International Journal of Innovation and Applied Studies

ISSN 2028-9324 Vol. 18 No. 4 Dec. 2016, pp. 1032-1038

* 2016 Innovative Space of Scientific Research Journals http://www.ijias.issr-journals.org/

**Procesamiento en paralelo de grandes volúmenes de información**

**[ Parallel processing of large volumes of information]**

***Gary Reyes Zambrano, María Fernanda Méndez, Luis Alberto Banchón and Héctor Alfredo Malucín Llamuca***

Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Guayaquil, Ecuador

Copyright © 2016 ISSR Journals. This is an open access article distributed under the ***Creative Commons Attribution License***, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

**ABSTRACT:** Parallel processing is defined as an efficient way to process information, favoring the exploitation of concurrent events in the computation process. Among the objectives of parallelism are improving the execution time and making competent use of computation resources, the use of uneven processing elements can cause poor operability or make the parallel time greater than the sequential one. This term also refers to the set of techniques used to provide simultaneous processing tasks with the purpose of increasing the computational speed. The purpose of this work is to present the conventional techniques used for data processing and briefly explain about the operation of the parallel processing algorithms on the GPU (Graphic Processing Unit). This expression refers to multi-core processors that provide high and efficient performance. It is worth mentioning that these algorithms use work platforms such as CUDA, which is a parallel calculation architecture of NVIDIA that uses the great potential of the GPU to offer a remarkable increase in system performance.

**KEYWORDS:** GPU, threads, speed, performance, data.

**RESUMEN:** El procesamiento paralelo se lo define como una manera eficiente de procesar información favoreciendo la explotación de los sucesos concurrentes en el proceso de computación. Entre los objetivos del paralelismo se encuentran mejorar el tiempo de ejecución y hacer uso competente de los recursos de cómputo, el uso desigual de los elementos de procesamiento puede causar pobre operatividad o hacer que el tiempo paralelo sea mayor que el secuencial. También este término hace mención al conjunto de técnicas utilizadas para proveer tareas simultáneas de procesamiento con el propósito de incrementar la velocidad computacional. La finalidad de este trabajo es dar a conocer técnicas convencionales utilizadas para el procesamiento de datos y explicar brevemente acerca del funcionamiento de los algoritmos de procesamiento en paralelo sobre la Unidad de procesamiento gráfico. Esta expresión hace referencia a los procesadores de múltiples núcleos que proporcionan un elevado y eficiente rendimiento. Cabe mencionar que estos algoritmos utilizan plataformas de trabajo como es el caso de CUDA, la cual es una arquitectura de cálculo paralelo de NVIDIA que utiliza el gran potencial de la unidad de procesamiento gráfico para ofrecer un notable aumento del rendimiento del sistema.

**PALABRAS CLAVES:** GPU, hilos, velocidad, rendimiento, datos.

|  |  |
| --- | --- |
| **Corresponding Author:** Gary Reyes Zambrano | **1032** |

**Gary Reyes Zambrano, María Fernanda Méndez Buenaño, *Luis Alberto Banchón and Héctor Alfredo Malucín Llamuca***

* **INTRODUCCIÓN**

El paralelismo es la ejecución concurrente, es decir, en el mismo intervalo de tiempo sobre distintos procesadores. En general dado un problema, se lo divide en subproblemas que son resueltos simultáneamente por los procesadores. [1]

El área de procesamiento paralelo se ha convertido en clave dentro de las Ciencias de la Computación, debido al creciente interés por el desarrollo de soluciones a problemas con muy alta demanda computacional y de almacenamiento. [2]

En el procesamiento en paralelo, diversas unidades de procesamiento dividen un problema en pequeñas partes y trabajan al mismo tiempo en él. Para lograr que un grupo de procesadores ataquen el mismo problema a la vez es necesario replantear los problemas un software especial para que permita dividir estos problemas entre diferentes procesadores de la manera más eficiente. proporcionando de esa manera los datos necesarios reensamblando posteriormente las diversas subtareas para llegar a una solución adecuada.

