CÓMPUTO PARALELO Y DISTRIBUIDO PARA HPC. FUNDAMENTOS, CONSTRUCCIÓN Y EVALUACIÓN DE APLICACIONES.

Marcelo Naiouf[(1)], Armando De Giusti[(1)(2)], Laura De Giusti[(1)(3)], Franco Chichizola[(1)], Victoria Sanz[(1)(2)(3)], Adrián Pousa [(1)], Enzo Rucci[(1)(2)], Silvana Gallo[(1)(2)], Erica Montes de Oca[(1)], Emmanuel Frati[(1)], Mariano Sánchez [(1)], María José Basgall[(1)(2)], Adriana Gaudiani[(4)]

[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 General Sarmiento
{mnaiouf, degiusti, ldgiusti, francoch, vsanz, apousa, erucci, sgallo, emontesdeoca,fefrati, msanchez, mjbasgall}@lidi.info.unlp.edu.ar, agaudi@ungs.edu.ar

RESUMEN

El eje central de la línea son los temas de procesamiento paralelo y distribuido para HPC (fundamentos y aplicaciones). Interesa la construcción, evaluación y optimización de soluciones sobre diferentes plataformas de software y arquitecturas con múltiples procesadores (multicore, clusters, cloud, aceleradores y placas de bajo costo), los lenguajes y paradigmas de programación paralela (puros e híbridos), los modelos de representación de aplicaciones paralelas, los algoritmos de mapping y scheduling, el balance de carga, las métricas de evaluación de complejidad y rendimiento computacional y energético, y la construcción de ambientes para la enseñanza de la programación concurrente y paralela.
Se propone aplicar los conceptos en problemas numéricos y no numéricos de cómputo intensivo y/o sobre grandes volúmenes de datos (búsquedas, simulaciones, n-body, big data, reconocimiento de patrones, bioinformática, etc), con el fin de obtener soluciones de alto rendimiento.
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, y con el Departamento de Arquitectura de Computadores y Automática de la Universidad Complutense de Madrid, entre otros.
Palabras clave: Cómputo paralelo y distribuido de altas prestaciones. Algoritmos paralelos y distribuidos. Clusters. Multicore. Aceleradores. Consumo energético. Balance de carga. Aplicaciones. Performance.

CONTEXTO

La línea de I/D que se presenta es parte del Proyecto "Computación de Alto Desempeño: Arquitecturas, Algoritmos, Métricas de rendimiento y Aplicaciones en HPC, Big Data, Robótica, Señales y Tiempo Real" del III-LIDI acreditado por el Ministerio de Educación, del proyecto "Computación de Alto Desempeño, Minería de Datos y Aplicaciones de interés social en la Provincia de Buenos Aires" financiado por la CIC PBA en la convocatoria a Proyectos de Innovación y Transferencia en Areas Prioritarias de la Pcia. de Buenos Aires (PIT-AP-BA), 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 por AECID, CyTeD, OEI y CIC y becas de Telefónica de Argentina. 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 [GIL14].
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 [JEF13] o los aceleradores basados en circuitos integrados reconfigurables (FPGAs, Field Programmable Gate Array) [SET13]. En la actualidad se comercializan placas

de bajo costo como Raspberry PI [RAS16] u Odroid [ODR16] que 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 [ANN12], así como las características de scheduling en los mismos [SAE15]. 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 [MCC12]. 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 paralla 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 [DEG10].

Muchos problemas algorítmicos se vieron impactados por los multicore y clusters de multicore. A partir de incorporar varios chips multicore dentro de un nodo y conectar múltiples nodos vía red, se puede crear una arquitectura NUMA, de modo que los cores en un chip compartan memoria principal, y puedan acceder remotamente a la memoria dedicada de otro chip, aunque ese acceso sea más costoso, surgiendo así varios niveles de comunicación. Esto impacta sobre el desarrollo de algoritmos que aprovechen adecuadamente las arquitecturas, y motiva el estudio de performance en sistemas híbridos. Además, es necesario estudiar la utilización de diferentes lenguajes y bibliotecas ya que aún no se cuenta con un standard, aunque puede mencionarse el uso de los tradicionales MPI, OpenMP y Pthreads [CHA08][HAG11] o los más recientemente explorados UPC, Chapel y Titanium del modelo PGAS [DEW15].

