Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

Tipo de artículo: Artículo de revisión
Temática: Programación paralela y distribuida
Recibido: 23/01/2015 | Aceptado: 31/08/2015

Técnicas de programación paralela aplicadas al procesamiento de datos ráster mediante la biblioteca GDAL

Parallel programming techniques applied to raster data processing using GDAL library

Grethell Castillo Reyes [1*]

[1*]Universidad de las Ciencias Informáticas. Carretera a San Antonio de los Baños, km 2 ½, Torrens, La Lisa, La Habana, CP. 19370.

[*] Autor para correspondencia: gcreyes@uci.cu

Resumen

El modelo de datos ráster es uno de los modelos de datos geoespaciales comúnmente utilizado en el almacenamiento y análisis de información de la superficie terrestre. Generalmente, para realizar operaciones sobre este tipo de datos, se emplea la Biblioteca de Abstracción de Datos Geoespaciales, más conocida como GDAL, capaz de manejar alrededor de cien formatos de archivos ráster. El tiempo de respuesta de dicha biblioteca durante el análisis de los datos, se ha visto condicionado por la tendencia al aumento gradual del volumen de los mismos, gracias al continuo perfeccionamiento de las técnicas de obtención de datos de la superficie terrestre. Por esta razón, este trabajo se centra en el estudio crítico de las principales contribuciones que marcan su interés en la aplicación de técnicas de programación paralela al proceso de análisis de la información ráster con el fin de incrementar el rendimiento en términos de velocidad. Como resultado del estudio realizado, se logró determinar que la utilización de la computación voluntaria para el aprovechamiento de los recursos de *hardware* disponibles en las organizaciones y el uso de técnicas de programación paralela que permitan garantizar la heterogeneidad entre plataformas de cómputo, emergen como alternativas interesantes a combinar e incluir en el diseño de estrategias para el procesamiento paralelo de información ráster. Estas variantes resultan aplicables en las entidades dedicadas al análisis de información geoespacial en Cuba, teniendo en cuenta las limitaciones del entorno computacional que las caracterizan.

Palabras clave: computación voluntaria, heterogeneidad, modelo de datos ráster, técnicas de programación paralela

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

Abstract

The raster data model is one of the geospatial data models commonly used to store and analyze information of the Earth's surface. Generally, to perform operations over these data, it is used the Geospatial Data Abstraction Library, known as GDAL, capable of handling about a hundred raster file formats. The response time of these library during data analysis, has been conditioned by the trend of gradually increasing the volume of these, thanks to the continuous improvement of the technical data acquisition of the land surface. For this reason, this work focuses on the critical study of the major contributions that mark their interest in applying parallel programming techniques to process raster data analysis in order to increase performance in terms of speed. As a result of the study, it was determined that the use of volunteer computing for the utilization of hardware resources available in organizations and the use of techniques to ensure the heterogeneity between computing platforms, emerge as interesting alternatives to combine and include in the design of strategies for parallel processing of raster data. These variants are applicable in institutions dedicated to the analysis of geospatial information in Cuba, considering the limitations of the computational environment that characterize them.

Keywords: *heterogeneity, parallel programming techniques, raster data model, volunteer computing*

Introducción

El vertiginoso desarrollo de las nuevas tecnologías satelitales de alta resolución, la teledetección y el incremento de la fotogrametría terrestre y aérea para la digitalización de la superficie, han posibilitado la obtención de imágenes que contienen información geográfica utilizada con diversos fines. Entre las aplicaciones más significativas de las imágenes satelitales, es notable su uso en la agricultura, la desforestación, la hidrología, la minería, la cartografía y de manera general en la vigilancia del medio ambiente (Romero, 2006). La información obtenida mediante estas tecnologías es almacenada y manipulada a través de un modelo de datos denominado modelo ráster. Su estructura constituye, en esencia, una matriz bidimensional conformada por celdas o píxeles. Cada celda representa un valor numérico que describe una característica del terreno en ese punto (Smith et al., 2013) y que puede estar sujeta a diferentes transformaciones.

Dentro del campo de los Sistemas de Información Geográfica (SIG), la información georreferenciada puede ser analizada mediante distintos modelos de datos (Peralta, 2008), entre ellos el modelo de datos ráster es uno de los más utilizados. Este modelo es la base para un gran número de algoritmos de análisis de superficies, que permiten extraer

características propias del terreno. Los mismos, revisten importancia para un porciento numeroso de los SIG rectores de la comunidad internacional.

Una solución ampliamente utilizada por estos sistemas para el procesamiento de datos geoespaciales en formato ráster es la Biblioteca de Abstracción de Datos Geoespaciales (GDAL). Entre los tipos de análisis ráster que son posibles realizar mediante la GDAL, son notables los algoritmos para el estudio de la inclinación de una superficie, útiles por ejemplo, en la determinación de zonas de poca pendiente favorables para la construcción, o zonas de mucha pendiente que determinan erosión o deslizamientos de tierra (Rodríguez and Suárez, 2010). Además, destacan los algoritmos para la generación de mapas de sombra y de rugosidad, empleados en la percepción de la profundidad de una superficie en tres dimensiones (Jenny, 2001) y para determinar la variabilidad de un relieve en un entorno determinado respectivamente (Seitavuopio et al., 2005). Actualmente, estos procesos se ven afectados indistintamente por dos limitantes fundamentales que giran en torno a la tendencia en el crecimiento del volumen de los datos ráster y la forma de procesamiento de estos mediante la biblioteca GDAL.

Las investigaciones realizadas por (Nikolakopoulos et al., 2006) demuestran que con los avances tecnológicos la disponibilidad de datos de superficies del terreno almacenados en formato ráster ha ido en aumento sostenido y cada vez con un mayor nivel de resolución espacial y precisión. Este elemento es directamente proporcional al tamaño que pueden llegar a alcanzar dichos datos, lo cual tiene sus consecuencias. A medida que aumenta la resolución y precisión, entonces mayor tamaño tendrá el modelo utilizado para el almacenamiento, ya que el número de celdas necesarias crece significativamente. Teniendo en cuenta esto, se considera, que uno de los problemas cuando se trata de procesar datos ráster, es que precisamente el tamaño alcanzado por estos suele ser significativo, incluso en término de gigabytes (Gao et al., 2013).

Adicionalmente, como segunda limitante, los datos de gran tamaño pueden condicionar el tiempo de respuesta del actual modelo de procesamiento de la biblioteca GDAL. Mediante este modelo, el acceso y la modificación de los datos se realiza cargando en la memoria principal fila por fila de un *dataset* ráster de manera iterativa. Por cada una de las filas se realizan las transformaciones pertinentes y se escriben los datos actualizados en el *dataset* de salida. Esto propicia que mientras mayor sea el número de filas y columnas en la matriz a procesar, mayor costo computacional se necesita para transformar los datos, lo que se traduce en aumento del tiempo de procesamiento y por lo tanto pérdida en términos de rendimiento.

En este sentido, con la finalidad de incrementar el rendimiento en velocidad de las aplicaciones, en los últimos años los procesadores han migrado hacia el paralelismo como un cambio importante en busca de este factor (Fraire et al.,

2013). Por su parte, los paradigmas de la programación paralela están estrechamente condicionados por la arquitectura de *hardware* y la infraestructura de software con la que se cuenta, lo que ha dado lugar al surgimiento de la computación de altas prestaciones y con ello a las supercomputadoras. Sin embargo, en el caso de Cuba, no abundan los entornos con terminales de este tipo que brinden un desempeño elevado. Por lo general, las organizaciones dedicadas al procesamiento de información geoespacial en el país, se caracterizan por poseer una infraestructura de red compuesta por servidores de bajas prestaciones, a los que se accede simultáneamente para realizar operaciones que en ocasiones suelen ser costosas y afectan la ejecución de otras peticiones.

