UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"PARALELIZACIÓN DE LA CONSTRUCCIÓN DEL JACOBIANO EN UN SIMULADOR NUMÉRICO DE YACIMIENTOS DE HIDROCARBUROS"

T E S I S

QUE PARA OBTENER EL TÍTULO DE

I N G E N I E R O P E T R O L E R O

P R E S E N T A :

LUIS FELIPE ALCÁNTARA GARCÉS

DIRECTOR DE TESIS:

DR. VÍCTOR HUGO ARANA ORTIZ

MÉXICO, D.F. **SEPTIEMBRE, 2006**

VNIVERSIDAD NACIONAL AVFONOMA DE MEXICO

FACULTAD DE INGENIERÍA
DIRECCIÓN
60-I-738

SR. LUIS FELIPE ALCÁNTARA GARCÉS
P r e s e n t e

En atención a su solicitud, me es grato hacer de su conocimiento el tema que propuso el profesor Dr. Víctor Hugo Arana Ortiz y que aprobó esta Dirección para que lo desarrolle usted como tesis de su examen profesional de Ingeniero Petrolero:

PARALELIZACIÓN DE LA CONSTRUCCIÓN DEL JACOBIANO EN UN SIMULADOR NUMÉRICO DE YACIMIENTOS DE HIDROCARBUROS

Ruego a usted cumplir con la disposición de la Dirección General de la Administración Escolar en el sentido de que se imprima en lugar visible de cada ejemplar de la tesis el título de ésta.

Asimismo, le recuerdo que la Ley de Profesiones estipula que se deberá prestar servicio social durante un tiempo mínimo de seis meses como requisito para sustentar examen profesional.

A t e n t a m e n t e
"POR MI RAZA HABLARÁ EL ESPÍRITU"
Cd. Universitaria, D. F., a 30 de junio de 2006
EL DIRECTOR

M. en C. GERARDO FERRANDO BRAVO
GFB*IAGC*gtg

VNIVERSIDAD NACIONAL AVFNOMA DE MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA EN CIENCIAS DE LA TIERRA

"PARALELIZACIÓN DE LA CONSTRUCCIÓN DEL JACOBIANO EN UN SIMULADOR NUMÉRICO DE YACIMIENTOS DE HIDROCARBUROS"

TESIS PRESENTADA POR :

LUIS FELIPE ALCÁNTARA GARCÉS

DIRIGIDA POR:

DR. VÍCTOR HUGO ARANA ORTIZ.

JURADO DEL EXÁMEN PROFESIONAL:

PRESIDENTE: **ING. MANUEL VILLAMAR VIGUERAS.**

VOCAL: **DR. VÍCTOR HUGO ARANA ORTIZ.**

SECRETARIO: **M. I. MARIO BECERRA ZEPEDA.**

1ER SUPLENTE: **ING. MARTÍN C. VELÁZQUEZ FRANCO.**

2DO SUPLENTE: **M. I. RUTH V. WILSON LÓPEZ.**

AGRADECIMIENTOS.

Agradezco al Grupo de Simulación Numérica de Yacimientos de la Facultad de Ingeniería de la UNAM por todo el apoyo y facilidades prestadas para la realización y finalización de este trabajo.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

Con el avance continuo en la velocidad de los procesadores y la refinación en la caracterización de un yacimiento petrolero, es factible incrementar el número de celdas numéricas para discretizar el dominio activo de un yacimiento de hidrocarburos.

El uso de la programación en paralelo, a diferencia de la secuencial, consiste en utilizar varios procesadores y repartir las tareas de algún algoritmo de cálculo, de tal manera que se reducen los tiempos de cálculo. Los algoritmos que componen a un simulador numérico de yacimientos pueden ser programados de tal manera que se optimiza la velocidad de cómputo y por lo tanto se mejora considerablemente los tiempos de simulación. El objetivo de esta tesis es aplicar la programación en paralelo a un simulador semi-implícito de aceite negro monofásico en tres dimensiones para conocer la factibilidad del uso de esta técnica y reducir los tiempos de simulación. La parte del procedimiento que se paraleliza en este trabajo es la construcción de Jacobiano debido a que es la segunda instancia que consume el mayor tiempo en un paso de tiempo. Se presentan dos procedimientos para paralelizar la construcción del Jacobiano, los tiempos se comparan con el caso secuencial.

Los resultados muestran que para el tipo de simulador y la implicitud utilizada, la reducción del tiempo de cómputo acumulado es aproximadamente de 1/10 (para un incremento de tiempo).

Esta tesis está estructurada de la siguiente manera: primero se presenta una breve introducción, seguida del planteamiento del problema a resolver y posteriormente se presentan conceptos básicos de Ingeniería de Yacimientos. Antes de presentar la paralelización del simulador, en el capítulo cinco, se presenta la filosofía y algunos conceptos básicos de la programación en paralelo. En el capítulo seis se analizan los resultados obtenidos y por último se presentan las conclusiones y recomendaciones.

CAPÍTULO 1. INTRODUCCIÓN

El incremento en el desarrollo de computadores en paralelo ha vencido las restricciones de tamaño y resolución para la simulación de yacimientos que se tiene con el uso de computadoras secuenciales. El número de bloques que se utiliza en una simulación numérica ha pasado de miles a millones. Lo anterior es debido al uso de computadoras con multiprocesadores (clusters).

Un simulador numérico trabaja dividiendo al yacmiento en bloques y realizando un balance de masa entre los bloques que interactuan. **Coats (1969)** define un simulador numérico como un conjunto de ecuaciones diferenciales parciales que expresan la conservación de masa y/o energía.

La Simulación Numérica de Yacimientos (SNY) combina la física, matemáticas e ingeniería de yacimientos para obtener algoritmos que deben ser programados para desarrollar una herramienta confiable capaz de predecir el comportamiento de los yacimientos de hidrocarburos bajo diferentes condiciones de explotación. El objetivo de la simulación de yacimientos es proporcionar al ingeniero de diseño de explotación una herramienta confiable para predecir el comportamiento de los yacimientos de hidrocarburos bajo diferentes condiciones de operación. El modelar el comportamiento de un yacimiento de hidrocarburos bajo diferentes esquemas de producción reduce el riesgo asociado a la elección del plan de explotación y por lo tanto minimiza los flujos de efectivo negativos.

Matax y Dalton (1990) mencionan que la simulación es la única forma de describir cuantitativamente el flujo de fluidos en un yacimiento heterogéneo, cuya producción se determina no solamente por las propiedades del yacimiento, sino también por la demanda del mercado, las estrategias de inversión y las políticas gubernamentales.

Los simuladores de yacimientos son usados principalmente porque son capaces de resolver problemas que no pueden ser resueltos analíticamente o de alguna otra manera.

Durante toda su vida la industria del petróleo se ha visto estrechamente ligada con la tecnología obligando a ésta a siempre estar innovando para cumplir con los grandes requerimientos que la industria presenta. Dentro del campo de la computación y la informática no ha sido la excepción, por ejemplo desarrollando computadoras poderosas capaces de simular el comportamiento de los yacimientos. Sin embargo como es de esperarse siempre surgen necesidades más complejas y dentro de la simulación numérica de yacimientos se presenta el de representar de la manera más aproximada que se pueda al yacimiento y sus fluidos, lo que se traduce en que los requerimientos computacionales cada día crecen más y más. Estos requerimientos se han logrado satisfacer pero a un costo muy alto, por ejemplo desde el ámbito económico directo, pues las supercomputadoras son extremadamente caras y no siempre cumplen con las expectativas de los ingenieros especializados en la simulación y, cuándo sí cumplen con estas expectativas, ocurre que el tiempo que emplean en la resolución de los problemas es demasiado lo cual puede provocar atrasos en el desarrollo de los campos petroleros, si se espera a que la simulación este completa, o una mala explotación de ellos.

