2023 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON) | 979-8-3503-6953-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/CHILECON60335.2023.10418716

# Accelerated Biometric Fingerprint Search on a multi-GPU environment

Ricardo J. Barrientos
*Laboratorio LITRP, Depto. DCI, Facultad de Ciencias de la Ingeniería, Universidad Católica del Maule*
Talca, Chile
rbarrientos@ucm.cl

Ruber Hernández-García
*Laboratorio LITRP, Facultad de Ciencias de la Ingeniería, Universidad Católica del Maule*
Talca, Chile
rhernandez@ucm.cl

Marco Mora
*Laboratorio LITRP, Depto. DCI, Facultad de Ciencias de la Ingeniería, Universidad Católica del Maule*
Talca, Chile
mmora@ucm.cl

Javier Riquelme Pizarro
*Doctorado en Ingeniería, Universidad Católica del Maule*
Talca, Chile
jriquelmep@ucm.cl

David Laroze
*Instituto de Alta Investigación, CEDENNA, Universidad de Tarapacá, Arica, Chile*
dlarozen@uta.cl

***Abstract*—One of the largest biometric databases in Chile is fingerprints, which is also the most widely used biometric in the national context. In this sense, it is relevant to have efficient algorithms in execution time on this biometrics. In the current context of High Performance Computing, there are co-processors, such as GPUs (Graphic Processing Units) capable of accelerating algorithms efficiently with low monetary cost, low electrical cost and physical space. In this work, we propose the acceleration of a biometric fingerprint algorithm on a multi-GPU hardware platform.**

***Index Terms*—High Performance Computing, Fingerprint, Parallel Computing, Biometrics, GPU**

## I. Introducción

Actualmente, los sistemas biométricos se desarrollan como respuesta a las crecientes demandas de seguridad actuales. La identificación y autenticación de personas a través de procedimientos biométricos son herramientas básicas en la sociedad actual, ya que pueden prevenir fraudes, suplantación, control de acceso al movimiento de seres humanos y acceso no deseado a una oficina, sin utilizar contraseñas, llaves, DNI, tarjetas magnéticas o cualquier otro dispositivo de identificación. Son muy útiles en el comercio electrónico ya que ayudan al consumidor a realizar transacciones seguras y sin molestias. En la situación actual del mundo, donde la movilidad digital y el comercio electrónico van en aumento, estos beneficios son cada vez más importantes. Cabe mencionar que según la literatura técnica, las pérdidas por fraude de identidad en el año 2020 alcanzaron un total de $56 mil millones de dólares (USD) en todo el mundo, lo que equivale a una pérdida de $106.545 dólares (USD) por minuto [1].

Un sistema biométrico tiene como objetivo reconocer la identidad de una persona en función de sus características fisiológicas u otros rasgos de comportamiento. El estado del arte en biometría, establece que un criterio que debe ser satisfecho por cualquier rasgo biométrico es: universalidad, peculiaridad, coleccionabilidad, desempeño, aceptabilidad y elusión. Así, se han propuesto varios tipos de rasgos biométricos, pero en particular, la biometría de huellas digitales presenta ciertas ventajas, tales como, que existe actualmente una gran cantidad de algoritmos que implementa esta biometría, hay una amplia gama de sensores capaces de capturar una huella digital, es fácil de capturar, entre otros.

En el presente artículo, se propone un trabajo preliminar de una búsqueda biométrica acelerada en huellas digitales, basado en un algoritmo multi-GPU, capaz de utilizar 4 GPUs con un comportamiento lineal al escalar en número de GPUs y en tamaño de la base de datos. Hemos logrado realizar un experimento con hasta 4.5 millones de huellas digitales, lo que de acuerdo a nuestro conocimiento, es la experimentación más grande en cuanto a cantidad de huellas digitales realizada en un artículo científico.

## II. Estado del Arte

Las líneas de las huellas dactilares están determinadas por su altura, y mientras más altura tenga una línea de una huella dactilar, más fácil será para el sensor detectarla y dibujarla en una imagen. Las líneas se denominan crestas y los espacios entre ellas, valles. A medida que se analizan las crestas y valles a diferentes grados, éstas presentan algunos patrones que pueden utilizarse para realizar una comparación entre huellas dactilares. Las características más relevantes, ordenadas desde lo más global a lo más local, son las siguientes [2]:

- Puntos singulares: son detectados en el nivel más global. Son puntos alrededor de los cuales se envuelven los patrones de cresta. Hay dos tipos de ellos: loops y deltas, y una huella dactilar puede tener entre cero y cinco puntos singulares.
- Mapa de orientación: pertenece al mismo nivel que los puntos singulares y contiene la dirección de las líneas de huellas dactilares para cada coordenada en la imagen.

