Método de externalización a plataformas Cloud Computing de procesamiento especializado GPU. Aplicación a sistemas de sensorización inteligente IoT

Antonio Maciá[1], Higinio Mora[1], Antonio Jimeno[1], y Victor Adsuar[2]

[1]Universidad de Alicante, Departamento de Tecnología Informática y Computación.
[2]Cloud Levante SL. Centro Creación de Empresas Universidad de Alicante. Parque Científico. 03005 Alicante.

Diciembre de 2023

Método de externalización a plataformas Cloud Computing de procesamiento especializado GPU. Aplicación a sistemas de sensorización inteligente IoT

Resumen:

El mundo de la computación está en constante cambio. Las tendencias principales que definen las aplicaciones de hoy en día son la Computación en la nube (Cloud computing) y la aceleración de aplicaciones mediante tarjeta gráfica (GPU). El paradigma del Internet de las Cosas (IoT) se apoya en la computación en la nube para proveer de capacidad de cómputo a dispositivos de baja capacidad de cómputo y energía. Esto se conoce como una arquitectura de externalización de cómputo, donde los dispositivos IoT (sensores, cámaras, bombillas) envían datos a la nube, donde son procesados para extraer información y tomar decisiones, ya que en la nube se dispone de mayor cantidad de recursos de cómputo. Es en este procesamiento de datos donde entra en juego el papel de las tarjetas gráficas. Debido a su arquitectura masivamente paralela, estos dispositivos son capaces de acelerar algoritmos de procesado de datos e inteligencia artificial (IA).

El método descrito debajo permite mejorar el aprovechamiento de los recursos de la GPU en arquitecturas de externalización de cómputo, permitiendo que múltiples aplicaciones aisladas mediante contenedores compartan la GPU de una manera segura. De esta manera se incrementa el uso de la tarjeta, reduciendo los costes necesarios para el funcionamiento de las aplicaciones.

Palabras clave

GPU, Computación en la nube, Arquitectura de externalización de cómputo, Internet de las Cosas (IoT).

1. Introducción

La computación en la nube es una de las tecnologías que están dando forma al mundo actual. Este modelo de computación ha colaborado en el desarrollo de la sociedad de la información y se está utilizando ampliamente en muchas áreas, transformando y creando nuevos modelos de negocios, donde se superan los desafíos de los sistemas tradicionales. En este sentido, las empresas deben hacer uso de esta tecnología para mantenerse competitivas en un mercado globalizado. Los sistemas en la nube ofrecen amplios beneficios a las empresas y usuarios en cuanto al acceso ubicuo a los datos, la gestión de recursos y el procesamiento de información.

Las arquitecturas de nube distribuida, como el Edge y otras capas intermedias, más cercanas al usuario y al lugar donde se recopilan los datos, se vislumbran como una tecnología clave que modelará el futuro de las Tecnologías de la Información y la Comunicación (TIC). Esta infraestructura se está mejorando mediante la adición de dispositivos informáticos específicos [1], como Unidades de Procesamiento Gráfico (GPU) [2]. Su uso es especialmente útil en arquitecturas de procesamiento externo, donde tareas intensivas en computación de campos especializados como CAD/CAM [3, 4] o Inteligencia Artificial (IA) [5] se transfieren a la nube, donde el uso de GPU proporciona un mejor rendimiento [6, 7] y permite que hardware estándar realice estas operaciones. Esto aporta capacidades superiores de cómputo paralelo que abordan nuevas necesidades intensivas en cómputo [8]. Sin embargo, esto conlleva nuevos desafíos, especialmente en la industria manufacturera [9], donde las pequeñas empresas tienden a diseñar estrategias en la nube incorrectas, desaprovechando las ventajas que ofrece este paradigma. Además, las arquitecturas cloud presentan una serie de vectores de ataque que son innatos a la arquitectura [10] y deben ser mitigados adecuadamente.