El propósito del procesamiento paralelo es acelerar la posibilidad de procesamiento de las computadoras elevar su rendimiento, esto es, la cantidad de procesamiento que puede lograr durante un cierto intervalo de tiempo. El número de circuitos aumenta con el procesamiento en paralelo y, con él, además el costo del sistema. Sin embargo, los descubrimientos tecnológicos han reducido el coso de la circuitería a un punto en donde las técnicas de procesamiento paralelo son económicamente factibles. [3]

El procesamiento paralelo puede considerarse desde diversos niveles de complejidad. En el nivel más bajo, distinguimos entre operaciones seriales y paralelo mediante el tipo de registro que utilizan. [3]

Cuando se habla de procesos paralelos en un computador, se hace referencia a aquellos procesos que se ejecutan y procesan a la vez, priorizando a los procesos lineales o secuenciales, que serán ejecutados de uno en uno.

La computación paralela por GPU es el uso de una unidad de procesamiento de gráficos (GPU, por sus siglas en inglés Graphics Processing Unit) junto a una CPU para acelerar el funcionamiento de las aplicaciones de aprendizaje profundo, análisis e ingeniería. [4]

Los primeros precursores de las unidades de procesamiento gráficos se remontan a la década de los 80. Se trataba de pequeños chips controladores, muy parecidos a los que se usan hoy día para la conectividad Ethernet, Bluetooth o Wifi. La unidad de procesamiento de gráficos (GPU) es uno de los componentes más importantes en los ordenadores modernos. Puede aligerar la carga de información que debe ser procesada por la unidad central, y esta última puede hacer su trabajo de manera más eficiente. [5]

Los multiprocesadores gráficos, GPU, se transformaron en una atractiva opción tanto para aplicaciones científicas como para otro tipo de cómputo no gráfico [6].

Esta arquitectura incluye cientos de arreglos de procesadores que son aptos de realizar una operación de punto flotante a la vez, por ciclo de reloj. Las arquitecturas modernas ya integran al hardware los multiprocesadores gráficos presentes en las placas de video, los cuales funcionan como coprocesadores paralelos de extremo poder de cómputo

* **RESULTADOS**

Como resultado de la investigación realizada se encontró las siguientes técnicas y algoritmos para el procesamiento en paralelo:

**Arquitectura Paralela**

La arquitectura paralela o de líneas paralelas (pipe-line), es una técnica en la que se descompone un [proceso](http://www.monografias.com/trabajos14/administ-procesos/administ-procesos.shtml#PROCE) secuencial en suboperaciones, y cada subproceso se ejecuta en un segmento dedicado especial que opera en forma concurrente con los otros segmentos. Una línea puede considerarse como un conjunto de segmentos de procesamiento por el que fluye información binaria [3] . Cada segmento ejecuta un procesamiento parcial, dictado por la manera en que se divide la tarea. El resultado obtenido del [cálculo](http://www.monografias.com/trabajos7/caes/caes.shtml) en cada segmento se transfiere al siguiente segmento en la línea. El resultado final se obtiene después de que los datos han recorrido todos los segmentos. El nombre "línea" implica un flujo reinformación similar a una línea de ensamblado industrial. Es característico de las líneas que varios cálculos puedan estar en proceso en distintos segmentos, al mismo tiempo. La simultaneidad de los cálculos es posible al asociar un [registro](http://www.monografias.com/trabajos7/regi/regi.shtml) con cada segmento en la línea. Los registros proporcionan aislamiento entre cada segmento para que cada uno pueda operar sobre datos distintos en forma simultánea.