La combinación de arquitecturas con múltiples núcleos con aceleradores dio lugar a plataformas híbridas con diferentes características. Más allá del tipo de acelerador utilizado, la programación de esta clase de plataformas representa un verdadero desafío. Para lograr aplicaciones de alto rendimiento, los programadores deben enfrentar dificultades como: estudiar características específicas de cada arquitectura y aplicar técnicas de programación y optimización particulares para cada una de ellas, lograr un balance de carga adecuado entre los diferentes dispositivos de procesamiento y afrontar la ausencia de estándares y 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 [EC213]. 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 aunque queda mucho por hacer en cuanto al diseño, lenguajes y programación

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 [BAL13]. 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 que componen la aplicación. Entre las aplicaciones de interés se encuentran las numéricas y no numéricas con alta demanda de cómputo.

2. LÍNEAS DE INVESTIGACIÓN, DESARROLLO E INNOVACIÓN

- Investigar en temas de cómputo paralelo y distribuido de alto desempeño, tanto en lo referido a los fundamentos como 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 cloud.
  - Se propone 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 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 (big data).

▪ Utilizar arquitecturas híbridas que combinan memoria compartida y pasaje de mensajes, evaluando performance para distintos modelos de comunicación.

▪ 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. Comparar estrategias 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 utilizadas.

▪ Respecto de las aplicaciones estudiadas y algoritmos implementados, se trabajó fundamentalmente con los siguientes problemas:

➢ **Best-first search (BFS) paralelo.** El algoritmo BFS es utilizado para resolver problemas combinatorios, los cuales requieren encontrar una secuencia de acciones que transformen una configuración inicial (problema) en una configuración final (solución). En particular, A* es una variante de BFS que permite encontrar soluciones de costo óptimo. Estos algoritmos requieren una alta capacidad de cómputo y gran cantidad de memoria, por esto su paralización es imprescindible. En los últimos años se ha re-impulsado el desarrollo de algoritmos paralelos BFS para aprovechar: (a) la potencia de cómputo de los procesadores *multicore* (b) la gran cantidad de RAM y potencia de cómputo de los *clusters de muticore.* En este sentido, HDA* [KIS13] paraleliza A* sobre clusters utilizando MPI: cada procesador realiza una búsqueda cuasi-independiente y se distribuyen los nodos en base a una función hash estándar. Otros autores [BUR10] adaptaron HDA* a máquinas multicore utilizando Pthreads: eliminan overheads existentes en la versión original al correr sobre una arquitectura de memoria compartida y utilizan menor cantidad de memoria. Para explotar eficientemente los recursos de un *cluster de multicore*, desarrollamos Hybrid HDA* (HHDA*) [SAN17], una versión híbrida de HDA* programada con MPI+Pthreads. El trabajo experimental demostró que HHDA* alcanza un rendimiento superior y consume menor cantidad de memoria, comparado con HDA* (versión original). Estas mejoras permitieron a HHDA* resolver una de las instancias más complejas del caso de estudio. Como trabajo futuro planeamos paralelizar algoritmos de búsqueda *sub-óptimos* usando nuestra estrategia de paralelización híbrida.

➢ **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. Para facilitar la etapa de desarrollo, en [POU16]

propusimos un esquema para estructurar código paralelo a ser ejecutado sobre un cluster heterogéneo de CPUs/GPUs, utilizando todos los recursos disponibles (CPUs y GPUs). Aplicamos este esquema al problema de suma por reducción y comprobamos que es posible incrementar el rendimiento de la aplicación utilizando CPUs y GPUs en forma colaborativa. Planeamos aplicar nuestro esquema de cómputo colaborativo a distintos tipos de problemas demandantes en cómputo.

- **Alineamiento de secuencias biológicas.** Esta operación consiste en comparar dos o más secuencias biológicas, como pueden ser las de ADN o las de proteínas, y resulta fundamental en investigaciones de la bioinformática y la biología molecular. El algoritmo de Smith-Waterman es considerado el método de alineamiento más preciso. Desafortunadamente, este algoritmo resulta costoso debido a su complejidad computacional cuadrática mientras que la situación se agrava aún más a causa del crecimiento exponencial de datos biológicos en los últimos años [RUC16]. El reciente surgimiento de aceleradores en HPC (GPU, Xeon Phi, FPGA, entre otros) da la oportunidad de acelerar los alineamientos sobre hardware comúnmente disponible a un costo accesible. En primer lugar, se exploró el empleo del modelo de programación OpenCL sobre FPGAs para acelerar el alineamiento de secuencias largas de ADN [RUC17a]. Luego, se desarrolló una herramienta capaz de procesar alineamientos de secuencias de ADN sin restricciones de tamaño y se analizó su rendimiento comparándolo con el de otras implementaciones basadas en multicores, Xeon Phi y GPUs [RUC18a]. Por otra parte, la salida al mercado de la segunda generación de procesadores Xeon Phi (Knights Landing) dio la oportunidad de evaluar su uso para búsquedas de similitud en bases de datos de proteínas [RUC17b]. A futuro, interesa explorar el uso de próximas generaciones de procesadores y aceleradores para esta aplicación, como pueden ser los procesadores Xeon *Skylake* de Intel y las nuevas generaciones de FPGAs.

