Ataques sobre métodos polialfabéticos.

Usando el Método de Kasiski y el Índice de Coincidencia

- Antonio Miguel Pozo Cámara
- Javier Bolivar Valverde

Granada, 22 de noviembre de 2015

Resumen

Con este programa (kasiskiAttack.py) puede realizar ataques sobre textos en castellano cifrados con el cifrado simétrico clásico de Vigenère. Se obtiene el texto descifrado, la posible clave con la que se ha cifrado el texto y si cada caracter de la clave es fiable o no haciendo uso del índice de coincidencia.

Introducción

Se ha desarrollado un proyecto en python para realizar ataques sobre métodos polialfabéticos utilizando el Método de Kasiski y el Índice de Coincidencia.

El Método de Kasiski fue introducido en 1863 por el militar Friedrich W. Kasiski. Este método analiza repeticiones en el texto cifrado para determinar el periodo que se usó para cifrarlo y así conseguir la longitud de la clave. Análogamente el índice de coincidencia (IC en lo que sigue) es un método desarrollado por William Friendman, en 1920. La idea se fundamenta en analizar la variación de las frecuencias relativas de cada letra, respecto a una distribución uniforme. En un texto cifrado, no se cuenta con información suficiente para hallar tal variación puesto que no. Sin embargo, se puede obtener por medio del IC. Al hacerlo, será posible aproximar el periodo de la clave.

Procedimiento

Buscamos en el texto cadenas de caracteres de 3 o más letras que se repitan en el texto. Una vez hecho esto, buscamos la separación que hay entre las dichas cadenas repetidas. El máximo común divisor de estas longitudes entre separaciones nos dará la posible longitud de la clave, pero no las letras que la forman.

Para encontrar la clave, dividimos el texto en “n” cadenas, siendo “n” la longitud de la clave anteriormente calculada. Como cada una de esas cadenas será el resultado de una cifra monoalfabética con un desplazamiento dado por la letra clave, contabilizamos en una tabla las veces que aparecen las letras del alfabeto en cada una de las cadenas y anotamos las de mayor ocurrencia.

La idea es buscar ahora a través de los tres caracteres más frecuentes en cada subcriptograma las posiciones relativas de las letras A, E y O (letras más frecuentes del castellano) que en castellano están separadas por 4 y 11 espacios (módulo tamaño del alfabeto). La letra de la posición que ocupe la letra A (A=0) será entonces la letra correspondiente de la clave.

El Índice de Coincidencia de un texto dado se define como la probabilidad de que, al escoger al azar dos letras cualesquiera del texto, éstas sean idénticas.

Según el libro 'The Codebreakers' de David Kahn, el Índice de Coincidencia del castellano es aproximadamente 0,07750.

Lo que nosotros hacemos es calcular el índice de coincidencia de cada cadena n (todas sus letras están cifradas con la misma letra de la clave). Si el resultado es mayor que el IC español, es que la letra obtenida es "fiable",es decir, es muy posible que corresponda con la clave. Sin embargo, si el resultado es menor que el IC español, decimos que el resultado "no es fiable", por lo que es probable que no coincida con la letra de la clave.

Prerequisitos

Tener instalado python [6]

Documentación del código

decrypt(cipher, key)

Descifra un texto cifrado con Vigenère, recibe como entrada el texto cifrado y la llave y devuelve el texto plano.

findusbs(text, l, lista_ocurrencias):

Busca todas las subcadenas de longitud 'l' en 'text' y almacena la longitud entre dos cadenas iguales en 'lista_ocurrencias'

filter(text):

Excluye los caracteres del texto pasado como argumento que no estén en el alfabeto "alphabet".

imprime(cadena):

Imprime la cadena pasada como argumento.

indexofCoincidence(cadena, i)

Calcula el índice de coincidencia de una cadena "cadena" en la posición "i"

splitText(ctext, mcd)

divide el texto del primer argumento en tantas cadenas como indique el segundo argumento

ktest(text):

Función principal del programa. Desarrolla los conceptos a tratar en el trabajo: Aplica el método de Kasiski y hace uso del Índice de Coincidencia.