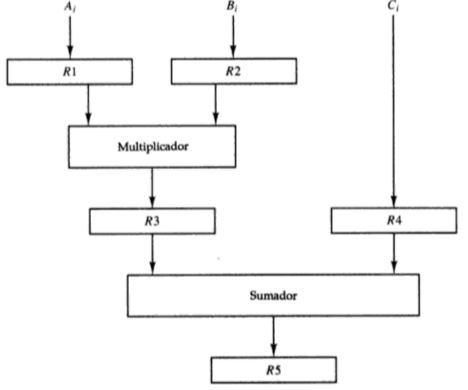


Figura 1. Procesamiento en paralelo mediante pipeline

Una manera sencilla de apreciar la arquitectura de líneas paralelas es imaginar que cada segmento consta de un registro de entrada seguido de un circuito combinatorio. Este registro contiene los datos y el circuito combinatorio ejecuta las suboperación en el segmento particular. La salida del circuito combinacional es un segmento dado se aplica al registro de entrada del siguiente segmento. Se aplica un reloj a todos los registros después de que se ha transcurrido un lapso suficiente para ejecutar toda la actividad del segmento. De esta manera la información fluye por la línea un paso a la vez.

La ordenación de la línea se muestra mediante un ejemplo:

Supongamos que se requiere ejecutar las operaciones multiplicar y sumar combinadas con un flujo de números.

Ai \* Bi + Ci para i=1,2, 3…7

Cada suboperación se va a implantar en un segmento dentro de la línea. En la figura 1 se puede observar que cada superación se implementa en un segmento internamente de la línea. Cada segmento a su vez posee uno o dos registros y un circuito combinatorio. Del registro R1 al R5 se recibe nuevos datos con cada pulso del reloj. El multiplicador el sumador son registros combinatorios. Las suboperaciones ejecutadas en cada segmento del conducto son las siguientes:

Estos cinco registros se cargan con datos nuevos mediante pulsos de reloj. El efecto que produce cada pulso es el siguiente: En el primer pulso se transfiere A1 y B1, dentro de R1 y R2. El segundo pulso de reloj transfiere el [producto](http://www.monografias.com/trabajos12/elproduc/elproduc.shtml) de R1 y R2 dentro de R3 y C1 dentro de R4. El mismo pulso de reloj transfiere A2 y B2 dentro de R1 y R2. El tercer pulso de reloj opera sobre los tres segmentos en forma simultánea. Coloca a A3 y B3 dentro de R1 y R2, transfiere el producto R1 y R2 dentro de R3, transfiere C2 dentro de R4 y coloca la suma de R3 y R4 dentro de R5. Se requiere de tres pulsos de reloj para llenar la línea y recuperar la primera salida de R5. De ahí en adelante, cada ciclo de reloj produce una nueva salida y mueve los datos un paso adelante en la línea. Esto sucede en tanto fluyan nuevos datos dentro del sistema cuando ya no hay datos de entrada disponibles, el reloj debe continuar hasta que emerge la última salida de la línea

**Parallel Query Option**

Esta nueva opción mejora el rendimiento de consultas a la base de datos, en cualquier tipo de arquitectura paralela soportada por Oracle. Con este componente, el optimizador, al analizar una petición a la base de datos, puede usar múltiples CPUs para procesar una simple query. [7]

Máquinas con la arquitectura SMP (Symetric Multi-processor), las cuales tienen de dos a treinta procesadores, pueden alcanzar una evidente mejora de rendimiento cercano a las treinta veces. Máquinas con arquitectura MPP (Massively Parallel Processor) que utilizan entre 50 y 1000 procesadores, pueden optimizar el rendimiento entre 50 y 1000 veces. Por ejemplo, una consulta que se ejecutaba en 24 horas en una CPU, puede ahora ejecutarse en menos de 2 minutos en una máquina masivamente paralela equipada con 1000 procesadores.

La opción paralela descompone automáticamente la consulta en múltiples sub-operaciones (procesos o threads) que son ejecutadas en paralelo a través de los diferentes procesadores de la máquina o máquinas en paralelo. Este mecanismo es totalmente transparente para los programadores y usuarios. (Ver control del grado del Paralelismo). Esta nueva técnica, puede mejorar extraordinariamente el rendimiento de operaciones que manipulen gran cantidad de datos, en aplicaciones diseñadas a la ayuda de toma de decisiones o sobre grandes bases de datos (VLDBs). [7]

El servidor Oracle puede usar procesamiento en paralelo en las siguientes operaciones:

- Accesos secuenciales a tablas (Table scans) - Ordenaciones (ORDER BY,GROUP BY,joins,DISTINCT) - Uniones de tablas (Merge--Join,Nested--Loops) - Funciones de grupo (GROUP BY,MIN,MAX,AVG, etc.) En adición al procesamiento en paralelo de consultas, y dentro de la gestión paralela de operaciones, también se paralelizan las siguientes operaciones en la base de datos:

- Creación de índices.