979-8-3503-6953-3/23/$31.00 ©2023 IEEE

- Minucias: son los puntos claves de las huellas reflejados en bifurcaciones y terminaciones de las crestas.

Entre los distintos tipos de patrones, las minucias son las características más utilizadas para el reconocimiento de huellas dactilares [2]. Algunos artículos afirman que son las características más confiables para estos fines [3], y que doce minucias perfectamente coincidentes entre dos huellas dactilares pueden asegurar que son los mismos [4]. Sin embargo, en las imágenes de mala calidad, la extracción de características puede implicar dificultades [5].

Una minucia $M_i$ se describe típicamente con cinco parámetros $(x_i, y_i, \theta_i, t_i, q_i)$:

- $(x_i, y_i)$: coordenadas en la imagen.
- $\theta_i$ : orientación o ángulo de la minucia.
- $t_i$ : tipo (terminación de cresta o bifurcación).
- $q_i$ : calidad.

Por lo tanto, una huella digital $F$ con $r$ minucias se puede representar como un vector de minucias $M_1, M_2, ..., M_r$. El número de minucias r es típicamente entre 30 y 100. Así, las minucias pueden almacenarse y manejarse fácilmente en un entorno de cómputo, y la comparación de huellas dactilares puede ser tratada como un cálculo de similitud entre conjuntos de minucias.

El proceso de comparación basado en minucias se puede realizar en tres niveles diferentes [2], [6]:

- Global: Se comparan las minucias de toda la imagen. Este tipo de coincidencia es más sensible a las distorsiones de la imagen, rotaciones y traslaciones, aunque el uso de información de toda la imagen al mismo tiempo proporciona una vista completa de la huella digital. Algunas propuestas se presentan en [52,53].
- Local: Se comparan pequeños grupos de minucias cercanas entre sí. Los problemas debidos a rotaciones y traslaciones se suavizan debido a que el uso de ángulos y coordenadas relativas hace que el método de rotación y traslación sea invariante. El problema de distorsión también se reduce porque las minucias cercanas son menos afectadas por las distorsiones. Sin embargo, no considerar la huella digital como un todo implica una pérdida de información que puede afectar la precisión del algoritmo. Algunos enfoques se describen en [54, 40].
- Híbrido: La mayoría de los algoritmos más robustos usan un enfoque híbrido, combinando ambas filosofías. En primer lugar, una correspondencia local extrae los grupos de minucias más similares de ambas huellas dactilares. Estas minucias se consideran iguales, y luego se realiza una correspondencia global basada en esta correspondencia. Entre las propuestas más relevantes destaca el Algoritmo de Jiang [7], que es el que utilizaremos como base para nuestra propuesta de algoritmo sobre un hardware paralelo. En Jiang, cada minucia se describe con un vector de características que depende de sus minucias vecinas, y los vectores de características de ambas huellas dactilares se comparan en pares. El algoritmo supone que el par más similar corresponde a la misma minucia en ambas huellas digitales, y compara el resto de las minucias utilizando coordenadas y ángulos relativos (evitando los problemas de traslación y rotación).

La Computación de Alto Rendimiento (HPC) es un área de la Ciencia de la Computación, que ha ido tomando cada vez más relevancia en un contexto científico, pero también en el mercado tecnológico, debido al aumento en los tamaños de las bases de datos que las compañías deben manejar actualmente. Esta área de la Ciencia de la Computación estudia y permite la ejecución de múltiples cálculos en un tiempo razonable [8], utilizando sistemas de cómputo masivo [9]. Dado el orden de complejidad de un AFIS (Sistema Automatizado de Identificación de Huellas Dactilares), las herramientas presentes en el área de HPC son recursos prometedores que ya han demostrado reducir el tiempo de identificación en este contexto [10]. Los tiempos de respuesta en tiempo real sólo pueden obtenerse mediante un diseño e implementación eficiente de algoritmos para aprovechar todos los recursos de hardware disponibles.

