Paralelización de algoritmos y evaluación de rendimiento en plataformas de cómputo de altas prestaciones. Aplicaciones

Marcelo Naiouf[(1)], Armando De Giusti[(1)(2)], Laura De Giusti[(1)(3)], Franco Chichizola[(1)], Victoria Sanz [(1)(3)], Adrián Pousa[(1)], Enzo Rucci[(1) (3)], María José Basgall[(1)(2)], Mariano Sánchez[(1)], Manuel Costanzo [(1)], Emmanuel Frati[(1)(4)], Adriana Gaudiani[(5)], Sergio Leandro Calderón [(1)]

[1]Instituto de Investigación en Informática LIDI (III-LIDI)
Facultad de Informática – Universidad Nacional de La Plata
50 y 115, La Plata, Buenos Aires
Comisión de Investigaciones Científicas de la Pcia. de Buenos Aires (CIC)
526 e/ 10 y 11 La Plata Buenos Aires

[2] CONICET – Consejo Nacional de Investigaciones Científicas y Técnicas
[3]CIC – Comisión de Investigación Científica de la Provincia de Buenos Aires
[4] Universidad Nacional de Chilecito
[5] Universidad Nacional de General Sarmiento
{mnaiouf, degiusti, ldgiusti, francoch, vsanz, apousa, erucci, mjbasgall, msanchez, mcostanzo, fefrati, scalderon}@lidi.info.unlp.edu.ar, agaudi@ungs.edu.ar

RESUMEN

El eje central de la línea de I/D es investigar en temas de cómputo paralelo y distribuido de alto desempeño, tanto en lo referido a los fundamentos como a la construcción, evaluación y optimización de las aplicaciones en arquitecturas multiprocesador. Se aplican los conceptos en problemas numéricos y no numéricos de cómputo intensivo y/o sobre grandes volúmenes de datos con el fin de obtener soluciones de alto rendimiento.
También incluye la construcción de ambientes para la enseñanza de la programación concurrente y paralela.
En la dirección de tesis de postgrado existe colaboración con el grupo HPC4EAS (High Performance Computing for Efficient Applications and Simulation) del Dpto. de Arquitectura de Computadores y Sistemas Operativos de la Universidad Autónoma de Barcelona; con el Departamento de Arquitectura de Computadores y Automática de la Universidad Complutense de Madrid; y con el grupo Soft Computing and Intelligent Information Systems (SCI2S) de la Universidad de Granada, entre otros.
Palabras clave: Cómputo paralelo y distribuido de altas prestaciones. Algoritmos paralelos y distribuidos. Arquitecturas multiprocesador. Ambientes de enseñanza.

CONTEXTO

La línea de I/D que se presenta es parte del Proyecto "Computación de Alto Desempeño y Distribuida: Arquitecturas, Algoritmos, Tecnologías y Aplicaciones en HPC, Fog-Edge-Cloud, Big Data, Robótica, y Tiempo Real" del III-LIDI acreditado por el Ministerio de Educación, y de proyectos acreditados y subsidiados por la Facultad de Informática de la UNLP. Además, existe cooperación con Universidades de Argentina, Latinoamérica y Europa a través de proyectos acreditados. Asimismo, el III-LIDI forma parte del Sistema Nacional de Cómputo de Alto Desempeño (SNCAD).