Los proveedores de servicios en la nube ofrecen GPUs como parte de la infraestructura virtual, generalmente utilizando el método de paso directo (passthrough), donde la GPU completa se utiliza exclusivamente para una única instancia virtual. Esta estrategia, aunque válida para grandes empresas, puede no ser óptima para las más pequeñas, como clientes de CAD/CAM en la industria manufacturera, videojuegos en línea y IA [11, 12], donde las aplicaciones individuales pueden infrautilizar todo el potencial proporcionado por las GPUs. Otro método es la virtualización de GPU (vGPU). Esto se basa en una estrategia de multiplexado temporal para proporcionar acceso concurrente a la GPU a múltiples aplicaciones independientes [13, 14, 15]. La virtualización de GPU a través de este método presenta ineficiencias, ya que la ejecución secuencial de programas que no utilizan todos los recursos de la GPU deja esos recursos adicionales sin usar. Sin embargo, nuevas tecnologías como Nvidia Multi-Instance GPU (MIG) [16] mejoran este escenario permitiendo ejecución concurrente de múltiples usuarios en la GPU. No obstante, esta tecnología solo está disponible un subconjunto concreto de las GPU actuales.

La necesidad de proporcionar un método eficiente para utilizar estas arquitecturas de procesamiento especializado en la nube para cargas de trabajo pequeñas y medianas ha abierto una línea de investigación centrada en optimizar la ejecución de código específico en dispositivos de procesamiento. Siguiendo esta tendencia, el método propuesto utiliza una configuración óptima de arquitectura en la nube que mejora la concurrencia de los dispositivos de procesamiento especializado. Esto permite utilizar mejor los recursos disponibles cuando las aplicaciones no ocupan todos los recursos disponibles en la GPU.

2. Trabajos previos

Esta sección realiza una revisión del estado actual de la investigación y tecnologías para las arquitecturas externalización de procesamiento en la nube, así como del estado actual del uso de GPU y virtualización en la nube, con el fin de establecer los límites del conocimiento en los campos relacionados con el método propuesto.

3. Externalización de procesamiento

El proceso de externalización para descargar parte de la carga de cómputo a la nube permite aumentar las capacidades de dispositivos [17]. En esta área, el paradigma de la informática en la nube móvil fue diseñado inicialmente para ampliar la vida útil de la batería de dispositivos IoT y móviles [18]. Sin embargo, esta tendencia ha evolucionado como una forma de proporcionar un rendimiento mejorado y acceso a recursos informáticos de alto rendimiento [19, 20, 21].

La externalización del procesamiento mediante una arquitectura de nube distribuida presenta algunos desafíos técnicos, ya que las numerosas posibilidades de externalización a infraestructuras edge y en la nube introducen nuevas variables que deben tenerse en cuenta [19]. Uno de los mayores retos consiste en la transmisión de datos entre los dispositivos y el servidor de cómputo, que debe ser reducida al mínimo posible [22]. Un modelo de cómputo distribuido en la nube tiene la ubicación física de los servidores que ejecutan la plataforma como parte de su definición [23]. La distancia a la que se sitúa el servidor de cómputo de los dispositivos impacta directamente en el rendimiento de la arquitectura [24]. El paradigma edge se basa en el uso de plataformas de cómputo en el mismo lugar donde se recopilan los datos, proporcionando procesamiento más barato, con mayor tiempo de respuesta y seguridad de datos al sistema [25].

La evolución del paradigma de la nube se vuelve más importante con la reciente introducción de redes 5G, que proporcionan una mayor conectividad y velocidad de transmisión de datos [26], y la proliferación de nuevas aplicaciones de Internet de las cosas (IoT) [27], como la iluminación inteligente [28] o Internet de vehículos (IoV), cuyo propósito es la externalización de tareas de decodificación de contenido en servidores (edge o nube) para obtener tiempos de respuesta más cortos.

En este sentido, gestionar el proceso de externalización a la nube es uno de los principales desafíos a abordar en la construcción de nuevos software para IoT, dispositivos móviles y otros sistemas distribuidos [29]. Por lo general, el caso más común es el uso de la CPU como la plataforma de ejecución final para la carga de trabajo externalizada.

En la actualidad, hay un número creciente de propuestas que intentan aplicar este paradigma para externalizar cargas de trabajo con necesidades específicas a dispositivos especializados y masivamente paralelos, como las GPUs [30]. Estas propuestas buscan optimizar o acelerar el cálculo de primitivas matemáticas utilizando el alto grado de paralelismo de estos dispositivos [31, 32].

4. Uso de GPU en arquitecturas en la nube

Las GPUs son una herramienta ampliamente utilizada para obtener un mejor rendimiento en cálculos intensivos en campos especializados, ya que proporcionan una alta potencia de cómputo para cálculos geométricos y de matrices, dado que su arquitectura "Single Instruction Multiple Threads"(SIMT) ofrece los mejores resultados con el paralelismo de datos.