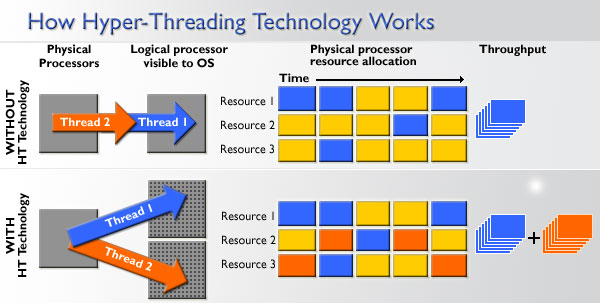
- Carga de datos(SQL\*LOADER).

**HyperThreading (HTT)**

La tecnología HyperThreading consiste en ‘simular’ dos procesadores en uno solo, dividiendo la carga de trabajo entre ambas y por tanto optimizando la celeridad de procesamiento.

La tecnología Hyper-Threading Intel® (Intel® HT) utiliza los recursos del procesador de manera más eficaz, posibilitando que se ejecuten múltiples subprocesos en cada núcleo. Como característica del desempeño, también aumenta la capacidad de procesamiento, lo que a su vez se traduce en un aumento del desempeño general del software de varios subprocesos. [8]

La tecnología Intel HT está disponible en los procesadores [Intel Core™ v Pro™](https://www.intel.la/content/www/xl/es/processors/vpro/core-processors-with-vpro-technology.html), los procesadores [Intel® Core™](https://www.intel.la/content/www/xl/es/processors/core/core-processor-family.html) los procesadores [Intel Core™ M](https://www.intel.la/content/www/xl/es/processors/core/core-m-processors.html) y los procesadores [Intel® Xeon®](https://www.intel.la/content/www/xl/es/servers/server-products.html)más recientes. [8] Al combinar uno de estos procesadores y chipsets Intel® con un sistema operativo y un BIOS compatible con la tecnología Hyper-Threading Intel, es posible:

* Ejecutar aplicaciones exigentes al mismo tiempo y mantener la capacidad de respuesta del sistema
* Mantener la protección, eficiencia y facilidad de administración de los sistemas, al tiempo que se reduce a un mínimo el impacto en la productividad.
* Aumentar la cantidad de operaciones que pueden procesarse de forma simultánea
* [](https://www.profesionalreview.com/wp-content/uploads/2017/02/Hyperthreading-5.jpg)Aprovechar las tecnologías de aplicaciones de 32 bits en uso y mantener el estado de preparación para las futuras aplicaciones de 64 bits

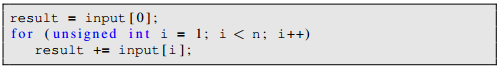
Diseño de tecnología Hyper-Threading 1

**ISSN : 2028-9324** **Vol. 18 No. 4, Dec. 2016** **1033**

**Algoritmo de Reducción Paralela**

El algoritmo de reducción paralela es un algoritmo de datos paralelos que se utiliza para resolver el problema de reducir todos los elementos en la matriz de entrada en un solo valor. Una parte integral del problema es una operación binaria asociativa que define cómo dos elementos de entrada se reducen en uno. [9]

Si la operación es una suma, multiplicación o valor máximo, la reducción paralela de todos los elementos da como resultado una suma, producto o máximo de la entrada completa, respectivamente. Tales cálculos se utilizan con frecuencia en muchas aplicaciones, por lo tanto, el rendimiento debe ajustarse al máximo. La implementación secuencial es sencilla: se necesita visitar todos los elementos y reducirlos a un valor único: la suma de la matriz de entrada:

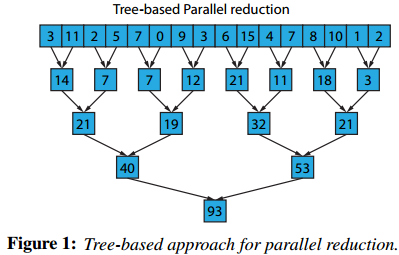


Aplicación del Algoritmo de reducción paralela

El código realiza n - 1 operaciones calculando la reducción secuencial con un número de pasos que se escala linealmente con el tamaño de la entrada. Le proporcionamos un esqueleto del código que ya contiene la funcionalidad de validar resultados y probar el rendimiento de implementaciones de GPU individuales.

**Implementación paralela**

Para dividir el cálculo en varias unidades de procesamiento, utilizaremos una reducción basada en árboles. Se debe tener en cuenta que, si el número de unidades de procesamiento es igual al tamaño de la entrada, podrá realizar la reducción en tiempo logarítmico. Para simplificar la implementación, solo consideraremos matrices con tamaños que sean iguales a las potencias de 2. Se pueden lograr matrices de manejo con un tamaño arbitrario mediante el relleno adecuado.



Enfoque basado en el esquema de árbol para división de cálculo.

Para realizar la reducción basada en árboles en una GPU, se puede usar un direccionamiento intercalado y siempre reduce dos elementos vecinos escribiendo el resultado nuevamente en uno de ellos. Siempre que no se necesite mantener el conjunto de GPU original, se puede realizar la reducción in situ directamente en la matriz de entrada colocada en la memoria global. Tener en cuenta que el resultado de un paso siempre se usa como entrada para el siguiente paso. Esto significa que se requiere sobre todo el NDRange, de lo contrario elementos de trabajo de un grupo podría realizar el próximo paso antes de que el paso actual sea terminado por otro grupo de trabajo.

**Procesamiento en paralelo de grandes volúmenes de información**

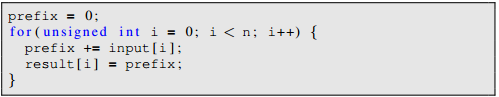
**Parallel Prefix Sum**

De manera similar a la reducción paralela, la suma de prefix paralela también pertenece a los populares algoritmos de datos paralelos. El PPS se usa a menudo en problemas tales como compactación de flujo, clasificación, recorridos eulerianos de un gráfico, cálculo de funciones de distribución acumulativa, etc. [9]

Dado un array de entrada X = [x0, x1, ... xn-1] y una operación binaria asociativa ⊕ una suma de prefijo inclusiva calcula una matriz de prefijos [x0, x0 ⊕ x1, x0 ⊕ x1 ⊕ x2, ..., x0 ⊕. . . ⊕ xn-1]. Si la operación binaria es suma, el prefijo de ith es simplemente una suma de todos los elementos anteriores más el elemento ith si queremos tener una suma de prefijo inclusivo.

**Implementación Secuencial**

La implementación secuencial es trivial: iteramos sobre todos los elementos de entrada y calcular cumulativamente los prefijos. Puede ser implementado como:



De forma similar a la reducción paralela, el esqueleto ya contiene todas las rutinas necesarias para asignar e inicializar OpenCL

**Implementación paralela**

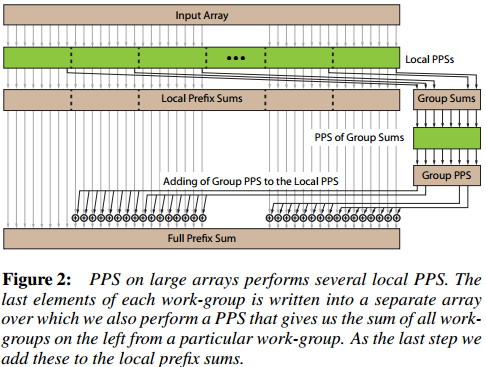
En cierto sentido, PPS es muy similar a la reducción paralela. Se realiza la operación binaria en forma de árbol, aunque en una manera más sofisticada.