Por esta razón, se profundiza en el análisis de los principales referentes teórico-prácticos de la bibliografía, que centran su atención en el desarrollo de soluciones para procesar la información geográfica almacenada en formato ráster aplicando técnicas de programación paralela. Además, se exponen los principales conceptos que sustentan la investigación y contribuyen a la comprensión de su contenido. El objetivo principal es realizar un análisis crítico de estos trabajos, en base a desarrollar un método que pueda hacer frente a las limitaciones actuales de las entidades cubanas que se sustentan en *hardware* de bajas prestaciones para el manejo de la información georreferenciada.

Desarrollo

El modelo de datos ráster

Un modelo puede definirse como una representación parcial y abstracta de la realidad. En el caso de los datos geográficos, (Longley et al., 2011) se refieren a modelo de representación geoespacial, como un *"(…) conjunto construido para la descripción y representación del aspecto de los objetos del mundo real en el ordenador (…)*". Estos modelos son usados en diversos campos de aplicación de la Geomática[1], entre los que destacan los SIG.

En el modelo ráster, según (Peralta, 2008), la información geográfica es representada en celdas o píxeles, generalmente cuadrados – aunque pueden utilizarse triángulos o hexágonos – ordenados conformando una estructura matricial conocida como malla regular, en la que cada celda tiene un valor y una localización determinada, como se muestra en la Figura 1. La malla regular se conoce en la bibliografía como estructura ráster. Mediante este modelo, la localización de las entidades geográficas se define como la referencia directa a la matriz de datos en la que cada celda está asociada a una porción del territorio (Llopis, 2008).

[1] Conjunto de ciencias donde se integran los medios para la captura, tratamiento, análisis, interpretación y almacenamiento de información geográfica a través de la informática.

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

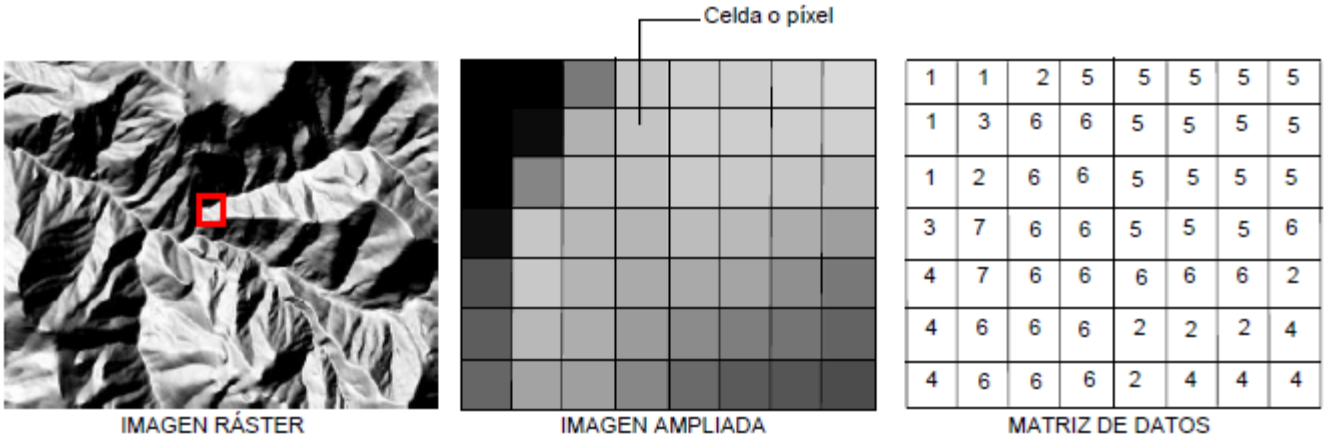

Figura 1. Esquema de representación de la información en el modelo ráster

Tamaño de celda y resolución espacial

La celda es la unidad mínima de información de un ráster, por lo tanto, su tamaño representa la precisión con la que son definidos los elementos geográficos en el modelo (Llopis, 2008). A su vez la resolución espacial de una superficie representada en un ráster, depende en gran medida del tamaño de las celdas de la matriz de datos, dado en metros sobre el terreno (ESRI, 2012). Cuanto menor sea el tamaño de dicha celda y por ende de la zona representada, mayor es el número de celdas que se representarán mediante el ráster. En el manual publicado por el Centro de Recursos de ArcGIS (ESRI, 2012), se afirma que los tamaños de celda más pequeños en modelos ráster grandes representan una superficie completa, por lo tanto, se necesita un espacio de almacenamiento mayor, elemento que implica también mayor tiempo de procesamiento de los datos.

Con el perfeccionamiento de las técnicas de adquisición de datos de la superficie terrestre (Max, 2005), la información almacenada en los modelos ráster es cada vez más detallada. Este hecho está dado como consecuencia de que el tamaño de las celdas en la matriz de datos se hace más pequeño, adquiriendo de esta forma mayores niveles de resolución espacial. La gráfica de la Figura 2 fue elaborada a partir de un estudio realizado sobre las características de algunas de las imágenes ráster obtenidas como resultado de la puesta en órbita de satélites de alta resolución y muestra la tendencia en el aumento de la misma en este tipo de dato.

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

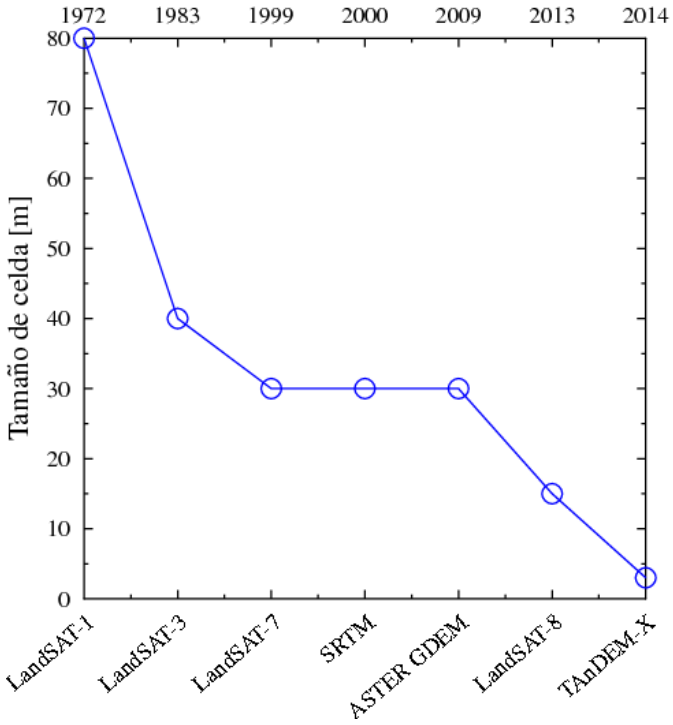

Figura 2. Tendencia en el aumento de la resolución espacial de las imágenes satelitales de la superficie terrestre[2].

Según la información representada, la evolución comienza en el año 1972, con el lanzamiento a la órbita de la primera versión de la serie de satélites LANDSAT, que logra obtener una imagen de la superficie terrestre con un nivel de resolución de 80m, resultado que fue mejorado posteriormente por la tercera versión de esta serie con una resolución de 40m. En los años 2000 y 2009, los proyectos SRTM y ASTER GDEM alcanzan un nivel de resolución de 30m cubriendo el 80% y el 99% de la Tierra respectivamente. Estos resultados son superados por la versión 8 de LANDSAT, al obtener imágenes con 15m de resolución. Según la revisión bibliográfica realizada hasta el momento, los resultados más recientes del año 2014 fueron arrojados por el satélite TAnDEM-X, puesto en órbita desde el año 2010 con el objetivo de obtener un Modelo Digital de Elevación (MDE) del 100% de la superficie terrestre, lo que se traduce en 149 millones de kilómetros cuadrados. La resolución alcanzada por este sensor supera a las anteriores, con un nivel de detalle de 3m sobre el terreno.