Sin embargo, como esto no es suficiente, los expertos en simulación de yacimientos han volteado la mirada hacia las computadoras multiprocesador así como también han recurrido a desarrollar mejores algoritmos de solución tales como la Discretización del Dominio (DD) y la Programación en Paralelo (PP), que esta última es una excelente alternativa pues una computadora de este estilo es mucho más barata que una supercomputadora y, con la ayuda de estos nuevos algoritmos, los tiempos de solución y capacidad de las máquinas se ven reducidos en gran cantidad e incrementada respectivamente, acarreando así dos beneficios de primera mano: ahorro de dinero en la computadora y ahorro de tiempo con la manera de operar del equipo.

González-Escribano, A. y otros (2006) establecen que para entender lo referente a la programación en paralelo, hay que conocer cuado menos las arquitecturas de las máquinas paralelas así como los modelos de programación que pueden ser utilizados en cada una de ellas.

Una computadora paralela es un sistema de cómputo multiprocesador que soporta la programación en paralelo. Hay dos categorías importantes de computadoras paralelas: multicomputadoras y multiprocesadores. Una multicomputadora es una máquina paralela construida con computadoras conectadas con una red de trabajo de interconexión. Los procesadores de las computadoras interactúan por medio del envío de mensajes de uno a otro. Esta arquitectura también es llamada cluster de computadoras o sistema de memoria distribuida. Un cluster de computadoras puede construirse usando computadoras comunes pero para lograr una buena eficiencia se tiene que emplear una red de trabajo de alta velocidad. En contraste, un multiprocesador es un sistema altamente integrado en el cual todos los CPU's comparten acceso a una sola memoria global. La memoria compartida brinda comunicación y sincronización en los procesadores. Esta arquitectura también es llamada multiprocesador simétrico o sistemas de memoria compartida **(González-Escribano, A. y otros, 2006)**. Estas definiciones se desarrollan más a fondo adelante en el trabajo**.**

Las diferentes arquitecturas sugieren diferentes modelos de programación. Algunos de ellos están enfocados en la eficiente explotación de la arquitectura subyacente, sacrificando la programación sin dificultad y reutilizable. Mientras que otros son más orientados al desarrollo de software paralelo sistemático

La paralelización no es una tarea sencilla, la idea de ejecutar de manera paralela un algoritmo secuencial siempre es atractiva, sin embargo no es fácil escoger la mejor alternativa y depende en gran medida de la arquitectura de la máquina de trabajo. Cada arquitectura tiene sus ventajas y desventajas y no es fácil desarrollar implementaciones paralelas que hagan un buen uso de lo anterior mitigando el efecto en lo siguiente.

El entender la programación en paralelo y desarrollar un programa bajo este esquema involucra la apertura del programador a nuevas ideas de estructura, pues si hacemos una analogía con la forma en como desempeñamos tareas comunes como solucionar cinco ejercicios de cálculo en los que en cada uno emplearemos 20 minutos y no nos importa como se solucionaron sino únicamente el resultado, en total serían 100 minutos de trabajo

para tener los 5 resultados. Pero si en lugar de ser sólo una persona la que soluciona los ejercicios son cinco quienes obtienen los resultados, uno por cada ejercicio, al mismo tiempo y al final sólo comparten el resultado, el tiempo que se emplea es sólo de 20 minutos y se ha obtenido lo que se desea; en otras palabras, se han realizado las mismas tareas pero en menor tiempo; es así entonces como la programación en paralelo consiste en poner a trabajar varios procesadores para que compartan el trabajo y terminen el proceso en menos tiempo; cada procesador toma la información que se le asigna, trabaja con ella y después la comparte con los otros, esto se repite hasta que llegan al resultado final. Esta es la idea que se debe tener cuando se trabaja con algoritmos paralelos y, entonces, el trabajo consiste en que el programador debe definir qué parte del trabajo se puede paralelizar y cómo hacerlo. También se debe tomar en cuenta de que la comunicación de datos entre los procesadores ocupa tiempo y memoria por lo que es un parámetro que se tiene que optimizar de la mejor manera posible ya que puede darse el caso en el que exista un número exagerado de intercambios de información, dando como resultado que el tiempo que se emplea es igual o mayor al que se ocuparía con un comportamiento secuencial.

Es conocido que los tiempos empleados en la solución de problemas por los simuladores es grande, entonces el trabajar con la PP es un camino para disminuir este tiempo, que para la industria del petróleo es muy importante, sin sacrificar los resultados finales de la simulación; pudiendo ser inclusive mejores que los que se obtienen con una simulación bajo algoritmo secuencial, ya que cuándo la cantidad de información numérica es muy detallada las computadoras tienen una capacidad determinada por lo que utilizan truncamiento o redondeo acarreando errores de precisión numérica. Al trabajar bajo una arquitectura paralela estos truncamientos o redondeos se ven disminuidos pues cada procesador trabaja con sólo una parte de la información que es menor que el total de ella.

No podemos quedarnos con la idea de que la programación en paralelo sólo sirve para el disminuir tiempo de las aplicaciones, aunque esta es una de las principales ventajas que presenta, también es utilizada en aquellos casos en los que las aplicaciones computacionales son demasiado grandes y no pueden ser soportadas por una computadora

sin importar que sea de las conocidas como supercomputadoras; en estos casos la programación en paralelo es la gran alternativa de solución.

Dogru (2000) menciona en su trabajo que la velocidad en el proceso de simulación está estrechamente ligada y prácticamente está regida por el número de celdas con las que se trabaja en la malla de simulación. Cuándo el número es pequeño la simulación es rápida y entre más aumenta el número de celdas es mayor el tiempo empleado, así como la cantidad de información que se maneja. Como es sabido, entre mayor sea el número de celdas en la malla es mejor la simulación y mejores los resultados. En estos tiempos hablar de un millón de celdas es muy normal pero a pesar de ello muchas de las máquinas utilizadas no son capaces de desarrollar eficientemente el trabajo. Cuando se trabaja con programación en paralelo este problema se ve resuelto, pues como se ha dicho, no solo sirve para acelerar sino para soportar trabajos muy grandes.

En concreto, la programación en paralelo ofrece ventajas como: reducir la carga de trabajo de las máquinas al distribuirlas en varias en lugar de una sola, optimizar el tiempo de ejecución de programas robustos, puede reducir la complejidad de algunos programas, optimizar los recursos de cómputo, alcanzando e incluso superando el rendimiento de las supercomputadoras con máquinas paralelas más económicas; entre otras mas.

El tiempo que emplea un simulador en llevar a cabo el proceso de simulación está dividido como lo muestran **Dogru y otros** en su trabajo, en donde la mayor parte del tiempo empleado ocurre en el proceso de solución del Jacobiano por los distintos métodos que se han desarrollado para esta tarea, en segundo lugar tenemos a la formación del Jacobiano que es donde se reúne la información del comportamiento del yacimiento y sus fluidos, se continua con la revisión de tablas que puede involucrar interpolaciones, extrapolaciones o simplemente tablas de correspondencia, todo esto hecho por la misma computadora; lo que sigue en la cantidad de tiempo empleado es la lectura y procesamiento de la información referente a los pozos que son colocados y por último se engloba el tiempo de otros procesos en un solo apartado.

A diferencia de otros trabajos donde la paralelización se lleva acabo en la solución del Jacobiano, el objetivo de este trabajo es desarrollar la paralelización de la etapa que consume más tiempo después de la solución del Jacobiano en el proceso de simulación, la creación del Jacobiano. Para la formulación de este trabajo, el jacobiano es una matriz héptadiagonal en donde cada elemento es un escalar, debido a las características del simulador empleado: aceite negro monofásico en tres dimensiones; sin embargo cuándo se utiliza un simulador más complejo los elementos de la matriz dejan de ser escalares y se convierten en arreglos matriciales. Se proponen distintos tamaños de la malla de simulación desde miles hasta cientos de miles de celdas, con el fin de apreciar mejor el desempeño del trabajo propuesto. Se utilizó un cluster de memoria distribuida y los resultados son comparados con los que arroja un simulador desarrollado de manera secuencial que ya ha sido probado y validado. El simulador utilizado es un modelo monofásico para yacimientos naturalmente fracturados. Para validar el correcto funcionamiento del sistema de doble porosidad se utiliza el estudio que realizan **Warren y Root (1963)**. El trabajo esta organizado de la siguiente manera, primero se presenta una breve introducción seguida por la definición y el análisis del problema propuesto, continua con conceptos de ingeniería de yacimientos y de programación en paralelo, para así continuar con el capítulo cinco dedicado a la paralelización del simulador y por último se presentan las conclusiones obtenidas del análisis de resultados presentado en el capítulo seis.

