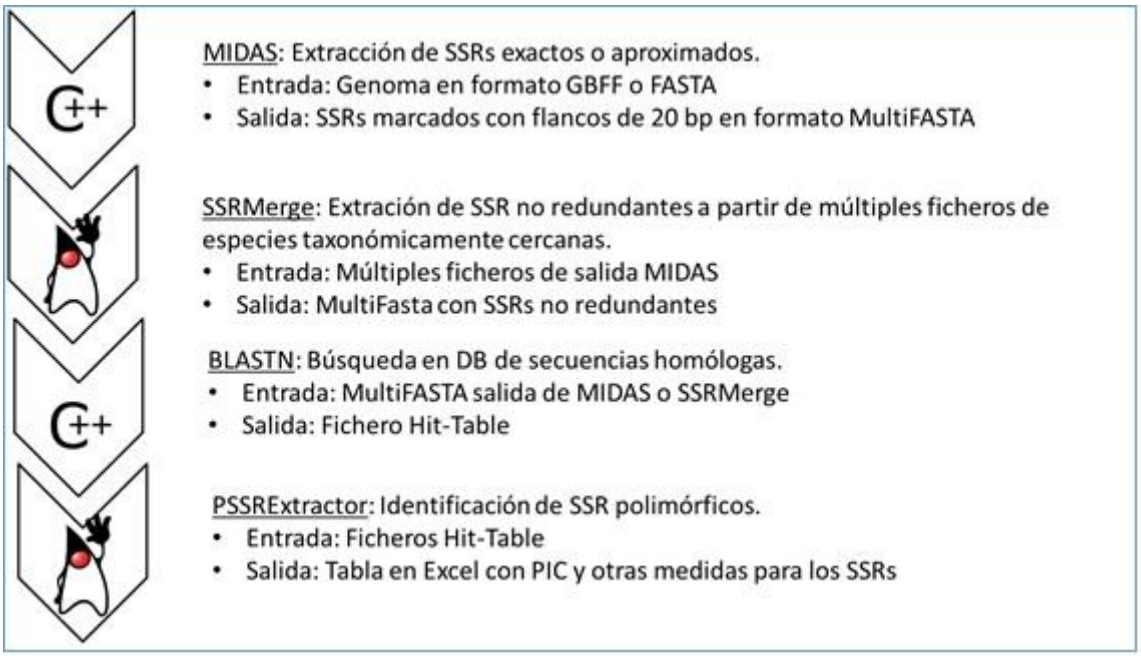


Fig. 2-Secuencia general de etapas para el minado in silico de SSRs polimórficos.

La tercera etapa es una corrida BLAST (NCBI server, http://blast.ncbi.nlm.nih.gov/). La entrada al servidor es el fichero MultiFASTA de la etapa previa. BLAST tiene muchos parámetros de configuración para hacer una corrida. Los parámetros que debemos modificar en nuestro caso particular son los siguientes: En el conjunto de búsqueda (Choose Search Set) especificar el organismo para el cual queremos hacer la búsqueda (e.g. Mycobacterium tuberculosis (taxid:1773)); en el tipo de programa seleccionar blastn, programa diseñado para encontrar homologías más remotas; en parámetros generales del algoritmo solo nos interesa modifica el umbral esperado (expect threshold) que por defecto es 10. El umbral esperado debe ser incrementado a >30 permitiéndonos encontrar hits que aunque tengan e-values grandes pueden ser loci de interés y la etapa siguiente ejecutada por PSSRextractor se encarga de depurar éstos. Otro parámetro que es necesario cambiar es el de enmascarado (mask). Aquí debemos marcar la opción de regiones de baja complejidad (low complexity regions) y enmascarar con letras minúsculas (mask lower case letters). Esto garantiza que el blastn intente reconocer todo el lucus incluyendo los dos flancos, aunque esto no siempre se logra, sobre todo cuando la región repetida es grande. El script PSSRextractor tiene implementadas reglas de negocio para solventar estos casos. La salida de esta etapa es el fichero texto de tabla de éxitos (hit-table) que genera el propio servidor.

Por último, la salida del BLAST es procesada por PSSRextractor. Este puede procesar uno o varias salidas pues su primer parámetro de entrada es un camino a un directorio, y procesará todos los ficheros de salidas BLAST que encuentre en su interior. Este script primeramente analiza sintácticamente (parser) la salida hit-table extrayendo toda su información. Los otros dos parámetros del script son porciento de identidad y porciento de cubrimiento que definen los criterios para tener en cuenta los flancos a considerar en cada entrada de la hit-table. Posteriormente procede a evaluar los polimorfismos por número variable de repeticiones presente en estos datos y lo hace de la manera siguiente:

Para un SSR (query en hit-table) habrá una o muchas entradas (subject en hit-table). Cada SSR tiene un tamaño de la unidad repetida (RUS). Cada entrada en hit-table tiene, entre otros, los siguientes valores: identificador de acceso del subject (SAV), longitud del alineamiento (AL), % de identidad (%I), posición inicial del query (q.start), posición final del query (q.end), posición inicial del subject (s.start) y posición final del subject (s.end). Entonces con estos datos se calcula el número de repeticiones (RN) de cada unidad repetida validando las siguientes condiciones:

1. Si AL > 40 Entonces: RN = (AL – 40) / RUS. Cuando el alineamiento es mayor o igual a 40 significa que cubrió ambos flancos del query, dado que la región repetida está marcada y no se tiene en cuenta, entonces la diferencia nos daría la cantidad de nucleótidos que están en la región repetida que dividida entre RUS nos devuelve el RN. Estos casos ocurren con muy baja frecuencia.
2. Si q.end <= 20 Entonces: La secuencia encontrada coincide con el flanco izquierdo. De lo contrario: La secuencia encontrada coincide con el flanco derecho. De modo que para

un mismo query, todas las entradas con el mismo SAV, se clasificaran a la izquierda o a la derecha del locus según esta condición. Estos casos son los que ocurren con mucha frecuencia. Ver Figura 3.

3. Si s.start izquierda < s.end izquierda && s.start derecha < s.end derecha && s.end izquierda < s.start derecha Entonces: RN = |s.start derecha – s.end izquierda| - 1 / RUS. Esta condición valida que el flanco izquierdo quede a la izquierda de la secuencia subject y que el flanco derecho está a la derecha, siendo la positiva dirección (es decir de 5´ a 3´). Similar condición se valida cuando la dirección es negativa solo hay que invertir las desigualdades estrictas. Esto ocurre porque el BLAST analiza también la complementaria de la secuencia en la BD (Fig. 3 y 4).

# blastn											
# Iteration: 0											
# Query: NC_021054.1|aatacg[1112818-1112843]|c:4|s:52|m:26|mm:0|i:0|ina:0|5e:1.69|3e:1.94											
# RID: Y692YMPN014											
# Database: nr											
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end, evalue, bit score											
# 200 hits found											
NC_021054.1	CP023640.1	100	20	0	0	1	20	1112840	1112859	0.25	37.4
NC_021054.1	CP023640.1	100	20	0	0	47	66	1112886	1112905	0.25	37.4
NC_021054.1	CP023639.1	100	20	0	0	1	20	1112791	1112810	0.25	37.4
NC_021054.1	CP023639.1	100	20	0	0	47	66	1112837	1112856	0.25	37.4
NC_021054.1	CP023638.1	100	20	0	0	1	20	1112753	1112772	0.25	37.4
NC_021054.1	CP023638.1	100	20	0	0	47	66	1112799	1112818	0.25	37.4

Fig. 3- Ejemplo de salida *hit-table* del BLAST. Se observa como para un *query* encuentra dos entradas para un mismo *subject* con **SAV** CP023640.1. La primera entrada tiene **q.start**=1 y **q.end**=20 (flanco izquierdo del *query*) y la segunda tiene **q.start**=47 y **q.end**=66 (flanco derecho del *query*).

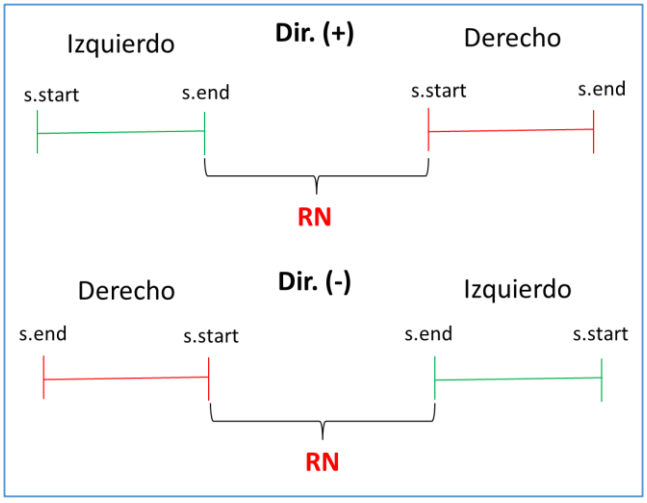


Fig. 4-Validación de la dirección (positiva o negativa de la comparación *query-subject*) y cálculo de RN.

