Un estudio de autenticación en dispositivos moviles

Ricardo Manuel Lazo Vásquez, *Estudiante, UCSP*

***Abstract*—El latente uso de las fuentes de información para las personas asi como las estructuras de datos que requieren almacenar grandes cantidades de datos para poder satisfacer una demanda en las consultas de la comunidad ha ido incrementando a lo largo de los años. Las posibilidades aumentaron cuando estas estructuras podian trasladarse a tarjetas graficas de manera completa, aumentando estas su eficiencia y distribución de recursos de manera notoria. El LSM es una estructura de datos bidimensionales que ayuda a almacenar datos de manera ordenada, en el presente trabajo se sacrificara el orden a cambio de paralelizar aun mas la inserción y busqueda de la estructura, obteniendo como resultado una mejora en velocidad de acuerda a la distribución deseada presentada por este algoritmo.**

***Index Terms*—LSM, GPU, NVIDIA, Algoritmos Paralelos**

I. Introduction

LA deficiencia de una estructura que almacene un diccionario fue superada en 2018 por [1] presentando el primer LSM como mejora de las estructuras tradicionales como el B-tree o el AVL-Tree, teniendo una mejora en su eficiencia y basando la elección en la arquitectura altamente paralelizable de la estructura de datos.
La pronta aparición de estructuras de datos multidimensionales no compite en eficiencia con este algoritmo por ser altamente paralelizable en todas sus operaciones (Actualización, inserción y eliminación) basandose en su estructura de arbol y mezcla mencionada en [2] como una opcion a tomar ante la estructura de dividir y mezclar, y la elegida mezclar y dividir pero en este caso juntar.
Las optimizaciones realizadas a esta estructura son basadas en la taxonomia de Flynn expuesta en [3] y que acoplaron una clasificación de 4 posibles arquitecturas que puede tener un programa. Siendo estas las señaladas en la imagen 1.

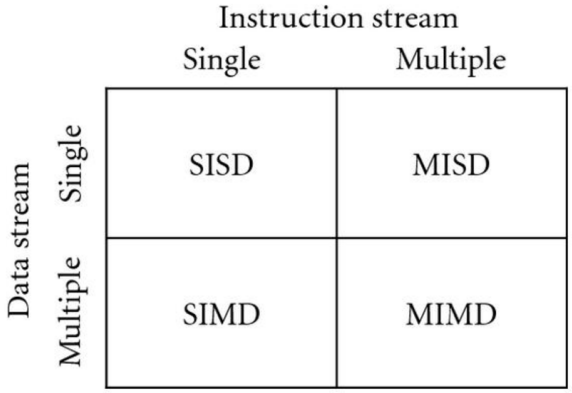

Figure 1. Taxonomia de Flynn

EL presente trabajo trata de paralelizar el LSM en diversas estructuras sacrificando la capacidad sencilla de actualizarse pero optimizando su busqueda, mas orientada a datos mas estaticos y con la posibilidad de amplificar la busqueda a una menor capacidad de memoria con una tarjeta de video menos potente.
El paper esta dividido de la siguiente manera; en la seccion II se hablara del LSM como avance en una GPU, en la sección III se hablara de la propuesta, finalmente en la sección IV se daran las conclusiones.

II. LSM

En [1] se propone el LSM como propuesta para almacenar un diccionario en la GPU teniendo mejoras notorias en su rendimiento por su capacidad de mezclar los datos de un primer nivel con el segundo y delegandolo a otro tambien expuesto en [2].
La arquitectura del LSM esta expuesta en la figura 2 siendo esta un arbol de delegación de tareas donde el primer nivel al encontrarse lleno puede transferir sus datos a otros subniveles inferiores que poseen mayor capacidad.

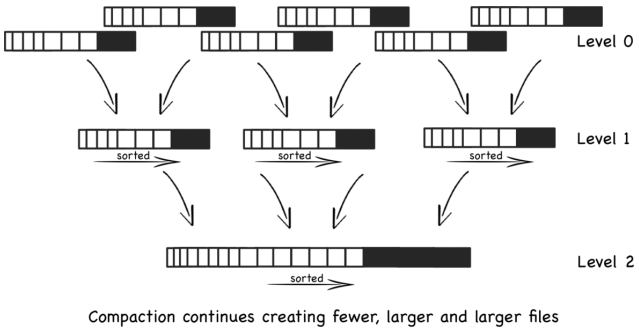

Figure 2. Arquitectura del LSM en 3 niveles

La ventaja de esta arquitectura es la posibilidad de incluir un dato adicional de fecha de actualización que contribuye en las tareas de actualización (Actualizar, eliminación) siendo estas no dependientes de la labor de busqueda, siendo solo actualizados los datos previos en el momento de la mezcla siendo tratados como registros adicionales hasta este momento.
La propuesta en [1] tiene un alto desempeño de $O(logn)$ en sus pruebas de actualización pero con la deficiencia de tiempo constante en las pruebas de inserción en su arreglo ordenado.

III. Propuesta y Resultados

Se presenta una alternativa a [1] tratando de pasar del modelo señalado en [3] tratando de sacrificar las operaciones de actualizacion a cambio de tener una mejora en el momento de insercion sin necesidad de mezclar y actualizar los datos.
Se opto entonces por tener una arquitectura **NVIDIA 940MX**

mucho menor a la usada en [1] para poder trabajar con la cantidad de hilos limite al momento de las operaciones de inserción y busqueda.
Siendo un sacrificio de ambas operaciones se opta por datos estaticos y cuya busqueda sea siempre la misma alrededor del tiempo, trabajando el LSM para una busqueda de la maxima cantidad de hilos con costo de $O(n/h)$ siendo este el tiempo para la inserción y costo de $O(log(n)/h)$ el costo de la busqueda.
El resultado de la investigación se encuentra en https://github.com/TheReverseWasp/Paralelos-CS-UCSP y se puede ubicar en la carpeta "At 2" y se encuentra en fase inicial.

IV. Conclusiones y trabajos futuros

Se abordo la propuesta expuesta en [1] de un LSM con operaciones de actualizacion, se sacrificaron estas operaciones para optimizar la busqueda de datos estaticos, logrando una mejora en los resultados de la propuesta anterior.
En trabajos futuros se espera implementar mejorias adicionales a las expuestas asi como inclusión del ordenamiento y busqueda binaria dentro de los niveles de los diversos LSMs.

References

[1] Saman Ashkiani, Shengren Li, Martin Farach-Colton, Nina Amenta, and John D Owens. Gpu lsm: A dynamic dictionary data structure for the gpu. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 430–440. IEEE, 2018.
[2] Thomas H Cormen. *Algorithms unlocked*. Mit Press, 2013.
[3] Steven Brawer. *Introduction to parallel programming*. Academic Press, 2014.