CAPÍTULO 2: DEFINICIÓN Y ANÁLISIS DEL PROBLEMA.

Para trabajar con un simulador numérico es importante que sepamos qué es un simulador y como funciona. Del mismo modo, y con la misma importancia, antes de utilizar cualquier método para la solución de un problema lo primero que debemos realizar es entender el problema que deseamos solucionar; es por esto que en este capítulo se describe el tema de discusión de esta tesis, para que ya comprendido el problema se pueda proseguir con el desarrollo de la solución.

El desarrollo de la tesis se enfoca en la paralelización de la construcción del jacobiano de un simulador numérico para yacimientos fracturados, por lo que es necesario conocer qué es un yacimiento fracturado desde el punto de vista de la simulación, cómo funcionan los simuladores para yacimientos de este tipo, que es la paralelización de un simulador y finalmente debemos conjuntar esa información para entender que es lo que se realizará.

2.1 Yacimientos Naturalmente Fracturados.

Los yacimientos fracturados se pueden idealizar, con el fin de estudiarlos de la mejor manera, haciendo pequeños bloques que representan la matriz de la roca separados por un pequeño espacio entre sí, que idealiza la fractura en las rocas.

El modelo idealizado de la **figura 2.1** es el modelo más general para simular dichos yacimientos, es conocido como modelo de doble porosidad en donde un sistema de fracturas esta superpuesto a un sistema de matriz e interactúan simultáneamente, como lo proponen **Barenblatt y Zheltov (1960)**; que después **Warren y Root (1963)** lo toman como base para presentar un sistema continuo de fracturas ortogonales mientras que el sistema de matriz es representado por bloques rectangulares idénticos, superpuestos de la misma manera. En un modelo de doble porosidad el fluido se almacena en los bloques de la matriz mientras que las fracturas son las encargadas de conducir este fluido hacia los pozos productores. En la **figura 2.2** se muestra la representación de este fenómeno.

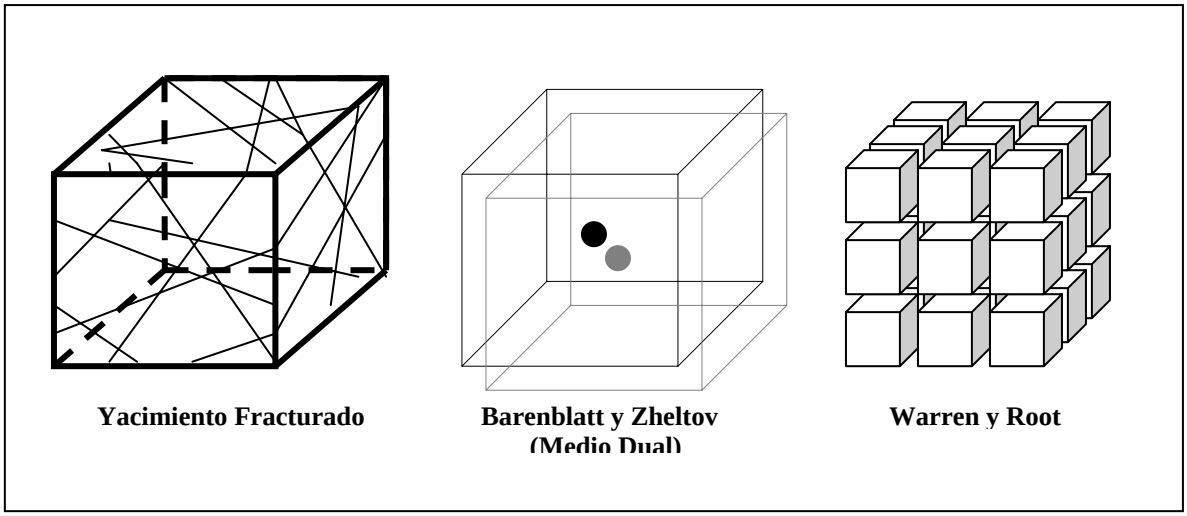


Figura 2.1. Idealización de un Yacimientos Fracturado **(Henn, N., y otros, 2004).**

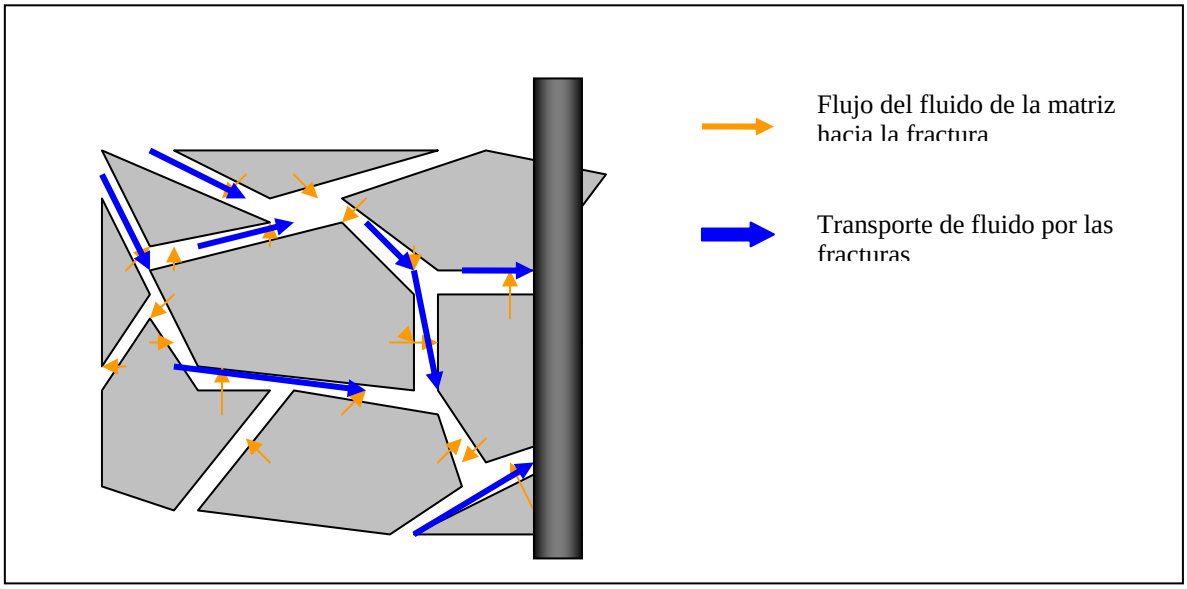


Figura 2.2. Desplazamiento del fluido hacia los pozos**.**

2.2 Simuladores de Yacimientos Naturalmente Fracturados.

Los simuladores de yacimientos fracturados son programas de cómputo capaces de predecir el comportamiento que tienen este tipo de yacimientos. Estos simuladores trabajan bajo el principio de que el yacimiento es dividido en celdas de trabajo y entre mayor sea el número de celdas es mucho mejor; esto con el fin de que la computadora, debido a sus principios de

operación los cuales no pueden asignar a un mismo nodo varios valores de las propiedades utilizadas en el estudio en el mismo paso de tiempo, pueda representar, de alguna manera, un rango de valores de estas propiedades en una porción del yacimiento (modelo idealizado); es decir cuando nosotros imaginamos que, por ejemplo, la presión en el yacimiento es gradualmente mayor a partir del pozo productor hacia la frontera del yacimiento lo hacemos como si fuera una línea que gradualmente se va ensanchando; sin embargo en la computadora esto no sucede así, se tienen que emplear las celdas con un valor de presión en cada una de ellas para hacer esa representación; es por eso que entre mayor sea el número de celdas será mucho mejor, como se aprecia en la **figura 2.3**.

Figura 2.3. Representación de la presión en el yacimiento de forma real y discretizada.

La **figura 2.4** representa como es que un simulador de doble porosidad trabaja e interactúa con las celdas ya definidas.