4. **RN escogido = *min* {RNs encontrados}**. Esta validación se realiza porque se pueden dar casos dudosos debido a duplicaciones de regiones en un mismo genoma. Por ejemplo, para el *query*

con la unidad repetida AATACG (zona roja) entre el flanco izquierdo (zona verde) y flanco derecho (zona azul) se pueden encontrar los siguientes casos (Fig.5):

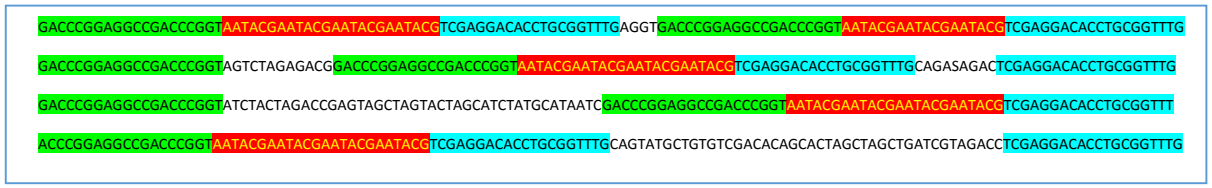


Fig. 5- Ejemplo de caso dudoso debido a duplicaciones de regiones en un mismo genoma

5. Solo se tienen en cuenta entradas donde los alineamientos tengan un **%I** (segundo parámetro del script) y un **% de cubrimiento** (tercer parámetro del script, (**AL**/20) x 100) mayores que 90. Estos dos parámetros garantizan que los flancos encontrados estén bien conservados.

Luego de tener el conjunto de NR para todos los *subjects* de un *query,* PSSRextractor genera dos ficheros de resultados, uno detallado y otro genérico, con nombres iguales al de la *hit-table* pero con los sufijos _specific.xls y _generic.xls respectivamente. El informe detallado brinda información sobre cada *subject* procesado y el genérico brinda la información relacionada al polimorfismo para cada *query*, es decir para cada SSR.

Entre la información relacionada al polimorfismo que brinda el reporte genérico se encuentran los siguientes valores:

I. **min_RN**, **max_RN** y **range**: Son tres columnas en el reporte que significan respectivamente el **RN** mínimo, el **RN** máximo y el rango (**max_RN** - **min_RN**).
II. **frecuency**: Frecuencia del alelo (número de repeticiones de la unidad repetida, **NR,** entre el total de todos los alelos) que presenta el SSR original a partir del cual se hizo la búsqueda.
III. **alleles**: Número de alelos encontrados para un *locus* (SSRs con RN diferentes).
IV. **PIC**: Contenido de Información Polimórfica ($1 - \sum_i p_i^2$). Este valor también se conoce en otros contextos como heterocigocidad promedio esperada o diversidad genética de Nei, y da una medida de la probabilidad de que, para un *locus* único, un par de alelos escogidos al azar en la población sean diferentes.

El resto de los valores que muestra el reporte genérico provienen de MIDAS (**access_number**, **pattern**, **pattern_length**, **RN**, **inaccuracy**, **entropy_5**, **entropy_3**), la cual aporta información valiosa al interpretar los SSR polimórficos. Por ejemplo el grado de inexactitud encontrado en el SSR o la entropía composicional de los flancos, nos permiten conocer respectivamente el grado de inexactitud del tracto repetido y cuán informativos pueden ser los flancos en la caracterización del *locus*.

Los valores de la última columna del reporte genérico (**exceptions**) muestran etiquetas que corresponden a excepciones en las validaciones del polimorfismo. Hay entradas en el reporte

genérico donde pueden aparecer más de una de estas etiquetas pues las excepciones se pueden dar simultáneas. Si todas las secuencias *subject* presentan excepciones, ya sea de unos o de otras, entonces se colocan las etiquetas. Las etiquetas son las siguientes:

i. **D** (*degenerated*): Los *subjects* tiene un **%I** y/o un **% cubrimiento** < 90%.
ii. **NF** (*not found*): No se encontró ningún *subject* en la base de datos con similitud.
iii. **O** (*outlier*): La cantidad de unidades repetidas entre los flancos de los *subjects* es dudosa por ser muy grande, siendo improbable que exista un microsatélite entre ellos. Los valores de corte establecidos para establecer esta excepción fueron mononucleótido: 157 bp, dinucleótido: 364 bp, trinucleótido: 109 bp, tetranucleótido: 45 bp, pentanucleótido: 150 bp y hexanucleótido: 193 bp. Estos valores fueron definidos después de procesar todos los SSR de más de 200 genomas bacterianos, registrando sus tamaños, y estableciendo el corte en 3 veces el rango intercuartil.
iv. **U** (*unpair*): Para una misma secuencia *subject* aparece un flanco y no el otro.

Resultados y Discusión

Mycobacterium tuberculosis (conocido también como bacilo de Koch) es un patógeno bacteriano, agente causal de la tuberculosis, infección contagiosa que afecta principalmente a los pulmones pero puede propagarse a otros órganos. La prevalencia de esta enfermedad es muy alta a nivel mundial, existiendo aproximadamente 12 millones de personas infectadas. El bacilo es también objeto de preocupación dentro de la comunidad médica y científica por presentar cepas con resistencia a múltiples antibióticos.

Los errores por deslizamiento en la replicación causantes de los microsatélites son reparados normalmente por tres enzimas mutL, mutS y mutH, sin embargo algunos genomas como los de micobacterias adolecen de este sistema enzimático [(9)]. Debido a ello estas especies bacterianas constituyen un valioso ejemplar para investigar las tasas de mutación de microsatélites y sus mecanismos regulatorios [(10)].

M. tuberculosis es un patógeno con una diversidad genética emergida de cepas más diversas, ganando así en sus mecanismos de virulencia [(11)]. Un ejemplo de importancia médica es la expansión de un microsatélite de micobacterias que ocurre en proteínas con el pentapétido-2 (PP2) [(12)].

En el presente estudio se analizaron 23 genomas completos (secuencias RefSeq), descargados del sitio ftp del NCBI (ftp://ftp.ncbi.nlm.nih.gov/refseq/release/bacteria/), pertenecientes a diversos aislamientos de *M. tuberculosis*. MIDAS detectó 4433 SSRs y de ellos se extrajeron con el *script* SSRMerge 414 *loci* no redundantes dentro de la especie. Se hizo el minado de polimorfismos para las salidas del servidor BLAST para estos SSRs con el script PSSRextractor. Las salidas de estos dos scripts pueden consultarse en los ficheros adjuntos (species_all.mfaa y Y692YMPN014-Alignment_generic_result.xls).

De los 414 SSR, fueron elegidos 288 que no mostraron ningún tipo de excepción, y de éstos, 104 mostraron **PIC** > 0, (36,1%). La Figura 6 muestra los valores promedios obtenidos para los diferentes tipos de SSRs clasificados por el tamaño de la unidad repetida (RUS).

RUS	Quantity	RN	inaccuracy	range	frequency	alleles	PIC
1	6	23.17	11.45	8.33	0.18	3.17	0.08
3	78	8.46	8.62	1.15	0.11	2.10	0.07
4	9	3.89	4.86	1.00	0.01	2.00	0.02
5	4	5.50	9.05	1.00	0.06	2.00	0.10
6	7	4.71	7.83	1.29	0.28	2.29	0.03

Fig. 6. Valores promedios para algunas medidas extraídas del reporte genérico en la especie *M. tuberculosis.*

Es significativa la cantidad de SSRs con 3bp como unidad repetida (76%). Esto es debido a que la mayor parte de los genomas bacterianos son regiones codificantes, las cuales presentan sesgo composicional en sus codones. Se detectó un único SSR para dinucleótidos y fue excluido del resultado por presentar excepciones en el análisis de polimorfismo. Como promedio, los SSR de 1bp fueron más inexactos y mostraron mayor polimorfismo con más cantidad de alelos y un PIC elevado (3.17 y 0.08 respectivamente). Los tractos de repeticiones (RN) fueron también significativamente mayores al resto.