Las pequeñas y medianas empresas del sector manufacturero tradicional hacen uso de aplicaciones CAD/CAM e Inteligencia Artificial que utilizan hardware GPU [33].

Las aplicaciones CAD/CAM realizan tareas de computación intensiva, por lo que las GPUs son esenciales para el desarrollo de herramientas de software eficientes que presenten un rendimiento optimo [33]. Algunos ejemplos de aplicaciones CAD/CAM que hacen uso de este tipo de hardware son el diseño colectivo en tiempo real con visualización y renderización de modelos 3D, algoritmos de cómputo para modelos sólidos, análisis y simulaciones en tiempo real, detección de colisiones e intercambio de datos entre diferentes partes [33]. Hay varios trabajos en estos campos. En [34], se presenta un método de optimización que utiliza GPUs para mejorar el cálculo de la trayectoria de herramientas y reducir el tiempo promedio de mecanizado. Hay trabajos que muestran el uso de GPUs para simulación de prendas de vestir [35], detección de colisiones de telas [36], animación [37, 38], y creación y simulación de textiles[39]. En [40], un algoritmo genético paralelizado implementado en la GPU encuentra la mejor orientación del modelo que minimiza el tiempo de construcción y el área de soporte. En [41], un algoritmo genético basado en CUDA logra una orientación óptima de la deposición de piezas.

En Inteligencia Artificial (IA), las tarjetas gráficas se están utilizando tanto para entrenar, como para realizar inferencias sobre el modelo final [42, 43, 44, 44].

A medida que estas aplicaciones utilizan arquitecturas de externalización de procesamiento, surge la necesidad inherente de incluir estos dispositivos especializados en las arquitecturas en la nube. Para satisfacer esta demanda, los proveedores de servicios en la nube han empezado a ofrecer GPUs como parte de sus servicios disponibles. En general, una GPU puede ofrecerse en un servicio en la nube de dos maneras:

- Dedicación exclusiva de la GPU a toda la plataforma (pass-through)
- Virtualización de GPU

El método de pass-through es válido para empresas con grandes necesidades de cómputo capaces de aprovechar todos los recursos de este hardware, pero este método desperdicia la potencia de cómputo de la GPU cuando se utiliza con aplicaciones pequeñas que no llegan a ocupar completamente los recursos de una GPU de altas prestaciones, pero aún se benefician de ellas con un mejor rendimiento.

Por otro lado, la virtualización de GPU tiene como objetivo compartir una GPU física entre diferentes clientes de máquinas virtuales de manera segura, donde los recursos de hardware pueden dividirse y asignarse a diferentes GPU virtuales, según las necesidades específicas de cada aplicación. Tradicionalmente, esto se hace compartiendo la utilización de la tarjeta gráfica, distribuyendo el tiempo entre varias instancias o procesos mediante una repartición de tipo Round Robin [45, 14, 15, 46]. Aunque compartir mediante multiplexado temporal ayuda a reducir los tiempos inactivos de la GPU, esto también puede ser ineficiente, ya que los kernels (programa que se ejecuta en la GPU) pueden no utilizar todos los recursos de la GPU durante su ventana de ejecución. Para lograr una virtualización de GPU más eficiente, NVIDIA lanzó su tecnología MIG para la nueva arquitectura Ampere. Esta tecnología permite la virtualización con soporte de hardware, donde cada instancia virtual de GPU puede ejecutarse simultáneamente de manera segura, con memoria y recursos informáticos asignados por separado, con garantía de calidad de servicio (QoS) y aislamiento de fallos [16]. No obstante, esta tecnología está limitada a un subconjunto específico de GPUs.

5. Arquitectura de externalización de cómputo GPU para Internet de las Cosas

A partir de estudio realizado, se destacan los problemas que trata de afrontar el método propuesto. Estos problemas se centran en el uso de arquitecturas de externalización de cómputo para aplicaciones IoT que requieren el uso de dispositivos GPU. A continuación se muestran los problemas encontrados por este tipo de aplicaciones y las principales contribuciones de método propuesto.

