HASHING USING FOURIER ALGORITHM

Charles Ndung'u

April 2023

1 Introduction To Hashing With Fourier Transforms.

The Fourier transform is a mathematical function that takes a time-based pattern as input and determines the overall cycle offset, rotation speed and strength for every possible cycle in the given pattern. Description we transform the time domain to a frequency domain.

Definations:
Frequency: is a measurement of how often a recurring event such as a wave occurs in a measured amount of time. commonly refered to as number of oscillations per second, which is the Unit time.

Odd Functions: this is a function that submits to the following condition, $f(x) = f(-x)$ to explain this further we shall use practical example.
NB a function is said to be odd iff it's symetrical about the origin after performing a rotation operation.
let our function be $f(-x)$ then we shall subject the function to test by choosing natural numbers thus we shall choose 1,2,3,4,5,6,7,8... Any number we choose and subject it to the function becomes negative that is
$f(-x), f(-1), f(-2), f(-3), f(-4)$.
example of odd functions are :
$sin(x), x^3$...

Even function : A function is said to be even if it's symetric along the $y-axis$, this is achieved by performing a reflection with the mirror line as the $y-axis$. Also this condition applies to even functions $f(-x) = f(x)$
Example:
$f(x^2), cos(x), sin^2x$...
taking the first equation and testing it with integers such as $-1, -3, 3, 5, 7$, this yields $f(-1^2) = f(1^2)$ note they are the same despite the input sign.

Fourier Series
Let's dig a little on the Fourier series. $f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n cos\ nx + \sum_{n=1}^{\infty} b_n sin\ nx$

where,
$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)dx$
$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)cos\ nx\ dx$
$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)sin\ nx\ dxn = 1, 2, 3. \ldots.$

Example 1:
Expand the function $f(x) = e^x$ in the interval $[-\pi, \pi]$ using Fourier series formula.
solution:
Applying the Fourier series formula, we know that $f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n cos\ nx + \sum_{n=1}^{\infty} b_n sin\ nx$
$a_0 = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^x dx$
$= \frac{e^{\pi}-e^{-\pi}}{2\pi}$
$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} e^x cos(nx)dx$
$= 1\pi\frac{e^x}{1+n^2}[\cos(nx) + n\sin(nx)]_{-\pi}^{\pi}$
$= \frac{1}{\pi(1+n^2)}[e^{\pi}(-1)^n) - e^{-\pi}(-1)^n)]$
$b_n = \frac{1}{\pi}\int e^x sin(nx)dx$
$= \frac{e^x}{\pi(1+n^2)}[\sin(nx) - n\cos(nx)]_{-\pi}^{\pi}$
$= \frac{1}{\pi(1+n^2)}[e^{\pi}(-n(-1)^n) - e^{-\pi}(-n)(-1)^n]$
$Thus f(x) = e^x = \frac{e^{\pi}-e^{-\pi}}{2\pi} + \sum_{n=1}^{\infty}[(\frac{(-1)^n(e^{\pi}-e^{-\pi})}{\pi(1+n^2)}cosnx + \frac{-n(-1)^n(e^{\pi}-e^{-\pi})}{\pi(1+n^2)}sinnx]$
Answer:Thus, the Fourier series for
$\frac{e^{\pi}-e^{-\pi}}{2\pi} + \sum_{n=1}^{\infty}\frac{(-1)^n(e^{\pi}-e^{-\pi})}{\pi(1+n^2)}[cosnx - nsinnx]$

Recall:
$sin(n\pi) = 0$
$cos(n\pi) = (-1)^n$
$cos(2n\pi) = 1$
$-cos(n\pi) = (-1)^n + 1$
$sin(2n\pi) = 0$

Hashing:
Suppose we have a sequence of data values x[0], x[1], ..., x[N-1], where N is the length of the sequence. We can represent this sequence as a sum of sinusoidal waves of different frequencies, amplitudes, and phases:
$x[k] = A_0 \cos\left(\frac{2\pi k}{N} + \phi_0\right) + A_1 \cos\left(\frac{2\pi k}{N} + \phi_1\right) + \cdots + A_{N/2} \cos\left(\frac{N\pi k}{N} + \phi_{N/2}\right)$

where A_i and ϕ_i are the amplitude and phase of the ith frequency component, respectively, and N is the length of the signal. The expression represents a discrete-time signal $x[k]$ that is composed of cosine waves of different frequencies and amplitudes. The cosine waves are scaled by the amplitude A_i and shifted in phase by ϕ_i. The frequencies of the cosine waves are integer multiples of the fundamental frequency $f_0 = \frac{1}{N}$, where k is the index of the discrete-time samples of the signal.
This is known as the discrete Fourier transform (DFT) of the sequence.

To compute the DFT, we can use an FFT algorithm, which is an efficient way to compute the DFT. The FFT algorithm has a time complexity of O(N log N), which is much faster than the naive approach of computing the DFT directly, which has a time complexity of $O(N^2)$.

Once we have computed the DFT, we can select a subset of the coefficients to use as the hash value. For example, we might choose the coefficients with the largest magnitudes, or the coefficients with the largest phases. The exact subset chosen will depend on the specific application and requirements.

To generate the hash value from the selected coefficients, we can use a hash function, such as SHA-256. The hash function takes the selected coefficients as input and produces a fixed-length string of characters that uniquely identifies the original sequence.

Example:
To hash the word "charles" using Fourier Hashing, we can follow these steps:

1) Convert the characters to their ASCII values: 99, 104, 97, 114, 108, 101, 115.

2) Compute the discrete Fourier transform (DFT) of the sequence of ASCII values using an FFT algorithm. Let's assume the length of the sequence is N=7 for simplicity.

3) Select a subset of the Fourier coefficients, based on some criteria. For example, we might choose the coefficients with the largest magnitudes. Let's say we choose the five coefficients with the largest magnitudes, which are:
A[0] = 721.9
A[1] = 36.4
A[2] = 11.2
A[4] = 6.4
A[3] = 5.5

4) Convert the selected coefficients back into the time domain using an inverse Fourier transform. This gives us a sequence of complex numbers that represent the original sequence of ASCII values.

5) Round the real parts of the complex numbers to integers. This gives us a sequence of integer values that we can use as the hash value.

6) Finally, we can apply a hash function, such as SHA-256, to the sequence of integer values to produce a fixed-length hash value.

Note that the exact values of the hash will depend on the specific subset of Fourier coefficients chosen and the hash function used. Also, this is just a simple example, and there are many variations and optimizations that can be made to improve the performance and security of the algorithm.

to be continued ...