pattern	RUS	RN	inaccuracy	entropy_5	entropy_3	min_RN	max_RN	range	frequency	alleles	PIC
g	1	15	6.67	1.55	1.72	14	23	9	0.989	3	0.021
c	1	20	15	1.88	1.68	20	25	5	0.01	2	0.02
g	1	27	14.8	1.74	1.86	27	35	8	0.01	2	0.02
g	1	23	17.4	1.68	1.24	21	23	2	0.025	2	0.049
g	1	27	7.41	1.72	1.94	14	27	13	0.01	5	0.185
g	1	27	7.41	1.72	1.94	14	27	13	0.01	5	0.185
acc	3	6	5.26	1.8	1.88	5	7	2	0.95	3	0.095
acc	3	4	0	1.69	1.95	4	5	1	0.01	2	0.02
agc	3	4	0	1.8	1.72	4	5	1	0.01	2	0.02
ccg	3	6	10	1.96	1.91	6	7	1	0.01	2	0.02
ccg	3	4	0	1.88	1.93	3	4	1	0.426	2	0.489
ccg	3	6	5.26	1.68	1.95	6	10	4	0.03	5	0.494
ccg	3	5	5.88	1.95	1.82	5	6	1	0.01	2	0.02
ccg	3	4	0	1.93	1.92	4	5	1	0.01	2	0.02
ccg	3	4	0	1.99	1.97	4	5	1	0.01	2	0.02
ccg	3	10	15.6	1.68	1.86	10	11	1	0.01	2	0.02
ccg	3	14	22.7	1.94	1.93	14	15	1	0.01	2	0.02
ccg	3	21	18.8	1.93	1.95	21	22	1	0.01	2	0.02
ccg	3	18	22.8	1.74	1.84	18	19	1	0.01	2	0.019
ccg	3	6	10	1.85	1.99	6	7	1	0.01	2	0.02
ccg	3	5	5.88	1.96	1.99	5	6	1	0.01	2	0.02
cgg	3	4	0	1.97	1.86	4	5	1	0.01	2	0.02
cgg	3	4	0	1.88	1.86	4	5	1	0.01	2	0.02
cgg	3	6	10.5	1.94	1.86	6	7	1	0.5	2	0.5
cgg	3	8	11.1	1.96	1.97	8	9	1	0.01	2	0.02
cgg	3	5	5.56	1.85	1.91	5	6	1	0.012	2	0.024
cgg	3	4	0	1.72	1.74	4	5	1	0.01	2	0.02
cgg	3	4	0	1.96	1.94	4	5	1	0.01	2	0.02
cgg	3	4	0	1.74	1.99	4	5	1	0.01	2	0.02
cgg	3	14	24.4	1.87	1.86	14	15	1	0.01	2	0.02
cgg	3	22	20.6	1.85	1.86	22	23	1	0.5	2	0.5
cgg	3	19	29.3	1.8	1.93	19	21	2	0.01	3	0.058
cgg	3	4	0	1.77	1.85	4	5	1	0.01	2	0.02
cgt	3	4	0	1.84	1.86	4	5	1	0.01	2	0.02
cgt	3	6	10	1.74	1.72	6	7	1	0.01	2	0.02
cgt	3	4	0	1.72	1.76	4	5	1	0.01	2	0.02
cgt	3	4	0	1.99	1.85	4	5	1	0.01	2	0.02
ggt	3	4	0	1.9	1.85	4	5	1	0.01	2	0.02
ggt	3	6	10	1.68	1.86	6	7	1	0.01	2	0.02
ggt	3	4	0	1.8	1.64	4	5	1	0.01	2	0.02
gtt	3	4	0	1.37	1.96	4	5	1	0.01	2	0.02
acc	3	11	11.8	1.68	1.69	11	12	1	0.01	2	0.02
ccg	3	4	0	1.58	1.95	4	5	1	0.02	2	0.039
cgg	3	8	16.7	1.99	1.97	7	8	1	0.941	2	0.112
cgg	3	4	0	1.79	1.93	4	5	1	0.02	2	0.039
ggt	3	7	13	1.8	1.78	7	9	2	0.01	3	0.039
ccg	3	10	15.6	1.88	1.79	10	11	1	0.99	2	0.02
cgg	3	4	0	1.88	1.86	4	5	1	0.01	2	0.02
ccg	3	32	22.7	1.88	1.91	31	32	1	0.088	2	0.16
act	3	6	10	1.37	1.92	5	7	2	0.822	3	0.296
ccg	3	16	16	1.91	1.84	16	17	1	0.01	2	0.02
ccg	3	16	21.6	1.77	1.84	16	17	1	0.015	2	0.029
aac	3	4	0	1.96	1.37	4	5	1	0.01	2	0.02
acc	3	4	0	1.64	1.8	4	5	1	0.01	2	0.02
acc	3	6	10	1.86	1.68	6	7	1	0.01	2	0.02
acc	3	4	0	1.85	1.9	4	5	1	0.01	2	0.02
acg	3	4	0	1.76	1.72	4	5	1	0.01	2	0.02

Fig. 7- SSRs provenientes de la especie *M. tuberculosis* con **PIC** > 0.

pattern	RUS	RN	inaccuracy	entropy_5	entropy_3	min_RN	max_RN	range	frequency	alleles	PIC
acg	3	6	10	1.72	1.74	6	7	1	0.01	2	0.02
acg	3	4	0	1.86	1.84	4	5	1	0.01	2	0.02
ccg	3	4	0	1.84	1.97	4	5	1	0.01	2	0.02
ccg	3	4	0	1.94	1.96	4	5	1	0.01	2	0.02
ccg	3	4	0	1.74	1.72	4	5	1	0.01	2	0.02
ccg	3	5	5.56	1.91	1.85	5	6	1	0.012	2	0.024
ccg	3	4	0	1.91	1.94	4	5	1	0.01	2	0.02
ccg	3	8	11.1	1.97	1.96	8	9	1	0.01	2	0.02
ccg	3	6	10.5	1.86	1.94	6	7	1	0.5	2	0.5
ccg	3	4	0	1.86	1.88	4	5	1	0.01	2	0.02
ccg	3	4	0	1.86	1.97	4	5	1	0.01	2	0.02
cgg	3	5	5.88	1.99	1.96	5	6	1	0.01	2	0.02
cgg	3	6	10	1.99	1.85	6	7	1	0.01	2	0.02
cgg	3	18	22.8	1.84	1.77	18	19	1	0.01	2	0.019
cgg	3	14	25	1.79	1.79	14	15	1	0.01	2	0.02
cgg	3	21	18.8	1.95	1.93	21	22	1	0.01	2	0.02
cgg	3	10	15.6	1.79	1.88	10	11	1	0.99	2	0.02
cgg	3	8	18.5	1.93	1.88	8	9	1	0.005	2	0.01
cgg	3	8	18.5	1.93	1.88	8	9	1	0.005	2	0.01
cgg	3	14	22.7	1.93	1.94	14	15	1	0.01	2	0.02
cgg	3	24	23.4	1.85	2	24	25	1	0.01	2	0.02
cgg	3	10	15.6	1.86	1.68	10	11	1	0.01	2	0.02
cgg	3	4	0	1.93	1.99	4	5	1	0.01	2	0.02
ggt	3	6	5.26	1.88	1.8	5	7	2	0.95	3	0.095
ccg	3	43	26.9	1.74	1.24	43	48	5	0.012	2	0.023
ccg	3	21	25.4	1.62	1.87	20	21	1	0.01	2	0.02
ggt	3	4	0	1.59	1.85	4	5	1	0.5	2	0.5
atcg	4	5	9.09	1.91	1.41	5	6	1	0.01	2	0.02
ccgg	4	5	13.6	1.99	1.76	5	6	1	0.01	2	0.02
ccgg	4	4	5.26	1.87	1.72	4	5	1	0.01	2	0.02
cggg	4	3	0	1.93	1.86	3	4	1	0.01	2	0.02
cggg	4	4	5.26	1.94	1.93	4	5	1	0.01	2	0.02
cggg	4	3	0	1.69	1.93	3	4	1	0.01	2	0.02
cccg	4	4	5.26	1.93	1.94	4	5	1	0.01	2	0.02
cccg	4	3	0	1.86	1.93	3	4	1	0.01	2	0.02
ccgg	4	4	5.26	1.72	1.87	4	5	1	0.01	2	0.02
cccgg	5	6	15.2	1.93	1.92	6	7	1	0.01	2	0.02
cgcgg	5	4	4.55	1.69	1.86	4	5	1	0.01	2	0.02
cgcgg	5	8	11.9	1.74	1.91	7	8	1	0.2	2	0.32
ccgcg	5	4	4.55	1.86	1.69	4	5	1	0.01	2	0.02
aatacg	6	4	0	1.69	1.94	3	5	2	0.97	3	0.058
accagc	6	4	7.41	1.77	1.94	4	5	1	0.01	2	0.02
accgcc	6	7	20	1.93	1.69	7	8	1	0.005	2	0.01
atgtcg	6	3	0	1.85	1.3	3	4	1	0.01	2	0.02
accgcc	6	7	20	1.85	1.69	7	8	1	0.005	2	0.01
attcgt	6	4	0	1.94	1.69	3	5	2	0.97	3	0.058
ctggtg	6	4	7.41	1.94	1.77	4	5	1	0.01	2	0.02

Fig. 7- (cont.): SSRs provenientes de la especie *M. tuberculosis* con **PIC** > 0.

En la Figura 7 se muestra el listado completo de los 104 SSR extraídos que mostraron algún nivel de polimorfismo (**PIC** > 0). Las secuencias íntegras de los marcadores, incluidas las secuencias flancos, los números de acceso y las posiciones en el genoma pueden ser obtenidas de las salida de MIDAS.

La metodología descrita tiene aspectos distintivos con respecto a otros procedimientos in silico reportados en la literatura:

(I) Se ha demostrado en ensayos experimentalmente que no todos los loci de SSRs muestran polimorfismo. Esto se debe, entre otros aspectos, a que el *locus* dentro de la población analizada no se encuentra sujeto a una dinámica de cambio en particular. En este sentido, la metodología nos permite seleccionar aquellos *loci* que sí manifiestan polimorfismos en los bancos de secuencia escogidos, reduciendo los costos que esto implica cuando se hace de forma experimental.

(II) La determinación del polimorfismo presente en el locus es totalmente automatizada. Los procedimientos comunes, no experimentales, emplean el alineamiento múltiple de los marcadores para luego, por inspección visual, detectar los polimorfismos. En este sentido, el procedimiento es ideal para el análisis a gran escala.

(III) El polimorfismo se define estrictamente como variación en el número de copias del SSR (PIC > 0), y no como simples inserciones o supresiones presentes en los marcadores que no correspondan con el tamaño de la unidad repetida.

(IV) El procedimiento puede hacerse partiendo de la detección de SSRs en una secuencia, pero opcionalmente, también puede partir de múltiples secuencias sin necesidad de hacer ensamblaje para obtener una secuencia consenso. Esto permite detectar SSRs que no pertenecen al mismo locus a pesar de estar en genomas muy emparentados. El script SSRMerge permite eliminar la redundancia de los loci comunes a todas las secuencias.