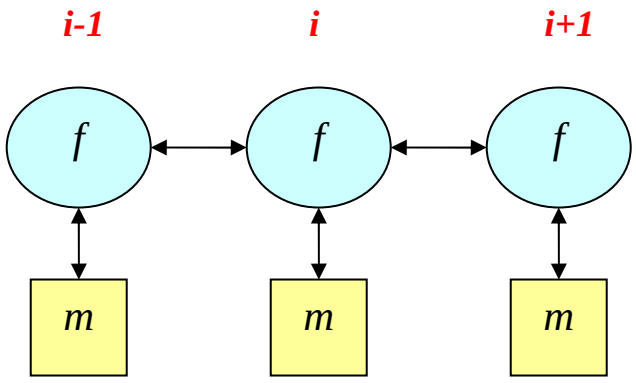


Figura 2.4. Modelo de doble porosidad **(Arana O., Víctor).**

Donde ***m*** es el bloque de matriz, ***f*** es la fractura, ***i*** es el bloque o nodo en estudio y las flechas indican la comunicación (movimiento de fluido) que se presenta entre los componentes.

Se debe entender que los nodos (puntos representativos de las celdas) en estudio cambian a lo largo del proceso de la simulación, es decir el nodo *i* no siempre es el mismo. De la misma manera los valores de las propiedades de estudio en cada nodo varían en cada intervalo de tiempo; es aquí donde los programas de los simuladores se encargan de hacer interactuar toda la información generada para que al final se muestren los datos-resultado los cuales corresponde al usuario del simulador entender y analizar.

2.2.1 Flujo de un fluido monofásico ligeramente compresible.

Para modelar el flujo de fluidos en medios porosos se utiliza la ecuación de difusión, la cual se obtiene a partir de la combinación de las ecuaciones de: Ecuación de continuidad (Ley de conservación de masa), Ley de conservación de momento (Ley de Darcy), Ecuación de estado y Ley de conservación de energía **(Aziz y Setari, 1979)**. Sin embargo, como el flujo de fluidos en el yacimiento se considera a temperatura constante, la ecuación de conservación de energía, frecuentemente, no es considerada. En el caso de flujo monofásico en una dimensión y con un fluido de baja compresibilidad la ecuación de difusión es (apéndice A):

$$\frac{\partial^2 p}{\partial x^2} = \frac{\mu}{k}\left[\phi c_t \frac{\partial p}{\partial t} + \tilde{q}_s\right] \tag{2.1}$$

y para tres dimensiones

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} = \frac{\mu}{k}\left[\phi c_t \frac{\partial p}{\partial t} + \tilde{q}_s\right] \tag{2.2a}$$

2.2.2 Discretización de la ecuación de difusión.

Si consideramos que en la **ecuación 2.2a** la permeabilidad y la viscosidad son función del espacio se tiene

$$\frac{\partial}{\partial x}\left(\frac{k_x}{\mu}\frac{\partial p}{\partial x}\right)+\frac{\partial}{\partial y}\left(\frac{k_y}{\mu}\frac{\partial p}{\partial y}\right)+\frac{\partial}{\partial z}\left(\frac{k_z}{\mu}\left(\frac{\partial p}{\partial z}-\rho g\right)\right)-\tilde{q}_s=\phi c_t\frac{\partial p}{\partial t} \qquad (2.2b)$$

esta ecuación es altamente no lineal por lo que sólo existen soluciones analíticas para casos muy simples, que no son aplicables en la mayoría de los casos reales. Sin embargo cuándo se quiere emplear esta ecuación en casos reales su solución se debe realizar por medio de algoritmos numéricos los cuáles cambian de un dominio continuo a un dominio discreto; por eso, cuándo se transforma una ecuación analítica a su forma numérica se dice que se ha discretizado.

En esta tesis no se muestra el proceso de discretización de la ecuación de difusión, ya que no es el objeto de estudio, solamente se toma la parte final del proceso, de la referencia indicada. **(Hernández R., O. y Del Valle M., P., 2005).**

Se busca que en el simulador que se utilizara en este trabajo se ocupen diferencias centrales para discretizar en el espacio y diferencias progresivas para discretizar en el tiempo, la formulación será implícita en presión, es decir, será calculada en un tiempo *n+1*, mientras que las transmisibilidades (T) se evaluarán al tiempo *n*. de esta forma una vez discretizada la **ecuación 2.2b**, desarrollando y reagrupando términos se tiene **(Hernández R., O. y Del Valle M,. P., 2005).**

$$\begin{aligned} &l_{i,j,k}\,p^{n+1}_{i,j,k-1}+s_{i,j,k}\,p^{n+1}_{i,j-1,k}+o_{i,j,k}\,p^{n+1}_{i-1,j,k}+c_{i,j,k}\,p^{n+1}_{i,j,k}\\ &+e_{i,j,k}\,p^{n+1}_{i+1,j,k}+n_{i,j,k}\,p^{n+1}_{i,j+1,k}+u_{i,j,k}\,p^{n+1}_{i,j,k+1}=d_{i,j,k}-q^{n}_{i,j,k} \end{aligned} \qquad (2.3)$$

Se observa en esta nueva ecuación que existe el término fuente *q* así como un coeficiente solución *d* y siete coeficientes, resultado de la interacción del nodo en cuestión con sus seis caras como se observa en la **figura 2.5**.

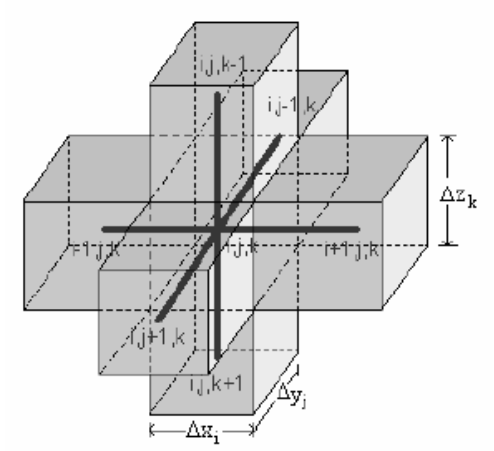


Figura 2.5. Modelo de análisis de un núcleo típico de malla para yacimientos no fracturados **(Hernández R., O. y Del Valle M,. P.).**

Los coeficientes *d,l,s,o,c,e,n,u* dependen de las transmisibilidades al tiempo *n*, que existen entre el nodo de análisis y sus fronteras mientras que el término fuente *q* es un gasto de producción (-) o de inyección (+) si es que existiera algún pozo en el nodo de interés. Estos nueve elementos son conocidos, ya que están definidos al tiempo *n*. Los coeficientes están definidos como

$$l_{i,j,k} = T^{n}_{i,j,k-\frac{1}{2}}$$

$$s_{i,j,k} = T^{n}_{i,j-\frac{1}{2},k}$$

$$o_{i,j,k} = T^{n}_{i-\frac{1}{2},j,k}$$

$$c_{i,j,k} = -\left[T^{n}_{i,j,k-\frac{1}{2}} + T^{n}_{i,j-\frac{1}{2},k} + T^{n}_{i-\frac{1}{2},j,k} + T^{n}_{i+\frac{1}{2},j,k} + T^{n}_{i,j+\frac{1}{2},k} + T^{n}_{i,j,k+\frac{1}{2}} + \frac{Vr_{i,j,k}}{\Delta t}\left\{ \phi^{n}_{i,j,k}\left(b_o c_f\right)_{i,j,k} + b^{n}_{i,j,k}\left(\phi_o c_r\right)_{i,j,k} \right\} \right]$$

$$e_{i,j,k} = T^{n}_{i+\frac{1}{2},j,k}$$

$$n_{i,j,k} = T^{n}_{i,j+\frac{1}{2},k}$$

$$u_{i,j,k} = T^{n}_{i,j,k+\frac{1}{2}}$$

$$d_{i,j,k} = -\frac{Vr_{i,j,k}}{\Delta t}\left[\phi^{n}_{i,j,k}\left(b_o c_f\right)_{i,j,k} + b^{n}_{i,j,k}\left(\phi_o c_r\right)_{i,j,k}\right]p^{n}_{i,j,k} + T^{n}_{i,j,k+\frac{1}{2}}\gamma_{i,j,k+\frac{1}{2}}\left(D_{i,j,k+1} - D_{i,j,k}\right) - T^{n}_{i,j,k-\frac{1}{2}}\gamma_{i,j,k-\frac{1}{2}}\left(D_{i,j,k} - D_{i,j,k-1}\right)$$