1. INTRODUCCIÓN

El área de cómputo de altas prestaciones (HPC, High-Performance Computing) es clave dentro de las Ciencias de la Computación, debido al creciente interés por el desarrollo de soluciones a problemas con alta demanda computacional y de almacenamiento, produciendo transformaciones profundas en las líneas de I/D [1].
El rendimiento en este caso está relacionado con dos aspectos: las arquitecturas de soporte y los algoritmos que hacen uso de las mismas, y el desafío se centra en cómo aprovechar las prestaciones obtenidas a partir de la evolución de las arquitecturas físicas. En esta línea la mayor importancia está en los algoritmos paralelos y en los métodos utilizados para su construcción y análisis a fin de optimizarlos.
Uno de los cambios de mayor impacto ha sido el uso de manera masiva de procesadores con más de un núcleo (*multicore*), produciendo plataformas distribuidas híbridas (memoria compartida y distribuida) y generando la necesidad de desarrollar sistemas operativos, lenguajes y algoritmos que las usen adecuadamente. También creció la incorporación de placas aceleradoras a los sistemas multicore constituyendo plataformas paralelas de memoria compartida con paradigma de programación propio asociado como pueden ser las unidades de procesamiento gráfico (GPU, Graphic Processing Unit) de NVIDIA y AMD, los coprocesadores Xeon Phi de Intel [2] o los aceleradores basados en circuitos integrados reconfigurables (FPGAs, Field Programmable Gate Array) [3]. En la actualidad, se comercializan placas de bajo costo como Raspberry PI [4], Odroid [5], NVIDIA Jetson [6], entre otras. Estas placas, con arquitectura similar a la que podemos encontrar en dispositivos móviles, poseen múltiples núcleos de baja complejidad y en algunos casos son procesadores multicore asimétricos (AMPs) con el mismo repertorio de instrucciones. Es de interés estudiar como explotar el paralelismo en estos dispositivos para mejorar el rendimiento y/o consumo energético de las aplicaciones [7]. Asimismo, los entornos

de computación cloud introducen un nuevo foco desde el punto de vista del HPC, brindando un soporte "a medida" sin la necesidad de adquirir el hardware.

La creación de algoritmos paralelos en arquitecturas multiprocesador no es un proceso directo [8]. El costo puede ser alto en términos del esfuerzo de programación y el manejo de la concurrencia adquiere un rol central en el desarrollo. Si bien en las primeras etapas el diseñador de una aplicación paralela puede abstraerse de la máquina sobre la que ejecutará el algoritmo, para obtener buen rendimiento debe tenerse en cuenta la plataforma de destino. En las máquinas multiprocesador, se deben identificar las capacidades de procesamiento, interconexión, sincronización y escalabilidad. La caracterización y estudio de rendimiento del sistema de comunicaciones es de interés para la predicción y optimización de performance, así como la homogeneidad o heterogeneidad de los procesadores [9].

Muchos problemas algorítmicos se vieron impactados por los multicore y clusters de multicore. El desarrollo de algoritmos que aprovechen adecuadamente las arquitecturas,motiva el estudio de performance en sistemas híbridos. Es necesario estudiar la utilización de lenguajes y bibliotecas ya que aún no se cuenta con un standard, aunque puede mencionarse el uso de los tradicionales MPI, OpenMP y Pthreads [10][11] o los más recientemente explorados UPC, Chapel y Titanium del modelo PGAS [12].

La combinación de arquitecturas de múltiples núcleos con aceleradores dio lugar a plataformas híbridas con diferentes características. Más allá del acelerador utilizado, la programación de estas plataformas representa un desafío. Para lograr aplicaciones de alto rendimiento, los programadores enfrentan dificultades como: estudiar características específicas de cada arquitectura y aplicar técnicas de programación y optimización particulares de cada una, lograr un balance de carga adecuado entre los dispositivos de procesamiento y afrontar la ausencia de estándares para este tipo de sistemas.

Por otra parte, los avances en las tecnologías de virtualización han llevado a que Cloud Computing sea una alternativa a los tradicionales sistemas de cluster [13]. El uso de cloud para HPC presenta desafíos atractivos, brindando un entorno reconfigurable dinámicamente sin la necesidad de adquirir hardware, y es una excelente plataforma para testear escalabilidad de algoritmos.

Métricas de evaluación del rendimiento y balance de carga

La diversidad de opciones vuelve complejo el análisis de performance de los Sistemas Paralelos, ya que los ejes sobre los cuales pueden compararse dos sistemas son varios. Existe un gran número de métricas para evaluar el rendimiento, siendo las tradicionales: tiempo de ejecución, speedup, eficiencia. Por su parte, la *escalabilidad* permite capturar características de un algoritmo paralelo y la arquitectura en que se lo implementa. Posibilita testear la performance de un programa sobre pocos procesadores y predecirla en un número mayor, así como caracterizar la cantidad de paralelismo inherente en un algoritmo.