- **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 $O(n^3)$. Empezando por una implementación secuencial de base, se estudió y cuantificó cómo diferentes optimizaciones a nivel de datos, hilos y compilador permiten mejorar su rendimiento sobre los nuevos procesadores Xeon Phi Knights Landing [RUC18b]. Como trabajo futuro, interesa realizar un estudio comparativo con otros aceleradores (como GPUs), no sólo desde el punto de vista del rendimiento sino también de la eficiencia energética.

- **Simulación distribuida de modelos orientados al individuo.** Debido al incremento en la complejidad de los modelos es necesaria mayor cantidad de cómputo y comunicación para lograr resultados representativos. Actualmente, se busca analizar el consumo energético del simulador teniendo en cuenta el balance de cómputo, ya que al trabajar con grandes cantidades de individuos se producen desbalances en cuanto a la cantidad de trabajo de cada uno de los diferentes procesos lógicos. Se pretende desarrollar un modelo energético para estimar el consumo de energía para este tipo de algoritmos y que permita decidir el escenario adecuado.

- **Problemas de tipo N-body.** Se utilizaron las plataformas de memoria compartida GPU y cluster de multicore para la resolución de problemas con alta demanda computacional del tipo N-body. Se emplearon diferentes modelos de comunicación: memoria compartida (Pthreads en CPU y CUDA en GPU), pasaje de mensajes (MPI) y soluciones híbridas (MPI-Pthreads). Se han mostrado los beneficios del uso de la GPU en problemas con características similares al caso planteado. El trabajo experimental ha dado como resultado una buena aceleración obtenida utilizando cluster de GPU. Además, se observó claramente que el uso del cluster de GPU logró una aceleración proporcional al speedup conseguido con el cluster de CPU pero con tiempos de ejecución significativamente menores [MON14]. También se han desarrollado diferentes alternativas de distribución de trabajado usando un Cluster de GPUs heterogéneas [MON16]. Además del uso de un Cluster de GPU se está utilizando MultiGPU, pero haciendo énfasis en el consumo energético. El énfasis en las experimentaciones actuales, se centra en la determinación de un modelo de estimación de consumo energético, y la precisión de la medición de los contadores de hardware en GPU.

- **Problemas de simulación relacionados con fenómenos naturales (inundaciones)**. Análisis de diferentes soluciones para la paralelización de este tipo de aplicaciones que son intensivas en cómputo; y el tiempo de ejecución y la performance alcanzable son críticas dado que los resultados que se esperan determinarán alertas y toma de decisiones. La utilización de escenarios de simulación en entornos donde interesa estudiar el comportamiento en situaciones de desastres producidos por fenómenos naturales como las inundaciones. En este ámbito se avanza en dos temas: (1) La implementación de un método de sintonización de un simulador de inundaciones en ríos de llanura, mediante la técnica de simulación paramétrica. El proceso requiere lanzar miles de escenarios de simulación hasta encontrar un conjunto ajustado de parámetros de entrada del

simulador. La experimentación se lleva a cabo con un modelo master-worker sobre un cluster [GAU15]. (2) En colaboración con el Laboratorio de Hidrología de la UNLP se comenzó con la paralelización de la simulación de inundaciones producidas por lluvias (en particular en el ámbito de la ciudad de La Plata, donde una corrida "standard" es del orden de las 8 hs), a fin de reducir el tiempo de ejecución a pocos minutos y permitir establecer un sistema de alertas [GAU16].

- **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 [DEG17]. 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 [DEG16]. Se ha ampliado para incorporar conceptos básicos de computación en la nube (Cloud Computing) [DEG16].