donde

$$T^{n}_{i+\frac{1}{2},j,k} = \left(\frac{\Delta y \Delta z}{\Delta x}\right)_{i+\frac{1}{2},j,k}\left(\frac{k}{\mu B}\right)_{i+\frac{1}{2},j,k} \qquad T^{n}_{i-\frac{1}{2},j,k} = \left(\frac{\Delta y \Delta z}{\Delta x}\right)_{i-\frac{1}{2},j,k}\left(\frac{k}{\mu B}\right)_{i-\frac{1}{2},j,k}$$

$$T^{n}_{i,j+\frac{1}{2},k} = \left(\frac{\Delta x \Delta z}{\Delta y}\right)_{i,j+\frac{1}{2},k}\left(\frac{k}{\mu B}\right)_{i,j+\frac{1}{2},k} \qquad T^{n}_{i,j-\frac{1}{2},k} = \left(\frac{\Delta x \Delta z}{\Delta y}\right)_{i,j-\frac{1}{2},k}\left(\frac{k}{\mu B}\right)_{i,j-\frac{1}{2},k}$$

$$T^{n}_{i,j,k+\frac{1}{2}} = \left(\frac{\Delta x \Delta y}{\Delta z}\right)_{i,j,k+\frac{1}{2}}\left(\frac{k}{\mu B}\right)_{i,j,k+\frac{1}{2}} \qquad T^{n}_{i,j,k-\frac{1}{2},} = \left(\frac{\Delta x \Delta y}{\Delta z}\right)_{i,j,k-\frac{1}{2},}\left(\frac{k}{\mu B}\right)_{i,j,k-\frac{1}{2},}$$

Ahora bien, la **ecuación 2.3** es la que gobierna el comportamiento de un yacimiento no fracturado en tres dimensiones; como en este trabajo se busca el estudio de yacimientos fracturados debemos agregar los elementos que rigen el flujo de fluidos por las fracturas, de la siguiente manera:

Si la **ecuación 2.2b** la consideramos en doble porosidad tenemos, para el sistema de fracturas

$$\frac{\partial}{\partial x}\left(\frac{k_{xf}}{\mu}\frac{\partial p_f}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{k_{yf}}{\mu}\frac{\partial p_f}{\partial y}\right) + \frac{\partial}{\partial z}\left(\frac{k_{zf}}{\mu}\left(\frac{\partial p_f}{\partial z} - \rho g\right)\right) - q'_{mf} = \phi_f c_f \frac{\partial p_f}{\partial t} \qquad (2.4a)$$

y para el sistema de bloques de matriz

$$\frac{\partial}{\partial x}\left(\frac{k_{xm}}{\mu}\frac{\partial p_m}{\partial x}\right)+\frac{\partial}{\partial y}\left(\frac{k_{ym}}{\mu}\frac{\partial p_m}{\partial y}\right)+\frac{\partial}{\partial z}\left(\frac{k_{zm}}{\mu}\left(\frac{\partial p_m}{\partial z}-\rho g\right)\right)+q'_{mf}=\phi_m c_m\frac{\partial p_m}{\partial t} \qquad (2.4b)$$

Como no hay flujo entre los bloques de matriz, los términos de flujo de la **ecuación 2.4b** se cancelan, quedando

$$q'_{mf}=(\phi c)_m\frac{\partial p_m}{\partial t} \qquad (2.5a)$$

Nótese que para el sistema de fracturas el signo de q'_{mf} es negativo y, por balance de materia, el signo para el sistema de bloques debe ser positivo. El término q'_{mf} está dado por

$$q'_{mf}=\frac{\sigma k}{\mu B}\left(p_f-p_m\right)\Rightarrow\ q_{mf}=q'_{mf}\,Vr\ \Rightarrow\ q_{mf}=\frac{\sigma Vrk}{\mu B}\left(p_f-p_m\right) \qquad (2.5b)$$

La discretización de la **ecuación 2.4a** es la siguiente

$$\begin{aligned}&l_{i,j,k}\,p^{n+1}_{f,i,j,k-1}+s_{i,j,k}\,p^{n+1}_{f,i,j-1,k}+o_{i,j,k}\,p^{n+1}_{f,i-1,j,k}+c_{i,j,k}\,p^{n+1}_{f,i,j,k}\\&+e_{i,j,k}\,p^{n+1}_{f,i+1,j,k}+n_{i,j,k}\,p^{n+1}_{f,i,j+1,k}+u_{i,j,k}\,p^{n+1}_{f,i,j,k+1}=d_{i,j,k}-q^{n}_{i,j,k}\end{aligned} \qquad (2.6)$$

La **ecuación 2.5a** discretizada es

$$q_{mf,i,j,k}=\frac{\left(Vr\phi c_t\right)^n_{m,i,j,k}}{\Delta t}\left(p^{n+1}_{m,i,j,k}-p^{n}_{m,i,j,k}\right) \qquad (2.7)$$

Sustituyendo la **ecuación 2.5b** en las **ecuaciones 2.6** y **2.7** tenemos

$$
\begin{aligned}
&l_{i,j,k}\, p^{n+1}_{f,i,j,k-1} + s_{i,j,k}\, p^{n+1}_{f,i,j-1,k} + o_{i,j,k}\, p^{n+1}_{f,i-1,j,k} \\
&+ c_{i,j,k}\, p^{n+1}_{f,i,j,k} + e_{i,j,k}\, p^{n+1}_{f,i+1,j,k} + n_{i,j,k}\, p^{n+1}_{f,i,j+1,k} \\
&+ u_{i,j,k}\, p^{n+1}_{f,i,j,k+1} - \left(\frac{\sigma Vrk}{\mu B}\right)_{i,j,k} \left(p^{n+1}_{f,i,j,k} - p^{n+1}_{m,i,j,k}\right) = d_{i,j,k} - q^{n}_{i,j,k}
\end{aligned}
\tag{2.8}
$$

y

$$
+\left(\frac{\sigma Vrk}{\mu B}\right)_{i,j,k} \left(p^{n+1}_{f,i,j,k} - p^{n+1}_{m,i,j,k}\right) = \frac{\left(Vr\phi bc_t\right)^{n}_{m,i,j,k}}{\Delta t}\left(p^{n+1}_{m,i,j,k} - p^{n}_{m,i,j,k}\right) \tag{2.9}
$$

Donde σ es conocido como el factor de forma matriz-fractura y se define como $\sigma = \frac{12}{L^2}$ cuando *L=Lx=Ly=Lz*, y como $\sigma = 4\left[\frac{1}{L_x^2} + \frac{1}{L_y^2} + \frac{1}{L_z^2}\right]$ si las longitudes son diferentes. *Lx, Ly, Lz* se refieren a las longitudes de la dirección correspondiente del bloque de matriz.

