Aplicación de Estructuras de Datos Probabilistico en

Samuel Leiva[1], Junior Micha[2] , Danilo Blas[3] Joel Janampa[4], Brener Bustillos[5]

***Abstract*—Este articulo matemtico-computacional presenta los resultados obtenidos mediante un proceso de investigacion sobre algunas estructuras de datos alternativas que pueden hacer que el trabajo de analisis de datos sea un ms eficaz, antes de usar tecnicas como un clster de cmputo para ejecutar herramientas distribuidas de cmputo paralelo como por ejemplo Hadoop y Spark que son herramientas correctas ,pero muy costosas.**

I. INTRODUCCIOŃ

Este artculo describe tres estructura de datos probabilsticos : Bloom Fitler , Count-min sketch e HyperLogLog , programaremos cada uno en lenguaje R y comprobaremos su funcionamiento para el conteo y las consultas referidas a elementos dentro de un conjunto.

A. Objetivos

- Usar la estructura bloom Fitler para hacer consultas sobre elementos que puedas pertenecer a la estructura.
- Usar la estructura Count-min sketch.
- Usar la estructura HyperLogLog para calcular el nmero aproximado de elementos en un multiconjunto.

En esta oportunidad usaremos cada una de las tres estructuras . En el primer (Bloom Filter) generaremos un conjuntos de n elementos aleatorios , seguidamente utilizaremos las funciones aadir y query(consultar) para aadir elementos y consultar si algn elemento pertenece al conjunto que hemos generado en el caso de la estructura Count-min sketch , finalmente con la estructura HyperLogLog generaremos un multiconjunto grande el cual calcularemos al cardinalidad de un elemento especfico con una excatitud del 2% usando 1,5kb de memoria.

B. BLOOM FITLER

- Estructura de datos espacio eficiente (Borton Howard Bloom -1970).
- Usado para probar si un elemento es miembro de un conjunto.
- Una consulta arroja “posiblemente en el conjunto” (falso positivo) o “definitivamente no en el conjunto” (falso negativo).
- Se puede agregar elementos al conjunto pero no removerlos.
- Mayores elementos con aadidos , mayores probabilidad de falsos positivos .
- Generalmente , menos de 10 bits por elemento son requeridos para un 1% de probabilidad falso positivo , independientemente del tamao o nmero de elementos del set .

1) **Descripcin de un algoritmo**: Arreglo de m bits , todo en 0 . Tambin debe haber k funciones hash , cada uno de los cuales mapea algn elemento a una de las m posiciones (distribucin aleatoria uniforme).
2) **Para aadir un elemento**: Se alimenta a cada una de las k funciones para obtener k posiciones en el arreglo . Todas esas posiciones se van a 1 .
3) **Para consultar(si es que el elemento est)**: Se alimenta a cada una de las k funciones para obtener k posiciones en el arreglo , si algunas de las posiciones son 0 el elemento definitivamente no se encuentra en el conjunto , si todos son 1, entonces o bien el elemento est o los bits han sidos puestos a 1 cuando fueron insertados.

C. COUNT-MIN SKETCH

– Sirve como una tabla de frecuencia de eventos en una data stream . Usa funciones hash para mapear eventos a frecuencias , pero solo usa espacios sublineales a costa de ”sobrecontar algunos eventos a causa de colisiones (2003-Graham Cormode , S.Muthu Muthokrishnan).
– Esencialmente es lo mismo que Bloom fitler, pero son usados de manera diferente y se ponen el tamao de manera diferente.

1) **Estructura**: El objetivo de count-min sketch es la de consumir un stream de eventos , uno a la vez , y contar la frecuencia de los diferentes tipos de eventos en el stream . En cualquier momento el sketch puede ser consultado para la frecuencia de un particular tipo de evento y regresar un estimado de esta frecuencia dentro de una cierta distancia de la frecuencia real , con una cierta probabilidad.

D. HYPERLOGLOG

– Algoritmo para el conteo distintivo , aproximando el nmero de distintos elementos en un multiconjunto .

Manuscrito creado el 11 de setiembre del 2018; cuya revisioń final sera el 363 de dicimebre .Este trabajo es compatible en formato IEEE y se distribuye bajo el Proyecto LaTeX.El manuscrito puede ser encontrado en los github de los autores

[1] S.Leiva es estudiante de ciencias de la computacion Universidad Nacional de Ingenieria,2015-2021, Lima,Peru. https://github.com/SamuelLeiva
[2] J.Micha es estudiante de pregrado de matematica ,Universidad Nacional de Ingenieria,2016-2022,Lima,Peru. https://github.com/JMicha23
[3] D.Blas es estudiante de ciencias de la computacion ,Universidad Nacional de Ingenieria,2015-2021,Lima,Peru. https://github.com/Sdann26
[4] J.Janampa es estudiante de matematica,Universidad Nacional de Ingenieria,2015-2021,Lima,Peru. https://github.com/JoelJanampaBautista
[5] B.Bustillos es estudiante de matematica,Universidad Nacional de Ingenieria,Universidad Nacional de Ingenieria,Lima,Peru. https://github.com/brenner-08

Es capaz de estimar cardinalidades mayor a 10^9 con una certeza del 2% usando 1,5 kb de memoria.

1) **Algoritmo**: Se basa en la observacin de que las cardinalidades de un multiconjunto de nmeros aleatorios uniformemente distribuidos pueden ser calculados estimando el mximo nmero de ceros en la representacin binaria de cada nmero en el conjunto . Si el mximo nmero de ceros observados es n , un estimado para el nmero de elementos en el conjunto es 2^n .
2) **Operaciones**
   Add: Aade un elenento al conjunto.
   Count: Nos arroja la cardinalidad el conjunto.
   Merge: Para obtener la unin de dos conjuntos.

II. ESTADO DEL ARTE

- **"Theory and Practice of Bloom Filters for Distributed Systems"**
  Este articulo presenta una serie de tcnicas probabilsticas como los Bloom-filters y sus variantes como stable Bloom Filter, Adaptative Bloom Filters, Filter Banks, etc. que se utilizan para reducir el procedimiento de la informacin y los costos de informacin.

- **"An Improved Data Stream Summary: The Count-Min Sketch and its Applications"**
  Este articulo presenta otras aplicaciones de la estructura de datos Count-min Sketch para problemas como encontrar cuartiles, elementos frecuentes, etc.

- **"HyperLogLog in Practice: Algorithmic Engineering of a State of The Art Cardinality Estimation Algorithm"**
  En este articulo se presentan mejoras al algoritmo HyperLogLog reduciendo sus requisitos de memoria y aumentar su precisin para un rango importante de cardinalidades.

- **"Falta UNO"**

III. OBJETIVOS

A. JSJSJS

B. UnDSDFs

- URWER
- AERWE
- DFSDF
- DDD,

C. Ecuaciones

sdajflas ecuanciones

$$\alpha + \beta = \chi \tag{1}$$

Note that the equation is centered using a

D. subitituo

-
-
-

IV. INTRODUCCION

RELLENAR

A. PIRMER

B. LLENAR SI SE DESEA

wiiiiisdshdfksbsdka

Fig. 1. descripcion de la imagen

V. CONCLUSIONS

APPENDIX

rellenear

ACKNOWLEDGMENT

REFERENCES

[1]
[2]
[3]