DIFERENCIAS FINITAS CON MALLA INTERCALADA PARA EL MODELADO SÍSMICO EN 3D EN MEDIOS ISÓTROPOS Y HETEROGENEOS HACIENDO USO DE ARQUITECTURAS HÍBRIDAS GPGPU

DIEGO ARMANDO NIÑO CASTELLANOS
JOSE AGUSTIN CARREÑO GALLEGO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA 2015

DIFERENCIAS FINITAS CON MALLA INTERCALADA PARA EL MODELADO SÍSMICO EN 3D EN MEDIOS ISÓTROPOS Y HETEROGENEOS HACIENDO USO DE ARQUITECTURAS HÍBRIDAS GPGPU

DIEGO ARMANDO NIÑO CASTELLANOS
JOSE AGUSTIN CARREÑO GALLEGO

Trabajo de grado para optar al título de
Ingeniero de Sistemas

Dirección:

CARLOS JAIME BARRIOS HERNÁNDEZ
INGENIERO DE SISTEMAS, PhD.

HERLING GONZÁLEZ ÁLVAREZ
FÍSICO, MsC.

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA 2015

AGRADECIMIENTOS

Este proyecto se hace posible gracias a la integración del grupo de Supercomputación y Cálculo Científico de la Universidad Industrial de Santander SC3 UIS con el área de Geofísica del Instituto Colombiano del Petróleo ICP.

De manera grata, se quiere resaltar la labor realizada por el MsC. Herling González, quién desempeñó como co-director y experto en el campo de geofísica, sísmica y computación. Fue el ente encargado de describir el problema desde el punto de vista físico y de plantear el problema bajo el punto de vista de las necesidades de la industria, la manera de dar solución por medio de métodos matemáticos, y de instruir sobre las plataformas de visualización utilizadas.

Igualmente, se destaca la labor realizada por el PhD. Carlos Jaime Barrios, director del Centro de Supercómputo y Cálculo Científico de la Universidad Industrial de Santander, quién fue la persona encargada de liderar labores de desarrollo y de gestionar el acceso de las máquinas de cómputo utilizadas, además de instruir sobre el uso de GPUs y la implementación con CUDA como paradigma de paralelización.

Finalmente, se agradece a todo el grupo SC3-UIS por la ayuda brindada para el desarrollo del proyecto, especialmente al Ingeniero de Sistemas Sergio Gelves y al Ingeniero Electrónico William Trigos, miembros del grupo de SC3-UIS por el rol desempeñado como entes de soporte de desarrollo , quienes fueron las personas encargadas de dar instructivo específico sobre C y CUDA como lenguajes de desarrollo y paradigma de paralelización; y el funcionamiento de los dispositivos utilizados.

Índice

Índice de figuras

Índice de tablas

RESUMEN

TITULO: DIFERENCIAS FINITAS CON MALLA INTERCALADA PARA EL MODELADO SÍSMICO EN 3D EN MEDIOS ISÓTROPOS Y HETEROGÉNEOS HACIENDO USO DE ARQUITECTURAS HÍBRIDAS GPGPU.[1]

AUTORES:
DIEGO ARMANDO NIÑO CASTELLANOS[2]
JOSE AGUSTIN CARREÑO GALLEGO[2]

PALABRAS CLAVE: Modelado sísmico, ecuación de onda elástica, medios isótropos, FDTD, geofísica computacional, arquitecturas híbridas, OpenMP, CUDA, visualización 3D.

DESCRIPCIÓN:

En el presente trabajo de grado, se muestra el estudio realizado para resolver la ecuación de una onda elástica en un medio isótropo y heterogéneo, mediante la técnica de diferencias finitas con malla intercalada en cuarto orden en el espacio y segundo orden en el tiempo.

En la primera parte, se muestra una descripción del problema de modelado sísmico, posterior a ello se describe el contexto y las característica propias del área en la cual se constituye el enfoque principal este trabajo. Consecuentemente, se plantea el esquema en que se ha aplicado e implementado la solución numérica del problema con un algoritmo computacional de malla intercalada FDTD.

Por otra parte se realizó el análisis de plataformas para soporte computacional masivo; desarrollo e implementación de códigos y visualización. La aceleración de códigos que implementan la técnica de diferencias finitas en el dominio del tiempo para ejecución en GPU con uso de CUDA, constituye una aplicación a estrategias de solución y simulación de propagación de ondas sísmicas en 3D basada en arquitecturas híbridas GPGPU.

La aplicación es creada bajo el API open source de Madagascar (Plataforma de procesamiento sísmico), haciendo uso del lenguaje de programación C, y los paradigmas de paralelización OpenMP y CUDA. Finalmente se hizo el análisis de rendimiento entre la ejecución sobre CPUs en serial y sobre arquitecturas híbridas CPU-GPU en paralelo.

[1] Trabajo de grado

[2] Escuela de Ingeniería de Sistemas, Facultad de Fisicomecánicas, Universidad Industrial de Santander. Directores: Carlos J. Barrios Ph.D., Herling González MsC.

ABSTRACT

TITLE: FINITE DIFERENCE WITH STAGGERED GRID FOR 3D SEISMIC MODELLING IN ISOTROPICS AND HETEROGENEOUS MEDIA USING HYBRID ARCHITECTURES GPGPU.[1]

AUTHORS: DIEGO ARMANDO NIÑO CASTELLANOS [2]
JOSE AGUSTIN CARREÑO GALLEGO[2]

KEYWORDS: Seismic modelling, elastic wave equation, isotropic media, FDTD, computational geophysics, hybrid architectures, OpenMP, CUDA, 3D visualization.

DESCRIPTION:

In this project, a study was conducted to solve the elastic wave equation in isotropic and heterogeneous media, using the finite differences method over a staggered mesh with 4th order in space and 2nd order in time.

In the first part, it show a description of the seismic modelling problem, then, the context and features of the seismic modelling are described; that idea is the principal focus in this work. Hence, the scheme how the problem is converted to a numerical solution, then with a computational algorithm of staggered mesh, then the way how this is implemented, was lay out.

Also, an analysis about platforms of high performance for support, development, implementation of codes and visualization of the resulting data was done. The codes used to implement finite difference method with time domain for the execution were accelerated in GPU using CUDA. it make an application for the 3D solution and simulation for the propagation of elastic wave based on hybrid architectures GPGPU was created.

The application was created using the open source Madagascar API (platform of seismic processing); C as the programming language; the OpenMP standard and CUDA like parallelization paradigms. Finally a performance analysis about the execution using CPUs and the hybrid architectures CPU-GPU was done.

[1]Degree work

[2]Escuela de Ingeniería de Sistemas, Facultad de Fisicomecánicas, Universidad Industrial de Santander. Directores: Carlos J. Barrios Ph.D., Herling González MsC.

INTRODUCCIÓN

La búsqueda de recursos naturales, representa retos mayores hoy día a Colombia. La industria de Petróleo y Gas desea plantear estrategias con el objetivo de mejorar la calidad de la imagen sísmica del subsuelo.

Sin embargo, crear una buena imagen sísmica del subsuelo conlleva a hacer uso de modelos de propagación de ondas sísmicas y el entendimiento de fenómenos ondulatorios. Esta descripción un poco mas completa de la naturaleza implica el uso de un sistema de ecuaciones diferenciales para su solución. Evacuar la solución numérica de este conjunto de ecuaciones no es una tarea fácil para el uso de computadoras convencionales, debido a la cantidad de variables físicas, como variables espaciales y temporales a evaluar. Y el uso de esta gran cantidad de variables y datos, exige una gran capacidad de cómputo.

En este punto, la ciencia de la computación y el cálculo científico, desea plantear alternativas de desarrollo adecuadas con el uso de tecnologías emergentes.

Éste documento esquematiza el desarrollo paso a paso de la solución de ecuación de onda elástica 3D en medios isótropos y heterogéneos, también evalúa el estado actual de la solución numérica y la manera de hacer uso de los recursos hardware ofrecidos por el Centro de Supercómputo y Cálculo Científico de la Universidad Industrial de Santander, haciendo uso de GUANE-1 como arquitectura de desarrollo para cómputo masivo. Para explicar el desarrollo del proyecto, se presentan cinco secciones, en la primera sección se introduce la teoría de las ondas sísmicas, seguido a esto, en la segunda sección se menciona la importancia del uso de las arquitecturas híbridas para realizar cálculos científicos; en la tercera sección se explica el trabajo realizado, desde la solución de la ecuación diferencial, hasta la visualización y validación de los resultados obtenidos. En la cuarta sección, se presentan las conclusiones y en la quinta sección se hacen recomendaciones que se pueden tomar en cuenta para mejorar el trabajo realizado.

1. TEORÍA DE PROPAGACIÓN DE ONDAS SÍSMICAS

El modelado computacional de datos sísmicos ha sido usado para soportar interpretaciones en datos de campo, proveer datos sintéticos para el desarrollo de técnicas de procesamiento, y para la simulación del fenómeno de propagación de ondas sísmicas. [1], [2], [3]

Basado en los modelos, se pretende determinar la estructura del subsuelo haciendo uso de los ecos obtenidos de las reflexiones cuando las ondas se propagan a través de los estratos con cambios de impedancia.

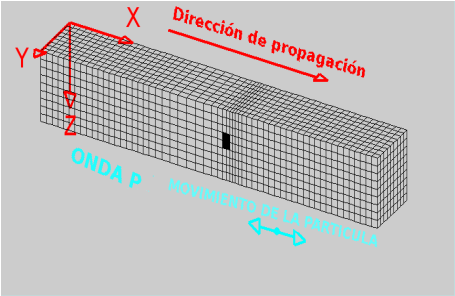
Hay modelos matemáticos de propagación de ondas que pueden ser simplificados o complejos. Estos pueden depender al medio sobre el que se propagan:

- De acuerdo al medio: homogéneo o heterogéneo.
- Acústico con campo de ondas-P.
- Acústico con campo de ondas-P y densidad del medio.
- Elástico con campos de onda-P, onda-S y densidad.
- Elástico con campos de onda-P, onda-S, densidad y anisotropía.
- Visco-elástico con campos de onda-P, onda-S, densidad, factor Q, etc.

La complejidad computacional aumenta hacia la base de la lista y con mayor grado cuando se va de 2D a 3D.

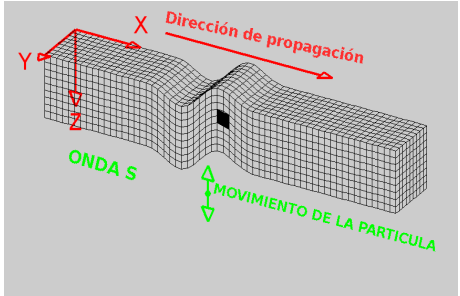
Las ondas de mayor importancia en imágenes sísmicas son las Ondas-P y las Ondas-S. Las ondas-P (ver *Figura 1*) son las usadas para exploración, sin embargo, las ondas-S (ver *Figura 2*)permiten extraer otro tipo de información: estimar efectos de atenuación, hacer análisis de anisotropía, caracterizar propiedades del subsuelo, etc. Las ondas que se muestran en las siguientes imágenes se propagan dentro de un medio elástico y homogéneo.

Figura 1: Ondas-P (Primarias), con velocidades entre 3.5 y 7 [Km/s].



Tomado de: Braile, L. «What Are Seismic Waves?» [4]

Figura 2: Ondas-S (Secundarias), con velocidades entre 1.5 y 3 [Km/s].

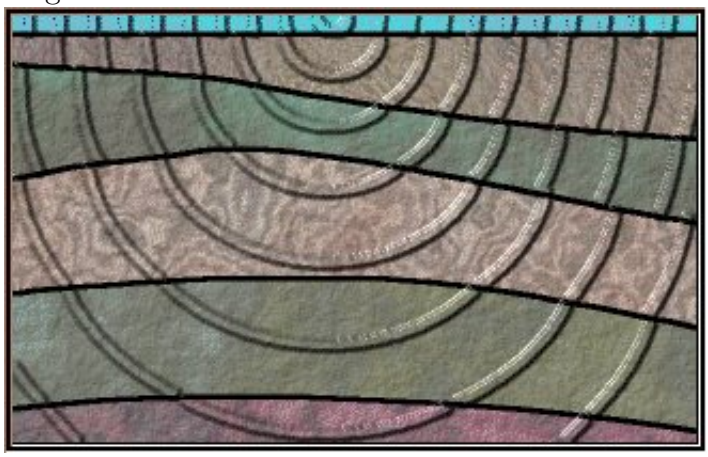


Tomado de: Braile, L. «What Are Seismic Waves?» [4]

Para la generación de una imagen sísmica, se debe tener en cuenta ciertas propiedades de la propagación de ondas y las características del medio de propagación. [5]

Una de estas características es el frente de onda, definido como la superficie donde el movimiento relativo de las partículas del material se desplazan al mismo tiempo y con igual fase. Los frentes de onda son planos a medida que la distancia aumenta, esto se puede ver en la *Figura 3*

Figura 3: A grandes distancias de la fuente es un frente de onda plano.

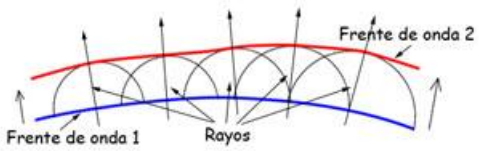


Tomado de: Intergeo «Frentes de onda» [6]

Analizando lo propuesto por el principio de Huygens, se establece que cada punto del avance del frente de onda puede ser considerado como una fuente de un segundo frente de onda.

La envolvente de todos los frentes de onda secundarios construyen el frente de onda principal en el mismo tiempo y fase, de tal manera que el frente de onda principal es tangente a los frentes de onda secundarios.

Figura 4: Frentes de onda y rayos.



Tomado de: e-ducativa «Tema 2: Movimiento Ondulatorio» [7].

Los rayos que se muestran en la *Figura 4*, son líneas perpendiculares al frente de onda que representan de manera abstracta la trayectoria seguida del frente. Es un concepto para la construcción compleja del Principio de Huygens en un medio determinado sobre el que se propagan las ondas.

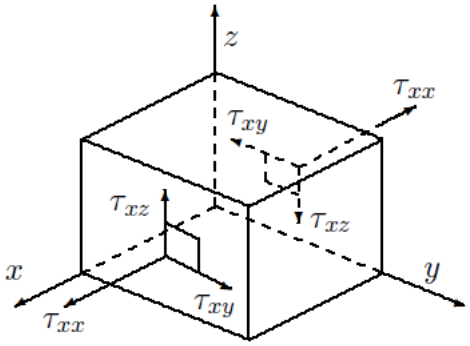
El Principio de Huygens nos ayuda a entender el fenómeno de reflexión, el cual se define como la razón de amplitud de una onda reflejada en función de una incidente es llamado coeficiente de reflexión R. Un $R<0$ representa un cambio de fase de la onda incidente, mientras que un $R>0$ no (esto es válido para ondas incidentes de manera normal en la interfaz). El coeficiente de reflexión se representa por la ecuación (1):

$$R = \frac{\rho_2 v_2 - \rho_1 v_1}{\rho_2 v_2 + \rho_1 v_1} \tag{1}$$

Donde ρ_1 y v_1 corresponden a la densidad del medio de donde proviene la onda incidente y la velocidad de la onda en ese medio respectivamente. ρ_2 y v_2 corresponden a la densidad del medio en donde incide la onda, y la veloidad de la onda en ese medio, respectivamente.