## III. Algoritmo paralelo propuesto

Este trabajo está basado en el uso de descriptores Jiang [7] para realizar cálculos de similitud entre una huella consulta y las huellas de la Base de Datos. Como en nuestro caso utilizaremos la GPU como un co-procesador para acelerar nuestra propuesta, deberemos distribuir el cómputo que realizará cada hilo de cada bloque de la GPU (CUDA-Block), siguiendo una estrategia de acceso alineado a los datos alojados en memoria de la GPU.

Primero, se estableció una conexión a la base de datos mediante la librería "mysql.h", para poder recuperar los descriptores xyt (obtenido con el comando mindtct del software NBIS [11]) y Jiang. Luego, se solicitó memoria en la GPU para poder alojar los descriptores, y posteriormente copiar esta información desde la memoria de la CPU a la GPU. Debido a las conclusiones de los trabajos previos [12]–[14], se optó por almacenar los datos por columnas en matrices, es decir, cada columna de la matriz representa un descriptor. Posteriormente, se lanza un kernel (función ejecutada en la GPU) con una cantidad de CUDA-Blocks igual a la cantidad de huellas en la Base de Datos (cada CUDA-Block en GPU está formado por un conjunto de $T$ hilos, donde $T$ es la máxima cantidad de minucias en una huella digital de la base de datos).

Debido a que un cálculo de similitud entre dos huellas necesita un proceso de sincronización, y además sabiendo que la sincronización en GPU es solo posible para hilos del mismo CUDA-Block, entonces se decidió que cada CUDA-Block en GPU procese el cálculo de similitud de la huella consulta contra un único elemento de la Base de Datos.

La primera etapa del kernel en GPU es realizar el cálculo local de similitud entre las minucias descritas en el descriptor Jiang. Cada i-ésimo hilo del b-ésimo CUDA-Block realiza un cálculo de similitud local, entre la huella consulta y la i-ésima minucia de la b-ésima huella de la Base de Datos. Cada hilo utiliza un heap (cola de prioridad) almacenado en

una matriz en memoria compartida de la GPU, donde la i-ésima columna de esta matriz almacena los elementos del heap del i-ésimo hilo. Luego de esta etapa, hay una barrera de sincronización, y además en la matriz de heaps se encuentran las menores similitudes. La segunda etapa del kernel en GPU utiliza solamente el primer *warp* (conjunto de 32 hilos) de cada CUDA-Block, para que cada hilo de este warp acceda a los elementos de los heaps anteriores, y utilizando una segunda matriz de heaps en memoria compartida, obtenga allí las 32 similitudes más pequeñas. Luego, la tercera etapa del kernel en GPU utiliza el primer hilo de cada bloque para obtener las distancias más pequeñas finales entre las minucias de la huella consulta y las minucias de la huella de la Base de Datos.

Luego, el kernel continúa obteniendo una segunda similitud por cada minucia (de la huella consulta y de las huellas de la Base de Datos), tomando como referencia los pares de minucias más similares encontrados en el paso previo. Estas nuevas similitudes, se almacenan en dos matrices alojadas en memoria compartida. Luego, hay una barrera de sincronización para los hilos del mismo bloque. Estas nuevas similitudes son utilizadas para obtener una similitud global entre las huellas, definida por:

$$ml(i,j) = \begin{cases} 0,5 + 0,5sl(i,j) & \text{si } |Fg_i^l - Fg_j^T| < Bg, \\ 0 & \text{Otros} \end{cases}$$

donde $sl(i,j)$ representa la similitud local entre la minucia $i$ de la huella consulta y la minucia $j$ de la huella de la Base de Datos. $Fg_i^l$ es la similitud ajustada de la minucia $i$ de la huella consulta, y $Fg_j^T$ es la similitud ajustada de la minucia $j$ de la huella de la Base de Datos. El valor de $Bg$ permite establecer que dos tomas de la misma huella puedan tener una pequeña deformación tolerable, y su valor se definió como $(8\ \pi/6\ \pi/6)^T$. Con estas similitudes $ml$ se calcula un promedio con los valores mayores encontrados entre dos minucias, y este promedio es la similitud final. Para esto último, dos etapas son utilizadas, una para encontrar el mayor valor, y otra para realizar la sumatoria. Este proceso es iterativo, y se puede aumentar el número de iteraciones con la finalidad de obtener una mejor similitud, pero mientras más iteraciones se realicen, mayor será el tiempo de ejecución. Según nuestra experimentación, se decidió por utilizar 5 iteraciones, debido a que este número permite obtener un error (EER) bajo y no aumenta demasiado el tiempo de ejecución.