[2] La información mostrada en la gráfica constituye una recopilación de datos de los siguientes sitios oficiales: http://landsat.gsfc.nasa.gov/,https://directory.eoportal.org/web/eoportal/satellite-missions/t/tandem-x, http://asterweb.jpl.nasa.gov/gdem.asp, ttp://www2.jpl.nasa.gov/srtm/

Como conclusión del estudio realizado, es evidente que la disponibilidad de información ráster de la superficie terrestre es amplia y su perfeccionamiento no se detiene, elemento que conlleva a innumerables consecuencias benéficas en la rama de la Geomática, sobre todo para su integración con SIG, pero que también tiene su contraparte, pues exige al unísono el perfeccionamiento de las técnicas utilizadas para su procesamiento y análisis.

Procesamiento ráster con la GDAL

A partir de la creciente masividad de los datos geoespaciales almacenados en estructuras ráster, han surgido diversas teorías sobre cómo llevar a cabo su procesamiento. Varias de ellas (Guan and Clarke, 2010) (Maulik and Sarkar, 2012) (Zhan and Qin, 2012), plantean una limitante fundamental que existe en cuanto al tema: la diversidad de formatos ráster que han surgido como resultado de su notable utilización como modelo de datos geoespaciales y la dificultad que existe en el procesamiento homogéneo de estos.

Con el objetivo de lograr el acceso genérico - ya sea para la lectura, escritura o traducción - de la variedad de formatos de datos ráster existentes, en el año 1998, Frank Warmendan (Warmerdam, 2008) introduce la biblioteca GDAL. Una de sus características fundamentales es que presenta un único modelo de datos abstracto que permite manipular la mayoría de formatos de datos ráster, pero que además se basa en el enfoque de la extensibilidad permitiendo la implementación de nuevos formatos en caso de que así se requiera. Como parte de sus utilidades, cuenta con un conjunto de herramientas de línea de comandos para el procesamiento ráster con diferentes propósitos (GDAL, 2014).

Proceso secuencial de los datos

Generalmente, un amplio rango de los algoritmos para el procesamiento ráster se ejecutan de forma secuencial (Qin et al., 2014a), o sea, durante su ejecución, una instrucción no comienza hasta que no termine la anterior. Este es el caso de los algoritmos para el procesamiento ráster que implementa la GDAL. La Figura 3 representa un esquema que describe el flujo secuencial de las operaciones realizadas por la biblioteca durante el procesamiento de los datos.

La primera operación que se ejecuta para procesar los datos con la GDAL, es el registro de los manejadores de formatos. La biblioteca cuenta con un manejador para cada formato de datos soportado, incluyendo los nuevos formatos que sean incluidos. En este caso, se registran todos los manejadores, con el objetivo de asegurar su disponibilidad en las operaciones siguientes. Posteriormente, se realiza la lectura de los metadatos del *dataset* ráster a procesar. En (GDAL, 2014) se definen como metadatos: el formato de archivo en el que están almacenados los datos, la dimensión de la estructura matricial, las bandas que componen el ráster, el sistema de referencia espacial que utiliza, así como su extensión espacial. A partir de estos metadatos se crea el *dataset* de salida.

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

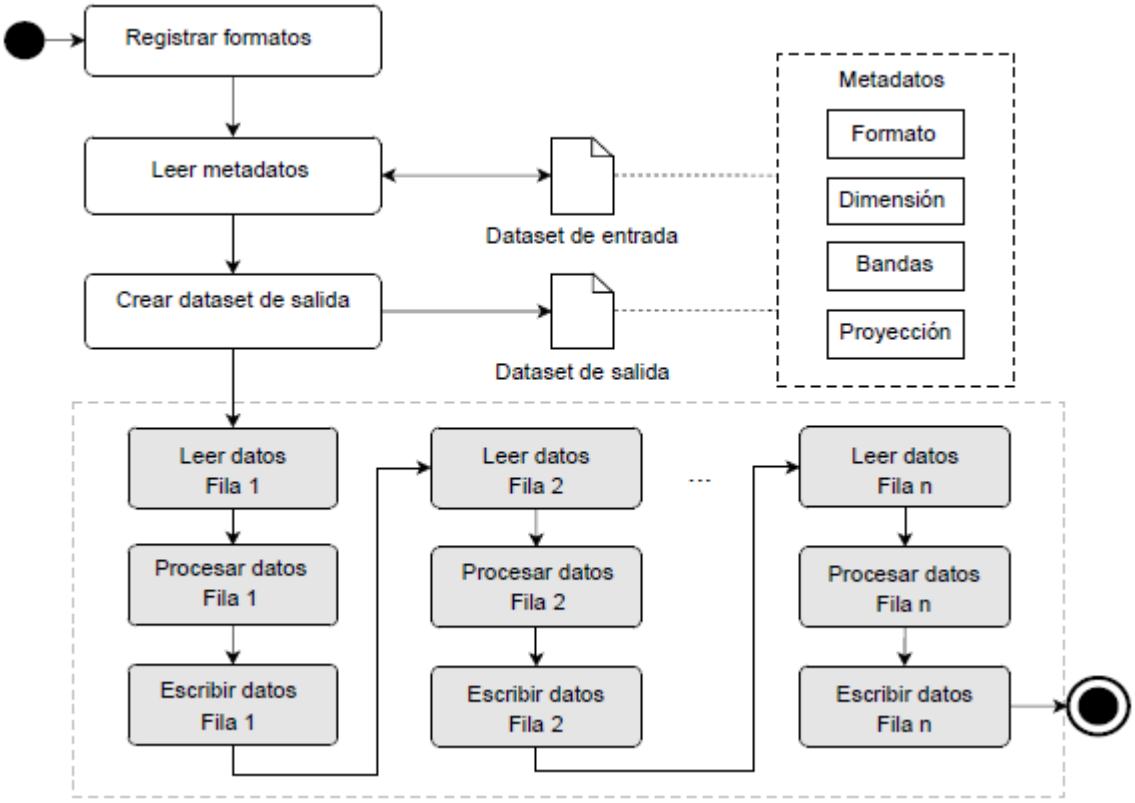

Figura 3. Esquema del procesamiento secuencial de datos ráster con la biblioteca GDAL.

Una vez realizadas las operaciones anteriores, se procede a la lectura de los datos almacenados en la estructura matricial, su procesamiento y luego su escritura en el *dataset* de salida. A los efectos de esta investigación, estas secuencias de operaciones son las más importantes. Precisamente esto se debe a que, independientemente del tamaño de la estructura a analizar, la GDAL realiza el procesamiento de los datos fila por fila, como se ilustra en el esquema de la figura anterior. Para ello ejecuta una serie de iteraciones recorriendo la matriz de entrada. En cada iteración, por cada una de las filas realiza la lectura, procesamiento y escritura de los datos.

A pesar de que varios autores (Armstrong and Densham, 1992; Wagner and Scott, 1995; Guan, 2008; Guan, 2009; Guan and Clarke, 2010; Zhan and Qin, 2012; Guan et al., 2013; Ouyang et al., 2013; Qin et al., 2014a; Qin et al., 2014b) se han inclinado por analizar y mejorar el procesamiento de los datos ráster en cuestiones de rendimiento, existe poca literatura publicada donde se realice un análisis profundo con respecto al comportamiento de la GDAL en relación con el tiempo de procesamiento secuencial a medida que aumenta la resolución y el tamaño de los datos que

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

son analizados. Por esta razón, se decidió realizar pruebas que demuestren lo anteriormente expuesto. La Tabla 1 resume los resultados obtenidos.

Las pruebas fueron realizadas en dos entornos con las siguientes configuraciones de *hardware*:

- Entorno 1 (E1): HP Pavilion g4-1174la con 4 GB de memoria RAM, un procesador AMD Dual-Core A4-3300M APU a 1.90 GHz y una tarjeta gráfica integrada Radeon HD 6480G de 512 MB.
- Entorno 2 (E2): ASUS x550L con 4 GB de memoria RAM, un procesador Intel Core i5-4200U APU a 1.6 - 2.6 GHz y una tarjeta gráfica integrada Intel HD Graphics 4400 de 1600 MB.