Para la generación de ondas sísmicas se requiere que se aplique una fuerza puntual externa que implica un esfuerzo. El esfuerzo está definido como fuerza por unidad de área, de tal manera que si la fuerza aplicada varía de un punto a otro, el esfuerzo también varía. Su valor puede ser calculado en cualquier punto tomando un elemento infinitesimal-mente pequeño de área en el punto aplicado. En la *Figura 5* se observa como se aplica el esfuerzo normal T_{xx} y perpendicular a este, los esfuerzos de cizalla T_{xz} y T_{xy} sobre una superficie.

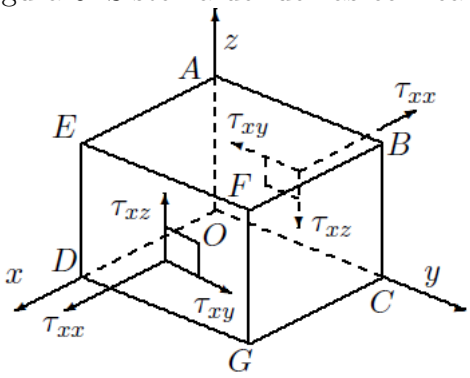
Figura 5: Sistema de fuerzas colineales.



Tomado de: GONZÁLEZ, H «Modelado Sísmico. Teoría de Propagación de Ondas.» [8]

Cuando el medio se encuentra en equilibrio (estático), los esfuerzos deben ser balanceados. Esto significa que los tres esfuerzos τ_{xx}, τ_{xy},τ_{xz} que actúan sobre la cara OABC deben ser iguales y opuestos a los esfuerzos de la cara DEFG, e igual manera para las demas relaciones en las otras 4 caras. También un par de esfuerzos cizallantes como τ_{xy}, constituyen una cupla que tiende rotar el elemento de volumen alrededor del eje z. (ver *Figura 6*).

Figura 6: Sistema de fuerzas colineales



Tomado de: GONZÁLEZ, H «Modelado Sísmico. Teoría de Propagación de Ondas.» [8]

Antes de producirse una onda sísmica, se mantiene un equilbrio entre esfuerzo y deformación. Sea $\overrightarrow{u}$ (con componentes,$u; v; w$) el campo de desplazamiento de un punto

material localizado en $x; y; z$. Las deformaciones pueden estar definidas en función de desplazamientos u_i de acuerdo a la expresión (2) (Lavergne 1989):

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{2}$$

Donde ∂u es el diferencial del campo de desplazamiento en una dirección y ∂x es el diferencial de la posición donde se aplica la deformación en una dirección.

El cambio relativo en dimensión y forma para el cubo, esta definido por el tensor de deformación, mostrado en la ecuación (3):

$$\varepsilon_{ij} = \begin{pmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{pmatrix} \tag{3}$$

Donde la matriz de deformaciones está compuesta por las deformaciones en dirección normal ε_{xx}, ε_{yy} y ε_{zz} y las deformaciones en dirección de cizalla ε_{xz} ε_{xy}, ε_{zy}.

Los cambios en las dimensiones dados por las deformaciones normales resultan en cambios de volumen por unidad de volumen. Estos cambios de volumen por unidad de volumen se llaman dilatación, se representa como $\Delta = \Delta v/v$

Los esfuerzos y deformaciones tienen una relación lineal (directamente proporcional). El orden proporción de deformación es de 1×10^{-8} (excepto en cercanías de la fuente), de tal manera que la Ley de Hooke se mantiene.

Continuando con el fundamento requerido para la generación de una imágen sísmica, es igualmente importante tener en cuenta las propiedades físicas del medio de propagación.

Se definen ciertas propiedades del medio como la rigidéz, caracterizada por una constante denomidada *Parámetro de Lamé* (μ) (ver *ecuación (4)*), con la que se mide la razón entre esfuerzo tangencial y deformación:

$$\mu = \frac{1}{2}\frac{\tau_{xy}}{\varepsilon_{xy}} \tag{4}$$

Donde τ_{xy} y ε_{xy} es el esfuerzo y la deformación aplicados en la dirección de cizalla xy, respectivamente.

Igualmente, se encuentra la razón entre esfuerzo normal y la dilatación (o compresión), más conocido como *Módulo de Young* (Y), expresado en la ecuación (5):

$$Y = \frac{\tau_{xx}}{\varepsilon_{xx}} \tag{5}$$

Donde τ_{xx} y ε_{xx} es el esfuerzo y la deformación aplicados en la dirección normal xx, respectivamente. También se puede expresar como la ecuación (6):

$$Y = \frac{\mu \quad (\quad 3\lambda \quad + \quad 2\mu \quad)}{\lambda \quad + \quad \mu} \tag{6}$$

Donde λ es una constante de Lamé.

Por otra parte, se contempla la razón entre la variación de presión y la dilatación volumétrica, definido como el *módulo volumétrico* (K), como se muestra en la ecuación (7):

$$K = \lambda + \frac{2}{3}\mu \tag{7}$$

También es importante tener en cuenta la razón de proporción de compresión en una dirección y dilatación en la dirección perpendicular; esto se conoce como el *Radio de Poisson* (σ), expresado en la ecuación (8):

$$\sigma = -\frac{\varepsilon_{yy}}{\varepsilon_{xx}} \tag{8}$$

También vista por la expresión de la ecuación (9):

$$\sigma = \frac{\lambda}{2(\lambda + \mu)} \tag{9}$$

Cada una de estas características definen el tipo de subsuelo, en donde se propaga la onda sísmica. Tradicionalmente se hace estudio de dos tipos de medios: el medio isótropo y el medio anisótropo.

Se define un medio isótropo cuando las propiedades físicas del terreno, no cambian

dependiendo de la dirección donde sea analizado. Por otra parte, se define un medio anisótropo cuando las propiedades físicas del terreno varían cuando se analiza en direcciones diferentes.

Cuando una onda se propaga en un medio con anisotropía, su frente de onda cambia respecto de uno con isotropía donde el frente de onda es esférico, como se observa en la *Figura 7*.

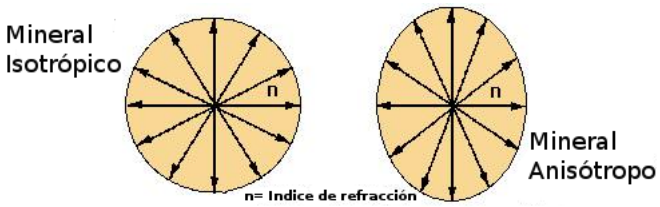


Figura 7: Comparación de los frentes de Onda para un medio isótropo y otro anisótropo, aquí el frente de onda es esférico para el caso isótropo . Mientras que para el caso anisótropo es elipsoidal.

Tomado de: DORRONOSO, C. «Isotropic and anisotropic minerals » [9]

Las características del medio en el que se propaga la onda elástica simulada en este trabajo, es la isotropía, definida anteriormente. Además el medio es heterogeneo, lo que significa que existe un contraste entre capas.

Cuando en un medio rocoso, el equilibrio entre deformidad y esfuerzo termina, por ejemplo, que el esfuerzo cese, se propaga la deformidad a lo largo del medio como ondas elásticas.[10]

Una vez identificadas y estudiadas las caracterísitcas necesarias para generar una imágen sísmica de la propagación de una onda elástica, se introduce el concepto e interpretación de la ecuación diferencial que describe analíticamente el fenómeno.

El fenómeno de la propagación de ondas sísmicas en un medio como la corteza terrestre corresponde a lo que describe la dinámica de las leyes de Newton y la teoría de elasticidad (Aki and Richards 2002)[2], donde el material como la corteza puede estar sujeto a seis componentes tensoriales de esfuerzos y deformaciones.

En las ecuaciones (10) se expresan los esfuerzos normales; en las ecuaciones (11), los esfuerzos de cizalla y en las ecuaciones (12) las componentes de velocidades. Este conjunto de ecuaciones corresponden a una onda elástica en un medio isótropo:

$$\begin{aligned}
\frac{\partial T_{xx}}{\partial t} &= C_{11}\frac{\partial v_x}{\partial x} + C_{12}\frac{\partial v_y}{\partial y} + C_{12}\frac{\partial v_z}{\partial z} \\
\frac{\partial T_{yy}}{\partial t} &= C_{12}\frac{\partial v_x}{\partial x} + C_{11}\frac{\partial v_y}{\partial y} + C_{12}\frac{\partial v_z}{\partial z} \\
\frac{\partial T_{zz}}{\partial t} &= C_{12}\frac{\partial v_x}{\partial x} + C_{12}\frac{\partial v_y}{\partial y} + C_{11}\frac{\partial v_z}{\partial z}
\end{aligned} \tag{10}$$

$$\begin{aligned}
\frac{\partial T_{xz}}{\partial t} &= C_{44}\left[\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right], \quad C_{11} = \lambda + 2\mu \\
\frac{\partial T_{xy}}{\partial t} &= C_{44}\left[\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right], \quad C_{12} = \lambda \\
\frac{\partial T_{yz}}{\partial t} &= C_{44}\left[\frac{\partial v_y}{\partial z} + \frac{\partial v_z}{\partial y}\right], \quad C_{44} = \mu
\end{aligned} \tag{11}$$

$$\begin{aligned}
\rho\frac{\partial V_x}{\partial t} &= \frac{\partial T_{xx}}{\partial x} + \frac{\partial T_{xy}}{\partial y} + \frac{\partial T_{xz}}{\partial z} \\
\rho\frac{\partial V_y}{\partial t} &= \frac{\partial T_{xy}}{\partial x} + \frac{\partial T_{yy}}{\partial y} + \frac{\partial T_{yz}}{\partial z} \\
\rho\frac{\partial V_z}{\partial t} &= \frac{\partial T_{xz}}{\partial x} + \frac{\partial T_{yz}}{\partial y} + \frac{\partial T_{zz}}{\partial z}
\end{aligned} \tag{12}$$

Donde T_{xx}, T_{yy} y T_{zz} son los esfuerzos normales y T_{xz}, T_{xy} y T_{yz} son los esfuerzos de cizalla que dependen de las velocidades V_x, V_y y V_z y viceversa. ρ es la densidad del medio. Debido a que la onda elástica se propaga en un medio isótropo, C_{11}, C_{12} y C_{44} corresponden a los parámetros de Lamé, por la simetría del medio. [11]

Cada uno de los componentes propios de la naturaleza de la propagación de una onda sísmica, ya mencionados, son descritos analíticamente por dichas ecuaciones. Para lograr generar una simulación de este fenómeno, se deben someter tales ecuaciones a una solución numérica y posteriormente a un algoritmo computacional que facilite el cálculo de los datos. Sin embargo, las máquinas de cómputo convencionales, no tienen

la suficiente capacidad para abarcar la gran cantidad de datos sísmicos que se requieren evaluar para obtener una imagen con una precisión adecuada. Por consiguiente, se hace necesario la búsqueda de nuevas tecnologías en donde se pueda implementar un algoritmo que permita evaluar la mayor cantidad de datos posibles con el mayor rendimiento.

Una de las soluciones computacionales emergentes que tiene mayor auge en el tratamiento de grandes cantidades de datos, es el uso de arquitecturas híbridas, en donde se logra hacer uso de las funcionalidades de una máquina de cómputo tradicional y un dispositivo encargado de la trata masiva de datos intercomunicados entre sí.

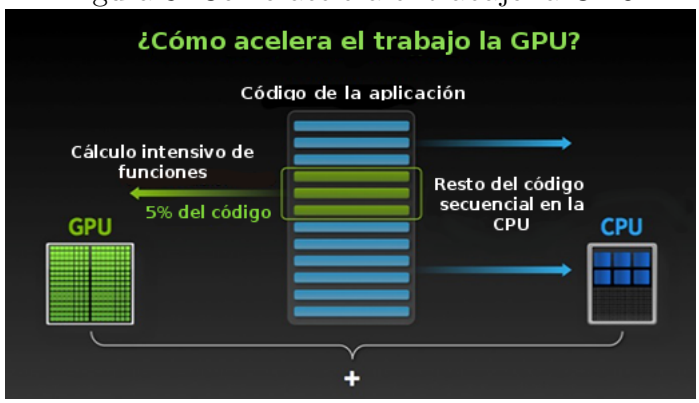
En la *sección 2*, se explica el uso de las arquitecturas híbridas para resolver problemas científicos.

2. USO DE ARQUITECTURAS HÍBRIDAS

Las arquitecturas híbridas CPU - GPU se han estado utilizando de manera masiva para acelerar aplicaciones de empresa, consumo, ingeniería, análisis y cálculo científico.

El cálculo acelerado en la GPU ofrece un rendimiento sin precedentes hoy día, ya que traslada las partes de la aplicación con mayor carga computacional a la GPU y deja el resto del código ejecutándose en la CPU. Desde la perspectiva del usuario, las aplicaciones simplemente se ejecutan más rápido. [12]

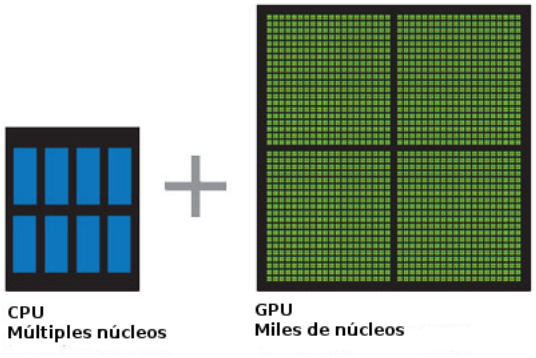
Figura 8: Cómo acelera el trabajo la GPU.



Nvidia «Qué es el GPU Computing» [12].

Una forma sencilla de entender la diferencia entre la CPU y la GPU es comparar la forma en que procesan las tareas. Como se mostró en la *Figura 8*, las aplicaciones se ejecutan de forma tradicional en la CPU, pero la GPU realiza los cálculos masivos. Una CPU está formada por varios núcleos optimizados para el procesamiento en serie, mientras que una GPU consta de millares de núcleos más pequeños y eficientes diseñados para manejar múltiples tareas simultáneamente (ver *Figura 9*). [12]

Figura 9: La CPU frente a la GPU. [12]



Nvidia «Qué es el GPU Computing» [12].

A continuación, en la *sección 3*, se muestra el procedimiento que se llevó a cabo para poder evaluar la solución, mediante la técnica FDTD, de la ecuación diferencial de una onda elástica en un medio isótropo y heterogéneo implementado para ejecutarse en arquitecturas híbridas y posteriormente, ser visualizada.