Podemos resolver para la incógnita $p^{n+1}_{m,i,j,k}$ de la **ecuación 2.9** y sustituirla en la **ecuación 2.8** haciendo

$$
W_{i,j,k} = \left[\sigma Vr \frac{k_m}{\mu B}\right]_{i,j,k}
$$

$$
Y_{i,j,k} = \frac{\left(Vr\phi bc_t\right)_{m,i,j,k}}{\Delta t}
$$

para *i* = 1,2,…,nx ; *j* = 1,2,…,ny ; *k* = 1,2,…,nz

La **ecuación 2.9** es entonces

$$
W_{i,j,k} = \left(p^{n+1}_{f,i,j,k} - p^{n+1}_{m,i,j,k}\right) = Y_{i,j,k}\left(p^{n+1}_{m,i,j,k} - p^{n}_{m,i,j,k}\right)
$$

Resolviendo para $p^{n+1}_{m,i,j,k}$

$$
p^{n+1}_{m,i,j,k} = \frac{W_{i,j,k}\, p^{n+1}_{f,i,j,k} + Y_{i,j,k}\, p^{n}_{m,i,j,k}}{W_{i,j,k} + Y_{i,j,k}} \tag{2.10}
$$

Sustituyendo la **ecuación 2.10** en la **ec. 2.8** nos queda entonces

$$
\begin{aligned}
&l_{i,j,k}\,p^{n+1}_{f,i,j,k-1} + s_{i,j,k}\,p^{n+1}_{f,i,j-1,k} + o_{i,j,k}\,p^{n+1}_{f,i-1,j,k} \\
&+ c_{i,j,k}\,p^{n+1}_{f,i,j,k} + e_{i,j,k}\,p^{n+1}_{f,i+1,j,k} + n_{i,j,k}\,p^{n+1}_{f,i,j+1,k} + u_{i,j,k}\,p^{n+1}_{f,i,j,k+1} \\
&-\left(\frac{\sigma Vrk}{\mu B}\right)_{i,j,k}\left[\,p^{n+1}_{f,i,j,k} - \left(\frac{W_{i,j,k}\,p^{n+1}_{f,i,j,k} + Y_{i,j,k}\,p^{n}_{m,i,j,k}}{W_{i,j,k} + Y_{i,j,k}}\right)\right] = \\
&d_{i,j,k} - q^{n}_{i,j,k}
\end{aligned}
\qquad (2.11)
$$

La **ecuación 2.11** tienen entonces a $p^{n+1}_{f,i,j,k}$ como incógnitas. Agrupando los términos semejantes se tiene

$$
\begin{aligned}
&l_{i,j,k}\,p^{n+1}_{f,i,j,k-1} + s_{i,j,k}\,p^{n+1}_{f,i,j-1,k} + o_{i,j,k}\,p^{n+1}_{f,i-1,j,k} + c_{i,j,k}\,p^{n+1}_{f,i,j,k} \\
&+ e_{i,j,k}\,p^{n+1}_{f,i+1,j,k} + n_{i,j,k}\,p^{n+1}_{f,i,j+1,k} + u_{i,j,k}\,p^{n+1}_{f,i,j,k+1} = d_{i,j,k} - q^{n}_{i,j,k}
\end{aligned}
\qquad (2.12)
$$

para i = 1,2,…,nx ; j = 1,2,…,ny ; k = 1,2,…,nz

Donde, redefiniendo, los coeficientes *l,s,o,c,e,n,u* y *d* son los siguientes

$$
\begin{aligned}
l_{i,j,k} &= T^{n}_{i,j,k-\frac{1}{2}} \\
s_{i,j,k} &= T^{n}_{i,j-\frac{1}{2},k} \\
o_{i,j,k} &= T^{n}_{i-\frac{1}{2},j,k} \\
c_{i,j,k} &= -\left[\begin{array}{l} T^{n}_{i,j,k-\frac{1}{2}} + T^{n}_{i,j-\frac{1}{2},k} + T^{n}_{i-\frac{1}{2},j,k} + T^{n}_{i+\frac{1}{2},j,k} + T^{n}_{i,j+\frac{1}{2},k} + T^{n}_{i,j,k+\frac{1}{2}} \\ + \dfrac{Vr_{i,j,k}}{\Delta t}\left\{\phi^{n}_{i,j,k}\left(b_o c_f\right)_{i,j,k} + b^{n}_{i,j,k}\left(\phi_o c_r\right)_{i,j,k}\right\} - \left[\dfrac{W^{2}_{i,j,k}}{W_{i,j,k} + Y_{i,j,k}} - W_{i,j,k}\right] \end{array}\right]
\end{aligned}
$$

$$e_{i,j,k} = T^{n}_{i+\frac{1}{2},j,k}$$

$$n_{i,j,k} = T^{n}_{i,j+\frac{1}{2},k}$$

$$u_{i,j,k} = T^{n}_{i,j,k+\frac{1}{2}}$$

$$d_{i,j,k} = -\left[\begin{array}{l} \dfrac{Vr_{i,j,k}}{\Delta t}\left[\phi^{n}_{i,j,k}\left(b_o c_f\right)_{i,j,k} + b^{n}_{i,j,k}\left(\phi_o c_r\right)_{i,j,k}\right]p^{n}_{f,i,j,k} + \\ \qquad T^{n}_{i,j,k+\frac{1}{2}}\gamma_{i,j,k+\frac{1}{2}}\left(D_{i,j,k+1} - D_{i,j,k}\right) - T^{n}_{i,j,k-\frac{1}{2}}\gamma_{i,j,k-\frac{1}{2}}\left(D_{i,j,k} - D_{i,j,k-1}\right) \\ + \dfrac{W_{i,j,k}Y_{i,j,k}}{Y_{i,j,k} + W_{i,j,k}}\, p^{n}_{m,i,j,k} \end{array}\right]$$

La representación del análisis de los nodos en yacimientos fracturados es el que se muestra en la **figura 2.6**, donde también se aprecia que no hay flujo entre bloques, solamente lo hay por la fractura, representada por el espacio entre los cubos.

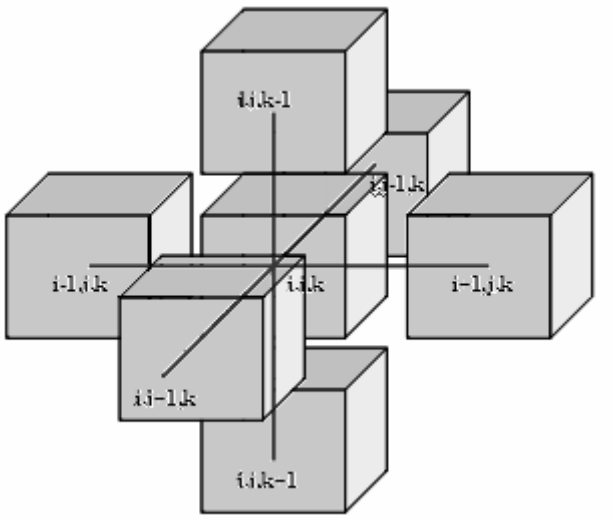


Figura 2.6. Modelo de análisis de un núcleo típico de malla cartesiana para yacimientos fracturados

Una vez definido cada bloque o celda, se genera un conjunto de ecuaciones que debe resolverse simultáneamente. Este sistema se puede representar de manera matricial. Por ejemplo para una malla de 3x3x3 en el sistema cartesiano, el arreglo matricial sería el que se aprecia en la **figura 2.7**.