Retos que aborda el método propuesto

a Las aplicaciones IoT que utilizan arquitecturas de externalización de cómputo se benefician del uso de dispositivos GPU. La arquitectura masivamente paralela de este tipo de dispositivos permite incrementar el rendimiento de estas aplicaciones, lo que se reduce en una reducción del tiempo de cómputo de estas aplicaciones, lo que aumenta el tiempo de respuesta de estas aplicaciones y reduce el coste de cómputo.

b La virtualización de las GPUs es de vital importancia para el correcto despliegue y la utilización óptima de estos dispositivos. No obstante, este problema aún no está resuelto. La ejecución secuencial de programas GPU que no lleguen a utilizar todos los recursos de la GPU, tanto utilizando virtualización GPU como sin usarla, dejan estos sin usar durante el tiempo que están usando la GPU.

c Además de la eficiencia, las aplicaciones que se ejecutan en un servidor en la nube necesitan garantías de aislamiento tanto de recursos de cómpu-

to como de los datos. Por esta razón, las aplicaciones que utilizan estos dispositivos en arquitecturas de externalización de cómputo a la nube no implementan técnicas existentes para aumentar la concurrencia de las aplicaciones en la GPU.

Contribuciones del método propuesto

a El método propuesto provee de una arquitectura de externalización de cómputo con recursos GPU a aplicaciones IoT. La arquitectura está pensada para soportar múltiples aplicaciones compartiendo los recursos disponibles de manera eficiente.

b La arquitectura permite que múltiples aplicaciones que necesitan utilizar un dispositivo GPU compartan su uso eficientemente de manera concurrente. Se pueden establecer límites de uso a las diferentes aplicaciones para garantizar un reparto justo de los recursos. Esto permite garantizar una calidad de servicio (QoS) para cada aplicación.

c Se establece un nivel de seguridad de las aplicaciones asumible para el caso de uso de la arquitectura. Los mecanismos de calidad de servicio permiten asegurar un reparto justo de los recursos de cómputo. La seguridad de los datos también está garantizada, ya que la memoria reservada en la GPU por las aplicaciones es privada y no puede ser accedida por otro proceso.

6. Descripción del método multi-teenant de compartición de recursos GPU

La presencia de múltiples aplicaciones con arquitecturas de externalización de cómputo que necesitan de dispositivos GPU hace necesario la inclusión de este tipo de dispositivos en arquitecturas en la nube.

El escenario contemplado en el método propuesto está orientado a proveedores de servicios IoT que utilizan una arquitectura de externalización de cómputo que necesitan GPUs. En este escenario el proveedor debe reservar los recursos en la nube necesarios para el funcionamiento de múltiples aplicaciones.

La arquitectura en la nube para la externalización de cómputo está formada por tres capas: los dispositivos IoT, el edge y la nube. La primera capa consiste en dispositivos inteligentes como sensores, cámaras, etc. Estos dispositivos son los que se encargan de capturar los datos a procesar. El edge es una capa opcional intermedia entre la nube y los dispositivos, cuya función es proporcionar de recursos de cómputo a menor coste y en un lugar geográficamente más cercano a los dispositivos que la nube (generalmente conectado mediante una red local), lo que reduce los costes de operación y el tiempo de respuesta de las aplicaciones. El edge puede estar implementado mediante un servidor de cómputo localizado en el mismo lugar que los dispositivos IoT, o incluso puede consistir en tareas de cómputo que realizan los mismos dispositivos. El edge puede encargarse tanto de

realizar el cómputo completo, y externalizar a la nube solo cuando la cantidad de carga supera los recursos disponibles, o realizar parte de la tarea para reducir la cantidad de trabajo que llega a la nube. Por último, la nube la forman los recursos remotos de cómputo. Estos recursos son alquilados a un proveedor de servicios en la nube mediante un modelo de pago por uso. La figura 1 muestra esta arquitectura.