Se procesaron cuatro MDE utilizando la herramienta *gdaldem*[3], con el objetivo de generar los mapas de sombra y rugosidad correspondientes a cada uno de los modelos proporcionados. Los tiempos de respuesta obtenidos, son el resultado del promedio de los tiempos arrojados en la ejecución de varias iteraciones por cada uno de los modelos utilizados.

Tabla 1. Pruebas experimentales del procesamiento secuencial con la GDAL.

Modelo	Tamaño (n x m)	Tiempo de procesamiento (s)			
		Mapa de sombra		Mapa de rugosidad	
		E1	E2	E1	E2
Modelo 1	2049 x 2049	8	5	6	5
Modelo 2	4097 x 4097	36	17	30	16
Modelo 3	21601 x 10801	97	55	88	30
Modelo 4	86404 x 28804	1 122	552	1 086	523

A partir de los resultados obtenidos se deduce que la deficiencia de este mecanismo, radica en que independientemente de las características de *hardware* de los entornos de procesamiento, mientras mayor sea la resolución y el tamaño del ráster, entonces la cantidad de filas y columnas - y por consiguiente las celdas - de la matriz a procesar aumenta, por lo que se incrementa el costo de cómputo y el tiempo de procesamiento, disminuyendo el rendimiento del proceso en cuestión.

Uno de los enfoques de solución más utilizados cuando se trata el problema de procesar grandes volúmenes de datos en el menor tiempo posible, se basa en explotar al máximo las posibilidades de ejecución en paralelo que brindan las arquitecturas multinúcleos actuales, siempre y cuando las instrucciones a ejecutar y los datos a procesar así lo

[3] Herramienta de la GDAL para analizar y visualizar MDE.

permitan. Como plantean (Guan and Clarke, 2010) en su teoría, desde la perspectiva de computación, el procesamiento de datos ráster puede ser paralelizado, dado que la estructura matricial utilizada para su almacenamiento puede ser particionada en varias submatrices y asignadas a múltiples procesadores para su cómputo.

Elementos de computación paralela

A partir de la demanda de problemas que exigen un costo computacional elevado para su resolución, ha proliferado la existencia de computadores capaces de solucionar estos problemas en un tiempo razonable. En este sentido, la velocidad de procesamiento de los computadores secuenciales se ha incrementado continuamente para adaptarse a las necesidades actuales, pero se han alcanzado los límites físicos en el incremento de su capacidad para resolver problemas con alto costo computacional.

Guiado por la insaciable demanda de un poder computacional superior, como solución a la problemática anterior surge el concepto de paralelismo, que como plantea (Gove, 2011), su esencia consiste en usar múltiples recursos computacionales simultáneamente para resolver un problema dado, con el fin de lograr menor tiempo de procesamiento.

Arquitecturas paralelas

Existen dos variantes principales para el intercambio de datos en entornos paralelos (Petersen and Arbenz, 2004) (Breshears, 2009) (Gebali, 2011), como muestra la Figura 4.

- Arquitectura de memoria compartida: Según (Petersen and Arbenz, 2004), mediante esta arquitectura todos los elementos de proceso acceden al mismo espacio de memoria común, Figura 4a. El acceso a la memoria debe ser controlado por los propios algoritmos a través de diferentes métodos de sincronización. La comunicación entre los procesadores se realiza utilizando la memoria, en el momento en que un proceso determinado escribe en un espacio de memoria y éste es leído por otro proceso. Bajo esta arquitectura, se han realizado diferentes implementaciones como es el caso de *Pthreads* (Lewis and Berg, 1997), *Threading Building Blocks* (Contreras and Martonosi, 2008) y el estándar *Open Multi – Processing* (OpenMP) (Dagum and Menon, 1998) (Chapman and Huang, 2007) utilizado en diversas investigaciones relacionadas con el procesamiento de información geográfica en entornos paralelos (Chen et al., 2013) (Cui and Zhang, 2013).
- Arquitectura de memoria distribuida: Mediante esta variante, cada procesador está asociado a un espacio de memoria físico propio o local (Gebali, 2011) y existe una red de interconexión que une los procesos, Figura 4b. Ante los usuarios, los sistemas implementados bajo esta arquitectura, aparecen como una única máquina para resolver un determinado problema. En este caso, la comunicación entre los procesadores se realiza

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

mediante el mecanismo de paso de mensajes. Este mecanismo se encuentra implementado a través de diversas bibliotecas como es el caso de la Máquina Virtual Paralela (PVM) (Geist et al., 1994) y PARMACS (Calkin et al., 1994). En los últimos años, son varias las investigaciones referidas al procesamiento de datos ráster en paralelo que optan por el uso de la Interfaz de Paso de Mensajes (MPI) (Guan and Clarke, 2010; Zhan and Qin, 2012) debido a su actualidad y disponibilidad.

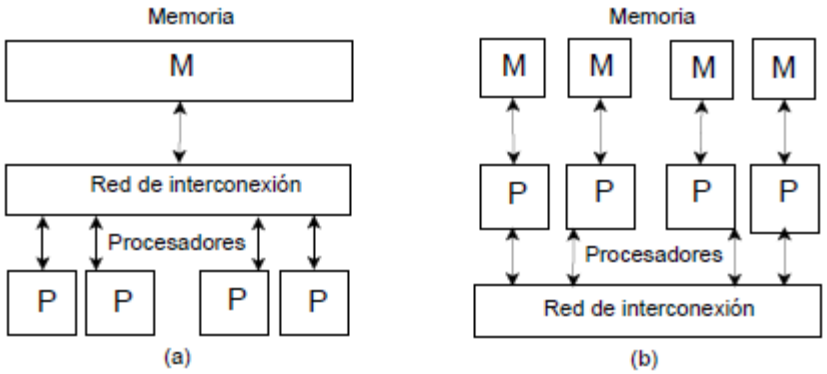

Figura 4. Variantes principales para el intercambio de datos en entornos paralelos. (a) Arquitectura de memoria compartida. (b) Arquitectura de memoria distribuida.

Sistemas distribuidos por computación voluntaria

Una de las variantes de implementación de sistemas distribuidos, es la computación voluntaria, a través de la cual el procesamiento se realiza en terminales identificadas en la red y no en máquinas bajo el control y la gestión de un proyecto determinado (Beberg et al., 2009) como es el caso de los clúster de computadoras de alto rendimiento. Esta tecnología, toma ventaja del tiempo de inactividad de una PC, ofrecida como voluntaria, para utilizar el poder de su Unidad Central de Procesamiento (CPU) y hacer simulaciones significativas (Kwesi and Amoako-Yirenkyi, 2008). El esquema básico de funcionamiento de la computación voluntaria está compuesto por un servidor encargado de administrar la distribución del procesamiento y que mantiene comunicación periódica con los voluntarios para informar nuevas tareas y obtener los resultados de las ya completadas.

Varios autores (Sarmenta, 2001; Anderson and Fedak, 2006; Kwesi and Amoako-Yirenkyi, 2008; Beberg et al., 2009), plantean que mediante este enfoque se proporciona una potencia de cálculo y procesamiento elevada, lo que permite reducir el tiempo requerido por las aplicaciones en condiciones normales de ejecución, obteniendo resultados con alto rendimiento. En los estudios realizados por Anderson y Fedak (Anderson and Fedak, 2006) se demuestra que

el potencial de este tipo de sistemas, se extiende mucho más allá de las tareas intensivas de la CPU y abarca aplicaciones que requieren elevado uso de la memoria y el disco.

Uno de los factores que a los efectos de esta investigación y a consideración de la autora, constituye una de las características más relevantes de la computación voluntaria, es que dicha tecnología emerge como una vía económica para la implementación de sistemas distribuidos para el procesamiento paralelo en entornos de bajas prestaciones, si se tiene en cuenta que el poder de cómputo no recae sobre un grupo de computadoras de altas prestaciones que conforman un clúster dedicado, sino que utiliza el poder de cálculo y procesamiento de los voluntarios distribuidos a través de una red.