- Filtramos el texto pasado como argumento [línea 108].
- Buscamos la distancia entre cadenas repetidas [líneas 114-115]
- Calculamos el máximo común divisor [línea 121]
- Dividir el texto en "x" cadenas, siendo "x" el máximo común divisor calculado en el paso anterior [línea 128].
- Contar las letras que más se repiten en cada una de las subcadenas creadas en el paso anterior, intercambiarlas por las más frecuentes del castellano y calcular el índice de coincidencia [líneas 137-187].
- Calcular las 3 letras de cada subcadena con más posibilidades de ser la clave.
- Imprimir la posible clave [líneas 192-196

Ejemplo de ejecución

```
jnb@jnb-pcbuntu ~/MisRepos/SPSI $ python kasiskiAttack.py
Introducir el texto cifrado:
NLKMARJLNSVVAOLCNNJBREIMAAWWYOMYDEMIVOIZJMDZNZNMUOCILOMBJDJBAENBAINBNSOQPRZALOHMWTMQPOZVDNOZRGVTNLXQNLJMBTVMWLVLAIGTJ
DJYDIZVUOYMBEITJDMQULVZJEGLNSZVUAYZRLGIMOMYDEGWMENMWLVLAIGTNBPMWDZANNGIMRDTUAYWASZZJ
MWLVLAIGT   60
OMYDE      128
LNS        136
JDJ         60
TJD         16
ZVU         24
UAY         44
GIM         32
Posible longitud de la clave = 4

cadena[0]== NANANRAYDVJNULJAANPLWPDRNNBWAJDUBJUJNURMDMWANWNMUA
Primera letra más probable: J
Segunda letra más probable: W
Tercera letra más probable: Z
Indice de coincidencia de cadena[0] 0.130 (0.129795918367)
(FIABLE)

cadena[1]== LRSONEAOEOMZOODEISROTONGLLTLIDIOEDLESALOEELIBDNRAS
Primera letra más probable: A
Segunda letra más probable: L
Tercera letra más probable: O
Indice de coincidencia de cadena[1] 0.089 (0.0889795918367)
(FIABLE)

cadena[2]== KJVLJIWMMIDNCMJNNOZHMZOVXJVVGJZYIMVGZYGMGNVGPZGDYZ
Primera letra más probable: V
Segunda letra más probable: R
Tercera letra más probable: Z
Indice de coincidencia de cadena[2] 0.073 (0.0734693877551)
(NO FIABLE)

cadena[3]== MLVCBMWYIZZMIBBBBQAMQVZTQMMLTYVMTQZLVZIYWMLTMAITWZ
Primera letra más probable: I
Segunda letra más probable: X
Tercera letra más probable: M
Indice de coincidencia de cadena[3] 0.083 (0.0832653061224)
(FIABLE)

POSIBLE CLAVE   ***** JAVI *****
Posible texto plano: ELPERRODESANROQUENOTIENERABOPORQUERAMONRAMIREZSELOHACORTADOTRESTRISTESTIGRESCOMENTRIGOENUNTRIGAL
ELCIELOESTAENLADRILLADOQUIENLODESENLADRILLARAELDESENLADRILLADORQUELODESENLADRILLEBUENDESENLADRILLADORSERA
```

Referencias

[1] http://www.robota.net/index.rsws?seccion=5&submenu=1&articulo=744
[2] http://mikelgarcialarragan.blogspot.com.es/2015/03/criptografia-i.html
[3] https://es.wikipedia.org/wiki/M%C3%A9todo_Kasiski
[4] https://www.youtube.com/watch?v=A7p2ydEPg1k
[5] https://es.scribd.com/doc/4679929/22/El-indice-de-coincidencia-IC
[6] https://wiki.python.org/moin/BeginnersGuide/Download

EL PROYECTO COMPLETO SE ENCUENTRA EN LA SIGUIENTE URL:
https://github.com/javibolibic/SPSI