La metodología es excelente para el análisis puramente computacional de loci de SSRs, en estudios evolutivos, de identificación genotípica o estudios funcionales a partir de los genes involucrados. Para su uso en el análisis experimental utilizando PCR, la metodología brinda toda la información necesaria (identificador de acceso a la secuencia, posiciones en el genoma, secuencias flancos, etc.) que permite el diseño de cebadores utilizando otras aplicaciones disponibles en internet.

Disponibilidad

Todos los software están disponibles en el material suplementario: MIDAS (distribución binaria midas_v1.1.exe), SSRMerge and PSSRExtractor se suministran en formato comprimido zip (ambos son Java NetBeans Projects para JDK 1.8 o superior, y los ficheros binarios jar, para ejecutar por línea de comandos, están en la carpeta \dist una vez descomprimidos).

Conclusiones

Se describe una metodología para la detección totalmente computacional de loci polimórficos de microsatélites. Como ejemplo de su utilización se procesaron 23 genomas completos pertenecientes a diversos aislamientos de *M. tuberculosis*. Se detectaron 4433 SSRs y de ellos se extrajeron 414 *loci* no redundantes dentro de la especie. Se hizo el minado de polimorfismos para las salidas de BLAST y 100 SSRs mostraron polimorfismos. La metodología es intuitiva y

viene acompañada de software para su aplicación. Su principal ventaja radica en los niveles de escalado que permite y la reducción de costos cuando se hacen análisis experimentales permitiendo la preselección de marcadores que han evidenciado polimorfismo en bancos de secuencias genómicas escogidos.

Bibliografía

1. Li YC, Korol AB, Fahima T, Beiles A, Nevo E. Microsatellites: Genomic distribution, putativefunctions and mutational mechanisms: A review. Molecular Ecology 2002; 11: 2453–65.
2. Ellegren, H. Microsatellites: Simple sequences with complex evolution. Nature Reviews. Genetics 2004; 5: 435–45.
3. Xu, J.S.,Wu,Y.T.,Ye,S.J.,Wang,L.,and Feng,Y.Z. SSR primer screening and assessment on pear germplasm resources. J. Central South Univ. Forest.Technol. 2012; 32, 80–5.
4. Hodel et al. Using Microsatellites in the 21st Century. Applications in Plant Sciences 2016 4(6)
5. Leclercq, S., Rivals, E., Jarne, P. Detecting Microsatellites Within Genomes: Significant Variation Among Algorithms. BMC Bioinformatics 2007, 8:125.
6. Grover A, Aishwarya V, Sharma PC. Searching Microsatellites in DNA Sequences: Approaches Used and Tools Developed. Physiol Mol Biol Plants (January–March 2012) 18(1):11–19
6. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic Local Alignment Search Tool. J. Mol. Biol. (1990) 215:403-410.
7. Martínez CM. MIDAS: Computer Application for the Identification of Exact and Inaccurate Microsatellites in Genomic Sequences. *Revista Cubana de Informática Médica*, Vol. 18, No. 2 (2018).
8. Fleischmann RD, Alland D, Eisen JA, Carpenter L, White O, Peterson J et AL. Whole-genome Comparison of Mycobacterium Tuberculosis Clinical and Laboratory Strains. J Bacteriol 2002, 184(19):5479-5490.
9. Sreenu V, Kumar P, Nagaraju J, Nagarajaram H. Microsatellite Polymorphism Across the M. Tuberculosis and M. Bovis Genomes: Implications on Genome Evolution and Plasticity. BMC Genomics 2006, 7:78.
10. Supply P, Marceau M, Mangenot S, Roche D, Rouanet C, Khanna V, et al. Genomic Analysis of Smooth Tubercle Bacilli Provides Insights Into Ancestry and Pathoadaptation of Mycobacterium Tuberculosis. Nat Genet. 2013 Feb; 45(2):172–179.
11. Warholm P, Light S. Identification of a Non-Pentapeptide Region Associated with Rapid Mycobacterial Evolution. PLoS ONE (2016), 11(5): e0154059.

Artículo original

Methodology for *in silico* mining of microsatellite polymorphic *loci*

Metodología para el minado in silico de loci polimórficos en microsatélites

Carlos M. Martínez Ortiz[1*]
Alejandro Rivero Bandínez[1]

[1]Department of Biochemistry, University of Medical Sciences, ICPB "Victoria de Girón", Havana, Cuba

*Autor para la correspondencia: cmmo@infomed.sld.cu

SUMMARY

Polymorphisms with variable number of tandem repeats (VNTR), are genetic markers used in areas of genomics as evolutionary, epidemiological and population genetics studies. The growth of genomic sequences in data banks and the development of computational tools for bioinformatics allow the mining of these markers without the need to use experimental methods, extending the analysis to non-model organisms of medical or economic importance. Due to the low complexity of these sequences and the high number of candidates presented when inspecting one or several genomes in a scaled manner, difficulties arise in processing the volume of data that is generated and the detection of polymorphisms by visual inspection in candidate markers.

A methodology and its algorithmic specificities are described, implemented in a software pipeline, which allow the fast and reliable identification of polymorphic SSRs *loci*. The global processing is done by the concatenation of the programs MIDAS, BLAST and the PSSR-Extractor script. The inputs are directory paths where multiple sequence files are found in FASTA or GBFF format and the outputs are the SSRs, access codes to the databases, positions in the genome, number of repetitions and the degree of polymorphism expressed as range of variation, allelic frequency, allele number and polymorphic information content (PIC). An optional script, SSRMerge, allows the identification of unique (non-redundant) *loci* in the set of processed genome sequences with taxonomically closed relationship.

Twenty three complete genomes (RefSeq from NCBI) belonging to various isolates of *Mycobacterium tuberculosis* were processed, 4433 SSRs were detected and from them 414 non-redundant *loci* were extracted within the species. The polymorphisms for these SSRs were mined in the BLAST server outputs and different measures are reported that reflect *loci* variations.

Key words: SSR; VNTR; Molecular marker; Data mining; Algorithm.

RESUMEN

Los polimorfismos con número variable de repeticiones en tándem (VNTR), constituyen marcadores genéticos utilizados en áreas de la genómica como estudios evolutivos, epidemiológicos y de genética poblacional. Los bancos de secuencias genómicas y las herramientas computacionales como BLAST permiten el minado de estos marcadores sin utilizar métodos experimentales, extendiéndolo a organismos no modelos de importancia médica o económica. Debido a la baja complejidad de estas secuencias y el número de candidatos que se presentan al inspeccionar un genoma cuando el procedimiento es escalado, surgen dificultades para procesar el volumen de datos generado y detectar por inspección visual los polimorfismos en los marcadores candidatos.

Se presentan una metodología y varios software que permiten la identificación y extracción rápida y fiable de *loci* polimórficos de SSRs. El procesamiento se hace por la concatenación de los programas MIDAS, BLAST, y el *script* PSSR-Extractor. Las entradas son rutas de directorios donde se encuentren múltiples archivos de secuencia en formato FASTA o GBFF y las salidas son los SSRs, códigos de acceso al GenBank, posiciones en el genoma, número de repeticiones y el grado de polimorfismo expresado como rango de variación, frecuencia alélica, cantidad de alelos y contenido de información polimórfica (PIC). Un *script* opcional, SSRMerge, permite la identificación de *loci* únicos (no redundantes) a nivel de especie, de género o en general del conjunto las secuencias que se desee procesar.

Se procesaron 23 genomas completos (RefSeq del NCBI) pertenecientes a diversos aislamientos de *Mycobacterium tuberculosis*. Se detectaron 4433 SSRs extrayéndose 414 *loci* no redundantes dentro de la especie. Realizado el minado de polimorfismos en las salidas del servidor BLAST para estos SSRs se reportan medidas que reflejan las variaciones que presentan estos *loci*.

Palabras Clave: SSR; VNTR; marcador molecular; minería de datos; algoritmo

Introduction

Microsatellites, or simple sequence repeats (SSRs), are small DNA motifs (from 1 to 6 nucleotides) repeated in tandem, present in all the genomes of prokaryotic and eukaryotic organisms [1]. These sequences forms tracts that can range from a few copies to hundreds of them. The molecular mechanism that explains these sequences is the so-called replication slippage, being the mechanism itself the cause of variability observed in the number of copies [2]. Microsatellites present high levels of polymorphism, which is translated as variable number of tandem repeats (VNTR) [3]. They present mutation rates between 10^{-2} and 10^{-5} per *locus* per generation, contrasting with other markers, for example, simple nucleotide polymorphisms (SNP) that has mutation rate around 10^{-9}. The variation in the mutation rates of VNTRs also produces a wide range of allelic diversity, making these markers very valuable to determine the degree of biological relationship between populations at the same species.

When, for a given study, microsatellites are chosen as markers, researchers have two options for their detection and characterization: generating sequence data or mining in repositories of sequences, whether public or private. The first option requires the preparation of genomic libraries, sequencing platforms and software for subsequent detection. The second option makes the first two steps of the first option unnecessary, eliminating its high costs, leaving only the stage assisted by software [4].