3. IMPLEMENTACIÓN DE LA SOLUCIÓN

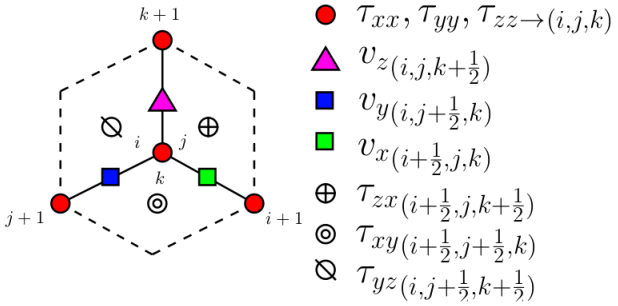
En el plan del proyecto se propuso la siguiente metodología, que se compone de seis fases. En la *Fase 1*, se desarrolló la ecuación diferencial de la onda elástica mediante la técnica FDTD; en la *Fase 2*, se muestran y se explican los software utilizados; en la *Fase 3*, se explica la implementación del algoritmo y su paralelización; en la *Fase 4*, se toman medidas de rendimiento y se analizan los resultados; en la *Fase 5*, se muestran las visualizaciones realizadas y en la *Fase 6*, se validan las soluciones.

A contiuación, en la *Fase 1*, se resuelve la ecuación de la onda elástica mediante el método numérico propuesto.

3.1. FASE 1: INVESTIGACIÓN Y DOCUMENTACIÓN DE LA NATURALEZA DE LA PROBLEMATICA A SOLUCIONAR.

De acuerdo a lo nombrado en la *sección 1*, se trabajaron las ondas elásticas en un medio isótropo y heterogeneo compuestas por seis componentes tensoriales de esfuerzos, (11) y (12) y sus velocidades (13). La ecuación diferencial se resolvió por el método de diferencias finitas con malla intercalada (ver *Figura 10*) con segundo orden de precisión en el tiempo y cuarto orden de precisión en el espacio. Éste método fué desarrollado por Kane Yee (1966) [13] y es utilizado para resolver ecuaciones de Maxwell. El esquema FDTD de malla intercalada esta basado en la aproximación de diferencias finitas de los operadores diferenciales tanto en el espacio como el tiempo de la ecuación de onda. [14]

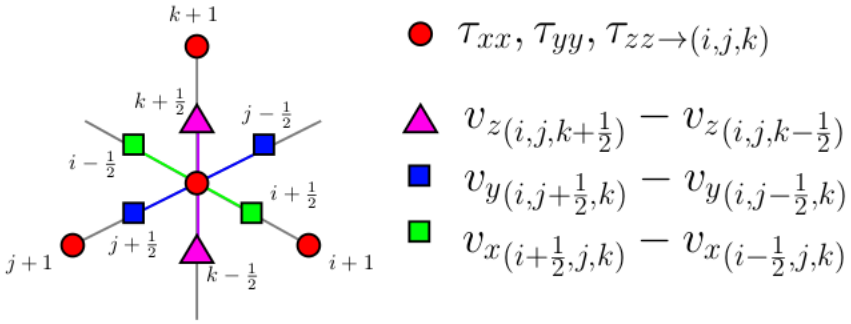
Figura 10: Método FDTD



GONZALEZ, H «Modelado Sísmico. Ecuación de Onda Elástica 2D/3D y Medios Anisóropos.» [15].

Se evalúan los próximos 4 puntos mas cercanos para obtener la ecuación discretizada de cuarto orden.

Figura 11: Método FDTD - Evaluando los esferzos.



GONZALEZ, H «Modelado Sísmico. Ecuación de Onda Elástica 2D/3D y Medios Anisóropos.» [15].

Aplicando el método de malla intercalada a las ecuaciones (11) y (12), guiado por la *Figura 11*, se obtienen las ecuaciones (13) a la (18):

$$\begin{aligned}
&\frac{T_{xx}^{n+\frac{1}{2}}(i,j,k) - T_{xx}^{n-\frac{1}{2}}(i,j,k)}{\partial t} = \\
&(\lambda+2\mu)\left[\frac{9}{8}\frac{V_x^n(i+\frac{1}{2},j,k) - V_x^n(i-\frac{1}{2},j,k)}{\partial x} - \frac{1}{24}\frac{V_x^n(i+\frac{3}{2},j,k) - V_x^n(i-\frac{3}{2},j,k)}{\partial x}\right] \\
&+(\lambda)\left[\frac{9}{8}\frac{V_y^n(i,j+\frac{1}{2},k) - V_y^n(i,j-\frac{1}{2},k)}{\partial y} - \frac{1}{24}\frac{V_y^n(i,j+\frac{3}{2},k) - V_y^n(i,j-\frac{3}{2},k)}{\partial y}\right] \\
&+(\lambda)\left[\frac{9}{8}\frac{V_z^n(i,j,k+\frac{1}{2}) - V_z^n(i,j,k-\frac{1}{2})}{\partial z} - \frac{1}{24}\frac{V_z^n(i,j,k+\frac{3}{2}) - V_z^n(i,j,k-\frac{3}{2})}{\partial z}\right]
\end{aligned} \tag{13}$$

$$\begin{aligned}
&\frac{T_{yy}^{n+\frac{1}{2}}(i,j,k) - T_{yy}^{n-\frac{1}{2}}(i,j,k)}{\partial t} = \\
&(\lambda)\left[\frac{9}{8}\frac{V_x^n(i+\frac{1}{2},j,k) - V_x^n(i-\frac{1}{2},j,k)}{\partial x} - \frac{1}{24}\frac{V_x^n(i+\frac{3}{2},j,k) - V_x^n(i-\frac{3}{2},j,k)}{\partial x}\right] \\
&+(\lambda+2\mu)\left[\frac{9}{8}\frac{V_y^n(i,j+\frac{1}{2},k) - V_y^n(i,j-\frac{1}{2},k)}{\partial y} - \frac{1}{24}\frac{V_y^n(i,j+\frac{3}{2},k) - V_y^n(i,j-\frac{3}{2},k)}{\partial y}\right] \\
&+(\lambda)\left[\frac{9}{8}\frac{V_z^n(i,j,k+\frac{1}{2}) - V_z^n(i,j,k-\frac{1}{2})}{\partial z} - \frac{1}{24}\frac{V_z^n|(i,j,k+\frac{3}{2}) - V_z^n|(i,j,k-\frac{3}{2})}{\partial z}\right]
\end{aligned} \tag{14}$$

$$\frac{T_{zz}^{n+\frac{1}{2}}(i,j,k) - T_{zz}^{n-\frac{1}{2}}(i,j,k)}{\partial t} =$$
$$(\lambda)\left[\frac{9}{8}\frac{V_x^n(i+\frac{1}{2},j,k) - V_x^n(i-\frac{1}{2},j,k)}{\partial x} - \frac{1}{24}\frac{V_x^n(i+\frac{3}{2},j,k) - V_x^n(i-\frac{3}{2},j,k)}{\partial x}\right]$$
$$+(\lambda)\left[\frac{9}{8}\frac{V_y^n(i,j+\frac{1}{2},k) - V_y^n(i,j-\frac{1}{2},k)}{\partial y} - \frac{1}{24}\frac{V_y^n(i,j+\frac{3}{2},k) - V_y^n(i,j-\frac{3}{2},k)}{\partial y}\right] \quad (15)$$
$$+(\lambda+2\mu)\left[\frac{9}{8}\frac{V_z^n(i,j,k+\frac{1}{2}) - V_z^n(i,j,k-\frac{1}{2})}{\partial z} - \frac{1}{24}\frac{V_z^n(i,j,k+\frac{3}{2}) - V_z^n(i,j,k-\frac{3}{2})}{\partial z}\right]$$

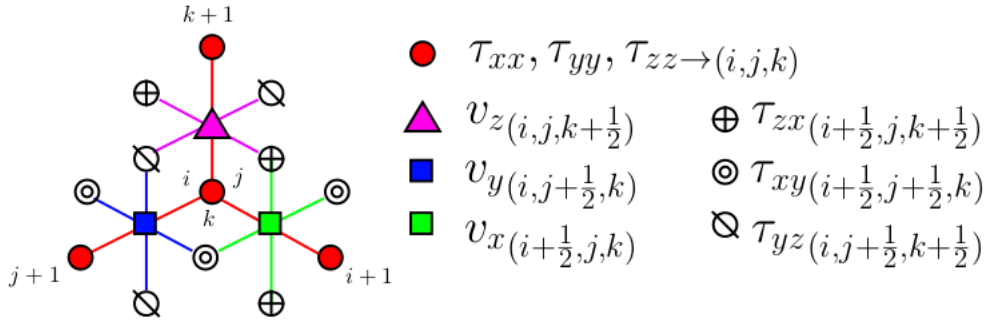
$$\frac{T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) - T_{xz}^{n-\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2})}{\partial t} =$$
$$\mu\left[\frac{9}{8}\frac{v_x^n(i+\frac{1}{2},j,k+1) - v_x^n(i+\frac{1}{2},j,k)}{\partial z} - \frac{1}{24}\frac{v_x^n(i+\frac{1}{2},j,k+2) - v_x^n(i+\frac{1}{2},j,k-1)}{\partial z}\right] \quad (16)$$
$$+\mu\left[\frac{9}{8}\frac{v_z^n(i+1,j,k+\frac{1}{2}) - v_x^n(i,j,k+\frac{1}{2})}{\partial x} - \frac{1}{24}\frac{v_z^n(i+2,j,k+\frac{1}{2}) - v_z^n(i-1,j,k+\frac{1}{2})}{\partial x}\right]$$

$$\frac{T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k) - T_{xy}^{n-\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k)}{\partial t} =$$
$$\mu\left[\frac{9}{8}\frac{v_x^n(i+\frac{1}{2},j+1,k) - v_x^n(i+\frac{1}{2},j,k)}{\partial y} - \frac{1}{24}\frac{v_x^n(i+\frac{1}{2},j+2,k) - v_x^n(i+\frac{1}{2},j-1,k)}{\partial y}\right] \quad (17)$$
$$+\mu\left[\frac{9}{8}\frac{v_y^n(i+1,j+\frac{1}{2},k) - v_x^n(i,j+\frac{1}{2},k)}{\partial x} - \frac{1}{24}\frac{v_y^n(i+2,j+\frac{1}{2},k) - v_y^n(i-1,j+\frac{1}{2},k)}{\partial x}\right]$$

$$\frac{T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) - T_{yz}^{n-\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2})}{\partial t} =$$
$$\mu\left[\frac{9}{8}\frac{v_y^n(i,j+\frac{1}{2},k+1) - v_y^n(i,j+\frac{1}{2},k)}{\partial z} - \frac{1}{24}\frac{v_y^n(i,j+\frac{1}{2},k+2) - v_y^n(i,j+\frac{1}{2},k-1)}{\partial z}\right] \quad (18)$$
$$+\mu\left[\frac{9}{8}\frac{v_z^n(i,j+1,k+\frac{1}{2}) - v_z^n(i,j,k+\frac{1}{2})}{\partial y} - \frac{1}{24}\frac{v_z^n(i,j+2,k+\frac{1}{2}) - v_z^n(i,j-1,k+\frac{1}{2})}{\partial y}\right]$$

Multiplicado por sus respectivos parámetros de Lamé, en la parte izquierda de la solución se evalúan mediante el método FDTD, los próximos vecinos, multiplicando por el coeficiente de diferencias finitas de cuarto orden $\frac{9}{8}$. Y en la parte derecha de la solución, se evaluan los segundos próximos vecinos, multiplicando por el coeficiente de diferencias finitas de cuarto orden $\frac{1}{24}$. De la misma manera se hace el cálculo de las velocidades, según la ecuación (13), siguiendo el esquema mostrado en la *Figura 12*.

Figura 12: Método FDTD - Evaluando las velocidades.



GONZALEZ, H «Modelado Sísmico. Ecuación de Onda Elástica 2D/3D y Medios Anisóropos.» [15].

Calculando las velocidades con respecto a los esfuerzos, obtenemos las ecuaciones (19), (20) y (21):

$$\begin{aligned}
&\frac{v_x^{n+1}(i+\frac{1}{2},j,k)+v_x^{n}(i+\frac{1}{2},j,k)}{\partial t} = \\
&\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{xx}^{n+\frac{1}{2}}(i+1,j,k)-T_{xx}^{n+\frac{1}{2}}(i,j,k)}{\partial x}-\frac{1}{24}\frac{T_{xx}^{n+\frac{1}{2}}(i+2,j,k)-T_{xx}^{n+\frac{1}{2}}(i-1,j,k)}{\partial x}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k)-T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j-\frac{1}{2},k)}{\partial y}-\frac{1}{24}\frac{T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{3}{2},k)-T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j-\frac{3}{2},k)}{\partial y}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2})-T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k-\frac{1}{2})}{\partial z}-\frac{1}{24}\frac{T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{3}{2})-T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k-\frac{3}{2})}{\partial z}\right]
\end{aligned} \tag{19}$$

$$\begin{aligned}
&\frac{v_y^{n+1}(i,j+\frac{1}{2},k)+v_y^{n}(i,j+\frac{1}{2},k)}{\partial t} = \\
&\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{xy}^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k)-T_{xy}^{n+\frac{1}{2}}(i-\frac{1}{2},j+\frac{1}{2},k)}{\partial x}-\frac{1}{24}\frac{T_{xy}^{n+\frac{1}{2}}(i+\frac{3}{2},j+\frac{1}{2},k)-T_{xy}^{n+\frac{1}{2}}(i-\frac{3}{2},j+\frac{1}{2},k)}{\partial x}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{yy}^{n+\frac{1}{2}}(i,j+1,k)-T_{yy}^{n+\frac{1}{2}}(i,j,k)}{\partial y}-\frac{1}{24}\frac{T_{yy}^{n+\frac{1}{2}}(i,j+2,k)-T_{yy}^{n+\frac{1}{2}}(i,j-1,k)}{\partial y}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2})-T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k-\frac{1}{2})}{\partial z}-\frac{1}{24}\frac{T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{3}{2})-T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k-\frac{3}{2})}{\partial z}\right]
\end{aligned} \tag{20}$$

$$
\begin{aligned}
&\frac{v_z^{n+1}(i,j,k+\frac{1}{2}) + v_z^{n}(i,j,k+\frac{1}{2})}{\partial t} = \\
&\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{xz}^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) - T_{xz}^{n+\frac{1}{2}}(i-\frac{1}{2},j,k+\frac{1}{2})}{\partial x} - \frac{1}{24}\frac{T_{xz}^{n+\frac{1}{2}}(i+\frac{3}{2},j,k+\frac{1}{2}) - T_{xz}^{n+\frac{1}{2}}(i-\frac{3}{2},j,k+\frac{1}{2})}{\partial x}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{yz}^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) - T_{yz}^{n+\frac{1}{2}}(i,j-\frac{1}{2},k+\frac{1}{2})}{\partial y} - \frac{1}{24}\frac{T_{yz}^{n+\frac{1}{2}}(i,j+\frac{3}{2},k+\frac{1}{2}) - T_{yz}^{n+\frac{1}{2}}(i,j-\frac{3}{2},k+\frac{1}{2})}{\partial y}\right] \\
&+\frac{1}{\rho}\left[\frac{9}{8}\frac{T_{zz}^{n+\frac{1}{2}}(i,j,k+1) - T_{zz}^{n+\frac{1}{2}}(i,j,k)}{\partial z} - \frac{1}{24}\frac{T_{zz}^{n+\frac{1}{2}}(i,j,k+2) - T_{zz}^{n+\frac{1}{2}}(i,j,k-1)}{\partial z}\right]
\end{aligned}
\tag{21}
$$

Cuando se hace una aproximación algebraica a una ecuación diferencial, el primer requerimiento que tiene que cumplir es la *consistencia*.