Para lograr el mayor rendimiento posible, se debe descomponer el cálculo y realizar ambos barridos en la memoria local (el concepto es similar a la descomposición del kernel en la tarea de reducción paralela). [9]

Las ventajas de rendimiento deben ser claras, solo se debe resolver algunos problemas menores de optimización. Se recomienda empezar con una pequeña matriz de entrada que se ajuste a la memoria local. Se describe brevemente la extensión para arreglos grandes, pero para una implementación correcta, es crucial que el PPS local funcione sin errores.

El kernel debería comenzar cargando datos de la memoria global a la local. Para un grupo de trabajo de tamaño N, recomendamos cargar y procesar 2N elementos para evitar una mala utilización del hilo: si se usara solo N elementos, la mitad de los hilos quedarían inactivos durante el primer y último paso del barrido ascendente y descendente, que no es bueno para el rendimiento. Obviamente, necesita asignar memoria local que pueda contener 2N elementos. [9]

Teniendo los datos en la memoria local, se puede realizar el barrido ascendente dentro de un único ciclo for. Luego, se escribe explícitamente cero en el último elemento de la matriz de memoria local y se realiza un barrido descendente. No se debe olvidar usar barreras cuando sea necesario. Para calcular la versión inclusiva, solo necesita cargar los valores de la memoria global, agregarlos a los resultados en la memoria local y volver a escribirlos en la memoria global del dispositivo.



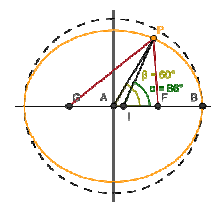
Arreglos PPS

**Algoritmo de conversión de coordenadas**

La entrada de esta etapa es un EPP, recibidas vienen definidas en latitud, longitud y altitud, comúnmente usadas en caso de las dos primeras, las unidades son grados, minutos y segundos. [10]

Es conveniente que se realice un cambio de coordenadas a cartesianas (x, y, z) puesto que los cálculos geométricos que describe el problema de la sincronización de trayectoria.

La salida que se espera de esta conversión que se recibe como entrada, pero en 3D (x, y, z), y manteniendo el resto de información recibida en el EPP. El algoritmo que se explicará con detalle a continuación aritméticas y trigonométricas simples para cada punto, usando los parámetros de diseño.



Conversión de Coordenadas

La conversión de coordenadas cuenta con buenas condiciones para ser paralelizable en cuanto a las operaciones a realizar: No tiene dependencia de datos entre ellas y no hay rehusó de datos.

Sin embargo, el paralelismo de datos debe ser tenido muy en cuenta en este caso, porque el número de puntos de entrada es bastante reducido.

**Algoritmos de interpolación**

Para obtener un gran número de puntos que describan la trayectoria de una aeronave con respecto al tiempo a partir del EPP compuesto de puntos con coordenadas cartesianas, esto se puede conseguir por medio de diferentes métodos de interpolación:

**Interpolación lineal**

La interpolación es un método sencillo y viable para obtener puntos intermedios de la trayectoria de una aeronave. Dados dos puntos conocidos de la trayectoria mediante un EPP recibido, la línea recta que los une aproxima los puntos intermedios que se quieren obtener. Se quiere obtener la interpolación de la posición de una aeronave en un tiempo t para x, y, z. Puesto que los 128 puntos recibidos en el EPP son por definición los puntos más destacables de la trayectoria (cambios de rumbo, de nivel, de velocidad, top of descent, top of climb), aun usando la sencilla interpolación lineal vamos a evitar grandes errores en la predicción de trayectoria. En este caso, la implementación de este algoritmo en GPU es eficiente. [10]

**Interpolación cuadrática**

A pesar de la buena aproximación que supone hacer una simple interpolación lineal, se puede mejorar el muestreo con respecto al verdadero movimiento de la aeronave incluyendo otras variables como la velocidad. [10]

El EPP brinda la velocidad estimada de la aeronave en cada punto de trayectoria, por lo que se puede mejorar el algoritmo anterior mediante un método de interpolación de 2º orden en el que se asume una aceleración constante. En caso de que la velocidad no sea disponible en alguno de los puntos del EPP, este método degenera en una interpolación lineal.