GPU: Plataformas de cómputo de propósito general

Por otro lado, impulsado por la demanda del mercado de los gráficos en tres dimensiones, el procesamiento de imágenes en tiempo real y la creciente industria de los video juegos, las Unidades de Procesamiento Gráfico (GPU) han evolucionado hasta convertirse en elementos significativos en comparación con las CPU, tomando gran ventaja sobre éstas en términos de capacidades de cómputo (NVIDIA, 2011).

Dado que las GPU poseen una arquitectura que toma como base la naturaleza altamente paralela de las operaciones de cómputo asociadas a la producción de gráficos por computadora, dedicando la gran mayoría de sus componentes a la realización de grandes volúmenes de cálculos matemáticos (Martínez, 2011), son utilizadas como plataformas de cómputo de propósito general, o *General Purpose GPU* (GPGPU). Este paradigma ha generado un creciente interés por parte de la comunidad científica dedicada a investigaciones (Lv et al., 2012; Steinbach and Hemmerling, 2012; Gao et al., 2013) relacionadas con el procesamiento de datos geográficos en paralelo.

A partir de la evolución de las GPU han surgido bibliotecas y plataformas que permiten el desarrollo de aplicaciones GPGPU. Este es el caso de CUDA, arquitectura de cómputo paralelo de propósito general, que incluye un modelo de programación y un conjunto de instrucciones que aprovechan el motor de cálculo paralelo de las tarjetas gráficas NVIDIA para resolver problemas computacionales complejos en una manera más eficiente (NVIDIA, 2011). Luego del surgimiento de CUDA y a partir de la colaboración entre el Grupo Khronos[4] y AMD[5] surge OpenCL, estándar abierto para la programación de colecciones heterogéneas de CPU, GPU y otros procesadores (Munshi et al., 2012).

[4] Consorcio industrial enfocado en la implementación de estándares abiertos para la creación y aceleración de la computación paralela y los gráficos.
[5] Compañía dedicada al desarrollo de procesadores de cómputo.

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

Varios autores (Komatsu et al., 2010) (Daga et al., 2011) (Fraire et al., 2013) coinciden en que una de las diferencias más notables entre OpenCL y CUDA, es que CUDA, al ser dependiente de la plataforma y del fabricante, hace difícil que el desarrollador aproveche completamente la potencia disponible en los sistemas modernos de computación. Sin embargo, OpenCL fue diseñado para explotar al máximo el paralelismo dentro de un mismo dispositivo, permitiendo la ejecución de tareas en múltiples núcleos de múltiples dispositivos, independientemente de la plataforma y su fabricante. Su objetivo, es lograr códigos eficientes y portables al mismo tiempo.

Estrategias de descomposición del dominio de los datos

Para obtener beneficios en un entorno de procesamiento paralelo, (Armstrong and Densham, 1992) plantean que se deben concebir procedimientos, a partir de los modelos secuenciales, en los que el paralelismo pueda ser explotado al máximo. Este proceso es conocido como descomposición de dominio, y se considera un elemento fundamental en el procesamiento de datos ráster en paralelo.

En la descomposición de dominio, los datos se dividen en subdominios y cada una de las particiones obtenidas es asignada a un procesador, donde se ejecuta el algoritmo secuencial correspondiente (Wagner and Scott, 1995) utilizando los datos del subdominio proporcionado. En los algoritmos paralelos para procesar información ráster, comúnmente los datos se descomponen en subdominios rectangulares utilizando tres estrategias comunes: descomposición por filas, descomposición por columnas o descomposición en bloques (Qin et al., 2014a) Figura **5**.

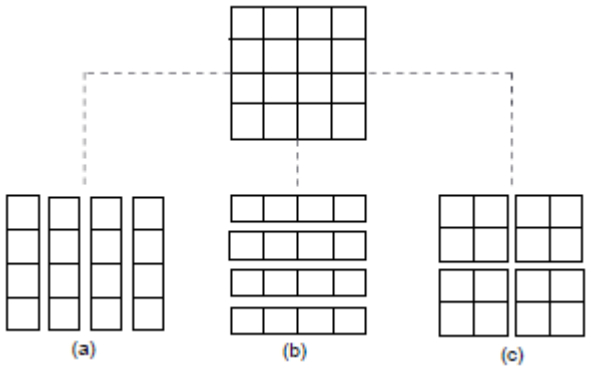

Figura 5. Principales estrategias de descomposición de dominio de los datos ráster. (a)Por columnas. (b) Por filas. (c) En bloques

En su investigación, (Wagner and Scott, 1995) plantean que el rendimiento de un algoritmo paralelo con respecto a la descomposición de los datos, depende en gran medida de la correspondencia que exista entre el tamaño de los subdominios obtenidos y el balance en la asignación de estos subdominios a cada uno de los elementos de procesamiento. En este sentido, el equilibrio entre los procesadores se logra si se particionan los datos en subdominios

del mismo tamaño, de manera que se logre uniformidad en la distribución y en consecuencia, que algunos procesadores no operen con mayor cantidad de datos que otros. Por su parte, los resultados obtenidos en experimentos realizados por (Guan and Clarke, 2010) demostraron que los enfoques de descomposición por filas y columnas son más adecuados que la descomposición por bloques en cuanto a rendimiento, tiempo de ejecución y eficiencia.

Tendencias del procesamiento ráster en paralelo

A partir del notable avance en las Ciencias de la Información Geográfica (GISciences), el incremento en el volumen y resolución de los datos geoespaciales almacenados en formato ráster, y la complejidad de los algoritmos y métodos utilizados para su procesamiento, han sido fundamentadas y aplicadas diversas teorías dentro del campo de la computación paralela con el fin de lograr mejores resultados si de rendimiento se trata. En esta sección se muestra el resultado de un análisis realizado sobre las principales investigaciones en el área del procesamiento de modelos ráster en entornos paralelos. Aunque en la presente investigación son de mayor interés las teorías relacionadas con el procesamiento en paralelo de datos ráster haciendo uso de la biblioteca GDAL, sí resulta importante destacar los trabajos que han marcado hitos en el área del procesamiento de modelos ráster en paralelo de manera general.

En (Guan, 2009; Guan and Clarke, 2010; Guan et al., 2013) los autores exponen el desarrollo de una biblioteca de programación denominada *parallel Raster Processing Library* (pRPL) aplicada a la aceleración de un modelo de un autómata celular. Por su parte en (Ouyang et al., 2013) se propone la utilización de una arquitectura de acceso paralelo a la información ráster basada en la utilización de MPI. Bajo el mismo concepto de utilizar MPI, en (Wang et al., 2012) se desarrolló una arquitectura para el procesamiento en paralelo de imágenes de teledetección, haciendo énfasis en las imágenes multiespectrales por el alto costo computacional que requieren.

En (Xia et al., 2011), los autores proponen una metodología genérica basada en la utilización de la GPU y el modelo de programación CUDA en el análisis geoespacial a través de algoritmos de interpolación. El aporte expuesto en (Lv et al., 2012) se enfoca en la utilización de la GPU para el diseño de algoritmos paralelos que aceleren el análisis geoespacial de la pendiente o inclinación de una superficie terrestre. Por su parte (Steinbach and Hemmerling, 2012) presentan una estrategia eficiente para el almacenamiento en caché de los datos ráster y la aceleración de su procesamiento utilizando la GPU.

Principales aproximaciones al procesamiento paralelo de la GDAL

Actualmente se pueden encontrar varios intentos de implementar el procesamiento de datos ráster con la biblioteca GDAL aplicando técnicas de programación paralela. Aunque no son tan abundantes las teorías abordadas, se

evidencia el interés de la comunidad científica sobre el área. Este es el caso de las investigaciones realizadas por (Zhan and Qin, 2012; Qin et al., 2014a; Qin et al., 2014b).

La principal contribución de (Zhan and Qin, 2012) consiste en el diseño de dos variantes, teniendo en cuenta las tres estrategias de descomposición de dominio de los datos ráster. En ambos casos se utilizó en la implementación el modelo de programación MPI.