In silico mining of polymorphic SSRs comprises two stages: 1st detect SSRs, with a wide range of applications developed, which, despite having the same purpose, use diverse statistical and computational criteria that influences the results obtained [5, 6]; 2nd determine if these SSRs has polymorphisms in repeat copy number, for which it is necessary to make sequence comparisons against repositories of sequences of related or the same species. The ideal applications for this purpose are those of BLAST type (Basic Local Alignment Search Tool, http://blast.ncbi.nlm.nih.gov/) [7]. This second stage, due the goal of BLAST of searching through local alignment homologous subsequences and the composition of the SSRs, which are regions of very low complexity, entails post-processing the outputs which presents serious difficulties when we want to do this mining in a scaled manner. This post-processing is usually done by manually editing the alignments and visually inspecting them when there are few SSRs candidates to explore, but when it comes to hundreds or thousands of candidates, the process necessarily requires a formalization and automation. In reviewed articles on the subject, the methodology for these purposes is not clarified, presenting itself in some cases in a low explicit manner and in others, omitting completely. Few references appear on this second stage and it is due to the usual procedure of pass directly to functional genotyping in experimental manner, for which an extensive and expensive detection system is required, which includes PCR, electrophoresis and sometimes sequencing.

In the case of microsatellites, the determination of polymorphism by variation in copy number, either experimentally or *in silico*, is supported in the specificity of the flanks at each *locus*. These 5'- and 3'-flanking sequences, which are normally reported by software that detect SSR with sizes around 20bp, are assumed to be conserved and unique in the genomes of each

species, allowing the unambiguous location of the *locus* and becoming candidates for primer sequences in PCR techniques.

When we do an extensive search with BLAST (blastn, for nucleotides) and the query sequences are SSRs with their respective 5'and 3' flanks, we have several complications due to the low complexity or redundancy of these sequences and also the possibility that flanks are not properly preserved. We must remember that BLAST is a system designed to detect homologous sequences that precisely includes filters to eliminate the high scores that produce these repeated regions. This system does not have a specific design to detect non-redundant homologous sequences separated by a redundant region and therefore it can't capture the *locus* as a whole in order to compare the variations in copy number.

Figures 1A - 1D show BLAST outputs for an SSR with these characteristics and the different types of alignments that produce the findings (hits) for different entries in the chosen database. It is an SSR extracted from a genome of *Vibrio cholerae* faced with a database of nucleotide sequences of this bacterial species. The SSR query is shown with flanks of 20 bp in capital letters and the repeat region, which is the region that must vary in length in case of being a polymorphic *locus*, in lowercase letters, with repetition pattern **aacaga**. Figure 1A shows how the blastn finds a sequence that is identical to the query, producing 100% identity and an e-value of 1e-44. In this case it is not ruled out that the sequence found (ID: AP018677.1) is the same one from which the SSR was extracted, although it does not necessarily have to be that way. Figure 1B shows an ideal case where an entry is found in the database (ID: CP026647.1) with a *locus* that presents variation in the number of repetitions, in this case it is a suppression of $(aacaga)_2$. Figure 1C shows two hits for the same sequence (ID: CP010812.1), which represent two entries in the BLAST output file (hit table) where the *locus* could not be completely covered by the system and two alignments are presented that overlap in a certain region. This represents a case where it is complicated to detect, visually or by another type of method, the variation in the number of copies. Finally, Figure 1D shows an entry for a sequence (ID: CP028892.1) where the alignment was truncated, not reaching the other flank and not reporting any other hit for that sequence with the right flank.

These are some examples where one can observe possible complications to interpret and detect polymorphism in a computational manner. When this is scaled to hundreds of markers it is totally impossible to do by simple visual inspection of alignments, even editing these to solve truncated entries.

The methodology that we present describes the stages and the algorithmic bases for the computational detection of polymorphisms in SSRs. The general procedure is done by the concatenation of software ranging from the detection of the SSRs, processing these by BLAST system and interpretation of BLAST outputs for detection of polymorphic markers.

In the following section (**Methods**) the sequence of steps followed by this methodology and the software used with explanation of its specificities are described in detail. In the **Results** section, the output corresponding to the detection of polymorphisms of SSRs in *Mycobacterium*

tuberculosis genomes is exposed and analyzed. It also describes input parameters, input and output formats and reported values.

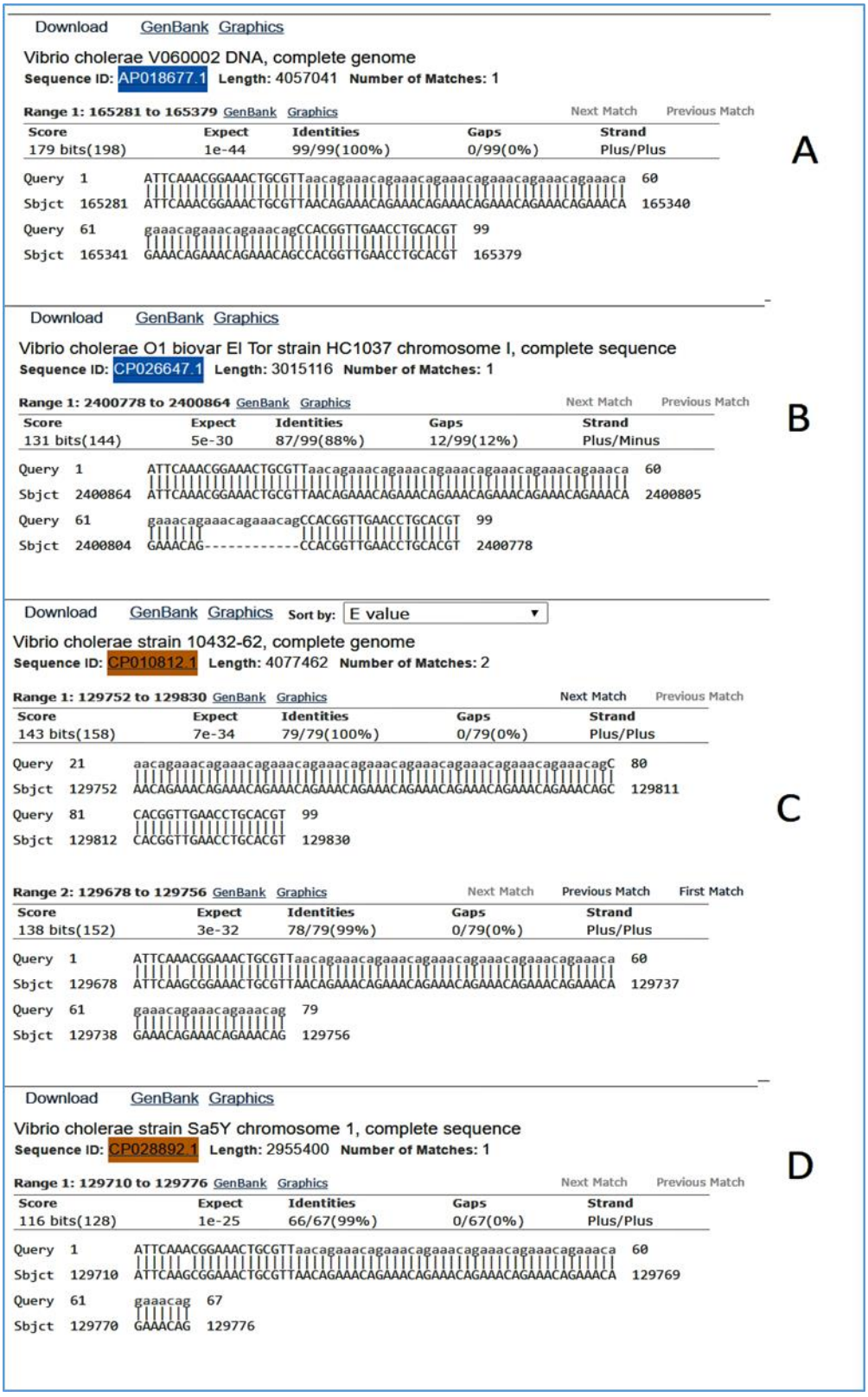


Figure 1. Examples of BLAST outputs for an SSR showing some complications that affect the identification of polymorphic *locus*. See explanations in the text for cases A, B, C, D.

Methods

Figure 2 shows the general sequence of stages for *in silico* mining of polymorphic SSRs. First a run of MIDAS [8] is done, entering as parameters a genomic sequence file that can be in FASTA or GBFF format (with simple or multiple entries), the minimum unit of the repeat to be detected and the alignment parameters for match, mismatch and indel. MIDAS detects all SSRs, exact or approximate, and generates a MultiFASTA file (extension .mfaa) with 20 bp flanking sequences added in capital letters and the repeat in lowercase letters. This format that marks the repeat region with lowercase letters is used by BLAST as masking procedure.

Then we run SSRMerge script, which allows extracting the set of non-redundant SSRs from multiple MIDAS output files applied to multiple genomes. Its first parameter is a path to a directory in the PC and it will process all the files with extension .mfaa. This is an optional stage and is applicable only when we are analyzing multiple genomes of related species. The greater the taxonomic link between these genomes, the greater the probability of finding similar conserved *loci* that are repeated in different genomes. The algorithmic principle of this script is an all against all flank comparison of SSRs present in all files processed. This comparison is made by Nedleman-Wunsch global sequence alignment. When comparing two SSRs and they present more than 90% identity, one of them is chosen and the other is rejected. The result is a MultiFASTA file containing a set of non-redundant SSRs, according to the defined parameters, and we assume that these SSRs belong to different *loci* in the set of genomes analyzed.