**Interpolación polinómica**

Para representar curvas es común el uso de polinomios por intervalos, cuya función es llamada spline. Esta técnica es usada habitualmente en informática y ha sido utilizada en el mundo aeronáutico por suavizar las trayectorias. Se usará la versión de splines cúbicas debido a que se obtienen resultados más adecuados a las curvas mediante polinomios de grado bajo y se evitan indeseables oscilaciones. La idea central es que en vez de usar un solo polinomio para interpolar los datos, se pueden usar segmentos de polinomios y unirlos adecuadamente (bajo ciertas condiciones de continuidad) para formar la interpolación. [10]

**ISSN : 2028-9324** **Vol. 18 No. 4, Dec. 2016** **1034**

* **DISCUSIÓN**

En la actualidad se continúa empleando los métodos tradicionales de procesamiento de datos un ejemplo de esto el proceso electrónico, para este tipo de proceso se emplean las computadoras, por lo cual la intervención humana no es requerida en cada etapa del procesamiento, una vez ingresados los datos, el computador efectúa los procesos requeridos automáticamente y emite el resultado deseado. Los procesos son realizados a velocidades altas, dependiendo de la cantidad de datos que se vaya a procesar un ejemplo práctico lo tenemos en el empleo de las computadoras personales que se usan en el hogar, para llevar los gastos diarios de la casa, realizar cálculos básicos y otras tareas.

A través de la aplicación las técnicas y los algoritmos de procesamiento paralelo para procesar grandes volúmenes de información se busca obtener mayores y mejores beneficios como la rapidez con la que se procesaran los datos y se obtendrá la información, además de mejor rendimiento del procesador. El funcionamiento y aplicación es diferente por un lado las técnicas de procesamiento en paralelo se pueden emplear en motores de base de datos que permiten reducir los tiempos de respuesta de las consultas que se realice o mediante la ejecución paralela de tareas con lo que se busca aprovechar al máximo la capacidades del procesador y reducir el tiempo de ejecución, en tanto los algoritmos de procesamiento en paralelo antes mencionados utilizan la GPU, que al contrario a la CPU, esta optimizada para trabajar con grandes cantidades de datos sobre los cuales se realiza las mismas operaciones de forma concurrente. Además, dispone cantidad de núcleos sencillos que se pueden ejecutar en paralelo para resolver problemas de una gran variedad de ámbitos como la inteligencia artificial o la minería de datos, la ejecución de un programa altamente paralelizable mediante el uso la GPU puede aumentar enormemente su rendimiento en comparación con la CPU. Estos programas normalmente realizan las mismas operaciones sobre una cantidad de datos muy elevada sin muchas dependencias entre ellos.

El uso de cualquiera de esas formas de procesamiento paralelo será decisión de quien las aplique ya que se deberá tener en cuenta cual de esas formas de procesamiento paralelo es las más óptimas y está orientada al trabajo que se esté llevando a cabo.

* **CONCLUSIÓN**

El manejo de grandes cantidades de datos es un gran reto para todas aquellas empresas hoy en día, y en un futuro, una gran oportunidad que le permita obtener ventaja sobre otras empresas. Cabe mencionar que en la actualidad se está remplazando la forma tradicional para realizar el procesamiento de información de una Big Data por nuevas técnicas y nuevos algoritmos que permiten almacenar y reorganizar dicha información, teniendo una mejor manipulación de los datos y proporcionando el menor tiempo posible de procesamiento. Todo esto es factible, gracias a las nuevas tecnologías desarrolladas para el área de la Big Data.

Se concluye además que, al utilizar procesamiento en paralelo, se obtiene una gran ventaja en cuanto a costos, puesto que resulta ser escalable, es decir que aumenta su capacidad de trabajo sin afectar su funcionamiento con un alto rendimiento. Otro beneficio de dicha forma procesamiento es que reduce el tráfico de bus, en el cual se distribuye paralelamente dependiendo del procesador que se utilice. El objetivo del procesamiento paralelo es acelerar el procesamiento de las computadoras, además elevar el rendimiento y la cantidad de datos se pueda procesar durante un intervalo de tiempo.