- **Aplicaciones en Big Data.** Para trabajar con Big Data, escenario que es cada vez más común debido al gran volumen de datos que se genera a diario, se requiere de cómputos de altas prestaciones y de herramientas específicas que ayuden al proceso de adaptación de los algoritmos iterativos tradicionales para ser ejecutados de manera paralela y distribuida. Una de las herramientas más utilizadas es Apache Spark, que se caracteriza principalmente por su velocidad de procesamiento debido a que hace uso intensivo de memoria RAM evitando continuas escrituras en disco y, a su vez, por poseer mecanismos eficientes de tolerancia a fallos. Apache Spark posee un módulo para el tratamiento de datos provenientes de un flujo de información potencialmente infinito llamado, Spark Streaming. En esta línea se está trabajando, por un lado, en la implementación de cálculos de índices económicos sobre un flujo de datos proveniente del mercado de las criptomonedas (Bitcoin, Ethereum, Litcoin, etc) utilizando Spark Streaming [BAR17] [BAS17]. Por otro lado, en la adaptación de técnicas tradicionales de Machine Learning para ser aplicadas a grandes volúmenes de datos mediante el uso del framework Spark.

4. FORMACIÓN DE RECURSOS HUMANOS

Dentro de la temática de la línea de I/D en el año 2017 se concluyó 1 tesis doctoral, 1 Trabajo Final de Especialización y 1 Tesina de Grado de Licenciatura. Se encuentran en curso en el marco del proyecto 3 tesis doctorales, 1 de maestría, 3 trabajos de Especialización y 2 Tesinas.

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.

Hay 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

[ANN12] 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.

[BAL13] 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

[BAR17] Bariviera, A. F., Basgall, M. J., Hasperué, W., & Naiouf, M. "Some stylized facts of the Bitcoin market". Physica A: Statistical Mechanics and its Applications, 484, 82?90. 2017.

[BUR10] Burns E, Lemons S, Ruml W, Zhou R. "Best First Heuristic Search for Multicore Machines". Journal of Artificial Intelligence Research, Vol.39, No.1, pp. 689-743, 2010.

[CHA08] Chapman, B., Jost, G. & Van der Pas. Using OpenMP – Portable Shared Memory Parallel Programming (2008). UK: MIT Press.

[DEG10] 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

[DEG16] L. C. De Giusti, F. Chichizola, S. Rodriguez Eguren, M. Sanchez, J. M. Paniego, A. E. De Giusti. "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. Octubre 2016. Pp 1357-1365.