Un aspecto de interés que se ha sumado como métrica, a partir de las plataformas con gran cantidad de procesadores, es el del *consumo* y la *eficiencia energética* [14]. Muchos esfuerzos están orientados a tratar el consumo como eje de I/D, como métrica de evaluación, y también a la necesidad de metodologías para medirlo.

El objetivo principal del cómputo paralelo es reducir el tiempo de ejecución haciendo uso eficiente de los recursos. El *balance de carga* es un aspecto central y consiste en, dado un conjunto de tareas que comprenden un algoritmo y un conjunto de procesadores, encontrar el mapeo (asignación) de tareas a procesadores tal que cada una tenga una cantidad de trabajo que demande aproximadamente el mismo tiempo, y esto es más complejo si hay heterogeneidad. Dado que el problema general de mapping es *NP*-completo, pueden usarse enfoques que dan soluciones subóptimas aceptables. Las técnicas de planificación a nivel micro (dentro de cada procesador) y macro (en un cluster) deben ser capaces de obtener buen balance de carga. Existen técnicas estáticas y dinámicas cuyo uso depende del conocimiento que se tenga sobre las tareas de la aplicación.

2. LÍNEAS DE INVESTIGACIÓN, DESARROLLO E INNOVACIÓN

- Investigar en temas de cómputo paralelo y distribuido de alto desempeño, en lo referido a los fundamentos y a la construcción y evaluación de las aplicaciones. Esto incluye los problemas de software asociados con el uso de arquitecturas multiprocesador:
  - Lenguajes, modelos y paradigmas de programación paralela (puros e híbridos a distintos niveles).
  - Asignación de procesos a procesadores optimizando el balance de la carga de procesamiento.
  - Métricas de evaluación de complejidad y rendimiento: speedup, eficiencia, escalabilidad, consumo energético, costo de programación.
- Construir, evaluar y optimizar soluciones utilizando algoritmos concurrentes, paralelos y distribuidos sobre diferentes plataformas de software y arquitecturas con múltiples procesadores:
  - Arquitecturas de trabajo homogéneas, heterogéneas e híbridas: multicores, clusters, GPU, Xeon Phi, FPGA, placas de bajo costo y entornos cloud.
  - Aplicar los conceptos en problemas numéricos y no numéricos de cómputo intensivo y/o sobre grandes volúmenes de datos (aplicaciones científicas, búsquedas, simulaciones, imágenes, realidad virtual y aumentada, bioinformática, big data, n-body).

- Analizar y desarrollar ambientes para la enseñanza de programación concurrente y paralela.
  - Caracterizar diferentes modelos de arquitecturas paralelas.
  - Representar distintos modelos de comunicación/sincronización.
  - Definir métricas de evaluación de rendimiento y eficiencia energética.

3. RESULTADOS OBTENIDOS/ESPERADOS

- Desarrollar y optimizar algoritmos paralelos sobre diferentes modelos de arquitectura. En particular, en aplicaciones numéricas y no numéricas de cómputo intensivo y tratamiento de grandes volúmenes de datos.
- Estudiar y comparar los lenguajes sobre las plataformas multiprocesador para diferentes modelos de interacción entre procesos.
- Investigar la paralelización en plataformas que combinan clusters, multicore y aceleradores. Compararestrategias de distribución de trabajo teniendo en cuenta las diferencias en potencias de cómputo y comunicación, dependencia de datos y memoria requerida.
- Evaluar la performance (speedup, eficiencia, escalabilidad, consumo energético) de las soluciones propuestas. Analizar el rendimiento de soluciones paralelas a problemas con diferentes características (dependencia de datos, relación cómputo / comunicación, memoria requerida).
- Mejorar y adecuar las técnicas disponibles para el balance de carga (estático y dinámico) entre procesos a las arquitecturas consideradas.