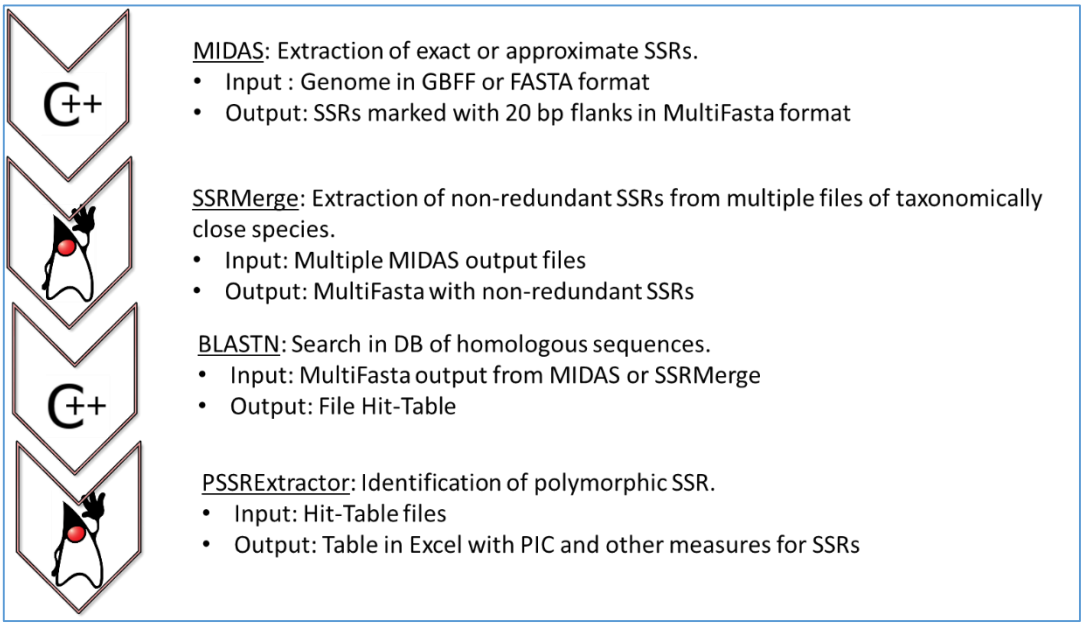


Figure 2. General sequence of steps for *in silico* mining of polymorphic SSRs.

The third step is a BLAST submitting (NCBI server, http://blast.ncbi.nlm.nih.gov/). The entry to the server is the MultiFASTA file of the previous stage. BLAST has many configuration parameters but the parameters that we must modify in that particular case are the following: In the search set (Choose Search Set) specify the organism for which we want to search (e.g. *Mycobacterium tuberculosis* (taxid: 1773)); in the program type select blastn, designed to find more remote homologies; in general parameters of the algorithm we are only interested in modifying the expected threshold (default threshold) which by default is 10. The expected threshold should be increased to > 30 allowing to find hits that although have large e-values they can be *loci* of interest. The next stage executed by PSSRextractor is responsible for debugging the wrong hits. Another parameter that needs to be changed is the masking procedure. We must mark the option of regions of low complexity and masking with lowercase letters. This guarantees that the blastn tries to recognize all *loci* including two flanks despite this is not always achieved, especially when the repeated region is large. The PSSRextractor script implements business rules to solve these cases. The output of this step is a hit table generated by the server.

Finally, the BLAST output is processed by PSSRextractor. It can process one or several outputs because its first input parameter is a path to a directory. It will process all the BLAST output files. This script first parses the hit table extracting all its information. The other two script parameters are identity and percentage of coverage that define criteria to consider the flanks in each entry of hit table. Subsequently it proceeds to evaluate the polymorphisms by variable number of repeat in the SSRs and it does so as follows:

For an SSR (query in hit-table) there will be one or many entries (subject in hit-table). Each SSR has a repeated unit size (RUS). Each entry in hit-table has the following values: subject access identifier (SAV), alignment length (AL), identity% (% I), initial position of the query (q.start), final position of the query (q.end), initial position of the subject (s.start) and final position of the subject (s.end). Then taking into account these data, the number of repetitions (RN) of each repeat unit is calculated by validating the following conditions:

1. If AL> 40 Then: RN = (AL - 40) / RUS. When the alignment is greater than or equal to 40 it means that it covered both sides of the query, since the repeated region is marked and not taken into account. Then the difference would give us the number of nucleotides that are in the repeat region that divided by RUS gives us the RN (Repeat Number). These cases occur with very low frequency.
2. If q.end <= 20 then the sequence found coincides with the left flank, otherwise, the sequence found coincides with the right flank. So for the same query, all the entries with the same SAV, will be classified to the left or to the right of the *locus* according to this condition. These cases are those that occur very frequently. See Figure 3.
3. If s.start left < s.end left AND s.start right < s.end right AND s.end left < s.start right, then RN = (|s.start right - s.end left| - 1) / RUS. This condition validates that the left flank is to the left of the subject sequence and that the right flank is

to the right, being positive the direction (i.e. from 5' to 3'). Similar condition is validated when the address is negative, you just have to reverse the strict inequalities. This occurs because BLAST also analyzes the complementary sequence in the BD. See Figures 3 and 4.

# blastn											
# Iteration: 0											
# Query: NC_021054.1|aatacg[1112818-1112843]|c:4|s:52|m:26|mm:0|i:0|ina:0|5e:1.69|3e:1.94											
# RID: Y692YMPN014											
# Database: nr											
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end, evalue, bit score											
# 200 hits found											
NC_021054.1	CP023640.1	100	20	0	0	1	20	1112840	1112859	0.25	37.4
NC_021054.1	CP023640.1	100	20	0	0	47	66	1112886	1112905	0.25	37.4
NC_021054.1	CP023639.1	100	20	0	0	1	20	1112791	1112810	0.25	37.4
NC_021054.1	CP023639.1	100	20	0	0	47	66	1112837	1112856	0.25	37.4
NC_021054.1	CP023638.1	100	20	0	0	1	20	1112753	1112772	0.25	37.4
NC_021054.1	CP023638.1	100	20	0	0	47	66	1112799	1112818	0.25	37.4

Figure 3. Example of BLAST hit-table output. It is observed how for a query it finds two entries for the same subject with SAV CP023640.1. The first entry has q.start = 1 and q.end = 20 (left side of the query) and the second has q.start = 47 and q.end = 66 (right side of the query).

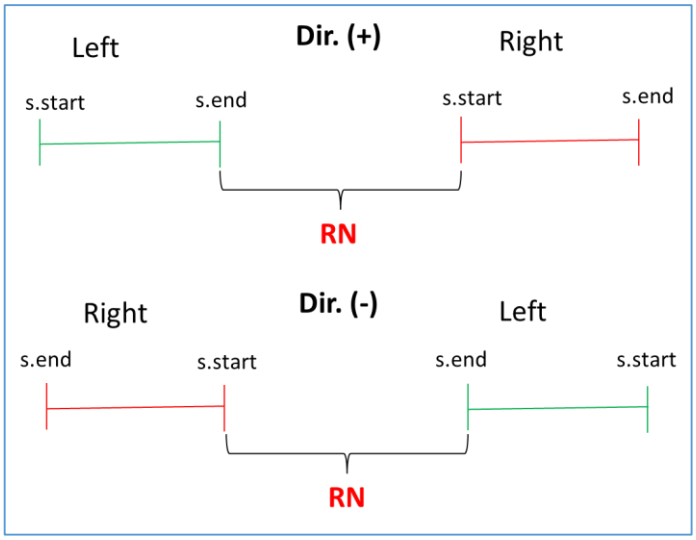
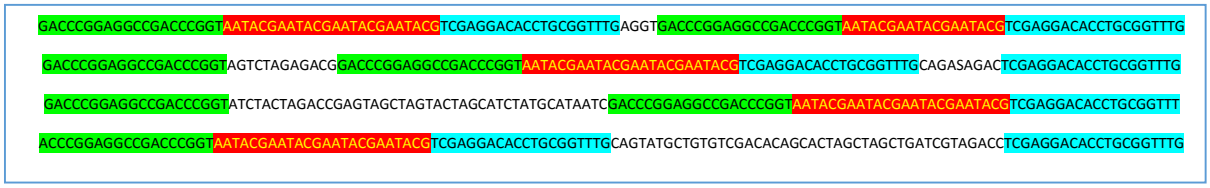


Figure 4. Validation of direction (positive or negative in query-subject comparison) and RN calculation.

4. RN chosen = *min* {RNs found}. This validation is done because doubtful cases can occur due to duplicate regions in the same genome. For example, for the query with the repeated unit AATACG (red zone) between the left flank (green zone) and the right flank (blue zone) the following cases can be found:



5. Only entries where the alignments have a % I (second parameter of the script) and a % of coverage (third parameter of the script, (AL / 20) x 100) greater than 90 are taken into account. These two parameters guarantee that the flanks found are well preserved.

After having the set of RN for all subjects, PSSRextractor generates two result files, one detailed and one generic, with names equal to the hit-table but with suffixes _specific.xls and _generic.xls respectively. The detailed one provides information on each subject processed and the generic provides the information related to the polymorphism for each query, i.e. for each SSR.