En una primera variante de solución, (Zhan and Qin, 2012) plantean la utilización de un proceso maestro encargado de almacenar todos los datos en la memoria del sistema, con el objetivo de que otros hilos de procesos accedan a los datos a través de la comunicación con el proceso maestro. La principal deficiencia de esta variante se presenta cuando el ráster a procesar excede en tamaño la capacidad de la memoria disponible en alguno de los nodos de cómputo.

A diferencia de esta variante, el segundo modelo que propone (Zhan and Qin, 2012) contempla la posibilidad de que cada proceso pueda leer y escribir datos en los archivos de entrada y salida respectivamente, como muestra la Figura 6. En este caso, el proceso maestro utiliza la biblioteca GDAL para obtener los metadatos útiles en el procesamiento, por ejemplo, la extensión espacial y la proyección. A partir de estos datos crea el archivo ráster de salida correspondiente. Luego, en concordancia con alguna de las estrategias de descomposición de dominio envía la información extraída a los procesos de trabajo. Basándose en la información del subdominio recibida, cada uno de los procesos de trabajo lee el subdominio correspondiente, realiza los cálculos y una vez obtenidos los resultados utiliza la GDAL para abrir y escribir dichos resultados en el archivo ráster de salida.

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

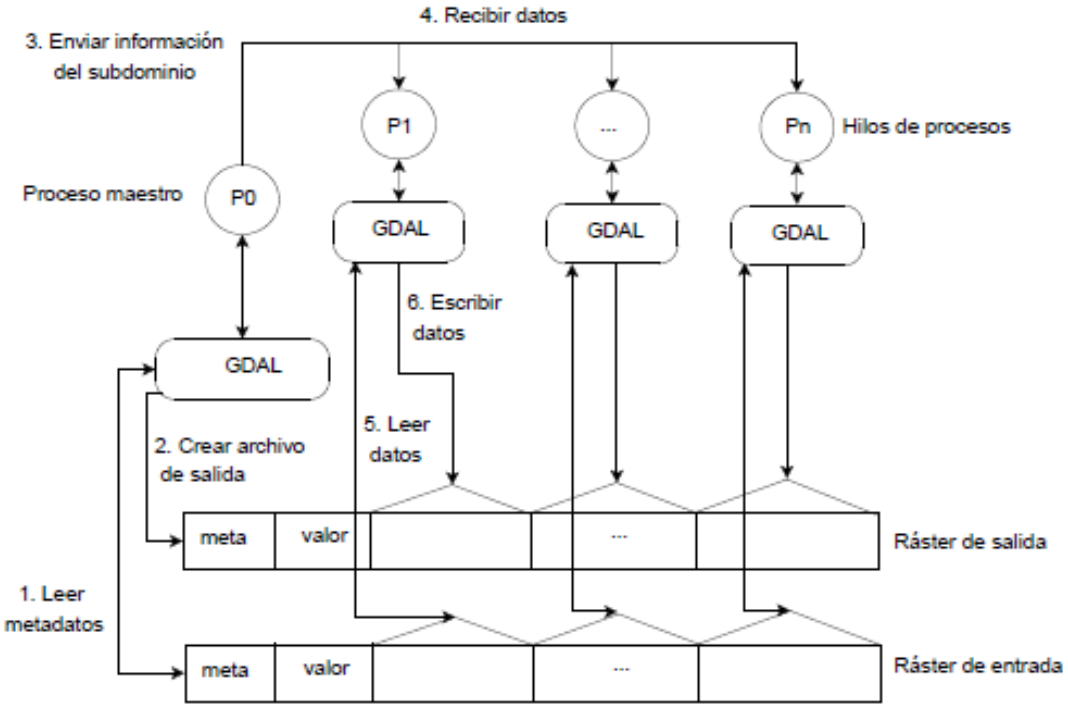

Figura 6. Esquema de E/S en paralelo para el procesamiento de datos ráster en paralelo utilizando GDAL propuesto por (Zhan and Qin, 2012)[6].

Para este caso, la dificultad radica en la carencia de flexibilidad en cuanto a la estrategia de descomposición de dominio y los formatos de archivo ráster que se emplean, pues los resultados obtenidos en el procesamiento no siempre son confiables cuando se utiliza la descomposición por columnas o por bloques, o cuando los datos ráster están almacenados en formato de imagen.

Los experimentos aplicados por (Qin et al., 2014b) permitieron corroborar que las deficiencias presentadas por la propuesta son atribuibles al mecanismo de almacenamiento en caché que implementa la GDAL. La dimensión del bloque de caché en GDAL es a menudo igual a la dimensión de la trama a procesar, lo que provoca que se obtengan resultados incorrectos cuando diferentes procesos intentan escribir en la misma región de un archivo ráster compartido. Esta característica afecta el procesamiento en paralelo en el caso de que los datos se dividan en columnas o en bloques, pues al realizar la lectura de los datos ráster estos son reorganizados como un flujo lineal contiguo de valores, que se inicia desde la celda de la esquina superior izquierda en el matriz ráster y termina con la celda de la

[6] Fuente: Elaboración propia basada en la figura 2 de (Zhan and Qin, 2012).

Revista Cubana de Ciencias Informáticas
Vol. 10, No. 1, Enero-Marzo, 2016
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 71-94

esquina inferior derecha. Como resultado, cada proceso debe acceder a varias secciones no contiguas del archivo ráster.

A partir de la determinación de las causas que provocan estos resultados incorrectos, (Qin et al., 2014b) propone el diseño de un módulo de redistribución de los datos. Este enfoque utiliza una estrategia de comunicación entre procesos para redistribuir los datos, de manera que cada proceso compute un trozo contiguo de datos.

Las investigaciones analizadas hasta el momento se enfocan en la utilización de MPI. Aunque la aplicación de esta teoría proporciona una solución eficiente, el proceso pudiera acelerarse aún más si se combina con la utilización de la arquitectura paralela que por naturaleza caracteriza a las GPU.

Sin lugar a dudas una de las soluciones más abarcadoras es la propuesta realizada recientemente por (Qin et al., 2014a), Figura 7. En su investigación, se enfocan en el diseño de una biblioteca para el procesamiento ráster que hace uso de sistemas paralelos a través de la combinación MPI/OpenMP en un clúster SMP[7], así como sistemas distribuidos mediante clústeres Beowulf[8] con MPI. Su principal contribución radica en utilizar además las características de las GPU para acelerar el procesamiento de los datos, tarea que realiza empleando la tecnología de cómputo de propósito general CUDA.

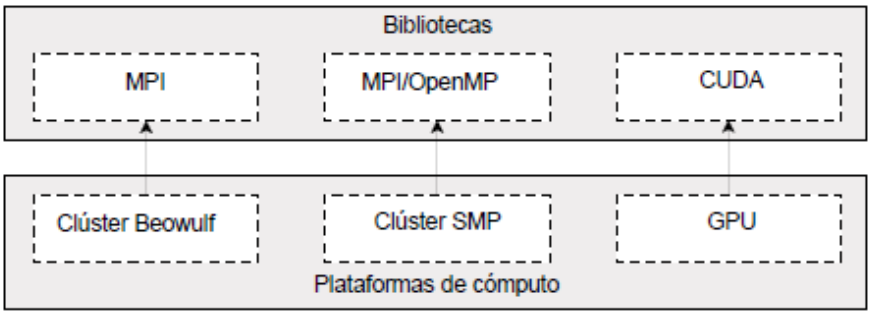

Figura 7. Principales elementos que intervienen en la propuesta presentada por (Qin et al., 2014a).

Aunque el empleo de CUDA se adentra ya en la utilización de la computación GPGPU para lograr mejores resultados en cuanto a la aceleración de los cálculos, presenta una limitación relevante. CUDA no está implementado para tener en cuenta la computación heterogénea entre diferentes plataformas de cómputo, en otras palabras, con su utilización no se aprovecha la potencia de los procesadores gráficos unida a la de las CPU y otros dispositivos de cómputo.