En este marco, pueden mencionarse los siguientes resultados:

- Para la experimentación se han utilizado y analizado diferentes arquitecturas homogéneas o heterogéneas, incluyendo multicores, cluster de multicores (con 128 núcleos), GPU y cluster de GPU, Xeon Phi y FPGA.
- Se experimentó la paralelización en arquitecturas híbridas, con el objetivo de estudiar el impacto del mapeo de datos y procesos, así como de los lenguajes y librerías.
- Respecto de las aplicaciones y temas estudiados, se trabajó fundamentalmente con los siguientes problemas:

➢ **Aceleración de aplicaciones con cómputo colaborativo CPU-GPU.** Las computadoras comerciales actuales incluyen decenas de cores y al menos una GPU. El uso de ambas unidades de procesamiento de forma colaborativa puede mejorar significativamente el rendimiento de una aplicación. Sin embargo, esto supone un desafío para los programadores ya que dichas unidades difieren en arquitectura, modelo de programación y rendimiento. En [15] se propuso un modelo híbrido para estructurar código a ser ejecutado sobre un sistema heterogéneo con múltiples cores y 1 GPU (utilizando todos los recursos disponibles). Utilizando este modelo se desarrolló un algoritmo paralelo de pattern matching para sistemas heterogéneos CPU-GPU [16]. Los resultados revelaron que este algoritmo supera en rendimiento a trabajos previos, desarrollados para sistemas multicore y GPUs, para datos de tamaño considerable. En [17] se presentó una solución al problema de pattern matching que aprovecha toda la potencia computacional de los procesadores Intel Xeon Phi KNL 7230 mediante el uso de SIMD y paralelismo de hilos. Se mostró que el algoritmo propuesto alcanza aceleraciones significativas. En trabajos futuros interesa investigar sobre el cómputo colaborativo incluyendo este tipo de arquitecturas.

➢ **Bioinformática y biología computacional.** Estas áreas incluyen aplicaciones que se nutren de las capacidades de HPC para brindar soluciones en un tiempo de respuesta aceptable, siendo cada día más debido al crecimiento exponencial de datos genómicos en los últimos años. Es posible encontrar aplicaciones en alineamiento de secuencias, acoplamiento y dinámica molecular, predicción y búsqueda de estructuras moleculares, entre otras.

➢ **Alineamiento de secuencias.** Si bien el algoritmo de Smith-Waterman es considerado el método de alineamiento más preciso, este algoritmo resulta costoso debido a su complejidad computacional cuadrática. Dado el surgimiento del estándar SYCL y de implementaciones como Intel oneAPI, se migró una versión CUDA de este algoritmo a su equivalente en SYCL, analizando el esfuerzo de programación requerido y la relación costo-beneficio entre overhead y portabilidad para diferentes arquitecturas CPU y GPUs [18][19][20]. A futuro, se propone extender el estudio realizado contemplando otras generaciones de GPUs y arquitecturas como FPGAs.

➢ **Identificación de biomarcadores.** Entre todas las plataformas, servicios y paquetes de software disponibles para esta aplicación, se encuentra Multiomix [21], quien se destaca del resto por facilitar el proceso de integración y minería de datos oncogenómicos públicos y cargados por los usuarios, proporcionando una interfaz gráfica de usuario amigable para los usuarios no expertos. A pesar de ello, Mutiomix aun funciona de forma secuencial por ser un desarrollo reciente (una ejecución habitual en Multiomix puede tomar varias horas para datasets de pequeño porte). Se propone acelerar el procesamiento de Multiomix de manera de disminuir los tiempos de respuesta y así mejorar la productividad de sus usuarios.

➢ **Cálculo de los caminos mínimos**. Es uno de los problemas básicos y de mayor antigüedad de la teoría de grafos teniendo aplicación en el dominio de las comunicaciones, del ruteo de tráfico, de la bioinformática, entre otros. El algoritmo de Floyd-Warshall (FW) permite computar la distancia mínima

entre todos los pares de un grafo. Además de poseer una alta demanda de ancho de banda, FW resulta costoso computacionalmente al ser $O(n^3)$. Habiendo desarrollado una versión optimizada para la arquitectura Intel Xeon Phi KNL [22], se propone tomarla como base para implementar una equivalente para procesadores multicore convencionales. Al mismo tiempo, se espera encontrar posibles oportunidades de optimización paralela [23].