Al finalizar el kernel en GPU, cada bloque transfiere a memoria de la CPU, la similitud entre una huella de la Base de Datos y la huella consulta. Una vez en CPU, estas similitudes son ordenadas utilizando el algoritmo Quicksort, para conocer el elemento más cercano.

## IV. Resultados Experimentales

### A. Plataformas de Hardware

Para los resultados experimentales, se utilizó un servidor multi-núcleo con 4 GPUs GPU NVIDIA GeForce GTX 1080Ti. El detalle de las GPU se muestra en la Tabla Ia, y el detalle del servidor host se muestra en la Tabla Ib.

TABLE I: Descripción de las plataformas de hardware utilizada en los experimentos.

(a) NVIDIA GPU.

| | |
|---|---|
| Coprocesador | NVIDIA GPU GeForce 1080Ti based on Pascal architecture. |
| Núcleos | 3.584 núcleos, 28 Multiprocesadores con 128 núcleos cada uno. |
| Device Memory | 12GB de memoria (ancho de banda 484 GB/s) |
| Shared Memory | 49 KB |
| Compilador | nvcc version 9.2.148, flags: -03 |

(b) Host Server

| | |
|---|---|
| Procesador | 2xIntel Xeon Gold 6140, 36 núcleos físicos, 25MB Cache, 2.20 GHz, Ivy Bridge. |
| Memoria | 64 GB, 59.7 GB/s |
| Sistema Operativo | GNU Debian System Linux kernel 4.19.181-1 |
| Compilador | gcc version 8.3.0, flags: -03 |

### B. Base de Datos

La base de datos utilizada corresponde a huellas digitales sintéticas creadas con el software SFinge [15], del Laboratorio Biometric Systems de la Universidad de Bolognia. Es el único software que existe actualmente para crear huellas digitales sintéticas a gran escala, y que además ha sido probado frente a bases de datos reales en diferentes competiciones de reconocimiento biométrico de huellas digitales [16]. Este software permite modificar una gran cantidad de parámetros en las huellas digitales a crear, por ejemplo, rotación, traslación, presión del dedo sobre el sensor, ruido en las crestas, ruido en los valles, cicatrices, ruido en las cicatrices, cantidad de semillas generadoras de minucias en una huella, por nombrar algunas. Se ha utilizado la distribución "Default" para generar las huellas digitales, que es la distribución que intenta imitar una base de datos real de huellas digitales. Se eligió como tipo de sensor de captura "óptico", siendo el área de adquisición de 21,1mm×28,4mm y una resolución de las imágenes de 500 DPI. Las imágenes son de extensión WSQ y la semilla de generación fue establecida en 1.

### C. Experimentación

Los resultados obtenidos en cuanto a precisión con son reflejados en la Figura 1, en donde se usaron 180.000 huellas en la base de datos y 180.000 huellas como consultas. Utilizamos la intersección de las medidas FRR (False Rejection Rate) y FAR (False Acceptance Rate) como el valor de Umbral seleccionado (0,3317). El Umbral es el valor utilizado por nuestro algoritmo para indicar si una huella es una coincidencia positiva o no.

El FAR (False Acceptance Rate) obtenido fue de 2,98%, el FRR (False Rejection Rate) 3,19%, y el TPR (True Positive Rate) 96,81%. Todas estas cifras son competitivas al momento de compararse con otros trabajos [17]. Manteniendo el TPR al 96,81%, nuestro sistema logra un alto rendimiento en cuanto a tiempo de ejecución.

La Figura 2 muestra el speed-up (o aceleración) de nuestro sistema. Esta medida es el tiempo del algoritmo secuencial

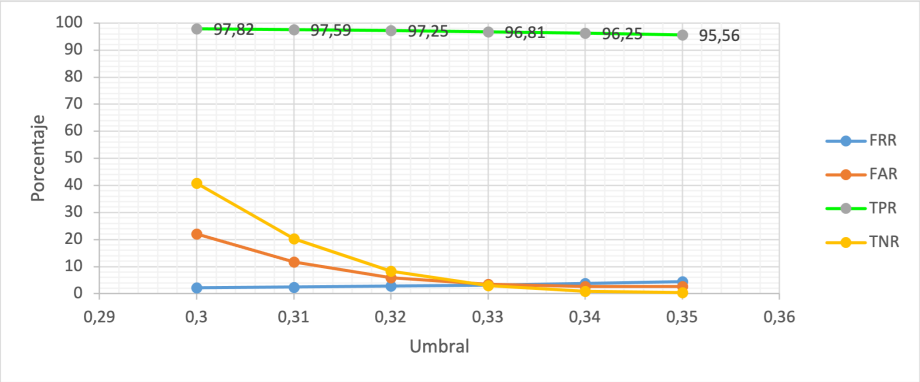


Fig. 1: Valores FRR (False Rejection Rate), FAR (False Acceptance Rate), TNR (True Negative Rate) TPR (True Positive Rate).