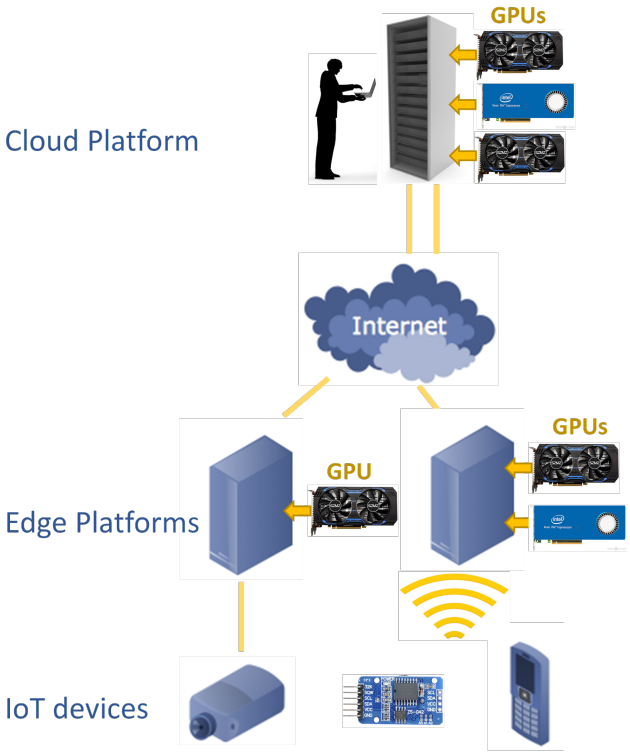

Figura 1: Arquitectura en la nube con edge para la externalización de cómputo con GPUs

Los dispositivos de cómputo específico, como las GPUs, pueden estar situadas tanto en el edge como en la nube. Dentro de los servidores edge y nube se sitúan los servicios de cómputo para los dispositivos IoT. Este método está orientado a el caso en que estas aplicaciones necesitan del uso de GPUs, por lo que compiten por este recurso.

El modelo presentado hace uso de la tecnología de NVIDIA "Multi-Process Service"(MPS) [47]. Esta tecnología permite a varios procesos utilizar la GPU de manera concurrente, de manera similar a como la tecnología de Streams permite concurrencia entre kernels dentro de un mismo proceso. Tiene una arquitectura de tipo cliente/servidor, donde los clientes se comunican con el servidor mediante puertos de tipo UNIX. MPS solo está disponible para sistemas operativos Linux.

Esta tecnología ofrece las siguientes características:

- Reserva de recursos de cómputo y memoria: Se puede limitar el uso de recursos de cómputo para los diferentes clientes de MPS. Las limitaciones se especifican como el porcentaje máximo de hilos activos de la GPU. También se puede limitar el tamaño máximo de la memoria disponible para cada cliente.
- Protección de memoria: Las aplicaciones disponen de su espacio privado de memoria, completamente aislado de procesos externos.
- Contención de errores: Cuando ocurre una excepción crítica, es reportada a todos los procesos que están utilizando la GPU en ese instante, sin indicar que la excepción haya sido provocada por un proceso externo. La GPU se bloquea hasta que todos los procesos que estaban ejecutándose terminan.

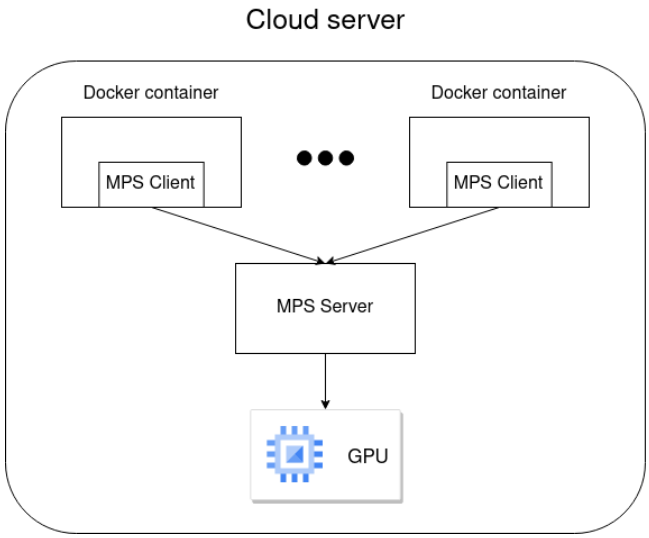

Figura 2: Método de configuración de las aplicaciones en los servidores de cómputo para un uso óptimo de los recursos GPU

En el modelo propuesto, las diferentes aplicaciones encargadas de procesar los datos de los dispositivos IoT se ejecutan dentro de contenedores para aumentar su aislamiento. Estas aplicaciones hacen uso de la GPU disponible en el servidor. Esta GPU es reservada para uso exclusivo al proveedor de la nube. El servidor, host de los contenedores, ejecuta un servidor MPS, al que se conectan las aplicaciones para utilizar la GPU. Esta conexión es posible enlazando el fichero que representa el puerto de tipo UNIX que utiliza MPS (/tmp/nvidia-mps). La figura 2 muestra la configuración interna del servidor.