- **Problemas de optimización de simulación de sistemas dinámicos complejos mediante heurísticas**. En una primera etapa se trabajó con una metodología de sintonización de modelos físicos usando para las experimentaciones un modelo de cauce de ríos. Estos modelos requieren de técnicas que detecten la falta de calibración de sus parámetros debido a muchas clases de incertidumbre que impactan en la calidad de los resultados. Se buscó un conjunto ajustado de parámetros que puedan sintonizar el simulador y se propuso un metamodelo de optimización mediante simulación basado en una heurística combinada entre un método de búsqueda Monte Carlo y un algoritmo de agrupamiento KMeans. Se lograron resultados alentadores que mejoraron la calidad de la simulación con gran reducción de los recursos y tiempo de cómputo. Se actuó "por proximidad" entre las distintas configuraciones de parámetros, lo cual fue incluido en el método combinado. En una segunda etapa se actuó por "similitud" de los valores de los parámetros, aprovechando que los valores de magnitudes físicas en regiones contiguas del dominio tienen poca variación. Se agrega conocimiento del sistema simulado, lo que disminuyó considerablemente el espacio de búsqueda de los parámetros y el costo computacional. Actualmente, se aumentó este conocimiento del sistema aprovechando el conocimiento de eventos disruptivos del pasado y su sintonización construyendo una base histórica de eventos que pueden aprovecharse en eventos actuales, disminuyendo aún más el costo computacional de calibración del modelo [24][25].

- **Ambientes para la enseñanza de concurrencia.** Se desarrolló el entorno CMRE para la enseñanza de programación concurrente y paralela a partir de cursos iniciales en carreras de Informática. Incluye un entorno visual que representa una ciudad en la que pueden definirse varios robots que interactúan. Combina aspectos de memoria compartida y distribuida mediante instrucciones para bloquear y liberar esquinas de la ciudad y el concepto de pasaje de mensajes a través de primitivas de envío y recepción. Además, se incluyen los conceptos de heterogeneidad (diferentes velocidades de los robots) y consumo energético [26]. Se ha integrado con el uso de robots físicos (Lego Mindstorm 3.0) que ejecutan en tiempo real las mismas instrucciones que los robots virtuales y se comunican con el entorno mediante bluetooth [27]. Se ha ampliado para incorporar conceptos básicos de computación en la nube (Cloud Computing) [27]. Actualmente, se está desarrollando una nueva herramienta para la enseñanza de programación concurrente en cursos avanzados. Su objetivo principal es visualizar los conceptos de sincronización y comunicación entre procesos.

- **Aplicaciones en Big Data.** En los últimos años, los escenarios de grandes volúmenes de datos (Big Data) son cada vez más comunes debido a la constante generación de datos a partir de fuentes como sensores o Internet, por ejemplo. Para poder trabajar con conjuntos de datos Big Data, se requiere de cómputos de altas prestaciones y de herramientas software específicas capaces de procesar significativos tamaños de datos en tiempos razonables, y que puedan ser ejecutados de manera paralela y distribuida.
  Una de las herramientas más populares para el procesamiento de Big Data es Apache Spark, la cual presenta varias características deseables que la hacen ser ampliamente elegida. El uso intensivo de memoria RAM, los mecanismos eficientes de tolerancia a fallos, las distintas estructuras de datos que provee altamente optimizadas, la amplia variedad de librerías que habilitan utilizar algoritmos de Machine Learning, procesamiento en Streaming, entre otros, son algunas de sus características más relevantes.
  En esta línea de trabajo, se están desarrollando nuevas técnicas para el preprocesamiento de datos para problemas de clasificación en Big Data, utilizando el framework Apache Spark sobre Cómputo de Altas Prestaciones. Como parte del trabajo realizado se presenta una propuesta metodológica para la reducción dual (reducción de instancias y de características) de un conjunto de datos tabulares que representa un problema de clasificación [28].
  Se trata de un diseño sencillo y escalable, capaz de analizar la reducción de datos de forma vertical y horizontal, recibiendo la visión del experto en datos mediante el establecimiento de dos simples condiciones: un umbral de calidad predictiva asociado al modelo obtenido a partir del conjunto reducido, y un rango de porcentajes de reducción a comprobar. Además, su implementación utiliza operaciones paralelas y utilidades totalmente optimizadas proporcionadas por Apache Spark.
  Esta línea de trabajo se lleva a cabo en colaboración con el Dr. Alberto Fernández Hilario del grupo de investigación "Soft Computing and Intelligent Information Systems" (SCI2S) de la Universidad de Granada, España.