Para solucionar problemas de estabilidad determinados por la consistencia del método numérico a través del tiempo, mientras se propagan los errores, se aplica la condición de estabilidad correspondiente, mostrada en el *Tabla 1* [16].

Tabla 1: Condición de estabilidad.

Dimensión	2- Orden $\frac{\partial t}{\partial x}$	2- Orden ∂t/4-Orden ∂x
1-D	1	$\frac{\sqrt{3}}{2}$
2-D	$\frac{1}{\sqrt{2}}$	$\sqrt{\frac{3}{8}}$
3-D	$\frac{1}{\sqrt{3}}$	$\frac{1}{2}$

Por otro lado, la *precisión* está relacionada con los errores locales de dos tipos, el primero con los errores de redondeo (tamaño para representar valores digitalmente), y los segundos errores por el truncamiento en la expansión de la serie de Taylor de los operadores diferenciales.

Para solucionar problemas de precisión, se hace:

$$\Delta x = \frac{\lambda_{min}}{10} = \frac{C_{min}}{10 \cdot 2f_0} \tag{22}$$

Se concluye que solucionando la ecuación de onda elástica mediante la técnica FDTD se calculan los campos en una malla intercalada, lo que aumenta la precisión de la solución, además se pueden corregir problemas de consistencia y presición para mejorar los resultados.

Una vez visto el modelo matemático, en la *Fase 2*, se tratarán las herramientas y mecanismos de solución implementados durante el desarrollo del proyecto.

3.2. FASE 2: ESTUDIO DE LAS SOLUCIONES EXISTENTES Y LAS HERRAMIENTAS A UTILIZAR

Como inducción al área del presente trabajo, nos relacionamos con la teoría de ondas sísmicas y realizamos prácticas de poca dificultad para relacionarnos con los software de trabajo. Se aumentó la dificultad de las prácticas a medida que se dominaban las herramientas.

El equipo utilizado fué GUANE-1, una plataforma de supercomputación híbrida basada en NVIDIA® TESLA GPUs, de la cual se utilizó una de las ocho GPUs que posee y cuenta con 2 procesadores XEON e5645 a 2.4 GHZ y 104 GB de RAM, 6 cores reales por procesador y con 2 hilos por core, para un total de 24 cores conectados mediante una red Ethernet 100/1000 /Infiniband/10 Gigabits.

Las soluciones se desarrollaron con ayuda de *Madagascar*, un software de código abierto para el análisis de datos multidimensionales y reproducibles para experimentos computacionales. Su objetivo es proporcionar un entorno de desarrollo y procesamiento de datos sísmicos eficaz.

Madagascar usa como formato estándar el *RSF* para intercambio, procesamiento, y tratamiento de datos. Conceptualmente, el formato *RSF* significa archivo de datos muestreados regularmente (Regularly Sampled File), cuya información está ordenada en forma de matriz hiperdimensional o hipercubo en un único arreglo.

Bajo el concepto de almacenamiento de datos, los archivos binario *RSF* de Madagascar manejan una representación multidimensional (hipercubo) en un único arreglo unidimensional. Esto tiene la ventaja de mantener la eficiencia y simplicidad para acceder la información.

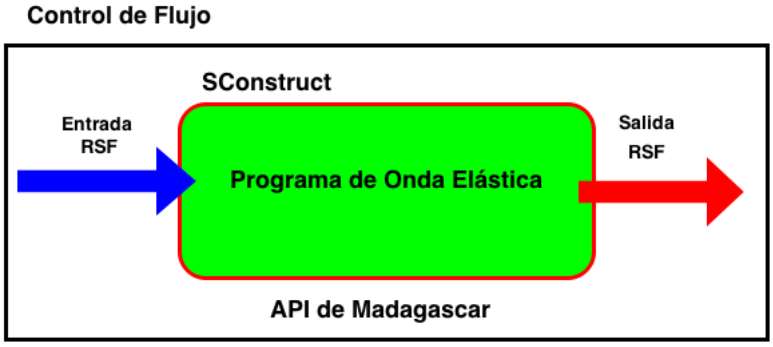
Los datos-hipercubo RSF están definidos por dos archivos, uno llamado *header* y otro binario que contiene los datos. El header contiene información sobre la dimensionalidad del hipercubo y su contenido. El archivo binario es almacenado remotamente (directorio separado), y contiene el hipercubo-dato referenciado por el header. Dado que el hipercubo-dato puede ser muy grande (Megas Bytes o Tera Bytes) usualmente se almacenan en un directorio remoto con el sufijo *.rsf@. [17]

Madagascar, además, posee una librería llamada *rsf.h*, la cual trabajamos con el lenguaje de programación C y el paradigma de paralelización CUDA. Para intercomunicar estas herramientas se utilizó *Scons*.

Scons es la parte central del armado de los módulos-programa de Madagascar, también es usado para la ejecución de flujos de datos entre programas y hacer reproducible experimentos computacionales de manera transparente. Es decir que cumple con dos objetivos de manera independiente [18]:

- Compilar programas codificados en C
- Definir flujos de ejecución entre los datos y programas de Madagascar.

Figura 13: Entradas y salidas de flujos rsf.

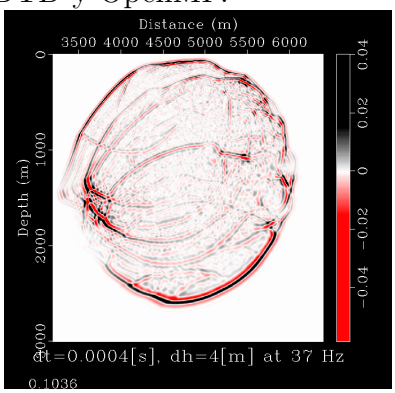


Fuente: Autores

En la *Figura 13* se observó el comportamiento que tiene la ejecución de los programas. El archivo *Sconstruct*, en este caso, sirve para el control de flujos. Al principio se ejecutan cálculos que se toman como flujos de entrada que va a tener el programa en C con el API Madagascar, y este a su vez, genera un archivo de salida que se manipuló en el archivo *Sconstruct*, para crear un archivo reproducible. Esta es la base para la ejecución de las aplicaciones que se trabajó durante el proyecto, con la que se realizaron prácticas para escalar la dificultad del problema.

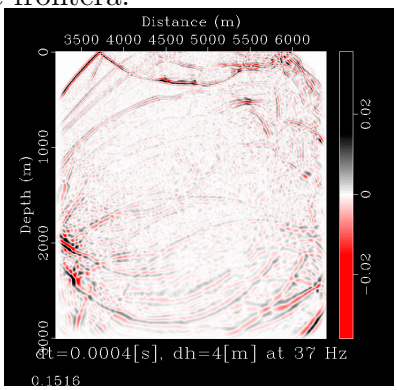
Una de las implementaciones más sobresalientes abarca la solución de una onda acústica en dos dimensiones, utilizando el método de diferencias finitas con malla intercalada y aplicando OpenMP. El resultado de la simulación se muestra en la *Figura 14* y *Figura 15*

Figura 14: Onda Acústica con FDTD y OpenMP.



Fuente: Autores

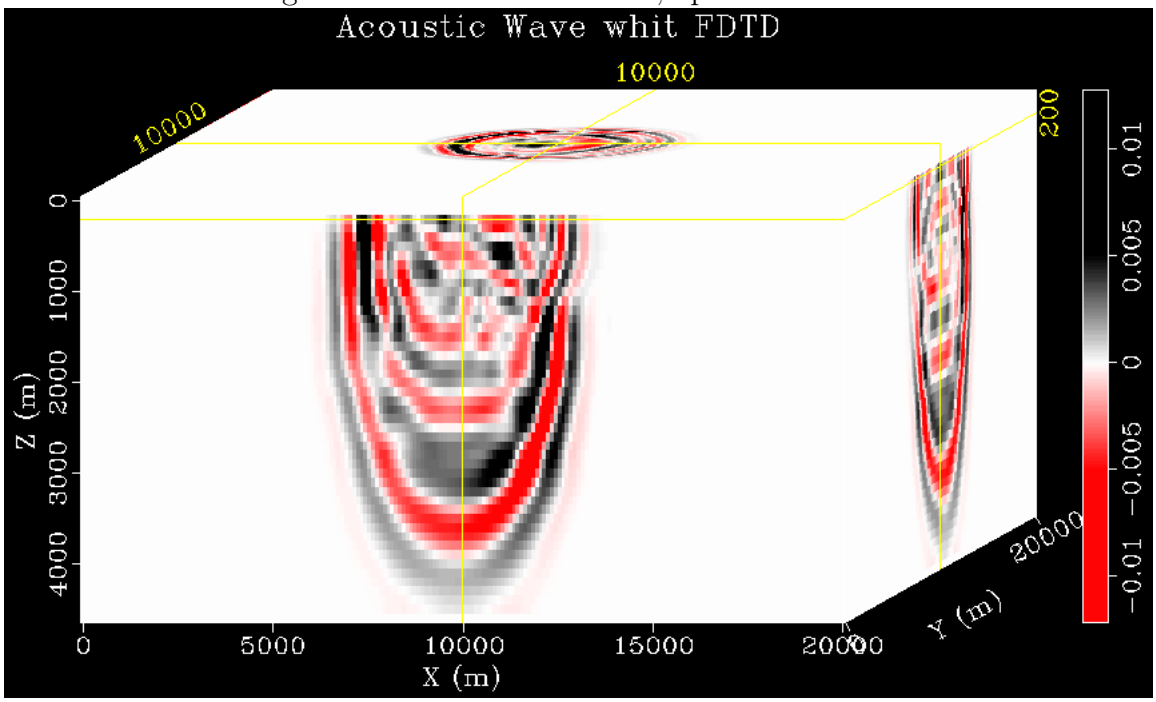
Figura 15: Se aplican condiciones de frontera.



Fuente: Autores

Otro de los problemas estudiados consistía en modelar la solución de la ecuación de una onda acústica en tres dimensiones, utilizando diferencias finitas con malla intercalada y acelerado con OpenMP. El resultado se puede observar en la *Figura 16*.

Figura 16: Onda Acústica 3D, aplicando FDTD.



Fuente: Autores

Las implementaciones trabajadas con OpenMP, aumentaron el rendimiento, ya que se comparte la memoria de todos los cores, aumentando el rendimiento de la aplicación. Cuando se aumenta la cantidad de procesadores en la máquina de cómputo, aumenta el rendimiento de la ejecución del algoritmo. [19]

Conocidas las herramientas utilizadas, se describe en la *Fase 3*, el uso de las GPUs y la implementación del algoritmo con el estandar de paralelización CUDA.

3.3. FASE 3: MEDICIÓN DEL GRADO DE PARALELIZACIÓN Y OPIMIZACIÓN DE LOS ALGORITMOS PARA EJECUTAR SOBRE GPU'S

Para programar la solución de una ecuación diferencial, primero se tiene que discretizar, para pasarla de su forma continua a una forma finita. Luego se diseña el algoritmo para ser implementado.

Para codificar el conjunto de ecuaciones comprendido entre la ecuación (13) y la ecuación (21), se remplazan los índices:

- $i + \frac{1}{2}$ por i.
- $i - \frac{1}{2}$ por $i - 1$
- $i + \frac{3}{2}$ por $i + 1$.
- $i - \frac{3}{2}$ por $i - 2$

Aplica también a los índices $j \pm \frac{1}{2}$, $k \pm \frac{1}{2}$, $j \pm \frac{3}{2}$ y $k \pm \frac{3}{2}$.

Los demás índices enteros como i, $i + 1$, $i - 1$, $i + 2$ ó $i - 2$ se mantienen.