Con esta arquitectura, las diferentes aplicaciones que se ejecutan en los contenedores pueden compartir la GPU de manera concurrente, incrementando su

uso. Es posible establecer restricciones de Calidad de Servicio, asignando una cantidad específica de recursos de cómputo, especificado como porcentaje de hilos activos disponible, y memoria de video a cada contenedor de la aplicación.

La memoria de video utilizada por los diferentes procesos está aislada. Esto es un elemento esencial, ya que las aplicaciones pueden trabajar con datos sensibles que deben ser protegidos. Sin embargo, esta tecnología no es capaz de aislar a las aplicaciones en el sistema de contención de errores. Una excepción crítica provocada por una de las aplicaciones se propagará al resto de aplicaciones que estén utilizando la tarjeta en ese mismo instante. No obstante, esta relajación del aislamiento es aceptable para este caso de uso, ya que las aplicaciones que se ejecutan pertenecen a un mismo usuario, que puede confiar en las aplicaciones que ejecuta, y aceptar el escenario de propagación de errores.

Referencias

[1] Mora, H., Signes-Pont, M.T., Jimeno-Morenilla, A., Sánchez-Romero, J.L.: High-performance architecture for digital transform processing. The Journal of Supercomputing 75(3), 1336–1349 (May 2018)

[2] Yeung, G., Borowiec, D., Friday, A., Harper, R., Garraghan, P.: Towards {GPU} utilization prediction for cloud deep learning. In: 12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 20) (2020)

[3] Efstathiou, C., Tapoglou, N.: Simulation of spiral bevel gear manufacturing by face hobbing and prediction of the cutting forces using a novel cad-based model. The International Journal of Advanced Manufacturing Technology 122(9-10), 3789–3813 (2022)

[4] Stavropoulos, P., Tzimanis, K., Souflas, T., Bikas, H.: Knowledge-based manufacturability assessment for optimization of additive manufacturing processes based on automated feature recognition from cad models. The International Journal of Advanced Manufacturing Technology 122(2), 993–1007 (2022)

[5] Kounta, C.A.K.A., Arnaud, L., Kamsu-Foguem, B., Tangara, F.: Review of ai-based methods for chatter detection in machining based on bibliometric analysis. The International Journal of Advanced Manufacturing Technology 122(5-6), 2161–2186 (2022)

[6] Rico-Garcia, H., Sanchez-Romero, J.L., Jimeno-Morenilla, A., Migallon-Gomis, H., Mora-Mora, H., Rao, R.V.: Comparison of high performance parallel implementations of tlbo and jaya optimization methods on many-core gpu. IEEE Access 7, 133822–133831 (2019)

[7] Khouzami, N., Michel, F., Incardona, P., Castrillon, J., Sbalzarini, I.F.: Model-based autotuning of discretization methods in numerical simulations of partial differential equations. Journal of Computational Science 57, 101489 (2022)

[8] Sfiligoi, I., Schultz, D., Riedel, B., Wuerthwein, F., Barnet, S., Brik, V.: Demonstrating a pre-exascale, cost-effective multi-cloud environment for scientific computing: Producing a fp32 exaflop hour worth of icecube simulation data in a single workday. In: Practice and Experience in Advanced Research Computing, pp. 85–90. ACM (2020)

[9] Lloret-Climent, M., Nescolarde-Selva, J.A., Mora-Mora, H., Jimeno-Morenilla, A., Alonso-Stenberg, K.: Design of products through the search for the attractor. IEEE Access 7, 60221–60227 (2019)

[10] Candel, J.M.O., Elouali, A., Gimeno, F.J.M., Mora, H.: Cloud vs Serverless Computing: A Security Point of View, p. 1098–1109. Springer International Publishing (Nov 2022)

[11] Chen, H., Lu, M., Ma, Z., Zhang, X., Xu, Y., Shen, Q., Zhang, W.: Learned resolution scaling powered gaming-as-a-service at scale. IEEE Transactions on Multimedia 23, 584–596 (2021)

