Simulación computacional de secuencias de ADN bajo Linux y Perl

por Carlos Andrés Pérez
<caperez/at/usc.edu.co>

Sobre el autor:
Carlos Andrés Pérez es especialista en Simulación Molecular, Doctorando en Biotecnología. Asesor técnico del Grupo de Investigación en Educación Virtual (GIEV). Dirección: Universidad Santiago de Cali, Calle 5ª carrera 62 Campus Pampalinda, Cali – Colombia.

Traducido al español por:
Carlos Andrés Pérez
<caperez/at/usc.edu.co>

Resumen:

En el presente artículo se expone la forma de generar “n” secuencias de ADN con “s” nucleótidos de manera aleatoria utilizando programación en Perl. A diferencia de otros programas en donde las secuencias de ADN se procesan como cadenas de texto aquí se propone trabajarlas como arreglos matriciales. Aunque las diferentes secuencias han sido generadas aleatoriamente, al realizar el análisis estadístico de la fracción promedio de nucleótidos idénticos que ocupan la misma posición de la comparación de secuencias se obtienen valores próximos a 0.25.

Simulación Computacional

La investigación científica esta enfocada al estudio y entendimiento de los procesos de la naturaleza. La investigación se ha abordado mediante la observación experimental y el modelamiento teórico, el cual por mucho tiempo ha estado enmarcado en el planteamiento y desarrollo de ecuaciones. Sin embargo, muchas de estas ecuaciones no se pueden resolver ni analítica ni numéricamente debido a aspectos técnicos o por la imposibilidad de representar un sistema natural.

El vertiginoso desarrollo de la informática permite aproximarnos a los fenómenos naturales mediante programas computacionales que en muchos aspectos pueden remplazar las ecuaciones matemáticas para describir sistemas naturales. Estos algoritmos facilitan la introducción de variables aleatorias lo cual posibilita la recreación de procesos físicos y biológicos en la computadora permitiendo encontrar aspectos determinísticos en las simulaciones, los cuales pueden ser traducidos a leyes.

¿PORQUÉ PERL ?

Perl (Practical Extraction and Report Language),es un lenguaje estructurado que tiene la posibilidad de manejar datos de tipo escalar, lista o arreglos, hashes y ficheros. Así mismo es muy versátil en su aplicabilidad debido a la facilidad que tiene de generar funciones que pueden ser evaluadas en cualquier tipo de datos, adicional a su capacidad de integrarse a sistemas de bases de datos, páginas Web dinámicas y manejo de sistemas operativos como Linux permitiendo asistir al usuario con tareas comunes muy pesadas para el Shell y muy complicadas para codificarlas en algún lenguaje para UNIX.

Perl genera un gran interés como herramienta analítica debido a que su programación puede manejar estructuras muy parecidas a Mathematica que es un lenguaje de manejo simbólico muy potente (http://www.xahlee.org/PerlMathematica_dir/perlMathematica.html), y sus módulos se pueden integrar con librerías de C, GTK, GGI, GGL, lo cual aumenta el rango de programas integrables en un solo lenguaje.

Existen varios programas con gran capacidad de manejo de datos con una estructuración de programación tipo matemática, lo cual facilita la evaluación de expresiones y el lenguaje es menos complejo, sin embargo, son software comerciales, característica que limita la reproducción y mejora de los programas, al mismo tiempo, su capacidad de comunicarse con otros lenguajes es muy limitada.

A diferencia de algunas herramientas de Linux, Perl no limita el tamaño de los datos, por tanto, estos depende de la capacidad de memoria que se disponga al igual que la recursión. El número de tablas hash usadas para arreglos asociativos no esta limitado.

En el presente artículo cada cadena de ADN ha sido tratada como un vector. Aunque se podía haber utilizado el módulo PDL (Perl Data Language, http://pdl.sourceforge.net/WWW−old/index_es.html), el cual esta orientado al tratamiento numérico de datos mediante la manipulación de matrices n−dimensionales, se tuvo problemas en definir cada elemento del vector como un nucleótido debido a que la función “pdl” sólo manipulaba órdenes numéricos, por ello se decidió utilizar las funciones básicas de Perl para la conformación de arreglos, sin embargo esto no quita de la posibilidad de representar cada nucleótido como un número.