[7] Arquitectura de computadores de alto rendimiento en la que varias unidades de procesamiento comparten una única memoria central.
[8] Conglomerado de elementos de cómputo, gestionado por un sistema Unix, donde cada nodo es una computadora personal sin teclado, mouse, tarjeta de video o monitor.

Además, CUDA ha sido implementado y optimizado para obtener alto desempeño en tarjetas gráficas NVIDIA solamente. Como consecuencia de esto, los resultados obtenidos no pueden ser aprovechados en cualquier plataforma o arquitectura, al contrario de esto, son dependientes del *hardware* fabricado por NVIDIA.

Adicionalmente, dichos resultados fueron evaluados en un clúster IBM SMP con 134 nodos de cómputo. Cada nodo consta de la siguiente configuración de *hardware*: dos CPUs Intel Xeon E5650 a 2.0 GHz con seis núcleos y 24 GB de memoria RAM DDR3, generalmente empleados como servidores por sus altas prestaciones. La GPU utilizada es una NVIDIA Corporation Tesla M2075 con 448 núcleos de procesamiento y 6 GB de memoria gráfica (Qin et al., 2014a). Sin embargo, a pesar del incremento en las velocidades de cómputo y la disponibilidad de memoria, terminales de estas características, en cuanto a sus altas prestaciones de *hardware*, no son las que predominan en el entorno de las entidades y proyectos dedicados al procesamiento y análisis de información geoespacial en Cuba.

Discusión

En términos generales, en la Tabla 2, se ilustra un análisis comparativo entre las estrategias más relevantes de la bibliografía que centran su atención en la implementación de alternativas para el procesamiento paralelo de información ráster mediante la GDAL.

Tabla 2. Análisis comparativo de las principales aproximaciones estudiadas.

Estudios	Plataforma de cómputo	Heterogeneidad entre plataformas	Confiabilidad de los resultados	Aplicación en entornos de bajas prestaciones
(Zhan and Qin, 2012) (E/S en serie)	CPU	Baja	Baja	Baja
(Zhan and Qin, 2012) (E/S en paralelo)	CPU	Baja	Baja	Baja
(Qin et al., 2014b)	CPU	Baja	Alta	Baja
(Qin et al., 2014a)	CPU (Clúster Beowulf y SMP), GPU	Parcial	Alta	Baja

A modo de resumen del estudio realizado, en la Tabla 3 se muestran las principales contribuciones de los últimos diez años en relación con el procesamiento de datos ráster en entornos paralelos, que a consideración de la autora, fundamentan propuestas de significación para la investigación.

Tabla 3. Estudios sobre procesamiento de datos ráster en entornos paralelos.

Estudios	Aportes
(Guan, 2008) (Guan, 2009) (Guan and Clarke, 2010) (Wang et al., 2012) (Guan et al., 2013) (Ouyang et al., 2013)	Uso de MPI en el procesamiento ráster.
(Zhao et al., 2010) (Xia et al., 2011) (Lv et al., 2012) (Qin and Zhan, 2012) (Osterman, 2012) (Gao et al., 2013)	Uso de la GPU con el modelo de programación CUDA.
(Zhan and Qin, 2012) (Qin et al., 2014b)	Uso de MPI y OpenMP en el procesamiento paralelo de datos ráster empleando la GDAL.
(Qin et al., 2014a)	Incorpora a la estrategia propuesta por (Qin et al., 2014b) el uso de la GPU con el modelo de programación CUDA para acelerar el procesamiento de los datos con la GDAL.

Conclusiones

El estudio realizado demostró que el perfeccionamiento de las tecnologías y mecanismos para la obtención de datos de la superficie terrestre ha propiciado la obtención de información ráster cada vez más detallada y con mayor nivel de resolución y precisión espacial, lo que ha generado un crecimiento gradual del volumen de los modelos ráster. Por lo tanto, se requieren altas prestaciones computacionales para apoyar su análisis y como alternativa, el reto fundamental es la implementación de técnicas que contribuyan a reducir el tiempo para su procesamiento. Como consideraciones finales del presente trabajo se relacionan las siguientes:

- El tiempo empleado por la GDAL en el procesamiento de los datos ráster, en ocasiones y teniendo en cuenta el nivel de resolución de los datos, no satisface las necesidades de los usuarios de SIG, en entornos computacionales con bajas prestaciones.
- A pesar de que se aprecia un avance científico en el área del procesamiento ráster, en su mayoría, las propuestas estudiadas, modelan el procesamiento paralelo en torno a la CPU a través de la utilización de estrategias paralelas mediante OpenMP y MPI o a la GPU mediante CUDA, sin explotar al máximo la heterogeneidad entre diferentes plataformas de cómputo.
- La aproximación propuesta por (Qin et al., 2014a) satisface este factor, sin embargo el empleo de CUDA limita su campo a las GPU del fabricante NVIDIA. Adicionalmente, la configuración de *hardware* del entorno computacional empleado para el despliegue de la solución propuesta no es la más asequible, en términos de costo, a las posibilidades de las organizaciones cubanas especializadas en el desarrollo de SIG y dedicadas al procesamiento de información geoespacial.

- Teniendo en cuenta estos elementos, la autora se inclina por el diseño de un método que permita garantizar la heterogeneidad entre diferentes plataformas de cómputo, tomando como base la arquitectura paralela de las GPU, además del bajo coste y las potencialidades de procesamiento que pueden llegar a alcanzar los sistemas distribuidos por computación voluntaria, con el fin complementar los medios disponibles para el procesamiento ráster.

Referencias

ANDERSON, D. P. AND G. FEDAK. The computational and storage potential of volunteer computing. In *Cluster Computing and the Grid. CCGRID 06. Sixth IEEE International Symposium on.* 2006.

ARMSTRONG, M. P. AND P. J. DENSHAM Domain decomposition for parallel processing of spatial problems. Computers, Environment and Urban Systems, 1992, 16, 497–513.

BEBERG, A. L., D. L. ENSIGN, J. GUHA, S. KHALIQ, et al. Folding@ home: Lessons from eight years of volunteer distributed computing. Parallel & Distributed Processing. IPDPS 2009. IEEE International Symposium on, 2009.

BRESHEARS, C. *The Art of Concurrency. First Edition*. edited by M. LOUKIDES. Edtion ed.: O'Reilly Media, 2009.

CALKIN, R., R. HEMPEL, H. C. HOPPE AND P. WYPIOR Portable programming with the PARMACS message-passing library. Parallel Computing, 1994, 20, 615–632.

CONTRERAS, G. AND M. MARTONOSI. Characterizing and improving the performance of Intel Threading Building Blocks. In *Workload Characterization. IISWC 2008. IEEE International Symposium.* 2008, p. 57 - 66.

CUI, S. AND S. ZHANG Parallel processing of topological operations by using a hybrid MPI/OpenMP approach. Natural Computation (ICNC), Ninth International Conference, Shenyang, IEEE, 2013, 1738 - 1742.

CHAPMAN, B. AND L. HUANG Enhancing OpenMP and Its Implementation for Programming Multicore Systems. Parallel Computing: Architectures, Algorithms and Applications. John von Neumann Institute for Computing, Julich, NIC Series, 2007, 38, 3-18.

CHEN, Z., C. ZHENJIE AND L. FEIXUE A Dynamic Data Partition Algorithm Oriented to MPI and OpenMP. Geocomputation, 2013.

DAGA, M., A. M. AJI AND W.-C. FENG. On the Efficacy of a Fused CPU+GPU Processor (or APU) for Parallel Computing. 2011.

DAGUM, L. AND R. MENON OpenMP: An Industry Standard API for Shared-Memory Programming. IEEE Computational Science and Engineering, 1998, 6, 46 - 55.

ESRI. ArcGIS Resource Center. 2012, [cited 25 de septiembre 2014]. Available from Internet:<http://help.arcgis.com/es/arcgisdesktop/10.0/help/index.html>.