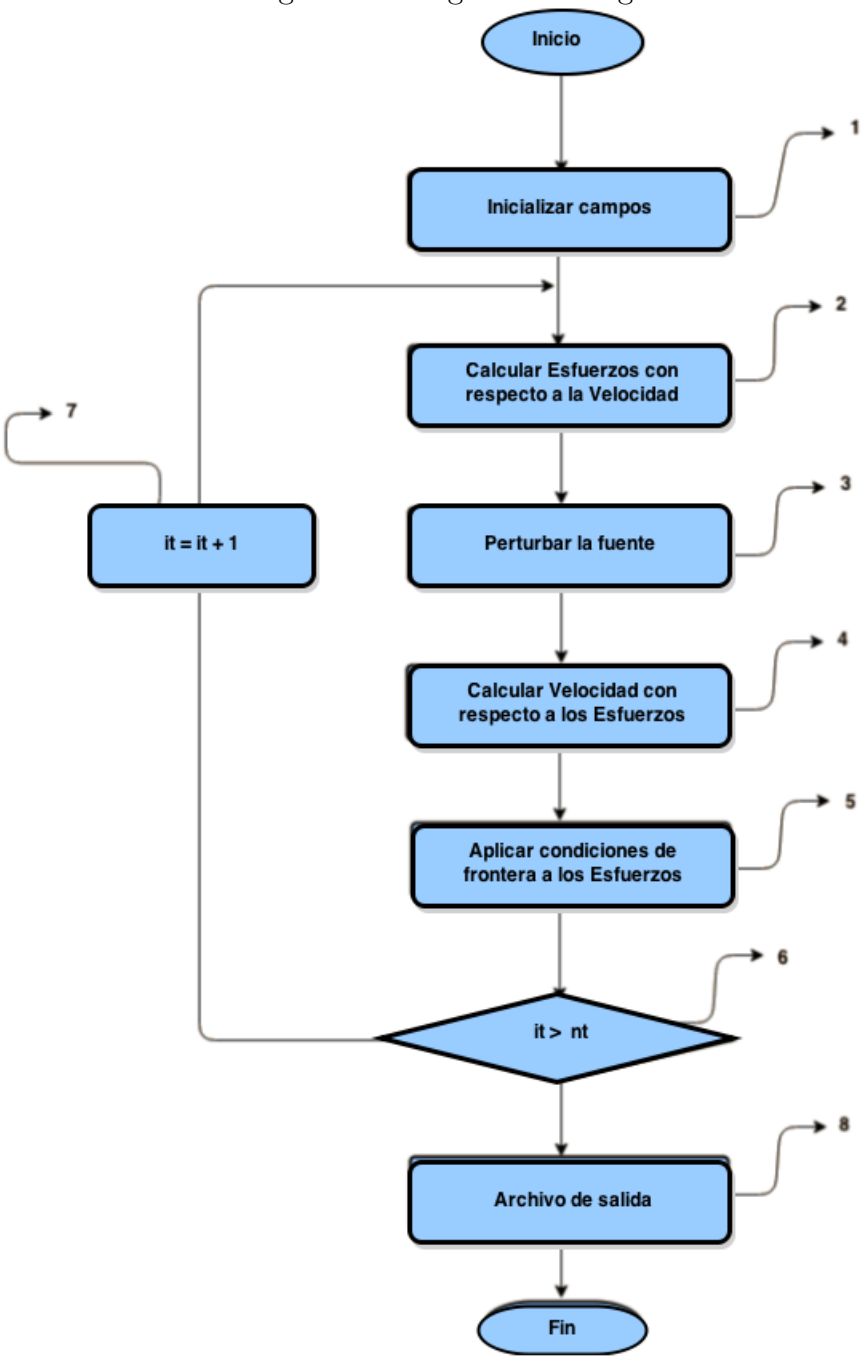
En el siguiente cuadro se ve cómo se escribiría el código en C, primero se define un macro que determina el recorrido de la matriz.

```
...
#define Id(A,i,j,k) (A)[ (i)*ny*nz + (j)*nz + (k) ]
...

// Inicio del ciclo de pasos de tiempo.
for (it=0; it<nt; it++) {

 /* Actualizacion de los esfuerzos con respecto a las velocidades*/
 for (i=2; i<nx-1; i++) {
  for (j=2; j<ny-1; j++) {
   for (k=2; k<nz-1; k++) {
    //coefficients
    cc = Id(svel,i,j,k)*Id(svel,i,j,k)*Id(rho,i,j,k); //mu
    cb = (Id(vel,i,j,k)*Id(vel,i,j,k)*Id(rho,i,j,k))-2*cc; //lambda
    ca = cb+(2*cc);

    Id(Txx,i,j,k) += (ca*dt*idx*(1.125f*(Id(Vx,i,j,k)-Id(Vx,i-1,j,k)) -
                      ((Id(Vx,i+1,j,k)-Id(Vx,i-2,j,k))/24.f)))   +
                      (cb*dt*idy*(1.125f*(Id(Vy,i,j,k)-Id(Vy,i,j-1,k)) -
                      ((Id(Vy,i,j+1,k)-Id(Vy,i,j-2,k))/24.f)))+
                      (cb*dt*idz*(1.125f*(Id(Vz,i,j,k)-Id(Vz,i,j,k-1)) -
                      ((Id(Vz,i,j,k+1)-Id(Vz,i,j,k-2))/24.f)));
    ...
   }
  }
 }
  ...

  /* Perturbacion de la onda en la fuente */

  ...

 /* Actualizacion de las velocidades con respecto a los esfuerzos */

 for (i=1; i<nx-2; i++) {
  for (j=1; j<ny-2; j++) {
   for (k=1; k<nz-2; k++) {

    rhodt = dt/Id(rho,i,j,k);
    Id(Vx,i,j,k) += rhodt*((idx*1.125f*(Id(Txx,i+1,j,k)-Id(Txx,i,j,k)))
                   -(idx*(Id(Txx,i+2,j,k)-Id(Txx,i-1,j,k))/24.f)+
                    (idy*1.125f*(Id(Txy,i,j,k)-Id(Txy,i,j-1,k))) -
                    (idy*(Id(Txy,i,j+1,k)-Id(Txy,i,j-2,k))/24.f)+
                    (idz*1.125f*(Id(Txz,i,j,k)-Id(Txz,i,j,k-1))) -
                    (idz*(Id(Txz,i,j,k+1)-Id(Txz,i,j,k-2))/24.f));

    ...
   }
  }
 }
 ...
```

Después de definir las variables y constantes, se empieza el ciclo del tiempo en donde se calculan los esfuerzos y las velocidades para cada paso de tiempo. En los ciclos de los esfuerzos y las velocidades se intercalan las mallas en el tiempo. Se puede apreciar también, la codificación de índices que se comentó anteriormente.

A continuación, se explica como se ejecuta el código en serial. En la *Figura 17* se puede ver como funciona el algoritmo implementado:

Figura 17: Diagrama del algoritmo.

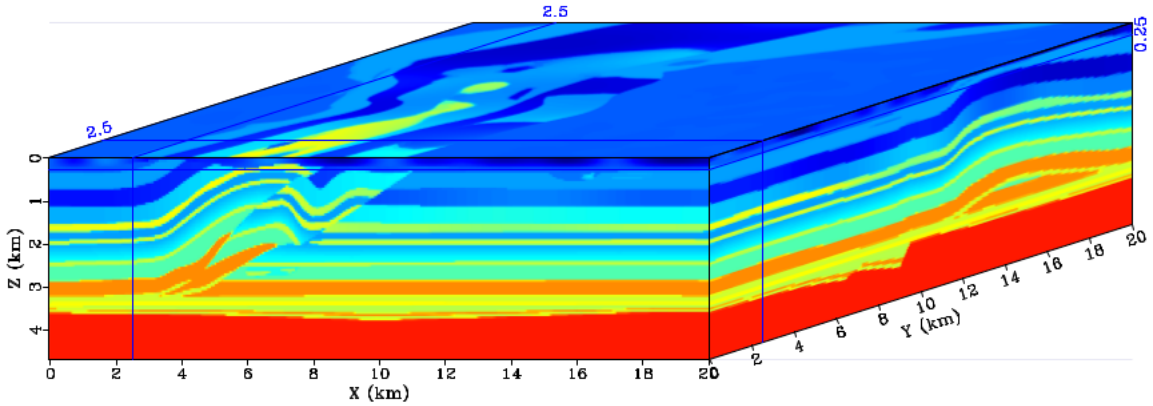


Fuente: Autores

1. Se asignan los ejes del hipercubo y se inicializan todas las variables que se van a trabajar: constantes de dimensiones, diferenciales, constantes auxiliares, los datos de la fuente, los vectores de esfuerzos normales y de cizalla y de velocidades.

   Algunos de estos datos, como el campo de velocidad 3D Overthrust (SEG/EAGE, mostrado en la *Figura 18*) y el modelo de Ricker (función de perturbación), se reciben desde el flujo de entrada.

Figura 18: Campo de velocidad 3D Overthrust SEG/EAGE.



Tomado de: SEG/EAGE EAGE, SEG «SEG/EAGE 3D Overthrust and Salt Models» [20].

2. Iterando en el tiempo, se actualizan los valores de las Esfuerzos con respecto a los valores de las Velocidades.

3. Se perturbó la fuente de la onda con el modelo de Ricker, utilizamos esta función para simular el disparo que genera la onda.

4. En una malla intercalada, se calculan los valores de las Velocidades con respecto a los Esfuerzos.

5. Se aplican las condiciones de frontera (Cerjan 1985)[21] para corregir errores de reflexión en los bordes del cubo sísmico.

6. El ciclo se repite en caso que la iteración sea menor al número de pasos totales de tiempo.

7. En caso de que la iteración sea menor al número de pasos de tiempo totales, se suma una iteración más.

8. En caso de completar los pasos de tiempo totales, se escribe el archivo para visualizar.

La condición de estabilidad para la ecuación en diferencias finitas de 4-Orden en el espacio y 2-Orden en el tiempo [16] es:

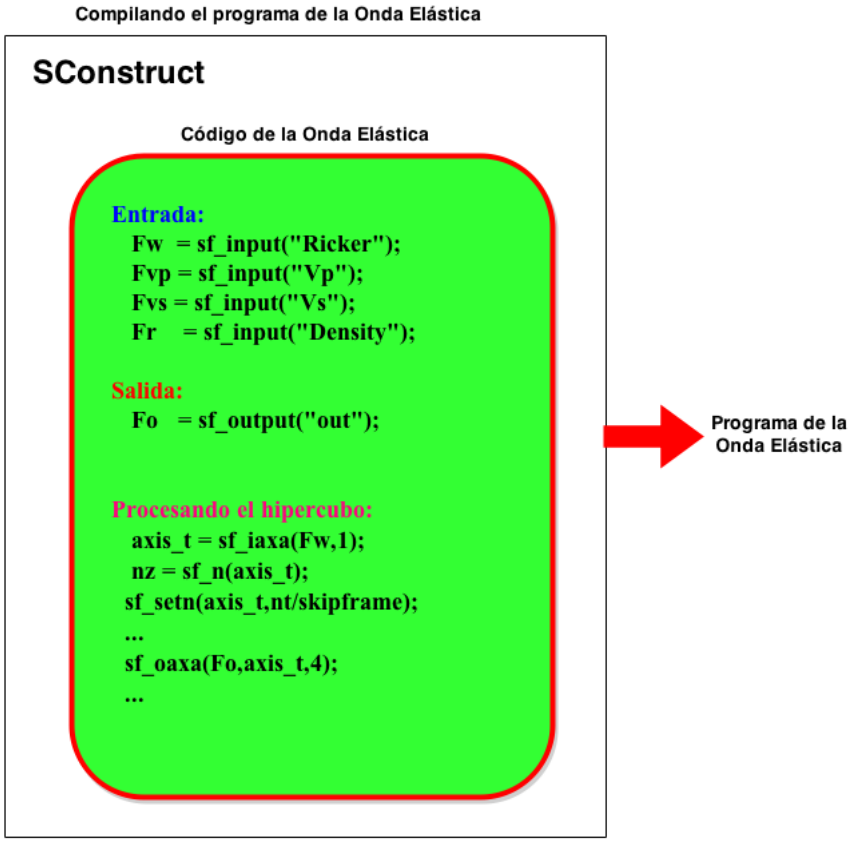
$$\Delta t = \frac{1}{2} \cdot \frac{\Delta x}{C_{max}} \tag{23}$$

Donde Δt es el tamaño del paso de tiempo, Δx es el tamaño de malla y C_{max} es el valor de la velocidad máxima del modelo.

Diversos ciclos anidados pueden ser paralelizados sin generar dependencias, entre ellos están el cálculo de los esfuerzos normales y de cizalla, el cálculo de las velocidades y las condiciones de frontera para cada lado del cubo simulado. Seguido a esto, se explica como se ejecutó el código implementado para ejecutarse sobre arquitecturas híbridas.

Utilizando Scons, se compila el código CUDA-C y se crea el archivo ejecutable (ver *Figura 19*):

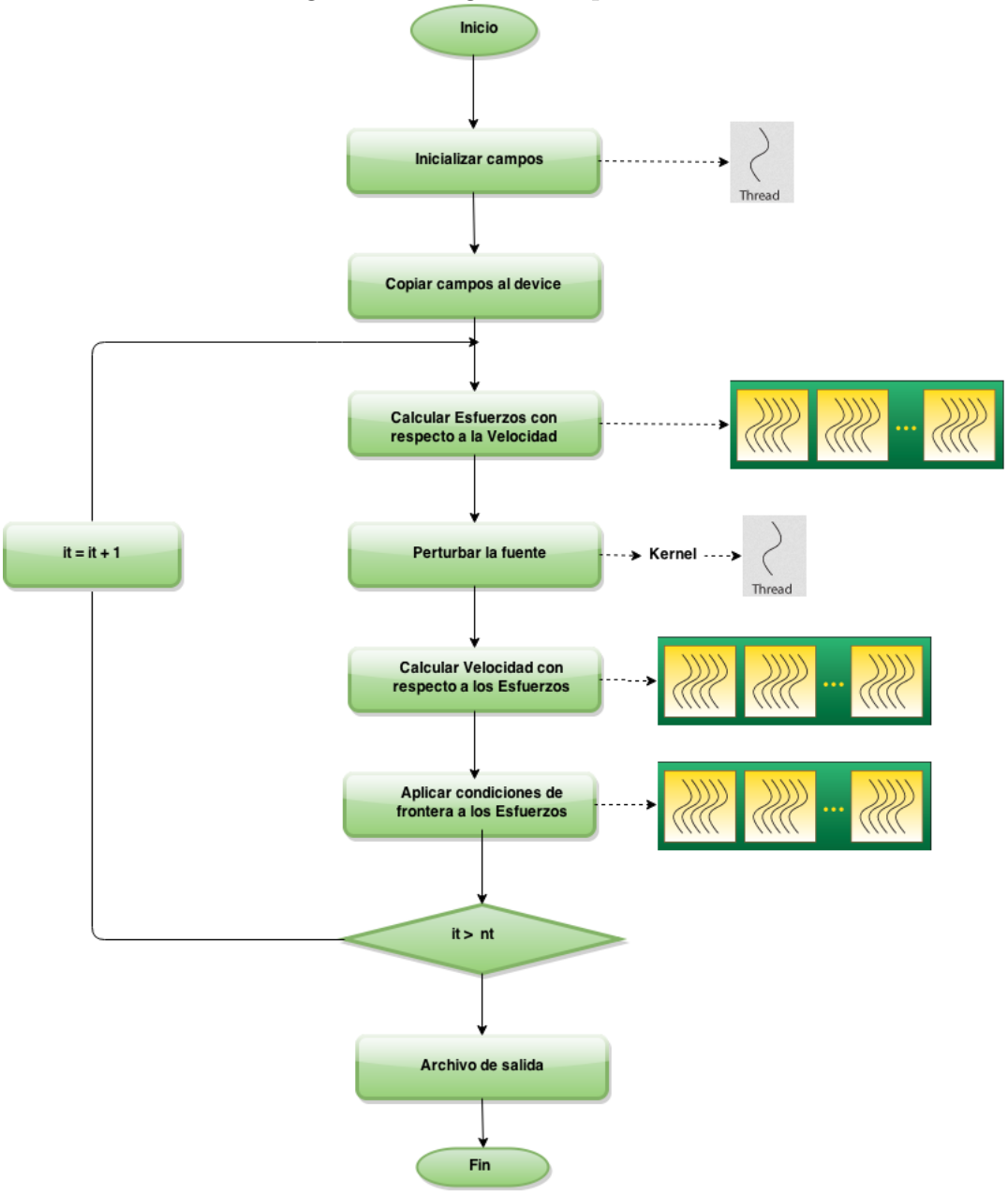
Figura 19: Compilación del programa.



Fuente: Autores

En la *Figura 20* se puede observar como se efectuó el código implementado en paralelo:

Figura 20: Diagrama de paralelización.



Fuente: Autores

1. Se define la fuente de perturbación, los pasos de tiempo, la velocidad mínima, la velocidad máxima, el tamaño de celda, el valor del paso de tiempo y la frecuencia.

2. Se particionan el modelo de velocidades (Campo de velocidad 3D Overthrust, el

cual va a ser nuestra velocidad de la onda P). Todos los cálculos se guardarán en archivos binarios RSF.