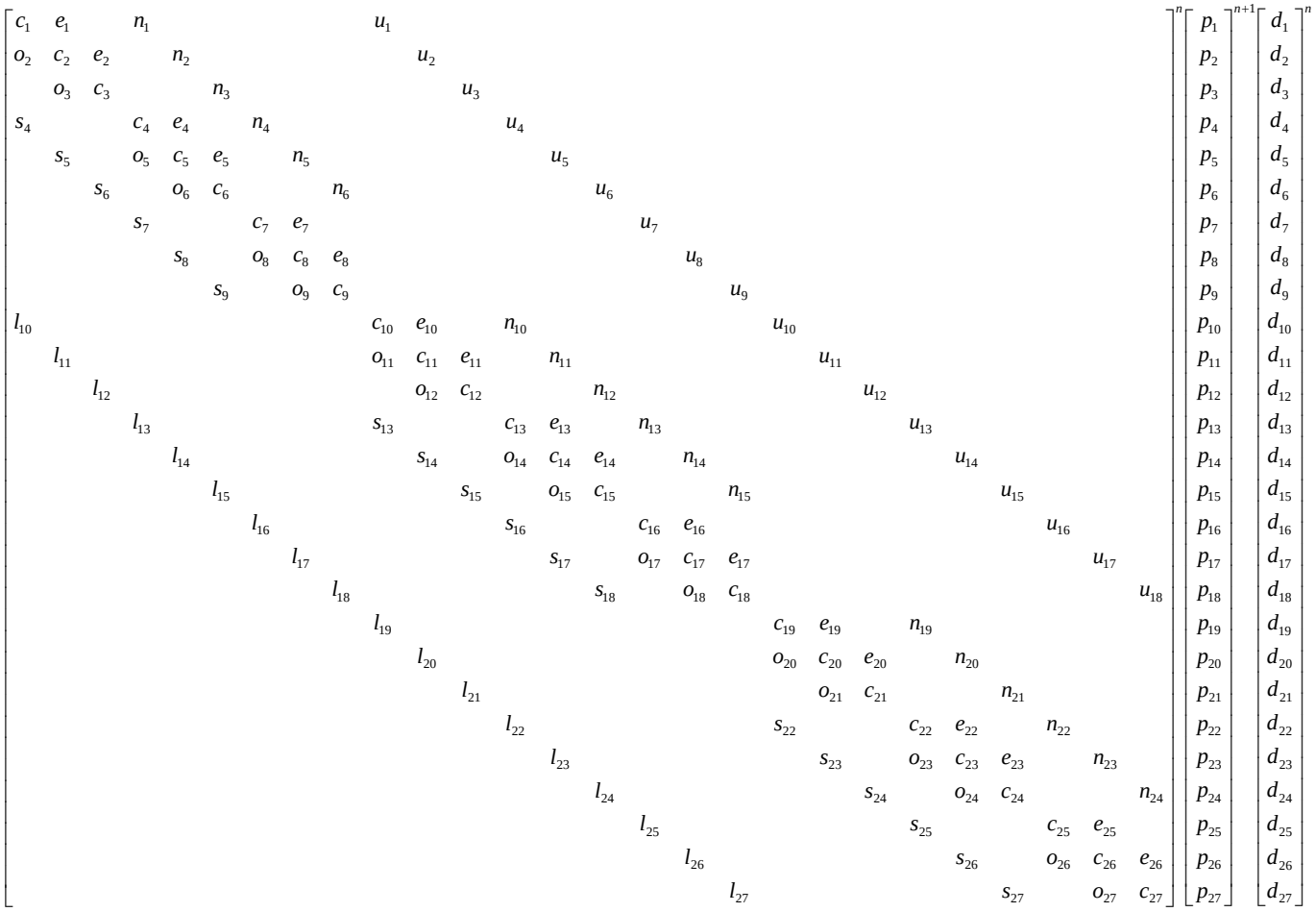


Figura 2.7. Esquema de una matriz de coeficientes para una malla de 3x3x3 **(Hernández R., O. y Del Valle M,. P.).**

El sistema de ecuaciones en diferencias que describe el comportamiento de flujo de fluidos en el yacimiento generado a partir de la educación 2.12, constituye un sistema de ecuaciones lineales que es resuelto, en este trabajo, empleando el método de GMRES especializado en la solución de ecuaciones de matrices dispersas, que corresponde a una de las técnicas directas analizadas para resolver sistemas matriciales grandes, debido a la relación directa, entre los requerimientos del sistema de computo, memoria y tiempo de ejecución, y el número de incógnitas que se resuelve simultáneamente. Entre mayor sea este número, mayores son los requerimientos del sistema y la eficiencia disminuye drásticamente, es por eso que el sistema de solución GMRES fue adaptado con un precondicionador para agilizar el proceso de solución.

Como se ha mencionado, los simuladores numéricos trabajan a base de celdas de análisis y entre más celdas sean serán mucho mejor los resultados; sin embargo esto también repercute en el tiempo de simulación requerido así como en la capacidad de trabajo de las maquinas, ya que entre mayor sea la cantidad de celdas de trabajo mayor deberá ser la capacidad de memoria de la maquina y mayor será el tiempo empleado en el proceso. Es por eso que una solución a estos problemas es el emplear procesos auxiliares como la programación en paralelo.

En este trabajo se ha definido una ecuación que representa el comportamiento de un yacimiento fracturado, desde el punto de vista de la simulación numérica, ec. 2.12. Esta ecuación se puede dividir en coeficientes (ya definidos arriba) y cada coeficiente calcularse de manera independiente, después compartir los resultados para que sean integrados nuevamente y obtener la solución total; es decir, cuando pensamos en paralelizar el programa de cómputo que da solución numérica a la ecuación 2.12 lo que queremos es que varios procesadores de computadora, comunicados entre sí, trabajen para que obtengan los coeficientes (*l,s,o,c,e,n,u*) por separado y, así cada procesador lleve a cabo el proceso que se la ha asignado y después interactúen con la información obtenida por cada uno de ellos para que obtengamos la solución final requerida, con todas las ventajas que nos dará el programar de manera paralela.

2.3 Análisis del Problema a Resolver.

Debido a todos los estudios que se han realizado, distintos investigadores han llegado a la conclusión en el consumo en el tiempo de cómputo en un proceso de simulación como se muestra en la **figura 2.8**, donde apreciamos que la parte a la que nos enfocaremos en este trabajo de paralelización es la que ocupa el segundo lugar en el consumo de tiempo total. Por lo que es un campo de trabajo que si se logra optimizar tendrá un beneficio considerable en la Simulación de Yacimientos. Ya se han desarrollado trabajos encaminados a la parte del proceso de solución, p.e. el que desarrollaron **Cedillo T., U. y Orantes L., R. (2005).**

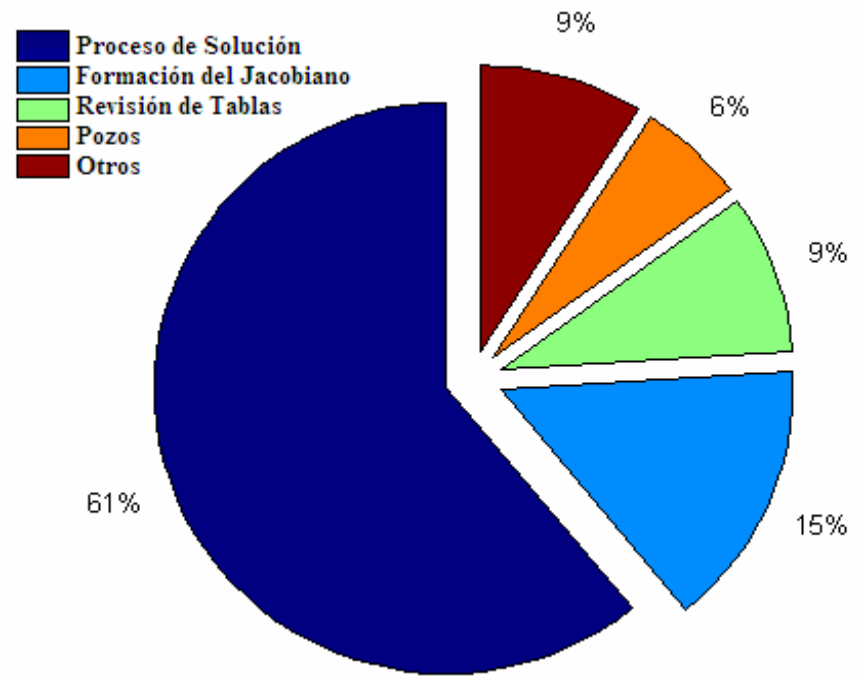


Figura 2.8. Repartición del tiempo de cómputo en la Simulación de Yacimientos **(Dogru, Ali H. y otros)**.

El trabajar con este problema es para proporcionar un fundamento, en la complicada tarea de la creación del jacobiano en un proceso de simulación numérica, para que se optimice el gasto de tiempo que se emplea en este rubro; buscando por consiguiente una disminución del tiempo empleado en toda la Simulación Numérica de Yacimientos, traduciéndose en más pruebas y mejores resultados. Como se mencionó existen trabajos que se enfocan a la parte de la solución, de los cuales muchos trabajan con esquema paralelo, entonces el trabajar en paralelo sobre la creación del jacobiano servirá además para adaptar y relacionar las optimizaciones de las dos tareas de manera más sencilla y eficiente.