FRAIRE, J. A., P. FERREYRA AND C. MARQUES OpenCL Overview, Implementation, and Performance Comparison. IEEE Latin America Trasactions, 2013, 11.

GAO, Y., H. YU, L. LIU AND X. GUO. General neighborhood analysis model for grid DEM on CUDA. 2013.

GDAL. GDAL - Geospatial Data Abstraction Library, Version 1.10. In *Development Team.* Open Source Geospatial Foundation, 2014.

GEBALI, F. *Algorithms and Parallel Computing*. Edtion ed.: John Wiley & Sons, Inc., Hoboken, New Jersey, 2011.

GEIST, A., A. BEGUELIN, J. DONGARRA, W. JIANG, et al. *PVM: Parallel Virtual Machine: a users' guide and tutorial for networked parallel computing*. Edtion ed.: MIT Press, Cambridge, MA, USA, 1994.

GOVE, D. *Multicore Application Programming. For Windows, Linux, and Oracle® Solaris*. edited by P. EDUCATION. Edtion ed.: Addison Wesley, 2011.

GUAN, Q. Parallel algorithms for geographic processing. Tesis doctoral. University of California, Santa Barbara, 2008.

GUAN, Q. pRPL: an open-source general-purpose parallel raster processing programming library. Newsletter SIGSPATIAL Special, 2009, 19, 57-62.

GUAN, Q. AND K. CLARKE A general-purpose parallel raster processing programming library test application using a geographic cellular automata model. International Journal of Geographical Information Science, 2010, 24, 695-722.

GUAN, Q., W. ZENG AND S. LIU. pRPL 2.0 Improving the Parallel Raster Processing Library. In *Proceedings of the 12th International Conference on GeoComputation, LIESMARS, Wuhan University, Wuhan, China.* 2013.

JENNY, B. An Interactive Approach to Analytical Relief Shading. Cartographica, 2001, 30.

KOMATSU, K., K. SATO, Y. ARAI, K. KOYAMA, et al. Evaluating Performance and Portability of OpenCL Programs. 2010.

KWESI, A. G. N. AND P. AMOAKO-YIRENKYI Volunteer Computing: Application for African Scientist. ICT AFRICA, 2008.

LEWIS, B. AND D. J. BERG *Multithreaded Programming With PThreads*. Edtion ed.: Prentice Hall PTR, 1997.

LONGLEY, P., M. F. GOODCHILD, D. MAGUIRE AND D. W. RHIND *Geographic information systems \& science. 3ra Edición*. edited by WILEY. Edtion ed.: Hoboken, 2011.

LV, M. H., X. WEI AND C. LEI A GPU-Based Parallel Processing Method for Slope Analysis in Geographical Computation. Advanced Materials Research, 2012, 538, 625-631.

LLOPIS, J. P. Sistemas de Información Geográfica aplicados a la gestión del territorio. Entrada, manejo, análisis y salida de datos espaciales. Teoría y práctica general para ESRI ArcGIS 9. Universidad de Alicante. Editorial Club Universitario, 2008.

MARTÍNEZ, Y. V. Modelo de Representación de Superficies de Terreno para su Visualización en Tres Dimensiones. Tesis doctoral Universidad de las Ciencias Informáticas, 2011.

MAULIK, U. AND A. SARKAR Efficient parallel algorithm for pixel classification in remote sensing imagery. Geoinformatica, 2012, 16, 391-407.

MAX, N. Progress in scientific visualization. The Visual Computer 21, 2005, 979–984.

MUNSHI, A., B. R. GASTER, T. G. MATTSON, J. FUNG, et al. *OpenCL Programming Guide*. edited by P. EDUCATION. Edtion ed.: Addison Wesley, 2012.

NIKOLAKOPOULOS, K. G., E. K. KAMARATAKIS AND N. CHRYSOULAKIS SRTM vs ASTER elevation products. Comparison for two regions in Crete, Greece. International Journal of Remote Sensing, 2006, 27, 481–483.

NVIDIA. NVIDIA CUDA C Programming Guide version 4.0. 2011. Available from Internet:<http://developer.download.nvidia.com/compute/DevZone/docs/html/C/doc/CUDA C Programming Guide.pdf>.

OSTERMAN, A. Implementation of the r.cuda.los module in the open source GRASS GIS by using parallel computation on the NVIDIA CUDA graphic cards. ELEKTROTEHNISKI VESTNIK. ENGLISH EDITION, 2012, 79, 19–24.

OUYANG, L., J. HUANG, X. WU AND B. YU Parallel Access Optimization Technique for Geographic Raster Data. Geo-Informatics in Resource Management and Sustainable Ecosystem. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2013, 398, 533-542.

PERALTA, F. J. R. *Análisis Espacial con Datos Raster en ArcGIS Desktop 9.2*. Edtion ed., 2008.

PETERSEN, W. P. AND P. ARBENZ *Introduction to parallel computing. A practical guide with examples in C.* Edtion ed.: Oxford University Press Inc., New York, 2004.

QIN, C.-Z., L.-J. ZHAN, A.-X. ZHU AND C.-H. ZHOU A strategy for raster-based geocomputation under different parallel computing platforms. International Journal of Geographical Information Science, 2014a, 37-41.

QIN, C.-Z. AND L. ZHAN Parallelizing flow-accumulation calculations on graphics processing units-From iterative DEM preprocessing algorithm to recursive multiple-flow-direction algorithm. Computers & Geosciences, 2012, 43.

QIN, C. Z., L. J. ZHAN AND A. X. ZHU How to apply the Geospatial Data Abstraction Library (GDAL) properly to parallel geospatial raster I/O? Transactions in GIS, 2014b.

RODRÍGUEZ, J. L. G. AND M. C. G. SUÁREZ Comparison of mathematical algorithms for determining the slope angle in GIS enviroment. Aqua-LAC, 2010, 2.

ROMERO, F. S. La Teledetección satelital y los sistemas de protección ambiental. AquaTIC. Revista científica de la Sociedad Española de Acuicultura, 2006, 24, 13-41.

SARMENTA, L. F. G. Volunteer Computing. Tesis doctoral. Department of Electrical Engineering and Computer Science. Massachusetts Institute Of Technology, 2001.

SEITAVUOPIO, P., J. RANTANEN AND J. YLIRUUSI Use of roughness maps in visualisation of surfaces. European Journal of Pharmaceutics and Biopharmaceutics, 2005, 59, 351-158s.

SMITH, M. J., M. F. GOODCHILD AND P. A. LONGLEY *Geospatial Analysis: A Comprehensive Guide to Principles, Techniques and Software Tools, 2nd edition*. Edtion ed.: Troubador Publishing, 2013.

STEINBACH, M. AND R. HEMMERLING Accelerating batch processing of spatial raster analysis using GPU. Computers & Geosciences, 2012, 45, 212–220.

WAGNER, D. F. AND M. S. SCOTT Improving the Performance of Raster GIS: A Comparison of Approaches to Parallelization of Cost Volume Algorithms. Autocarto-Conference, 1995, 164-176.

WANG, X., Z. LI AND S. GAO Parallel Remote Sensing Image Processing: Taking Image Classification as an Example. Computational Intelligence and Intelligent Systems. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2012, 159-169.

WARMERDAM, F. The Geospatial Data Abstraction Library. Brent H and Michael LG. Open Source Approaches in Spatial Data Handling. Springer, Berlin, 2008, 87-104.

XIA, Y., L. KUANG AND X.-M. LI Accelerating geospatial analysis on GPUs using CUDA. Journal of Zhejiang University-SCIENCE C (Computers & Electronics), 2011, 12, 990-999.

ZHAN, L.-J. AND C.-Z. QIN Parallel Geospatial Raster Processing by Geospatial Data Abstraction Library (GDAL) — Applicability and Defects. Transactions in GIS, 2012.

ZHAO, Y., Z. HUANG, B. CHEN AND Y. FANG Local acceleration in Distributed Geographic Information Processing with CUDA. Geoinformatics. 18th International Conference. IEEE Xplore Digital Library, 2010.