3. Se calcula la función perturbadora de Ricker.

4. Haciendo uso de un ajuste polinomial calculado por Brocher [22], relacionando la velocidad de las ondas-P y la densidad de la corteza terrestre, se calcula la densidad:

$$\rho\left[\frac{g}{cm^3}\right] = 0{,}000106V_p^5 - 0{,}0043V_p^4 + 0{,}0671V_p^3 - 0{,}4721V_p^2 + 1{,}6612V_p \qquad (24)$$

   válido para rangos de velocidad $1{,}5 < V_p\left[\frac{Km}{s}\right] < 8{,}5$

5. De la misma manera se relaciona la velocidad de las ondas P con la velocidad de las ondas S

$$V_s\left[\frac{Km}{s}\right] = 0{,}0064V_p^4 - 0{,}1238V_p^3 + 0{,}7949V_p^2 - 1{,}234V_p + 0{,}7858 \qquad (25)$$

   válido para rangos de velocidad $1{,}5 < V_p\left[\frac{Km}{s}\right] < 8$

6. Se ejecuta la aplicación incluyendo como parámetros las coordenadas de la fuente de perturbacion, los archivos binarios RSF de la velocidad de la onda P, la velocidad de la onda S y la densidad.

   En este paso empieza a ejecutarse el código CUDA-C del diagrama de la *Figura 20*.

   - Se leen los datos RSF que se tomaron desde el flujo de entrada.
   - Se crean los campos de los Esfuerzos y las Velocidades en el *Host*.
   - Se asignan los ejes del hipercubo.
   - Se crean los todos los campos que se utilizan en el *device* (velocidad de la onda P, velocidad de la onda S, densidad, Esfuerzos, Velocidades y función Ricker).
   - Se hace copia al *Device* de las matrices que se van a utilizar en el kernel.
   - Al iniciar los pasos de tiempo, se calculan los esfuerzos en paralelo en la GPU.
   - Se perturbó la onda en la fuente con la función Ricker, esto se hace dentro de una GPU en forma serial, para ahorrar pasos de memoria de *GPU* a *CPU* y viceversa.

- Se calculó la velocidad, con respecto a los esfuerzos, esto se calcula en paralelo en una GPU.
- Para terminar el cálculo del paso de tiempo, se le aplica a la onda las condiciones de frontera para que ésta no se refleje, esto tambíen se hace en paralelo, en un kernel diferente.
- El proceso se repite hasta completar los pasos de tiempo.
- Para ahorrar tiempo en la escritura de datos, se pasó un parametro entero *skipframe* para imprimir cada vez que el paso de tiempo sea múltiplo de este.

7. El resultado obtenido de la ejecución de la aplicación es un archivo binario de formato RSF que contiene la solución de la onda elástica en 3D para medios isótropos.

8. Haciendo uso de comandos de Madagascar, se toma como entrada el archivo RSF resultante y se crea un archivo reproducible en formato .vpl que será nuestra animación.

Conocida la forma como se implementó el algortimo para ser ejecutado sobre GPUs, se muestra en el siguiente punto, dos estrategias de separación de dominios definidas para comprobar el rendimiento que se puede obtener al ejecutar el algoritmo sobre múltiples GPUs y la forma de hacer una adecuada distribución de los datos para calcular.

3.3.1. Estrategias de separación de dominios.

La aplicación expuesta anteriormente calcula los esfuerzos y las velocidades en una GPU y en un kernel (función que ejecuta todos los procesos instanciados de forma paralela en la GPU), lo cual significa que el dominio no se separa.

Para la separación de los dominios, se siguen los pasos de la *Figura 20* pero al momento de calcular los esfuerzos, se divide el dominio (el eje x en el caso de las pruebas realizadas), luego se perturba la fuente en todo el dominio y se vuelve a dividir para calcular las velocidades.

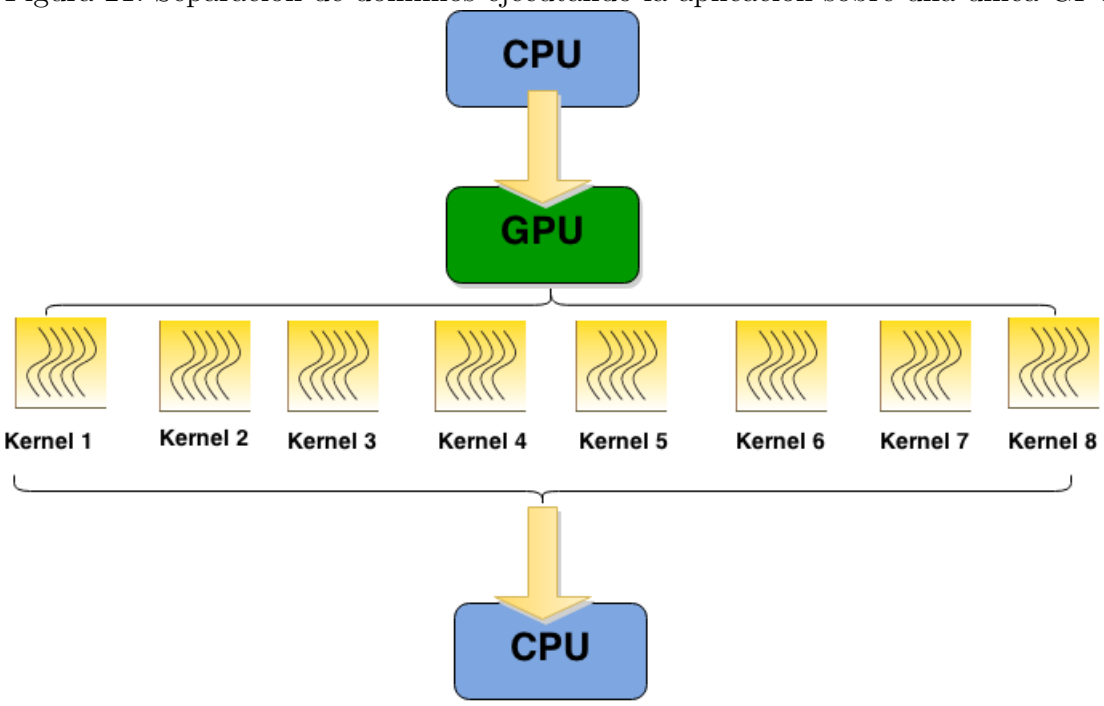
A continuación se exponen dos diferentes estrategias de separar los dominios:

1. **Separación de dominios ejecutando la aplicación sobre una única GPU.**

En este caso se crean tantos kernels como se requiera dividir el cáculo de los esfuerzos, y en cada uno de ellos se calcula las n partes de la división del dominio que se haya determinado; seguido a esto, se perturba la fuente y se invocan otros m kernels que calcularán las velocidades. Este método es poco ortodoxo, pero obtiene un rendimiento similar, aunque más elevado, en comparación con el código sin separación de dominios.

Hay que tener en cuenta que los kernels que se ven en la *Figura 21* se ejecutan uno despues de otro y no todos en el mismo momento porque no se están ejecutando con corrientes (*streams*) de cálculos debido a que aumenta la complejidad.

Figura 21: Separación de dominios ejecutando la aplicación sobre una única GPU.



Fuente: Autores

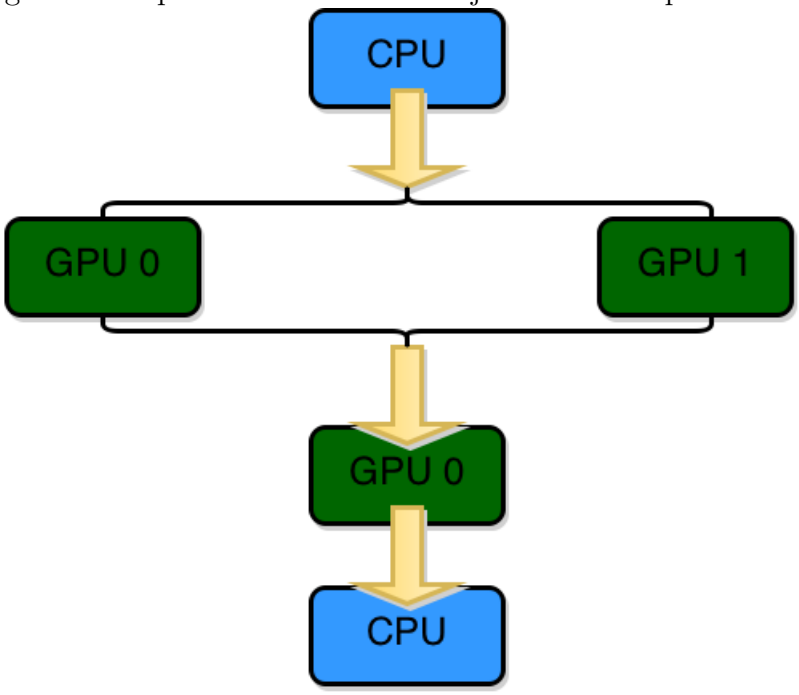
2. **Separación de dominios ejecutando la aplicación en múltiples GPUs.**

Aplicando esta estrategia, se crean todos los campos que se necesitan en las diferentes GPUs en donde se va a trabajar, en este caso, dos GPUs; se copia todo el dominio en las dos GPUs, pero se divide el cálculo del dominio (el eje x) en los kernels, se calculan los enfuerzos en dos kernels ejecutados en una GPU cada uno, luego se perturba la fuente en el dominio que cada GPU está manejando; para calcular las velocidades se sigue el mismo procedimiento; se aplican las condiciones de frontera para cada mitad del dominio desde cada una de las GPUs

y por último, en la denominada GPU 0, se crea un campo auxiliar en donde se retorna el valor del vector calculado en GPU 1, para posteriormente unificar los resultados de la tensión a visualizar.

Esta estrategia es más común para la división de los dominios, sin embargo en los resultados obtenidos, la aceleración es similar a la de la aplicación ejecutada en una GPU debido a la cantidad de pasos de memoria y comunicación que tiene que hacer una GPU con otra o con la misma CPU. Por otra parte, este método puede ser útil para calcular matrices de datos mucho más grandes que la trabajada en este proyecto, por lo que la aceleración aumentaría considerablemente.

Figura 22: Separación de dominios ejecutando la aplicación en múltiples GPUs.



Fuente: Autores

Los datos se intercomunican para cada paso de tiempo y se unifican en la GPU0 y no en la CPU, para evitar numerosos pasos de memoria entre CPU y GPU y viceversa. La ventaja de intercomunicar las GPUs que realizan los cálculos, es reducir el tiempo de ejcución del algorimtimo evitando procesos de lectura y escritura de datos.

Una vez descritas las estrategias de implementación del algoritmo ejecutado sobre GPUs, se muestran en la *Fase 4*, los resultados obtenidos y las respectivas medidas

de rendimiento realizadas, haciendo comparación entre las dos arquitecturas estudiadas.

3.4. FASE 4: ANÁLISIS Y COMPARACIÓN DE RESULTADOS OBTENIDOS

Para analizar los resultados obtenidos, se hace uso de métricas que ayuden a evaluar y comparar las diferentes arquitecturas.

La métrica más visible y que se registra con facilidad es el *tiempo de ejecución*, midiendo el tiempo que demora cada arquitectura en ejecutar la aplicación, con el cual se podría calcular también la *aceleración* o *speedup* [23].

La aplicación se ejecutó con diferente cantidad de datos para comprobar la hipótesis que cuantos más datos se calculen, la arquitectura mas rápida en tiempo de ejecución será la híbrida CPU-GPU, seguido de la CPU con múltiples cores y por último la CPU con un solo core.

La aceleración está definida formalmente, en la ecuación (26) como:

$$Speedup = \frac{T(1)}{T(N)} \tag{26}$$

Donde $T(1)$ es el tiempo de ejecución en serial y $T(N)$ es el tiempo de ejecución en paralelo.

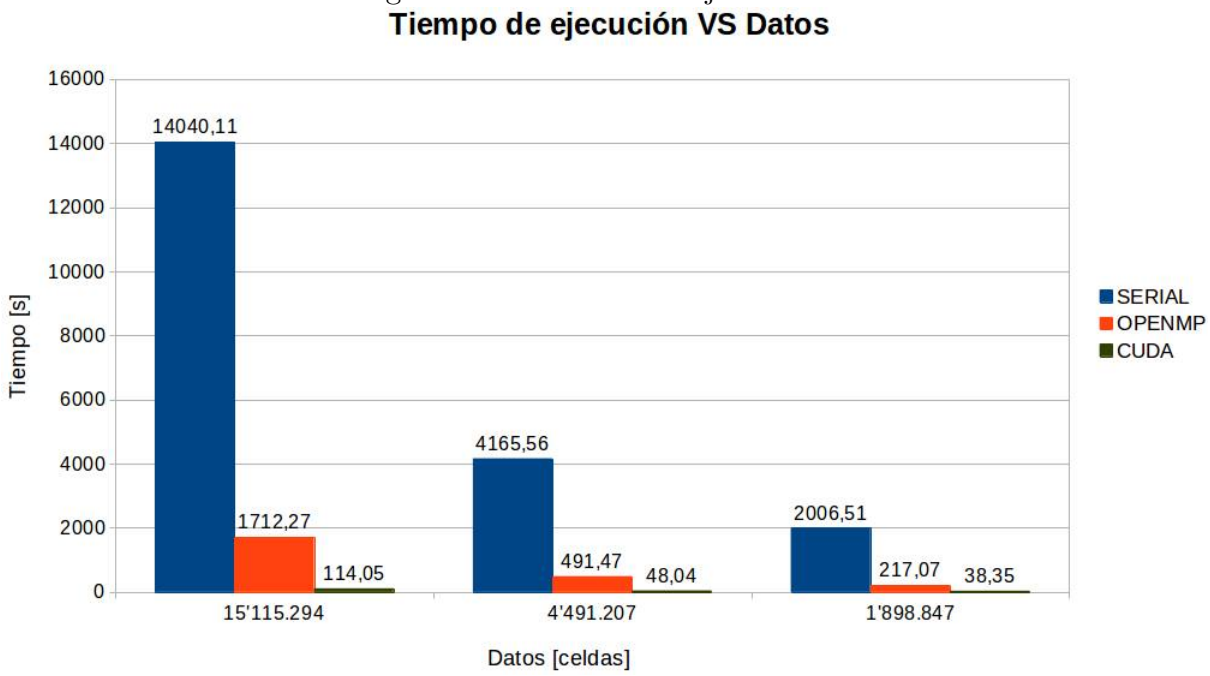
En el *Tabla 2*, se pueden ver los resultados de los tiempos de ejecución en las diferentes arquitecturas. Se tomó una muestra de seis tiempos, con 1.000 pasos de tiempo:

Tabla 2: Tiempos de ejecución en segundos.