- **Cifrado de grandes volúmenes de datos.** Hoy en día, la cantidad de datos sensibles que se generan para ser almacenados y/o transmitidos a través de la red aumenta constantemente. Para proteger los datos confidenciales de amenazas potenciales, se utilizan estrategias de encriptación. Además, el tiempo

involucrado en el cifrado de datos está directamente relacionado con la cantidad de datos que se cifrarán y puede ser significativo. Para reducir el tiempo de cifrado es natural recurrir a soluciones de encriptación paralelas, que explotan todo el poder computacional proporcionado por las arquitecturas emergentes. AES (Advanced Encryption Standard) es uno de los algoritmos de cifrado más utilizados y el gobierno de los Estados Unidos lo considera lo suficientemente seguro como para proteger la información nacional. Hay varias implementaciones de AES, tanto en hardware como en software. En [29] se presenta una comparación de rendimiento de una solución AES basada en hardware para CPU multinúcleo con el de otras dos soluciones AES basadas en software para CPU multinúcleo y GPU, respectivamente. El primero se implementa con las nuevas instrucciones de Intel AES y el segundo con la biblioteca OpenSSL. Los resultados revelan que utilizar cómputo paralelo para el proceso de cifrado reduce significativamente el tiempo de ejecución respecto a una solución secuencial. Los resultados también muestran que el mayor rendimiento se alcanza utilizando una solución multicore con AES basado en hardware. Sin embargo, la solución por software utilizando 2 GPUs puede ser una alternativa competitiva a la solución multicore por hardware cuando se tienen pocos núcleos o una CPU que no soporta AES. Asimismo, en [30] se muestra que una solución colaborativa, que utiliza al mismo tiempo múltiples cores y 2 GPUs para el cifrado de grandes volúmenes de datos, permite alcanzar mejoras en el rendimiento.

Organización de Eventos

En 2023 se organizaron las XI Jornadas de Cloud Computing, Big Data & Emerging Topics (JCC-BD&ET 2022), con participación de especialistas académicos del país y el exterior, además de empresas con experiencia en Cloud Computing [31][32]. En junio de 2024 se tiene prevista la organización de las XII JCC-BD&ET [33].

4. FORMACIÓN DE RECURSOS HUMANOS

Dentro de la temática de la línea de I/D se concluyó 1 tesis doctoral, 1 tesis de maestría y 1 Tesina de Grado de Licenciatura. Se encuentran en curso en el marco del proyecto 3 tesis doctorales, 2 de maestría, y 2 Tesinas de grado.

Se participa en el dictado de las carreras de Doctorado en Cs. Informáticas y Magíster y Especialización en Cómputo de Altas Prestaciones de la Facultad de Informática UNLP, por lo que potencialmente pueden generarse más Tesis y Trabajos Finales.

Asimismo, en grado, los integrantes de la línea de I/D dictan materias relacionadas como Programación Concurrente, Sistemas Paralelos y Taller de Programación sobre GPU.

Existe cooperación con grupos de otras Universidades del país y del exterior, y tesistas de diferentes Universidades realizan su trabajo con el equipo del proyecto.

5. BIBLIOGRAFÍA

[1]. Giles MB, Reguly I. “Trends in high-performance computing for engineering calculations”. Phil.Trans.R.Soc.A 372: 20130319. 2014. http://dx.doi.org/10.1098/rsta.2013.0319

[2]. Jeffers J., Reinders J. “Intel Xeon Phi Coprocessor High Performance Programming”. Morgan Kaufmann.

[3]. Sean Settle. “High-performance Dynamic Programming on FPGAs with OpenCL”. IEEE High Performance Extreme Computing Conf (HPEC ’13). 2013. https://doi.org/10.1016/j.physa.2017.04.159.

[4]. Raspberry PI. https://www.raspberrypi.org/ Accedido 21 de marzo de 2016.

[5]. Odroid http://www.hardkernel.com Accedido 2016.