dividido por el tiempo de nuestro sistema ejecutado sobre GPUs. El algoritmo secuencial utiliza como recurso de hardware solo 1 núcleo del procesador. Se observa un speed-up de hasta 102,2x con 4 GPUs, es decir, nuestro sistema utilizando 4 GPUs es 102 veces más rápido en tiempo real que el algoritmo secuencial. Nuestro algoritmo propuesto se demora 6,04 segundos en promedio en procesar una consulta, y obtiene un EER (Equal Error Rate) de 1,8%.

En el estado del arte, uno de los últimos trabajos en cuanto a huellas digitales que presenta un muy alto rendimiento utilizando una GPU como co-procesador es el trabajo descrito en [18]. Dado que en este trabajo se utiliza una arquitectura de hardware diferente, es difícil la comparación, pero igualmente la tabla II muestra que [18] es capaz de alcanzar hasta 24.290 MPS (matches per second) entre dos huellas digitales con una GPU GTX 1050Ti, mientras que nuestro trabajo alcanza hasta 93.091 MPS en una GPU GTX 1080ti. Un número mayor en cuanto a MPS era esperable dado que la 1080Ti es una tarjeta gráfica de mayor rendimiento, pero el EER es un valor teórico no dependiente del tiempo de ejecución, y es donde nuestra propuesta obtiene la ventaja.

## V. Conclusiones y futuros trabajos

En el presente artículo, se ha propuesto un trabajo preliminar de un algoritmo acelerado mediante el uso de hardware paralelo single-GPU y multi-GPU, para realizar una búsqueda biométrica mediante huellas digitales. Para esto, se realizó una distribución del cómputo de ambas fases del algoritmo Jiang, junto con una reducción de los resultados parciales usando heaps como estructuras auxiliares en GPU.

De acuerdo a nuestro conocimiento, se ha realizado experimentación con una cantidad de huellas digitales superior a la del estado del arte, llegando a utilizar 4.500.000 de huellas como base de datos. Se ha alcanzado un promedio de respuesta por consulta de 6,04 sobre la base de datos de 4,5 millones de huellas manteniendo un porcentaje de TPR (True Positive Rate) de 96,81%.

Como trabajo futuro se continuará expandiendo la base de datos para lo cual deberemos incluir una arquitectura de hardware multi-nodo, con el objetivo de mantener la escalabilidad del sistema.

## Agradecimientos

Este trabajo fue financiado por: 1) ANID Subdirección de Investigación Aplicada / Concurso IDeA I+D 2023 ID23i10242, Gobierno de Chile, 2) ANID FONDECYT INICIACIÓN 2022 No. 11220693 “End-to-end multi-task learning framework for individuals identification through palm vein patterns”, Gobierno de Chile, 3) ANID FONDECYT REGULAR 2020 No. 1200810 “Very Large Fingerprint Classification Based on a Fast and Distributed Extreme Learning Machine”, Gobierno de Chile.

## Referencias

[1] L. Kessem, “IBM Security: Future of Identity Study,” 2008.
[2] D. Maltoni, D. Maio, A. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*. Springer-Verlag Inc., 2009.
[3] H. Lee and R. Gaensslen, “Advances in fingerprint technology.”
[4] Q. Fang and N. Bhattacharjee, “Incremental fingerprint recognition model for distributed authentication,” in *Proceedings of the International Conference on Security and Management*, 2008, pp. 41–47.
[5] X. Chen, J. Tian, and X. Yang, “A new algorithm for distorted fingerprints matching based on normalized fuzzy similarity measure,” pp. 767–776.
[6] R. Cappelli, M. Ferrara, and D. Maltoni, “Minutia cylinder-code: A new representation and matching technique for fingerprint recognition,” pp. 2128–2141, 2010.
[7] X. Jiang and W. Yau, “Fingerprint minutiae matching based on the local and global structures,” in *15th International Conference on Pattern Recognition, ICPR’00, Barcelona, Spain, September 3-8, 2000*. IEEE Computer Society, 2000, pp. 6038–6041. [Online]. Available: https://doi.org/10.1109/ICPR.2000.906252
[8] R. Robey and Y. Zamora, *Parallel and High Performance Computing*. Manning, 2021.
[9] A. Singh, *Parallel And Distributed Computing*. Independently published, 2022.
[10] H. H. Le, N. H. Nguyen, and T.-T. Nguyen, “Speeding up and enhancing a large-scale fingerprint identification system on gpu,” pp. 147–162, 2018.
[11] C. I. Watson, M. D. Garris, E. Tabassi, C. L. Wilson, R. M. Mccabe, S. Janet, and K. Ko, “User’s guide to NIST biometric image software (NBIS).”
[12] R. J. Barrientos, F. Millaguir, J. L. Sánchez, and E. Arias, “GPU-based exhaustive algorithms processing knn queries,” *The Journal of Supercomputing*, vol. 73, no. 10, pp. 4611–4634, 2017.
[13] *CUDA C++ Best Practices Guide*, 12th ed., NVIDIA Corporation, June 2023.