Tamaño de Matriz	Tiempos de ejecución [s]	
201x201x47 1.898.847 celdas	Serial	2007.04 ± 2.82
	OpenMP	218.42 ± 3.87
	CUDA	38.4 ± 0.1
267x267x63 4.491.207 celdas	Serial	4152.73 ± 31.56
	OpenMP	491.48 ± 1.6
	CUDA	48.9 ± 0.33
401x401x94 15.115.294 celdas	Serial	14040.11 ± 578.93
	OpenMP	1712.01 ± 3.24
	CUDA	114.1 ± 0.46

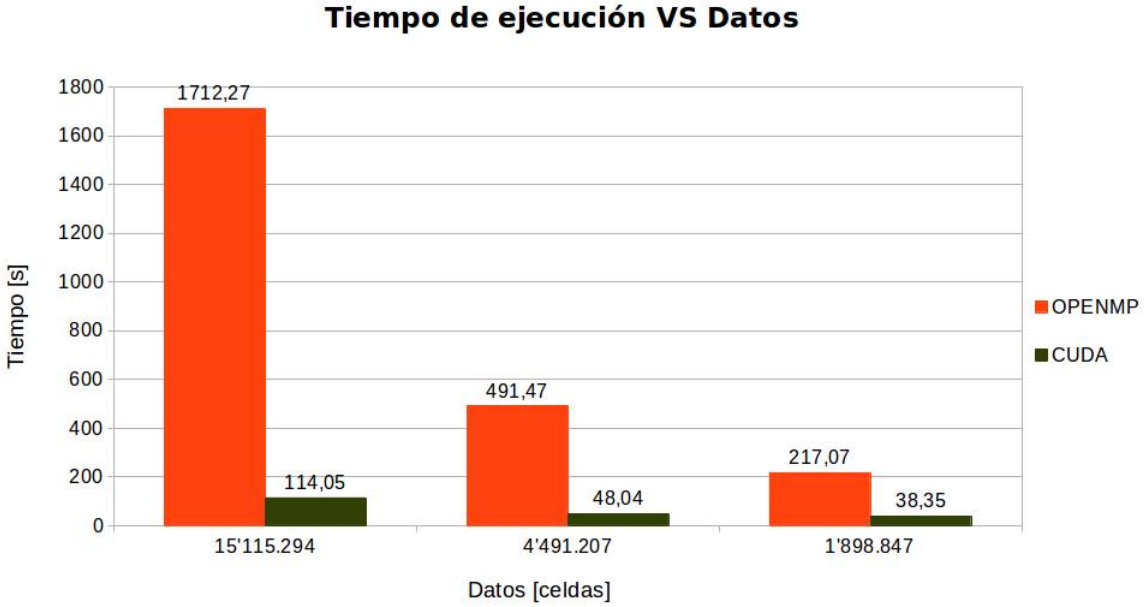
En la *Figura 23* y la *Figura 24*, se muestra la comparación de tiempos de ejecución de la aplicación en direrentes arquitecturas con diferente cantidad de datos calculados:

Figura 23: Velocidad de ejecución.



Fuente: Autores

Figura 24: Velocidad de ejecución: CUDA vs OMP.



Fuente: Autores

Se puede observar una disminución, en dos ordenes de magnitud, del tiempo ejecución de la aplicación acelerada con CUDA con respecto a la Serial, pero también se resalta el tiempo de ejecución que logró la aplicación acelerada con OpenMP con respecto a la serial. La razón por la que OpenMP también consiguió una mejora visible en un orden de magnitud, en el tiempo de ejecución se debe a la máquina en donde fueron ejecutadas las aplicaciones.

En el *cuadro 3* se muestra la aceleración que tiene la aplicación con OpenMP con respecto a la serial, aplicando la ecuación (26) como:

$$Speedup = \frac{T(1)}{T(24)}$$

Es considerable la aceleración de la aplicación con OpenMP se debe a que la máquina está utilizando sus 24 cores, un número de cores considerablemente alto si se compara con una computadora convencional. Por otra parte, cuando las dimensiones de la matríz con los datos sísmicos crece, aumenta la cantidad de cálculos hechos por el algoritmo y debido a esto la aceleración disminuye.

Tabla 3: Aceleración OpenMP vs Serial

Datos [celdas]	Aceleración
1898847	9.19
4491207	8.45
15115294	8.2

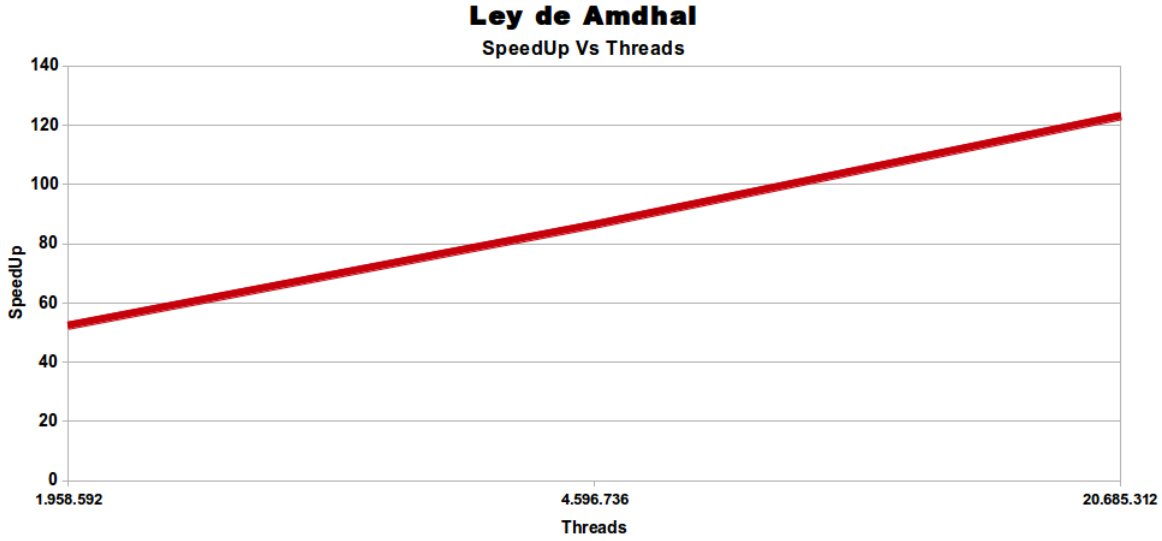
Tabla 4: Aceleración CUDA vs Serial

Datos [celdas]	Aceleración
1898847	52.27
4491207	86.36
15115294	123.05

Se puede observar que utilizando una GPU aumenta, un orden de magnitud más (y en el mejor caso, dos ordenes de magnitud) que con OpenMP, la aceleración de la aplicación CUDA con respecto a la serial.

En la *Figura 24* se observa el aumento de aceleración del la aplicación CUDA con respecto a la aplicación serial, mostrada en el *cuadro 4*. Se compara con respecto al número de hilos (*threads*) utilizados, haciendo referencia a un hilo como procesador para representar la Ley de Amdahl [23], que compara aceleración con número de procesadores utilizados. A medida que aumentan los datos, la aceleración de la aplicación CUDA aumenta de forma lineal.

Figura 25: Modificación de la Ley de Amdahl: Speedup (CUDA-Serial) Vs Threads.



Fuente: Autores.

Seguido a esto, en la *Fase 5*, se muestran las diferentes imágenes de las simulaciones realizadas con cada una de las arquitecturas.

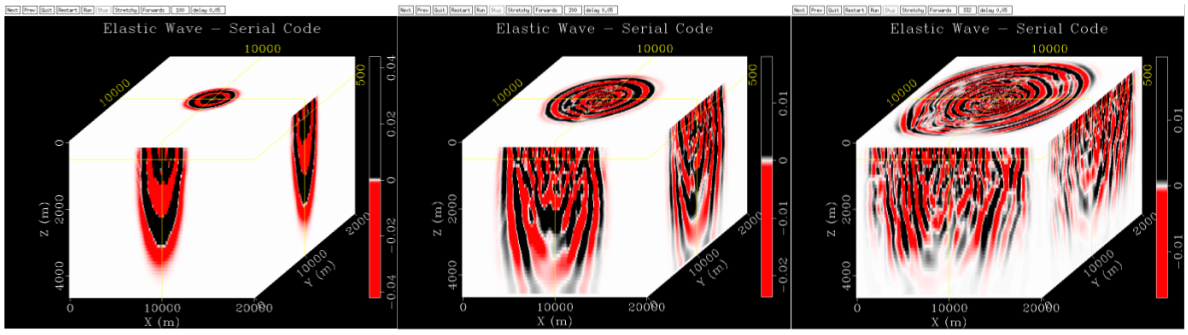
3.5. FASE 5: VISUALIZACIÓN DE LOS DATOS Y DESARROLLO DEL PROTOTIPO SOLUCIÓN

En esta fase se muestran las visualizaciones de los datos generados por los códigos programados:

Los 1.000 pasos de tiempo calculados en la ejecución de la aplicación, disminuyen en la visualización debido al tamaño del paso de tiempo y a la constante skipframe en el código, que representa que si el número de paso de tiempo actual es múltiplo del skipframe se guardán los datos para la visualización.

Código Serial con 1'898.847 datos y 333 pasos de tiempo.

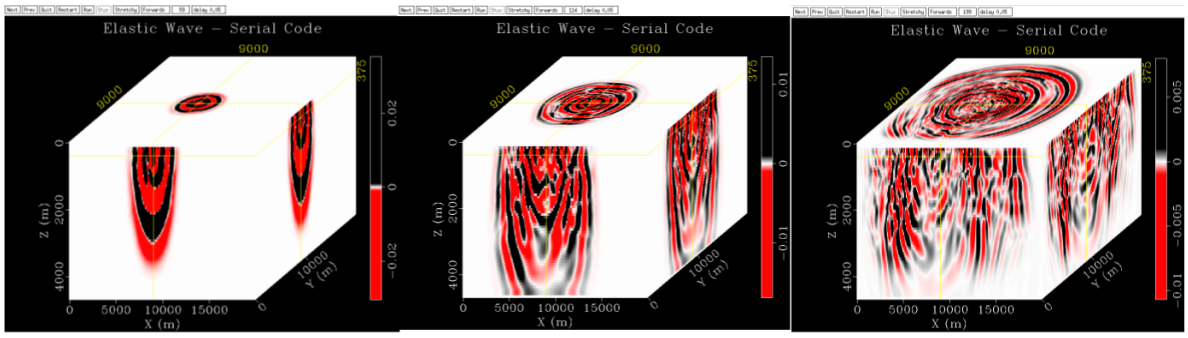
Figura 26: Visualización Código Serial, se muestran los pasos de tiempo 100, 200 y 333 respectivamente.



Fuente: Autores

Código Serial con 4'491.207 datos y 200 pasos de tiempo.

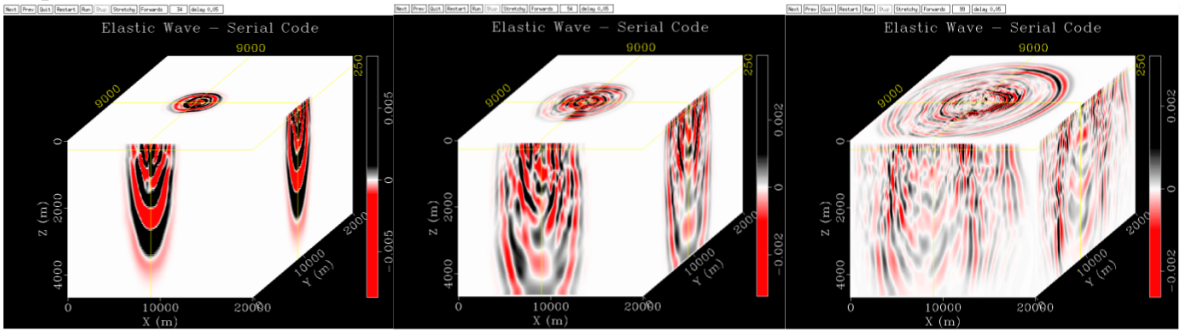
Figura 27: Visualización Código Serial, se muestran los pasos de tiempo 60, 115 y 200 respectivamente.



Fuente: Autores

Código Serial con 15'115.294 datos y 100 pasos de tiempo.

Figura 28: Visualización Código Serial, se muestran los pasos de tiempo 35, 55 y 100 respectivamente.



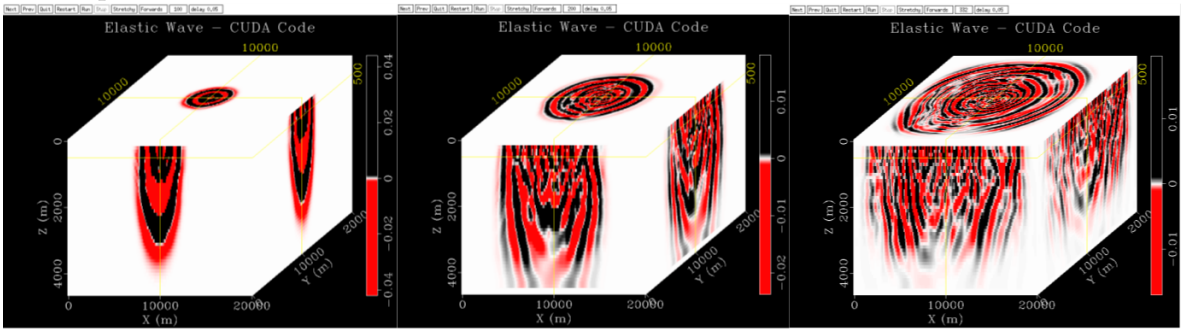
Fuente: Autores

Como se puede observar, la visualización con más datos, es la que tiene una mejor vista con respecto a las otras, puesto que se calculan más celdas y por lo tanto se grafican más puntos.

Ahora se muestran las visualizaciones de la aplicación con CUDA:

Código CUDA con 1'898.847 datos y 333 pasos de tiempo.

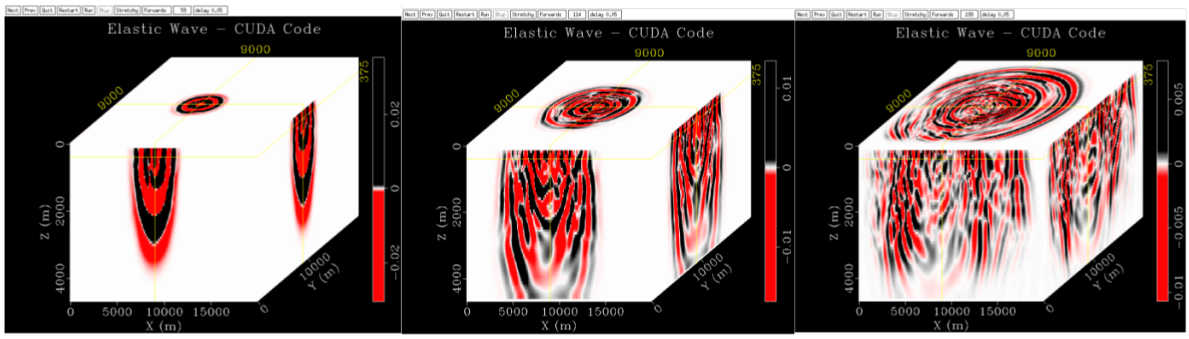
Figura 29: Visualización Código CUDA, se muestran los pasos de tiempo 100, 200 y 333 respectivamente.



Fuente: Autores

Código CUDA con 4'491.207 datos y 200 pasos de tiempo.