[12] Han, Y., Guo, D., Cai, W., Wang, X., Leung, V.C.M.: Virtual machine placement optimization in mobile cloud gaming through qoe-oriented resource competition. IEEE Transactions on Cloud Computing 10(3), 2204–2218 (2022)

[13] Peña, A.J., Reaño, C., Silla, F., Mayo, R., Quintana-Ortí, E.S., Duato, J.: A complete and efficient cuda-sharing solution for hpc clusters. Parallel Computing 40(10), 574–588 (2014), https://www.sciencedirect.com/science/article/pii/S0167819114001227

[14] Giunta, G., Montella, R., Agrillo, G., Coviello, G.: A gpgpu transparent virtualization component for high performance computing clouds. In: Euro-Par 2010-Parallel Processing: 16th International Euro-Par Conference, Ischia, Italy, August 31-September 3, 2010, Proceedings, Part I 16. pp. 379–391. Springer (2010)

[15] NVIDIA: Virtual GPU Software User Guide :: NVIDIA Virtual GPU Software Documentation — docs.nvidia.com. https://docs.nvidia.com/grid/13.0/grid-vgpu-user-guide/index.html (2022), [Accessed 16-Sep-2022]

[16] NVIDIA: NVIDIA Multi-Instance GPU (MIG) — nvidia.com. https://www.nvidia.com/es-es/technologies/multi-instance-gpu/ (2022), [Accessed 16-Sep-2022]

[17] Mora, H., Pujol, F.A., Ramírez, T., Jimeno-Morenilla, A., Szymanski, J.: Network-assisted processing of advanced iot applications: challenges and proof-of-concept application. Cluster Computing (Jun 2023)

[18] Waheed, A., Shah, M.A., Mohsin, S.M., Khan, A., Maple, C., Aslam, S., Shamshirband, S.: A comprehensive review of computing paradigms,

enabling computation offloading and task execution in vehicular networks. IEEE Access 10, 3580–3600 (2022)

[19] Mora, H., Mora Gimeno, F.J., Signes-Pont, M.T., Volckaert, B.: Multilayer architecture model for mobile cloud computing paradigm. Complexity 2019, 1–13 (2019)

[20] Dash, S., Ahmad, M., Iqbal, T.: Mobile cloud computing: a green perspective. In: Intelligent Systems: Proceedings of ICMIB 2020. pp. 523–533. Springer (2021)

[21] Mora Mora, H., Gil, D., Colom Lopez, J.F., Signes Pont, M.T., et al.: Flexible framework for real-time embedded systems based on mobile cloud computing paradigm. Mobile information systems 2015 (2015)

[22] Elouali, A., Mora Mora, H., Mora-Gimeno, F.J.: Data transmission reduction formalization for cloud offloading-based iot systems. Journal of Cloud Computing 12(1), 1–12 (2023)

[23] Yuan, H., Zhou, M.: Profit-maximized collaborative computation offloading and resource allocation in distributed cloud and edge computing systems. IEEE Transactions on Automation Science and Engineering 18(3), 1277–1287 (2021)

[24] Mora, H., Mora-Gimeno, F., Jimeno-Morenilla, A., Macia-Lillo, A., Elouali, A.: Serverless computing at the edge for aiot applications. In: 2022 International Conference on Artificial Intelligence of Things (ICAIoT). IEEE (Dec 2022), http://dx.doi.org/10.1109/ICAIoT57170.2022.10121879

[25] Qiu, T., Chi, J., Zhou, X., Ning, Z., Atiquzzaman, M., Wu, D.O.: Edge computing in industrial internet of things: Architecture, advances and challenges. IEEE Communications Surveys & Tutorials 22(4), 2462–2488 (2020)

[26] Zheng, G., Zhang, H., Li, Y., Xi, L.: 5g network-oriented hierarchical distributed cloud computing system resource optimization scheduling and allocation. Computer Communications 164, 88–99 (Dec 2020)

[27] Etemadi, M., Ghobaei-Arani, M., Shahidinejad, A.: Resource provisioning for iot services in the fog computing environment: An autonomic approach. Computer Communications 161, 109–131 (Sep 2020)

[28] Mora, H., Peral, J., Ferrandez, A., Gil, D., Szymanski, J.: Distributed architectures for intensive urban computing: a case study on smart lighting for sustainable cities. IEEE Access 7, 58449–58465 (2019)