[6]. Nvidia Jetson https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/ Accedido en febrero de 2023

[7]. Annamalai A., Rodrigues R., Koren I., Kundu S. "Dynamic Thread Scheduling in Asymmetric Multicores to Maximize Performance-per-Watt". 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum, pp. 964-971, 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum, 2012.

[8]. McCool M. “Structured Parallel Programming: Patterns for Efficient Computation”. Morgan Kaufmann.

[9]. De Giusti L, Naiouf M., Chichizola F., Luque E., De Giusti A. “Dynamic Scheduling in Heterogeneous Multiprocessor Architectures. Efficiency Analysis”. Computer Science and Technology Series – XV Argentine Congress of Computer Science Selected Papers. La Plata (Buenos Aires): Editorial de la Universidad de La Plata (edulp). 2010. p85 - 95. isbn 978-950-34-0684-7.

[10]. Chapman B., Jost G., Van der Pas. “Using OpenMP – Portable Shared Memory Parallel Programming”. 2008. UK: MIT Press.

[11]. Hager G., Wellein G. “Introduction to HPC for Scientists and Engineers”. 2011. EEUU: CRC Press.

[12]. De Wael M., Marr S., De Fraine B., Van Cutsem T., De Meuter W. “Partitioned Global Addess Space Languages”. ACM Computing Surveys 47 (4), 2015.

[13]. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/es/ec2/. Febrero 2013.

[14]. Balladini J., Rucci E., De Giusti A., Naiouf M., Suppi R., Rexachs D., Luque E. “Power Characterisation of Shared-Memory HPC Systems”. Computer Science & Technology Series – XVIII Argentine Congress of Computer Science Selected Papers. ISBN 978-987-1985-20-3. Pp 53-65. EDULP, La Plata (Argentina), 2013

[15]. Sanz V., Pousa A., Naiouf M., De Giusti A. “Accelerating Pattern Matching with CPU-GPU Collaborative Computing”. Proceedings of the