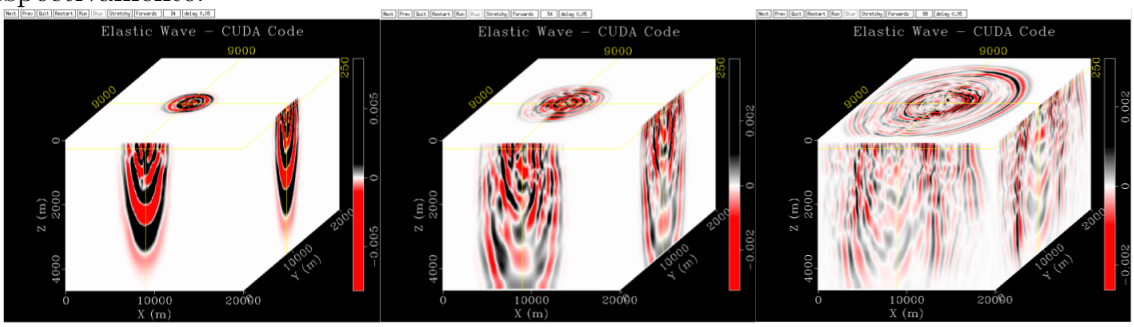
Figura 30: Visualización Código CUDA, se muestran los pasos de tiempo 60, 115 y 200 respectivamente.



Fuente: Autores

Código CUDA con 15'115.294 datos y 100 pasos de tiempo.

Figura 31: Visualización Código CUDA, se muestran los pasos de tiempo 35, 55 y 100 respectivamente.



Fuente: Autores

Una vez obtenidos los resultados de la simulación realizada, se muestra en la *Fase 6*, la validación que se hizo para corroborar que la simulación obtenida con el algoritmo ejecutado sobre GPU, corresponde a los datos reales. Además, se comprueba que el resultado de la ejecución en serial y en paralelo coinciden.

3.6. FASE 6: VALIDACIÓN DE LA SOLUCIÓN PLANTEADA

Cuando se habla de validación, se tienen en cuenta los alcances del proyecto. Como se vio anteriormente, se resuelve una onda elástica en 3D en un medio isótropo y heterogéneo usando arquitecturas híbridas. Para desarrollar la evaluación del desarrollo, se comparó el código serial, como modelo de referencia, con los conocimientos adquiridos sobre el área del proyecto, se estudió el modelo para identificar las partes que se podían paralelizar. Seguido a esto, después de paralelizar, se calculó un número de pasos de tiempo ideal para hacer las pruebas con los diferentes códigos.

Analizando las visualizaciones del código serial y el código en paralelo concuerdan con respecto a la vista, al paso de tiempo capturado y a la barra de valores lateral. Además, Madagascar puede realizar restas de archivos rsf, restando elemento por elemento de los archivos, la resta se utilizó para comparar los resultados del código serial y el código en paralelo, el resultado de esta operación, fué un archivo rsf que posteriormente se visualizó, mostrando un cubo sísmico en blanco, lo que quiere decir que la resta fue igual a cero en todos sus componentes. El uso del método FDTD da una mayor precisión a los resultados obtenidos que el de diferencias finitas. Con respecto a la arquitectura incorporada, utilizando CPU-GPU se obtienen iguales resultados para todas las ejecuciones con respecto al código serial ejecutado en una CPU.

También se hizo una prueba de sensibilidad, aumentando y disminuyendo la cantidad de datos calculados, obteniendo siempre los mismos resultados al modelo serial, con variaciones en las visualizaciones.

4. CONCLUSIONES

A lo largo del presente documento, se describió todo el proceso de estudio y recolección de información sobre el área de sísmica que contempla el proyecto, se estudió la ecuación diferencial de onda elástica que describe analíticamente el fenómeno específico y se introduce el concepto y la forma de abarcar una solución de dicha ecuación para lograr ver su comportamiento.

Se modeló la solución de la ecuación diferencial de la onda elástica en un medio isótropo y heterogéneo con la técnica FDTD con segundo orden de precisión en el tiempo y cuarto orden de presición en el espacio, ejecutado sobre una arquitectura híbrida.

El método de discretización utilizado es uno de los más precisos que se pudo utilizar y calcularlo en cuarto orden fue suficiente para el análisis del proyecto, pues se manejaron números flotantes y no dobles.

Se evaluó el nivel de paralelización del algoritmo dada la naturaleza de los datos y se implementa CUDA sobre el código para ejecutar sobre una GPU. En esta parte del proceso, se logra ver claramente el alto desempeño del uso de la GPU para tratar problemas que manejan una gran cantidad de datos, como en éste caso. Se observó una notable aceleración en la aplicación implementada con CUDA y se logra observar la gran utilidad de esta tecnología aplicada. También se resalta la comparación que se hizo de ejecutar la aplicación con OpenMP, pues se dió por hecho que utilizando 24 cores de GUANE-1, contra una arquitectura híbrida, utilizando tan solo una CPU y una GPU , es más potente en cuestion de rendimiento las arquitecturas hibridas. Hay que tener en cuenta que la arquitectura de GUANE-1 no es una tecnología actual, por lo tanto, se concluye que ejecutando la aplicación en una tecnología reciente, el rendimiento sería más alto, debido a que posee una memoria global de más capacidad y posee una arquitectura de comunicación más rápida, lo que solucionaría el problema de comunicación que se tiene en la implementación con mas de una GPU.

Con la Ley de Amdahl modificada, en donde se analiza el SpeedUP vs Threads, tomando cada thread como un proceador, se pudo ver un considerable aumento de la aceleración al aumentar los threads, por lo que se afirma que si se ejecuta el código en una tecnología actual, ésta dispone de una mayor cantidad de threads para calcular, por lo que sería mayor el rendimiento.

Los resultados obtenidos y las mediciones realizadas, con base en las comparaciones de las soluciones ya existentes estudiadas, muestran que el uso de GPU demuestran un mejor desempeño y unos resultados precisos de la aplicación acelerada con CUDA con respecto a la serial, esto se concluye de la *Fase 6: Validación de la solución planteada*

en la *sección 3* donde se restan las dos soluciones y se visualizan, obteniendo como resultado un cubo en 3D una visualización en blanco, dado que la resta de los vectores es igual a cero en todos los elementos, lo que significa que los datos generados por la aplicación en paralelo coinciden con los resultados de la aplicación en serial.

Se analizó que con la cuarta parte, o menos de los datos del modelo de velocidad calculados, la visualización se observaría muy pixelada, y se llegó al punto máximo de llegar a trabajar con la mitad de los datos del campo de velocidad haciendo uso de los softwares ya nombrados. Si se toman todos los datos del modelo de velocidad, no es posible visualizarlos con Madagascar debido a la cantidad de información.

Se concluye que con el equipo de trabajo y las herramientas software utilizadas, se logró cumplir con los objetivo del proyecto, pero para satisfacer las necesidades de la industria, se hace necesario incluir un software dedicado a la visualización, para poder aumentar la cantidad de datos visualizados, lo que es fundamental para hacer un buen análisis del campo estudiado por los expertos. Finalmente se confirma que las arquitecturas híbridas obtienen un mayor desempeño y son de mucha ayuda para la industria que tendría más posibilidades de analizar el campo debido al tiempo al que se disminuye la obtención de los resultados, por lo que podrían tambien desarrollar grandes proyectos en un periodo más corto.

5. RECOMENDACIONES

Las recomendaciones que se pueden hacer para seguir mejorando este proyecto son las siguientes:

1. Al aumentar el número de datos calculados, se presenta un error al gráficar toda la información, el error lo produce el visualizador de Madagascar, por lo que se recomienda un módulo en Madagascar como visualizador para grandes cantidades de datos.

2. Para aumentar aun más el rendimiento de la aplicación, se podría incorporar:

   - El uso de mas GPUs: Es importante para problemas con mayor tamaño de datos, donde se requiera particionar el dominio para lograr un mejor rendimiento.
   - El uso de memoria compartida: Para obtener más rapidez en el acceso de los datos, es una memoria más rápida que la memoria global utilizada en este proyecto y requiere una implementación distinta.
   - Definir una estrategia para disminuir los cuellos de botella GPU-CPU.
   - El uso de Streams: Definir el tamaño de datos más eficiente para enviar cálculos mediante streams.
   - El uso de la memoria de constantes y de memoria de textura: Implementar el uso de estas memorias, correspondientes a cachés de solo lectura. Así se utilizaría la mayoría de la jerarquía de memoria que posee una GPU.

3. Confrontar con otros software de modelado sísmico para comparar el manejo de datos y las visualizaciones. Se utilizó Madagascar debido a que es utilizado en el campo académico y que es un software libre, además la API ofrece un entorno de programación cómodo.

CITAS

[1] TELFORD, W. M., L. P. Gelart, R. E. Sheriff and D. A. Keys «Applied Geophysics», *Operations Research.Cambridge: Cambridge University Press.* 1976.

[2] AKI, K. and P. G. Richards «Quantitative Seismology.», *New York: W. H. Freeman and Co.* 1985.

[3] CLARERBOUT, J. «Imaging the Earth's Interior.», *Oxford: Blackwell Scientific Publications.* 1985.

[4] BRAILE, L. «What Are Seismic Waves?», *UP Seis, an educational site for budding seismologists.* [Enero 2015] Dispodible en la Web:
http://www.geo.mtu.edu/UPSeis/waves.html

[5] LANDAU, L. D. and E. M. Lifshitz «Theory of Elasticity.», *Oxford: Pergamon Press.* 1986.

[6] Intergeo «Frentes de onda», *Intergeo.* [Enero 2015] Dispodible en la Web:
http://inter-geo.org/Study/Seismic/Waves/Wave-Fronts.php?lang=es

[7] e-ducativa «Tema 2: Movimiento Ondulatorio», *E-DUCATIVA CATEDU.* [Enero 2015] Dispodible en la Web:
http://e-ducativa.catedu.es/

[8] GONZÁLEZ, H «Modelado Sísmico. Teoría de Propagación de Ondas.», *Seminario de Modelado Sísmico. Grupo de Geofísica. Instituto Colombiano de Petróleo*, págs.6-7. 2014.

[9] DORRONOSO, C. «Isotropic and anisotropic minerals », *Departamento de Edafología de la Facultad de Ciencias* [Enero 2015]. Dispodible en la Web:
http://edafologia.ugr.es/optmine/intro/isoaniw.htm

[10] GÁLVEZ, P «Ondas (P y S) (R y L)», *Mediciones Sismicas e Investigación.* [Enero 2015] Disponible en la Web:
http://geofisicasismospgf.blogspot.com/p/ondas-p-y-ondas-s.html

[11] SLAWINSKI, M. A. «Waves and Rays in Elastic Media.», *San Francisco, California: Creative Commons.* págs.118–119. 2007.

[12] Nvidia «Qué es el GPU Computing», [en línea],
[Enero 2015]. Dispodible en la Web:
http://www.nvidia.es/object/gpu-computing-es.html

[13] YEE, K. «Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media.», *IEEE Trans. Antennas Propagat.*,**14** págs.302–307. 1966.

[14] TAFLOVE, A. and M. E. Brodwin «Numerical solution of steady-state electromagnetic scattering problems using the time-dependent maxwell's equations.», *IEEE Trans. Microwave Theory and Techniques*,**23** págs.623–630. 1975.

[15] GONZÁLEZ, H «Modelado Sísmico. Ecuación de Onda Elástica 2D/3D y Medios Anisóropos.», *Seminario de Modelado Sísmico. Grupo de Geofísica. Instituto Colombiano de Petróleo*, págs.20-27. 2014.

[16] LINES, L. R., R. Slawinski and P. Bording «A recipe for stability analysis of finite-difference wave-equation computations.», *Geophysics*,**64** págs.967–969. 1999.

[17] Ahay «Madagascar», [en línea], Octubre 2008, Abril 2014, [Enero 2015]. Dispodible en la Web: http://www.ahay.org/wiki/Main_Page

[18] FOMEL, S. and G. Hennenfent «Reproducible computational experiments using scons.», *In 32nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP),* pp. 1257–1260. IEEE. 2007

[19] OpenMP «OpenMP Architecture Review Board», [en línea], 2014 [Enero 2015]. Dispodible en la Web: http://openmp.org/wp/about-openmp/

[20] SEG/EAGE EAGE, SEG «SEG/EAGE 3D Overthrust and Salt Models» [Enero 2015] Disponible en la web: http://geodus1.ta.tudelft.nl/seage3dm/

[21] CERJAN, C., D. Kosloft, R. Kosloff and M. Resheq «A nonreflecting boundary condition for discrete acoustic and elastic wave equations.», *Geophysics,* **50** pags. 1257–1260. 1985.

[22] BROCHER, T. M. «Empirical relations between elastic wavespeeds and density in the earth's crust.», *Bulletin of the Seismological Society of America*,**95** págs.2081–2092. 2005.

[23] THIÉBAUT, D. «MEASURING PERFORMANCE », *Parallel Programming in C for the Transputer*, Chapter 8. 1995.

BIBLIOGRAFÍA

AHAY «Madagascar», [en línea], Octubre 2008, Abril 2014, [Enero 2015]. Dispodible en la Web: http://www.ahay.org/wiki/Main_Page

AKI, K. and P. G. Richards «Quantitative Seismology.», *New York: W. H. Freeman and Co.* 1985.

BROCHER, T. M. «Empirical relations between elastic wavespeeds and density in the earth's crust.», *Bulletin of the Seismological Society of America*,**95** págs.2081–2092. 2005.

CERJAN, C., D. Kosloft, R. Kosloff and M. Resheq «A nonreflecting boundary condition for discrete acoustic and elastic wave equations.», *Geophysics,* **50** pags. 1257–1260. 1985.

CLARERBOUT, J. «Imaging the Earth's Interior.», *Oxford: Blackwell Scientific Publications.* 1985.

FOMEL, S. and G. Hennenfent «Reproducible computational experiments using scons.», *In 32nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP),* pp. 1257–1260. IEEE. 2007

GONZÁLEZ, H «Modelado Sísmico. Ecuación de Onda Elástica 2D/3D y Medios Anisóropos.», *Seminario de Modelado Sísmico. Grupo de Geofísica. Instituto Colombiano de Petróleo*, págs.20-27. 2014.

GONZÁLEZ, H «Modelado Sísmico. Teoría de Propagación de Ondas.», *Seminario de Modelado Sísmico. Grupo de Geofísica. Instituto Colombiano de Petróleo*, págs.6-7. 2014.

LANDAU, L. D. and E. M. Lifshitz «Theory of Elasticity.», *Oxford: Pergamon Press.* 1986.

LINES, L. R., R. Slawinski and P. Bording «A recipe for stability analysis of finite-difference wave-equation computations.», *Geophysics*,**64** págs.967–969. 1999.

Nvidia «Qué es el GPU Computing», [en línea], [Enero 2015]. Dispodible en la Web: http://www.nvidia.es/object/gpu-computing-es.html

TELFORD, W. M., L. P. Gelart, R. E. Sheriff and D. A. Keys «Applied Geophysics», *Operations Research.Cambridge: Cambridge University Press.* 1976.

THIÉBAUT, D. «MEASURING PERFORMANCE », *Parallel Programming in C for the Transputer*, Chapter 8. 1995.