Las [técnicas](http://www.monografias.com/trabajos6/juti/juti.shtml) de procesamiento paralelo se usan con el propósito de realizar tareas simultáneas de procesamiento de [datos](http://www.monografias.com/trabajos11/basda/basda.shtml) con el fin de aumentar la [velocidad](http://www.monografias.com/trabajos13/cinemat/cinemat2.shtml#TEORICO) computacional de un [sistema](http://www.monografias.com/trabajos11/teosis/teosis.shtml) de [computadoras](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml). En lugar de procesar cada instrucción en forma secuencial como se realiza en una [computadora](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml) convencional, un sistema de procesamiento paralelo puede ejecutar procesamiento concurrente de datos para conseguir un menor [tiempo](http://www.monografias.com/trabajos901/evolucion-historica-concepciones-tiempo/evolucion-historica-concepciones-tiempo.shtml) de ejecución.

Habitualmente las GPUs se utilizan en la computación gráfica, pero también son una alternativa de uso muy buena para programas con alto nivel de paralelismo. Su aplicación en ámbitos como la ciencia, simulación o inteligencia artificial resulta muy atractiva. El hardware actual tiene un gran potencial para la computación de general, es imperioso el desarrollo de un programa que permita explotarlo debidamente, y esto comprende un estilo de programación totalmente diferente al habitual. La plataforma CUDA, junto con su gran cantidad de librerías, facilita el desarrollo de estos programas y, herramientas tales como NVIDIA Visual Profiler, apoyan a estudiar su procedimiento para conocer y perfeccionar su productividad.

# Referencias

|  |  |
| --- | --- |
| [1] | M. Naiouf, *Procesamiento Paralelo Balance de Carga Dinamico,* La Plata, 2004. |
| [2] | M. Naiouf, A. De Giusti, L. De Giusti, F. Chichizola, V. Sanz, A. Pousa, E. Rucci, S. Gallo, E. Montes de Oca, E. Frati, M. Sánchez, J. Basgall, M. Carolina Actis y A. Gaudiani, «Fundamentos de cómputo paralelo y distribuido para HPC. Construcción y evaluación de aplicaciones,» *Instituto de Investigación en Informática LIDI (III-LIDI),* pp. 1-6, 2011. |
| [3] | M. Mano, Arquitectura de Computadoras, Los Angeles: Pearson, 2010. |
| [4] | «nvidia,» NVIDIA, [En línea]. Available: http://la.nvidia.com/object/what-is-gpu-computing-la.html. [Último acceso: 10 Mayo 2018]. |
| [5] | J. Torres, «Hipertextual,» 18 Diciembre 2013. [En línea]. Available: https://hipertextual.com/archivo/2013/12/hardware-gpu-grafica/. [Último acceso: 8 Mayo 2018]. |
| [6] | A. Gaudiani, «Análisis del rendimiento de algoritmos,» 2012. |
| [7] | A. Gutiérrez, «computerworld,» [En línea]. Available: http://www.computerworld.es/archive/procesamiento-en-paralelo. [Último acceso: 15 Mayo 2018]. |
| [8] | Intel, «Intel,» Intel, [En línea]. Available: https://www.intel.la/content/www/xl/es/architecture-and-technology/hyper-threading/hyper-threading-technology.html. [Último acceso: 15 Mayo 2108]. |
| [9] | J. Novak, G. Liktor y C. Dachsbacher, «GPU Computing: Data-Parallel Algorithms,» *Karlsruhe Institute of Technology,* pp. 2-9, 2014. |
| [10] | E. DE LA IGLESIA RAMÍREZ, «SINCRONIZACIÓN PARALELA DE TRAYECTORIAS PARA DETECCIÓN DE CONFLICTOS ENTRE AERONAVES,» Madrid, 2013. |

**ISSN : 2028-9324** **Vol. 18 No. 4, Dec. 2016** **1038**