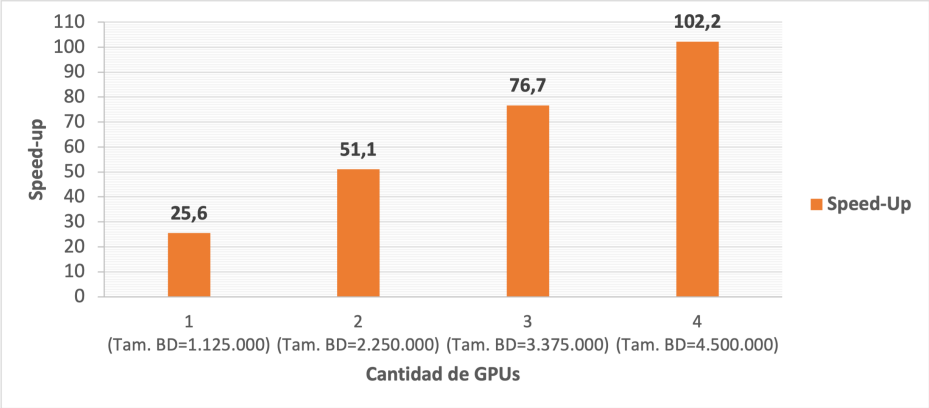


Fig. 2: Speed-up (tiempo algoritmo secuencial/tiempo algoritmo paralelo) al aumentar el número de GPUs junto con el número de elementos de la Base de Datos.

TABLE II: Comparación de nuestro trabajo (Par-Jiang) con el trabajo del estado del arte [18].

| Método | Modelo de GPU | Arquitectura de GPU | Base de Datos | EER | MPS (Match per second) |
|---|---|---|---|---|---|
| Par-Jiang | GeForce GTX 1080Ti | Pascal | Sintética (usando SFinge) | 1,8 | 93.091 |
| [18] | GeForce GTX 1050Ti | Pascal | Sintética (usando SFinge) | 2,91 | 24.290 |


[14] R. J. Barrientos, J. I. Gómez, C. Tenllado, M. P. Matias, and M. Marin, "Range query processing on single and multi GPU environments," *Computers & Electrical Engineering*, vol. 39, no. 8, pp. 2656 – 2668, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045790613001560

[15] "SFinge (synthetic fingerprint generator). Biometric System Laboratory." [Online]. Available: http://biolab.csr.unibo.it

[16] R. Cappelli, M. Ferrara, A. Franco, and D. Maltoni, "Fingerprint verification competition 2006 (fvc2006)," in *Biometric Technology Today*, vol. 15, August 2007, pp. 7–9.

[17] M. Galar, J. Derrac, D. Peralta, I. Triguero, D. Paternain, C. Lopez-Molina, S. García, J. M. Benítez, M. Pagola, E. B. Tartas, H. B. Sola, and F. Herrera, "A survey of fingerprint classification part I: taxonomies on feature extraction methods and learning models," *Knowl. Based Syst.*, vol. 81, pp. 76–97, 2015. [Online]. Available: https://doi.org/10.1016/j.knosys.2015.02.008

[18] A. J. Sanchez-Fernandez, L. F. Romero, D. Peralta, M. A. Medina-Pérez, Y. Saeys, F. Herrera, and S. Tabik, "Asynchronous processing for latent fingerprint identification on heterogeneous cpu-gpu systems," *IEEE Access*, vol. 8, pp. 124 236–124 253, 2020.