Secuencias aleatorias

El programa que se expone permite generar “n” cadenas de ADN conformadas por “s” nucleótidos. Cada cadena presenta la misma longitud y sus componentes son nucleótidos elegidos aleatoriamente. El programa convierte la lista de escalares pasada como parámetro por la función Dumper en una cadena conteniendo código Perl que describe la estructura de datos, esto nos permite visualizar cada vector.

```
#!/usr/bin/perl &ndash;w
# Usa el módulo Data::Dumper
use Data::Dumper;
# En $var1 se almacena el número de secuencias, en $var2 la longitud de la cadena.
print "Introduce el número de secuencias a generar aleatoriamente\n";
$var1 = <STDIN>;
```

```
print "Introduce el número de nucleótidos de la secuencia\n";
$var2 = <STDIN>;
# En la función aleatorio se define el arreglo @ns que contiene los nucleotidos,
# $lon es la variable local de la subrutina que guarda el valor del número de
# nucleótidos, $col es la variable local de la subrutina que guarda el valor del
# número de secuencias.
sub aleatorio {
local @ns = (a,g,c,t);
local $lon = $_[1];
local $col = $_[0];
# Se define un arreglo vacio @a en donde se guardaran los vectores.
@a = ();
# Las variables que indican los vectores y las posiciones de sus componentes.
local $i = 0;
local $u = 0;
while ($u <= $col - 1 && $i <= $lon - 1) {
# $result es la variable que guarda la elección aleatoria de cualquiera de los cuatro-
# nucleótidos.
$result = @ns[int(rand(4))];
# De esta manera se adicionan nucleótidos y se generan las cadenas que los contienen.
$a[$u][$i] = $result;
$i = $i + 1;
if ($i == $lon ) {
$u = $u + 1;
$i = 0;
}
}
return @a;
}
# Se muestra en pantalla cada vector y sus componentes.
print Dumper(&aleatorio($var1,$var2));
# Se define las posiciones y los vectores iniciales que se compararan para determinar-
# si tienen nucleótidos idénticos en las mismas posiciones.
$k = 0;
$count = 0;
$s1 = 0;
$s2 = 1;
while ($s1 <= $col - 2 && $s2 <= $col - 1 && $k <= $lon - 1 ) {
# Si los nucleótidos son idénticos $count aumenta una unidad.
if ($a[$s1][$k] eq $a[$s2][$k]) {
$count = $count + 1;
}
# $k indica los nucleótidos en el vector, $s1 y $s2 son los vectores que se comparan.
# Si el valor de $k es igual al número de nucleótidos es un indicativo
# que la comparación entre dos vectores ha finalizado.
$k = $k + 1;
if($k == $lon ) {
$k = 0;
$s2 = $s2 +1;
}
# Si $s2 es igual a $col, indica que uno de los vectores a sido
# comparado completamente con los demás vectores.
if ($s2 == $col ) {
$k = 0;
$s1 = $s1 + 1;
$s2 = $s1 + 1 ;
}

}
# Mediante este bucle se puede determinar el valor de $p que indica
# el número de comparaciones realizadas.
for ($p = $col - 1, $r = $col -2; $r >= 0 ; $r--){
$p += $r;
}
```

```
# Se muestran en pantalla los resultados.
print "El numero de nucleotidos identicos es: $count\n";
print "El numero de comparaciones es: $p\n";
$y = $count/$p;
# En la variable $cor se guarda el valor de la fracción promedio de
# nucleótidos idénticos que ocupan la misma posición de la
# comparación de secuencias.
$cor = $y/$lon;
print "La fraccion promedio de los nucleotidos identicos que estan en la misma posicion es: $cor"
```

Al estimar el valor de $cor para 10 secuencias de 30, 50, 70 y 90 nucleótidos se obtienen los siguientes resultados respectivamente: 0.2340, 0.26, 0.26031, 0.2661.

Para 40 secuencias de 100, 200, 300 y 500 nucleótidos se obtienen los siguientes valores de $cor respectivamente: 0.2507, 0.2482, 0.2480, 0.2489.

Lo cual nos permite concluir que para “n” número de secuencias de ADN conformadas por “s” nucleótidos y generadas aleatoriamente, presentan una fracción promedio de los nucleótidos idénticos que están en la misma posición de aproximadamente 0.25.

Bibliografia

- http://bioperl.org/
- http://pdl.perl.org/index_es.html
- http://www.unix.org.ua/orelly/perl/prog3/
- http://www.xahlee.org/PerlMathematica_dir/Matica.html
- Ejemplos de manipulación de ficheros en perl creados por el Dr. Antonio J. Pérez Pulido para la Maestría en Bioinformatica de la Universidad Internacional de Andalucía.

- linuxfocus.org April2005/article374

Archivo

Perl source code: ADNaleatorio_pl.txt

2005−08−22, generated by lfparser_pdf version 2.51