Among the information related to the polymorphism that generic report provides, are the following values:

I. min_RN, max_RN and range: These are three columns in the report that respectively mean the minimum RN, the maximum RN and the range (max_RN - min_RN).
II. frequency: Frequency of the allele that shows the original SSR from which the search was made.
III. alleles: Number of alleles found for one *locus* (SSRs with different RNs).
IV. PIC: Polymorphic Information Content $(1 - \sum_i p_i^2)$. This value is also known in other contexts as expected average heterozygosity or Nei genetic diversity, and gives a measure of the probability that, for a single *locus*, a pair of alleles chosen at random in the population are different.

The rest of values shown in generic report come from MIDAS (access_number, pattern, pattern_length, RN, inaccuracy, entropy_5, entropy_3), which provides valuable information when interpreting polymorphic SSRs. For example, the degree of inaccuracy found in SSRs or the compositional entropy of the flanks allow us to know, respectively, the degree of inaccuracy of the repeated tract and how informative the flanks may be in the characterization of the *locus*.

The values of the last column of the generic report (exceptions) show labels that correspond to exceptions in polymorphism validation. There are entries in the generic report where more than one of these labels can appear because the exceptions can occur simultaneous. When all the subject sequences have exceptions, the labels are place. Labels are the following:

i. **D** (degenerated): The subjects have a% I and/or a % coverage <90%.
ii. **NF** (not found): No subject was found in the database with similarity.
iii. **O** (outlier): The number of repeat units between the flanks of the subjects is doubtful because it is very large, being unlikely that there is a microsatellite

between them. The cut-off values established for this exception were mono: 157 bp, di: 364 bp, tri: 109 bp, tetra: 45 bp, penta: 150 bp and hexa: 193 bp. These values were defined after processing all the SSR from more than 200 bacterial genomes, registering their sizes, and establishing the cut-off in 3 times the interquartile range.

iv. **U** (unpair): For the same subject sequence, one edge appears and not the other.

Results and Discussion

Mycobacterium tuberculosis, also known as Koch's bacillus, is a bacterial pathogen, causative agent of tuberculosis, a contagious infection that mainly affects the lungs but can spread to other organs. The prevalence of this disease is among the highest in the world, with approximately 12 million people infected. The bacillus is also object of concern in medical and scientific communities for presenting strains with resistance to multiple antibiotics.

The errors due to sliding in replication that cause microsatellites are normally repaired by three enzymes *mutL, mutS and mutH*, however some genomes such as mycobacteria suffer from this enzymatic system [9]. Due to this, these bacterial species constitute a valuable example to investigate the rates of microsatellite mutations and the existence of regulatory mechanisms that govern them [10].

M. tuberculosis is a pathogen with a genetic diversity emerged from more diverse strains, gaining in virulence mechanisms [11]. An example of medical importance in the expansion of microsatellites is that which occurs in proteins of mycobacteria with pentapeptide-2 (PP2) [12].

In the present study, 23 complete genomes were analyzed (RefSeq sequences), downloaded from the NCBI ftp site (ftp://ftp.ncbi.nlm.nih.gov/refseq/release/bacteria/), belonging to various *M. tuberculosis* isolates. MIDAS detected 4433 SSRs, and from them, SSRMerge script extracted 414 non-redundant *loci* from the species. The polymorphisms were mined from the BLAST server outputs with PSSRextractor script. The outputs of these two scripts can be found in the supplementary files (species_all.mfaa and Y692YMPN014-Alignment_generic_result.xls).

From 414 SSRs, 288 did not show any type of exception, and of these, 104 showed PIC> 0 (36.1%). Figure-5 shows average values obtained for the different types of SSRs classified by the size of the repeat unit (RUS).

RUS	Quantity	RN	inaccuracy	range	frequency	alleles	PIC
1	6	23.17	11.45	8.33	0.18	3.17	0.08
3	78	8.46	8.62	1.15	0.11	2.10	0.07
4	9	3.89	4.86	1.00	0.01	2.00	0.02
5	4	5.50	9.05	1.00	0.06	2.00	0.10
6	7	4.71	7.83	1.29	0.28	2.29	0.03

Figure 5. Average values for some measurements taken from the generic report for M. tuberculosis.

The number of SSRs with 3bp repeat unit is significant. This is because most of the bacterial genomes are coding regions, which have a bias in the use of codons. A single SSR was detected for dinucleotide and was excluded from the result due to exceptions in the polymorphism validation. On average, SSRs of 1bp were more inaccurate and showed greater polymorphism with more alleles and elevated PIC (3.17 and 0.08 respectively). The repetition tracts (RN) were also significantly greater than the rest.

pattern	RUS	RN	inaccuracy	entropy_5	entropy_3	min_RN	max_RN	range	frequency	alleles	PIC
g	1	15	6.67	1.55	1.72	14	23	9	0.989	3	0.021
c	1	20	15	1.88	1.68	20	25	5	0.01	2	0.02
g	1	27	14.8	1.74	1.86	27	35	8	0.01	2	0.02
g	1	23	17.4	1.68	1.24	21	23	2	0.025	2	0.049
g	1	27	7.41	1.72	1.94	14	27	13	0.01	5	0.185
g	1	27	7.41	1.72	1.94	14	27	13	0.01	5	0.185
acc	3	6	5.26	1.8	1.88	5	7	2	0.95	3	0.095
acc	3	4	0	1.69	1.95	4	5	1	0.01	2	0.02
agc	3	4	0	1.8	1.72	4	5	1	0.01	2	0.02
ccg	3	6	10	1.96	1.91	6	7	1	0.01	2	0.02
ccg	3	4	0	1.88	1.93	3	4	1	0.426	2	0.489
ccg	3	6	5.26	1.68	1.95	6	10	4	0.03	5	0.494
ccg	3	5	5.88	1.95	1.82	5	6	1	0.01	2	0.02
ccg	3	4	0	1.93	1.92	4	5	1	0.01	2	0.02
ccg	3	4	0	1.99	1.97	4	5	1	0.01	2	0.02
ccg	3	10	15.6	1.68	1.86	10	11	1	0.01	2	0.02
ccg	3	14	22.7	1.94	1.93	14	15	1	0.01	2	0.02
ccg	3	21	18.8	1.93	1.95	21	22	1	0.01	2	0.02
ccg	3	18	22.8	1.74	1.84	18	19	1	0.01	2	0.019
ccg	3	6	10	1.85	1.99	6	7	1	0.01	2	0.02
ccg	3	5	5.88	1.96	1.99	5	6	1	0.01	2	0.02
cgg	3	4	0	1.97	1.86	4	5	1	0.01	2	0.02
cgg	3	4	0	1.88	1.86	4	5	1	0.01	2	0.02
cgg	3	6	10.5	1.94	1.86	6	7	1	0.5	2	0.5
cgg	3	8	11.1	1.96	1.97	8	9	1	0.01	2	0.02
cgg	3	5	5.56	1.85	1.91	5	6	1	0.012	2	0.024
cgg	3	4	0	1.72	1.74	4	5	1	0.01	2	0.02
cgg	3	4	0	1.96	1.94	4	5	1	0.01	2	0.02
cgg	3	4	0	1.74	1.99	4	5	1	0.01	2	0.02
cgg	3	14	24.4	1.87	1.86	14	15	1	0.01	2	0.02
cgg	3	22	20.6	1.85	1.86	22	23	1	0.5	2	0.5
cgg	3	19	29.3	1.8	1.93	19	21	2	0.01	3	0.058
cgg	3	4	0	1.77	1.85	4	5	1	0.01	2	0.02
cgt	3	4	0	1.84	1.86	4	5	1	0.01	2	0.02
cgt	3	6	10	1.74	1.72	6	7	1	0.01	2	0.02
cgt	3	4	0	1.72	1.76	4	5	1	0.01	2	0.02
cgt	3	4	0	1.99	1.85	4	5	1	0.01	2	0.02
ggt	3	4	0	1.9	1.85	4	5	1	0.01	2	0.02
ggt	3	6	10	1.68	1.86	6	7	1	0.01	2	0.02
ggt	3	4	0	1.8	1.64	4	5	1	0.01	2	0.02
gtt	3	4	0	1.37	1.96	4	5	1	0.01	2	0.02
acc	3	11	11.8	1.68	1.69	11	12	1	0.01	2	0.02
ccg	3	4	0	1.58	1.95	4	5	1	0.02	2	0.039
cgg	3	8	16.7	1.99	1.97	7	8	1	0.941	2	0.112
cgg	3	4	0	1.79	1.93	4	5	1	0.02	2	0.039
ggt	3	7	13	1.8	1.78	7	9	2	0.01	3	0.039
ccg	3	10	15.6	1.88	1.79	10	11	1	0.99	2	0.02
cgg	3	4	0	1.88	1.86	4	5	1	0.01	2	0.02
ccg	3	32	22.7	1.88	1.91	31	32	1	0.088	2	0.16
act	3	6	10	1.37	1.92	5	7	2	0.822	3	0.296
ccg	3	16	16	1.91	1.84	16	17	1	0.01	2	0.02
ccg	3	16	21.6	1.77	1.84	16	17	1	0.015	2	0.029
aac	3	4	0	1.96	1.37	4	5	1	0.01	2	0.02
acc	3	4	0	1.64	1.8	4	5	1	0.01	2	0.02
acc	3	6	10	1.86	1.68	6	7	1	0.01	2	0.02
acc	3	4	0	1.85	1.9	4	5	1	0.01	2	0.02
acg	3	4	0	1.76	1.72	4	5	1	0.01	2	0.02

Figure 6. SSRs from the *M. tuberculosis* with PIC> 0.