[DEG17] J. Castro, L. D. Giusti, G. Gorga, M. Sánchez, and M. Naiouf,. "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, enero de 2018.
[DEW15] De Wael, M; Marr, S; De Fraine, B; Van Cutsem, T; De Meuter, W. "Partitioned Global Addess Space Languages". ACM Computing Surveys (CSUR) 47 (4), 2015.
[EC213] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/es/ec2/. Febrero 2013.
[GAU15] Adriana Gaudiani. "Simulación y optimización como metodología para mejorar la calidad de la predicción en un entorno de simulación hidrográfica". Tesis de Doctorado en Ciencias Informáticas (Facultad de Informática – UNLP). 2015.
[GAU16] A.Gaudiani, E. Luque, P. García, M. Re, M. Naiouf, A. De Giusti. "How a Computational Method Can Help to Improve the Quality of River Flood Prediction by Simulation". Advances and New Trends in Environmental and Energy Informatics (part V). ISBN 978-3-319-23455-7. Pp337-351. 2016.
[GIL14] Giles MB, Reguly I. 2014 "Trends in high-performance computing for engineering calculations". Phil.Trans.R.Soc.A 372: 20130319. http://dx.doi.org/10.1098/rsta.2013.0319
[HAG11] Hager, G. & Wellein, G. Introduction to HPC for Scientists and Engineers. (2011) EEUU: CRC Press.
[JEF13] Jeffers, James; Reinders, James. "Intel Xeon Phi Coprocessor High Performance Programming", Morgan Kaufmann, 2013.
[KIS13] A. Kishimoto, A. Fukunaga, and A. Botea, "Evaluation of a simple, scalable, parallel best-first search strategy," Artificial Intelligence, vol. 195, pp. 222–248, 2013.
[MCC12] McCool, Michael. "Structured Parallel Programming: Patterns for Efficient Computation", Morgan Kaufmann, 2012
[MON14] E. Montes de Oca, L. De Giusti, F. Chichizola, A. De Giusti, M. Naiouf. "Utilización de Cluster de GPU en HPC. Un caso de estudio". Proceedings del XX Congreso Argentino de Ciencias de la Computación (CACIC 2014), pp. 1220-1227, 2014.
[MON16] E. Montes de Oca, L. C. De Giusti, F. Chichizola, A. E. De Giusti, M. Naiouf. "Análisis de uso de un algoritmo de balanceo de carga estático en un Cluster Multi-GPU Heterogéneo". Proceedings del XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016), pp. 159-168, 2016.
[ODR16] Odroid http://www.hardkernel.com Accedido 21 de Marzo de 2016.
[POU16] Pousa A, Sanz V, De Giusti A. "Estructurando código paralelo para clusters heterogéneos de CPUs/GPUs". Proceedings del XXII Congreso Argentino de Ciencias de la Computación. ISBN: 978-987-733-072-4. Pp. 139-148
[RAS16] Raspberry PI. https://www.raspberrypi.org/ Accedido 21 de Marzo de 2016.
[RUC15] "Smith-Waterman Protein Search with OpenCL on FPGA". Enzo Rucci, Armando De Giusti, Marcelo Naiouf, Carlos García Sanchez, Guillermo Botella Juan, Manuel Prieto-Matías. Proceedings of 2014 IEEE Symposyum on Parallel and Distributed Processing with Applications (ISPA). 20 al 22 de Agosto de 2015. Helsinki, Finlandia. ISBN: 978-1-4673-7952-6. Pp. 208-213. DOI: 10.1109/Trustcom.2015.634.
[RUC16] "State-of-the-art in Smith-Waterman Protein Database Search". Enzo Rucci, Armando De Giusti, Marcelo Naiouf, Carlos García Sanchez, Guillermo Botella Juan, Manuel Prieto-Matías. Big Data Analytics in Genomics. Ka-Chun Wong (Editor). ISBN: 978-3-319-41278-8 (print) 978-3-319-41279-5 (online), Springer, págs. 197-223, 2016.
[RUC17a] "Accelerating Smith-Waterman Alignment of Long DNA Sequences with OpenCL on FPGA". E. Rucci, C. Garcia, G. Botella, A. De Giusti, M. Naiouf, and M. Prieto-Matias, En: Bioinformatics and Biomedical Engineering. IWBBIO 2017. Lecture Notes in Computer Science, vol 10209., ISBN: 978-3-319-56154-7, Springer, Cham, págs. 500-511, 2017.
[RUC17b] "First Experiences Accelerating Smith-Waterman on Intel's Knights Landing Processor". E. Rucci, C. Garcia, G. Botella, A. De Giusti, M. Naiouf, and M. Prieto-Matias, En: Algorithms and Architectures for Parallel Processing. ICA3pp 2017. Lecture Notes in Computer Science, vol 10393, ISBN: 978-3-319-65482-9, Springer International Publishing, págs. 569-579, 2017.
[RUC18a] "SWIFOLD:Smith-Waterman Implementation on FPGA with OpenCL for Long DNA Sequences". E. Rucci, C. Garcia, G. Botella, A. De Giusti, M. Naiouf, and M. Prieto-Matias. BMC Systems Biology. ISSN: 1752-0509. In press. 2018.
[RUC18b] "Blocked All-Pairs Shortest Paths Algorithm on Intel Xeon Phi KNL Processor: A Case Study". E. Rucci, A. De Giusti, and M. Naiouf, En: 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. 47-57, 2018.
[SAN15] V. Sanz, A. De Giusti, and M. Naiouf, "Performance tuning of the HDA* algorithm for multicore machines." in Computer Science & Technology Series - XX Argentine Congress of Computer Science - Selected Papers. Argentina: EDULP, 2015, pp. 51-62.
[SAN17] Sanz V., De Giusti A., Naiouf M. (2017) "A Hybrid Parallel Search Algorithm for Solving Combinatorial Optimization Problems on Multicore Clusters". Algorithms and Architectures for Parallel Processing. ICA3PP 2017. Lecture Notes in Computer Science, vol 10393. Springer, Cham.
[SET13] High-performance Dynamic Programming on FPGAs with OpenCL. Sean Settle. 2013 IEEE High Performance Extreme Computing Conf (HPEC '13), 2013. https://doi.org/10.1016/j.physa.2017.04.159
[BAS17] Basgall, M. J., Hasperué, W., Naiouf, M., & Bariviera, A. F."Cálculo del exponente de Hurst utilizando Spark Streaming: enfoque experimental sobre un flujo de transacciones de criptomonedas. Presentado en XXIII Congreso Argentino de Ciencias de la Computación (La Plata, 2017).