International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2018), ISBN: 978-3-030-05051-1, págs. 310-322, doi. https://doi.org/10.1007/978-3-030-05051-1_22, 2018.
[16]. Sanz V., Pousa A., Naiouf M., De Giusti A. "Efficient Pattern Matching on CPU-GPU Heterogeneous Systems". In: Wen S., Zomaya A., Yang L. (eds) Algorithms and Architectures for Parallel Processing. ICA3PP 2019. Lecture Notes in Computer Science, vol 11944. Springer, Cham. 2020.
[17]. Sanz V., Pousa A., Naiouf M., De Giusti A. "Accelerating Pattern Matching on Intel Xeon Phi Processors". In: Qiu M. (eds) Algorithms and Architectures for Parallel Processing. ICA3PP 2020. Lecture Notes in Computer Science, vol 12452. Springer, Cham. https://doi.org/10.1007/978-3-030-60245-1_18.
[18]. Costanzo M., Rucci E., García-Sánchez C., Naiouf M., Prieto-Matías M.. "Migrating CUDA to oneAPI: A Smith-Waterman Case Study." International Work-Conference on Bioinformatics and Biomedical Engineering 2022, ISBN 978-3-031-07802-6, págs. 103-116, Cham, doi: 10.1007/978-3-031-07802-6_9. 2022.
[19]. Costanzo M., Rucci E., García-Sánchez C., Naiouf M., Prieto-Matías M. "Comparing Performance and Portability Between CUDA and SYCL for Protein Database Search on NVIDIA, AMD, and Intel GPUs." 2023 IEEE 35th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). ISBN 979-8-3503-0548-7. Págs. 141-148. Porto Alegre, Brasil. doi: 10.1109/SBAC-PAD59825.2023.00023. Octubre 2023..
[20]. Costanzo M., Rucci E., García-Sánchez C., Naiouf M., Prieto-Matías M. "Assessing opportunities of SYCL for biological sequence alignment on GPU-based systems." J Supercomput (2024). https://doi.org/10.1007/s11227-024-05907-2
[21]. Camele G., Menazzi S., Chanfreau H., Marraco A., Hasperué W., Butti M. D., Abba M.C. "Multiomix: a cloud-based platform to infer cancer genomic and epigenomic events associated with gene expression modulation". Bioinformatics, Volume 38, Issue 3, 1 February 2022, Pages 866–868, https://doi.org/10.1093/bioinformatics/btab678
[22]. Costanzo M., Rucci E., Costi U., Chichizola F., Naiouf M. "Comparison of HPC Architectures for Computing All-Pairs Shortest Paths. Intel Xeon Phi KNL vs NVIDIA Pascal". En: Computer Science – CACIC 2020. Revised Selected Papers., Springer International Publishing, doi. 10.1007/978-3-030-75836-3_3, 2021.
[23]. Calderón S. L., Rucci E., Chichizola F. "Adaptación de Algoritmo OpenMP para Computar Caminos Mínimos en Grafos en Arquitecturas x86". XXIX Congreso Argentino de Ciencias de la Computación – CACIC 2023. Octubre de 2023. En prensa.
[24]. Trigila, M., Gaudiani, A., Wong, A., Rexachs, D., Luque, E. (2023). Reduction of the Computational Cost of Tuning Methodology of a Simulator of a Physical System. In: Mikyška, J., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M. (eds) Computational Science – ICCS 2023. ICCS 2023. Lecture Notes in Computer Science, vol 10475. Springer, Cham. https://doi.org/10.1007/978-3-031-36024-4_49.
[25]. Gaudiani, A., Wong, A., Luque, E. et al. A computational methodology applied to optimize the performance of a river model under uncertainty conditions. J Supercomput 79, 4737–4759 (2023). https://doi.org/10.1007/s11227-022-04816-6
[26]. Castro J., De Giusti L., Gorga G., Sánchez M., Naiouf M. "ECMRE: Extended Concurrent Multi Robot Environment". Computer Science – CACIC 2017. Communications in Computer and Information Science, vol 790, ISBN: 978-3-319-75213-6 978-3-319-75214-3, Springer, Cham, págs. 285-294. 2018.
[27]. De Giusti L., Chichizola F., Rodríguez Eguren S., Sánchez M., Paniego J. M., De Giusti A. "Introduciendo conceptos de Cloud Computing utilizando el entorno CMRE". Proceedings del XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016) – Workshop de Innovación en Educación en Informática. 2016.
[28]. Basgall, M. J., Naiouf, M., & Fernández, A. "Análisis y diseño de técnicas de preprocesamiento de instancias escalables para problemas no balanceados en Big Data. Aplicaciones en situaciones de emergencias humanitarias". Tesis Doctoral. 2022.
[29]. Sanz V., Pousa A., Naiouf M., De Giusti A. "Comparison of Hardware and Software Implementations of AES on Shared-Memory Architectures". In: Naiouf M., Rucci E., Chichizola F., De Giusti L. (eds) Cloud Computing, Big Data & Emerging Topics. JCC-BD&ET 2021. Communications in Computer and Information Science, vol 1444. Springer, Cham. 2021. https://doi.org/10.1007/978-3-030-84825-5_5.
[30]. Sanz, V., Pousa, A., Naiouf, M., De Giusti, A. "Performance Analysis of AES on CPU-GPU Heterogeneous Systems". In: Rucci, E., Naiouf, M., Chichizola, F., De Giusti, L., De Giusti, A. (eds) Cloud Computing, Big Data & Emerging Topics. JCC-BD&ET 2022. Communications in Computer and Information Science, vol 1634. Springer, Cham. 2022. https://doi.org/10.1007/978-3-031-14599-5_3
[31]. Naiouf M., Rucci E., Chichizola F., De Giusti L. "Cloud Computing, Big Data & Emerging Topics: 11th Conference, JCC-BD&ET 2023, La Plata, Argentina, June 27-29, 2023, Proceedings". Communications in Computer and Information Science. Springer Cham. 2023. https://doi.org/10.1007/978-3-031-40942-4.
[32]. De Giusti A., Naiouf M., Chichizola F., Rucci E., De Giusti L. (eds). "Short papers of the 11h Conference on Cloud Computing, Big Data & Emerging Topics (JCC-BD&ET 2023)". 2023. https://doi.org/10.35537/10915/155281.
[33]. XII Jornadas de Cloud Computing, Big Data & Emerging Topics. https://jcc.info.unlp.edu.ar/