pattern	RUS	RN	inaccuracy	entropy_5	entropy_3	min_RN	max_RN	range	frequency	alleles	PIC
acg	3	6	10	1.72	1.74	6	7	1	0.01	2	0.02
acg	3	4	0	1.86	1.84	4	5	1	0.01	2	0.02
ccg	3	4	0	1.84	1.97	4	5	1	0.01	2	0.02
ccg	3	4	0	1.94	1.96	4	5	1	0.01	2	0.02
ccg	3	4	0	1.74	1.72	4	5	1	0.01	2	0.02
ccg	3	5	5.56	1.91	1.85	5	6	1	0.012	2	0.024
ccg	3	4	0	1.91	1.94	4	5	1	0.01	2	0.02
ccg	3	8	11.1	1.97	1.96	8	9	1	0.01	2	0.02
ccg	3	6	10.5	1.86	1.94	6	7	1	0.5	2	0.5
ccg	3	4	0	1.86	1.88	4	5	1	0.01	2	0.02
ccg	3	4	0	1.86	1.97	4	5	1	0.01	2	0.02
cgg	3	5	5.88	1.99	1.96	5	6	1	0.01	2	0.02
cgg	3	6	10	1.99	1.85	6	7	1	0.01	2	0.02
cgg	3	18	22.8	1.84	1.77	18	19	1	0.01	2	0.019
cgg	3	14	25	1.79	1.79	14	15	1	0.01	2	0.02
cgg	3	21	18.8	1.95	1.93	21	22	1	0.01	2	0.02
cgg	3	10	15.6	1.79	1.88	10	11	1	0.99	2	0.02
cgg	3	8	18.5	1.93	1.88	8	9	1	0.005	2	0.01
cgg	3	8	18.5	1.93	1.88	8	9	1	0.005	2	0.01
cgg	3	14	22.7	1.93	1.94	14	15	1	0.01	2	0.02
cgg	3	24	23.4	1.85	2	24	25	1	0.01	2	0.02
cgg	3	10	15.6	1.86	1.68	10	11	1	0.01	2	0.02
cgg	3	4	0	1.93	1.99	4	5	1	0.01	2	0.02
ggt	3	6	5.26	1.88	1.8	5	7	2	0.95	3	0.095
ccg	3	43	26.9	1.74	1.24	43	48	5	0.012	2	0.023
ccg	3	21	25.4	1.62	1.87	20	21	1	0.01	2	0.02
ggt	3	4	0	1.59	1.85	4	5	1	0.5	2	0.5
atcg	4	5	9.09	1.91	1.41	5	6	1	0.01	2	0.02
ccgg	4	5	13.6	1.99	1.76	5	6	1	0.01	2	0.02
ccgg	4	4	5.26	1.87	1.72	4	5	1	0.01	2	0.02
cggg	4	3	0	1.93	1.86	3	4	1	0.01	2	0.02
cggg	4	4	5.26	1.94	1.93	4	5	1	0.01	2	0.02
cggg	4	3	0	1.69	1.93	3	4	1	0.01	2	0.02
cccg	4	4	5.26	1.93	1.94	4	5	1	0.01	2	0.02
cccg	4	3	0	1.86	1.93	3	4	1	0.01	2	0.02
ccgg	4	4	5.26	1.72	1.87	4	5	1	0.01	2	0.02
cccgg	5	6	15.2	1.93	1.92	6	7	1	0.01	2	0.02
cgcgg	5	4	4.55	1.69	1.86	4	5	1	0.01	2	0.02
cgcgg	5	8	11.9	1.74	1.91	7	8	1	0.2	2	0.32
ccgcg	5	4	4.55	1.86	1.69	4	5	1	0.01	2	0.02
aatacg	6	4	0	1.69	1.94	3	5	2	0.97	3	0.058
accagc	6	4	7.41	1.77	1.94	4	5	1	0.01	2	0.02
accgcc	6	7	20	1.93	1.69	7	8	1	0.005	2	0.01
atgtcg	6	3	0	1.85	1.3	3	4	1	0.01	2	0.02
accgcc	6	7	20	1.85	1.69	7	8	1	0.005	2	0.01
attcgt	6	4	0	1.94	1.69	3	5	2	0.97	3	0.058
ctggtg	6	4	7.41	1.94	1.77	4	5	1	0.01	2	0.02

Fig. 6 (cont.) SSRs from the *M. tuberculosis* with PIC> 0.

Figure 6 shows the complete list of 104 extracted SSRs that showed levels of polymorphism. The entire sequences of the markers, including the flanking sequences, the access numbers and positions in the genome can be obtained from the MIDAS outputs.

The methodology described has distinctive features with respect to other *in silico* procedures reported in the literature:

(I) It has been experimentally demonstrated that not all SSRs *loci* show polymorphism. Among other aspects, this is because the *locus* within the analyzed population is not subject to a particular changing dynamic. In this sense, the methodology allows us to select those *loci* that do show polymorphisms in the selected sequence data banks, reducing the costs when it is done experimentally.

(II) Determination of *locus* polymorphism is fully automated. The common procedures to detect polymorphisms, not the experimental ones, use visual inspection of multiple alignments for markers. In this sense, the procedure is ideal for large-scale analysis.

(III) Polymorphism is defined strictly as variation in copy number (PIC> 0), and not as simple insertions or deletions present in markers that do not correspond to the size of the repeat unit.

(IV) The procedure can be done starting from SSRs detection in one sequence, but optionally, it can also start from multiple sequences without the need to assemble a consensus one. This allows a SSRs detection that do not belong to the same *locus* despite being in highly related genomes. SSRMerge script allows to eliminate the redundancy of common *loci* in all sequences.

The methodology is excellent for purely computational analysis of SSR *loci*, applied to evolutionary studies, genotypic identification or functional studies with the genes involved. For experimental analysis using PCR, the methodology provides all the necessary information (sequence access identifier, positions in the genome, flanking sequences, etc.) that allows the design of primers using applications available on internet.

Availability

All the software are available in supplementary material. MIDAS (binary distribution midas_v1.1.exe). SSRMerge and PSSRExtractor are compressed in zip format. Both are *Java NetBeans Projects* for JDK 1.8 platform or above. The binary jar files, for command line execution, are in \dist folder once decompressed.

Conclusions

A methodology for the fully computational detection of microsatellite polymorphic *loci* is described. As example of use, 23 complete genomes belonging to various isolates of *M. tuberculosis* were processed. 4433 SSRs were detected and from them 414 non-redundant *loci* were extracted within the species. The polymorphisms were mined from BLAST outputs and 104 SSRs showed polymorphisms. The methodology is intuitive and comes with software for application. Its main advantage lies in the levels of scaling it allows and the reduction of costs when experimental analyzes are made, allowing preselection of markers showing polymorphism in chosen genomic sequence data banks.

References

1. Li YC, Korol AB, Fahima T, Beiles A, Nevo E. Microsatellites: Genomic distribution, putative functions and mutational mechanisms: A review. Molecular Ecology 2002; 11: 2453–2465.
2. Ellegren, H. Microsatellites: Simple sequences with complex evolution. Nature Reviews. Genetics 2004; 5: 435–445.
3. Xu, J.S.,Wu,Y.T.,Ye,S.J.,Wang,L.,and Feng,Y.Z. SSR primer screening and assessment on pear germplasm resources. J. Central South Univ. Forest.Technol. 2012; 32, 80–85.
4. Hodel et al. Using microsatellites in the 21st century. Applications in Plant Sciences 2016 4(6)
5. Leclercq, S., Rivals, E., Jarne, P. Detecting microsatellites within genomes: significant variation among algorithms. BMC Bioinformatics 2007, 8:125.
6. Grover A, Aishwarya V, Sharma PC. Searching microsatellites in DNA sequences: approaches used and tools developed. Physiol Mol Biol Plants (January–March 2012) 18(1):11–19
7. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J. Mol. Biol. (1990) 215:403-410.
8. Martínez CM. MIDAS: Computer application for the identification of exact and inaccurate microsatellites in genomic sequences. *Revista Cubana de Informática Médica*, Volúmen 18, No. 2 (2018).
9. Fleischmann RD, Alland D, Eisen JA, Carpenter L, White O, Peterson J, DeBoy R, Dodson R, Gwinn M, Haft D, Hickey E, Kolonay JF, Nelson WC, Umayam LA, Ermolaeva M, Salzberg SL, Delcher A, Utterback T, Weidman J, Khouri H, Gill J, Mikula A, Bishai W, Jacobs Jr WR, Venter JC, Fraser CM: Whole-genome comparison of Mycobacterium tuberculosis clinical and laboratory strains. J Bacteriol 2002, 184(19):5479-5490.
10. Sreenu V, Kumar P, Nagaraju J, Nagarajaram H. Microsatellite polymorphism across the M. tuberculosis and M. bovis genomes: Implications on genome evolution and plasticity. BMC Genomics 2006, 7:78.
11. Supply P, Marceau M, Mangenot S, Roche D, Rouanet C, Khanna V, et al. Genomic analysis of smooth tubercle bacilli provides insights into ancestry and pathoadaptation of Mycobacterium tuberculosis. Nat Genet. 2013 Feb; 45(2):172–179. doi: 10.1038/ng.2517 PMID: 23291586.
12. Warholm P, Light S. Identification of a Non-Pentapeptide Region Associated with Rapid Mycobacterial Evolution. PLoS ONE (2016), 11(5): e0154059. doi:10.1371/journal.pone.0154059