[29] Li, M.: Computation offloading and task scheduling on network edge. University of Waterloo (2021)

[30] Ribes, V.S., Mora, H., Sobecki, A., Gimeno, F.J.M.: Mobile cloud computing architecture for massively parallelizable geometric computation. Computers in Industry 123, 103336 (2020)

[31] Martinez-Noriega, E.J., Yazaki, S., Narumi, T.: Cuda offloading for energy-efficient and high-frame-rate simulations using tablets. Concurrency and Computation: Practice and Experience 33(2), e5488 (2021)

[32] Tsog, N., Mubeen, S., Bruhn, F., Behnam, M., Sjödin, M.: Offloading accelerator-intensive workloads in cpu-gpu heterogeneous processors. In: 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). pp. 1–8. IEEE (2021)

[33] Jimeno-Morenilla, A., Azariadis, P., Molina-Carmona, R., Kyratzi, S., Moulianitis, V.: Technology enablers for the implementation of industry 4.0 to traditional manufacturing sectors: A review. Computers in Industry 125, 103390 (Feb 2021)

[34] Morell-Giménez, V., Jimeno-Morenilla, A., García-Rodríguez, J.: Efficient tool path computation using multi-core gpus. Computers in Industry 64(1), 50–56 (Jan 2013)

[35] Tang, M., Tong, R., Narain, R., Meng, C., Manocha, D.: A gpu-based streaming algorithm for high-resolution cloth simulation. In: Computer Graphics Forum. vol. 32, pp. 21–30. Wiley Online Library (2013)

[36] Vassilev, T.I.: Garment simulation and collision detection on a mobile device. International Journal of Mobile Computing and Multimedia Communications (IJMCMC) 7(3), 1–15 (2016)

[37] Xue, C., Dong, H., Zhang, M., Pan, Z.: Real-time simulation on virtual dressing based on virtual human body model. Journal of System Simulation 29(11), 2847–2855 (2020)

[38] Hui, Z., Zhen, L., Yanjie, C., et al.: Real-time collision detection method for fluid and cloth. J Comput Aided Des Graph 30(4), 602–610 (2018)

[39] Leaf, J., Wu, R., Schweickart, E., James, D.L., Marschner, S.: Interactive design of periodic yarn-level cloth patterns. ACM Transactions on Graphics 37(6), 1–15 (Dec 2018)

[40] Li, Z., Xiong, G., Zhang, X., Shen, Z., Luo, C., Shang, X., Dong, X., Bian, G.B., Wang, X., Wang, F.Y.: A gpu based parallel genetic algorithm for the orientation optimization problem in 3d printing. In: 2019 International Conference on Robotics and Automation (ICRA). IEEE (May 2019)

[41] Huang, R., Dai, N., Li, D., Cheng, X., Liu, H., Sun, D.: Parallel non-dominated sorting genetic algorithm-ii for optimal part deposition orientation in additive manufacturing based on functional features. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 232(19), 3384–3395 (Oct 2017)

[42] Talib, M.A., Majzoub, S., Nasir, Q., Jamal, D.: A systematic literature review on hardware implementation of artificial intelligence algorithms. The Journal of Supercomputing 77, 1897–1938 (2021)

[43] Jimeno-Morenilla, A., Sanchez-Romero, J.L., Migallon, H., Mora-Mora, H.: Jaya optimization algorithm with gpu acceleration. The Journal of Supercomputing 75, 1094–1106 (2019)

[44] Chen, Z., Wang, J., He, H., Huang, X.: A fast deep learning system using gpu. In: 2014 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE (Jun 2014)

[45] Peña, A.J., Reaño, C., Silla, F., Mayo, R., Quintana-Ortí, E.S., Duato, J.: A complete and efficient cuda-sharing solution for hpc clusters. Parallel Computing 40(10), 574–588 (Dec 2014)

[46] vikancha MSFT: Serie NVv4 - Azure Virtual Machines — learn.microsoft.com. https://learn.microsoft.com/es-es/azure/virtual-machines/nvv4-series (2022), [Accessed 26-Sep-2022]

[47] Corporation, N.: Multi-Process Service :: GPU Deployment and Management Documentation — docs.nvidia.com. https://docs.nvidia.com/deploy/mps/index.html (2021), [Accessed 15-Sep-2022]