Se debe aclarar que, en términos estrictos, el jacobiano es la matriz que se genera en un simulador numérico totalmente implícito resultado de hacer las derivadas de las funciones que involucran a la p, S_g y S_w como incógnitas, generando que cada elemento de esta matriz sea una submatriz; pero en este estudio se entenderá como jacobiano a la matriz de coeficientes conformada por elementos escalares resultado de manejar un simulador semi-implícito donde la única incógnita es la presión p. De aquí en adelante se manejaran los términos jacobiano y matriz de coeficientes indistintamente.

CAPÍTULO 3. CONCEPTOS DE INGENIERÍA DE YACIMIENTOS.

El objetivo de la ingeniería es la optimización. El objetivo de la ingeniería de yacimientos es proveer los factores, información y conocimientos a las operaciones de control para obtener la máxima recuperación posible de un yacimiento al menor costo posible. Ya que generalmente una recuperación máxima no es obtenida por una inversión mínima, el ingeniero debe buscar la combinación ideal de recuperación, costo y otros factores pertinentes.

Desde el punto de vista de un operador, cualquier procedimiento o acción que resulte en un beneficio óptimo es ingeniería eficiente y aquella que no lo haga no lo es. Para obtener buenos beneficios, todas las operaciones deben ser iniciadas en el tiempo adecuado. Una ingeniería de yacimientos efectiva debe proveer los factores necesarios suficientemente temprano para permitir un control más efectivo del yacimiento.

3.1 Ingeniería de yacimientos. Un esfuerzo grupal.

Calhoun (1963) ha descrito el sistema de ingeniería concerniente al ingeniero petrolero como la integración de tres subsistemas: (1) la creación y operación de pozos, (2) el procesamiento de los fluidos en superficie y (3) los fluidos y su comportamiento dentro del yacimiento. Los dos primeros están gobernados por el último. La naturaleza del yacimiento y de los fluidos determina cuántos pozos son necesarios, dónde deben ser perforados, cómo deben ser terminados y puestos en producción y que tratamiento de equipo es necesario para obtener el mejor beneficio. Los distintos subsistemas y sus especialistas no pueden estar aislados, deben ser considerados como partes interrelacionadas de un sistema unificado y un grupo de trabajo. El sistema es controlado completamente por el desempeño del yacimiento, que es sólo una pequeña distinción entre la ingeniería petrolera y la ingeniería de yacimientos **(Essley, P.L., 1963)**.

La ingeniería de yacimientos no debe comenzar con el inicio del desarrollo del yacimiento, sino con el descubrimiento para que la efectividad sea máxima.

3.2 El Proceso de Ingeniería.

La ingeniería de yacimientos aplica un conocimiento general del comportamiento de yacimientos a un sistema particular para producir el resultado deseado. Los sistemas de yacimientos con los cuales el ingeniero debe trabajar generalmente son complejos, involucrando yacimientos múltiples, barreras de flujo, fallas y distribuciones irregulares de propiedades petrofísicas.

La primera consideración en la ingeniería de yacimientos y la función principal del ingeniero es definir y evaluar el sistema del yacimiento. Definir quiere decir determinar la extensión areal, espesor, inclinación, limites de producción y ambiente geológico. El evaluar significa determinar las propiedades físicas del yacimiento y sus fluidos, la variación de estas propiedades a través del sistema y la ubicación de las heterogeneidades, barreras de flujo, fracturas, etc.; que afectan el flujo de los fluidos. Sólo cuando los límites y las propiedades del yacimiento están determinados adecuadamente el ingeniero tendrá el conocimiento suficiente para deducir el desempeño futuro del yacimiento.

Las técnicas modernas de ingeniería proveen al ingeniero de numerosas herramientas y procedimientos para el estudio de los yacimientos. Usadas juiciosamente, y en conjunto con datos geológicos y de producción, estas herramientas pueden aportar una buena idea de las condiciones internas del yacimiento.

3.2.1 El Programa de Evaluación del Yacimiento.

Un programa de evaluación coordinado no sólo brinda información para una mejor ingeniería y su costo es menor que un programa fortuito muy grande.

Ocasionalmente un programa de evaluación temprana presentará pruebas razonables de la comunicación del yacimiento y el drene sobre áreas extensas. Así, una pronta evaluación también provee datos para la unificación y oportunidades óptimas de operaciones de mantenimiento de presión **(Essley, P.L., 1963)**.

La definición y evaluación temprana del sistema del yacimiento es el requerimiento básico para una ingeniería efectiva. La ingeniería debe permitir la obtención de datos necesaria para la evaluación del sistema y debe participar en decisiones de operación con la consideración del yacimiento. Entonces el trabajo del ingeniero es obtener así como interpretar los factores necesarios para evaluar el sistema; es su responsabilidad conocer que datos son requeridos e idear un plan para obtenerlos al mínimo costo.

3.2.2 El Estudio Geológico.

Para definir y evaluar el sistema del yacimiento, el ingeniero debe considerar el ambiente de depósito, continuidad, litología y limites de la roca almacén. El ambiente de depósito ofrece pistas concernientes tanto a las grandes unidades geológicas como a las pequeñas no conformidades presentes en estas unidades, las cuales afectan significativamente el flujo y el desempeño del yacimiento **(Essley, P.L., 1963).**

Las heterogeneidades del yacimiento pueden ser la clave par interpretar el comportamiento del yacimiento o los sucesos en un proyecto de recuperación secundaria o mejorada. Lutita o líneas de arena o laminaciones delgadas, las cuales restringen o afectan el flujo, pueden o no ser continuas dentro de un área extensa. Tales irregularidades generalmente son muy delgadas para aparecer en registros geofísicos y rara vez son notadas en análisis de núcleos pero pueden ser observadas en afloramientos y son descritas frecuentemente en las descripciones geológicas de los núcleos.

Elkins (1963) comentó sobre el efecto de las heterogeneidades en la reducción de la permeabilidad vertical en aparentes arenas limpias. Estas barreras previenen la conificación de agua y gas de manera efectiva y también el drene hacia arriba o hacia abajo de la zona productora. La identificación de las heterogeneidades en los núcleos, o la deducción de sus efectos desde las pruebas de pozos, no indican que tales barreras sean continuas. En muchos yacimientos se conoce donde están estas delgadas líneas impermeables, colocadas aleatoriamente dentro de un gran cuerpo arenoso, que evitan la conificación pero que tienen un pequeño efecto al evitar la segregación vertical de los fluidos del yacimiento.

El conocimiento de la extensión y especie de las no uniformidades presentes puede ayudar al ingeniero en la interpretación de los datos del yacimiento o en el diseño de pruebas especiales para evaluar el comportamiento de éste. Sin embargo el efecto de estas no uniformidades en el desempeño del sistema, usualmente, es ignorado por los ingenieros.

El objetivo del trabajo geológico de campo, es la rectificación de las hipótesis de trabajo que se planea el explorador a partir del análisis de las diversas fuentes de información que consulta en las etapas previas al trabajo de campo, la toma de datos estratigráficos, estructurales y geoquímicas y la toma de muestras para estudios petrográficos, paleontológicos y de geoquímica. Con este trabajo geológico se pueden evaluar: las relaciones espaciales y temporales de los cuerpos rocosos para definir la historia geológica de la región, la distribución y el espesor de las unidades litoestratigráficas, los arreglos estructurales o estratigráficos de cada una de estas unidades, el contenido de sustancias de interés económico y las propiedades de la rocas.

En el proceso geológico de exploración directa se desarrollan diferentes etapas, las cuales quedan definidas de la siguiente manera **(Arellano G., Javier, 2003)**:

- Etapa 1 (de gabinete), en esta etapa se realizan las actividades de recopilación, selección, análisis y síntesis de información antecedente.
- Etapa 2 (de campo), queda definida con el levantamiento de secciones estratigráficas, estructurales, verificación de la cartografía y de las estructuras, toma de muestras representativas, construcción de un mapa geológico y la construcción de secciones geológicas.
- Etapa 3 (de gabinete), las actividades realizadas en esta etapa son el análisis de datos estructurales de rocas, la elaboración del informe técnico en forma de texto, la elaboración del mapa geológico definitivo (**figura 3.1**) y la elaboración de las secciones geológicas definitivas (**figura 3.2**).

Figura 3.1. Mapa Geológico **(Arellano G., Javier)**

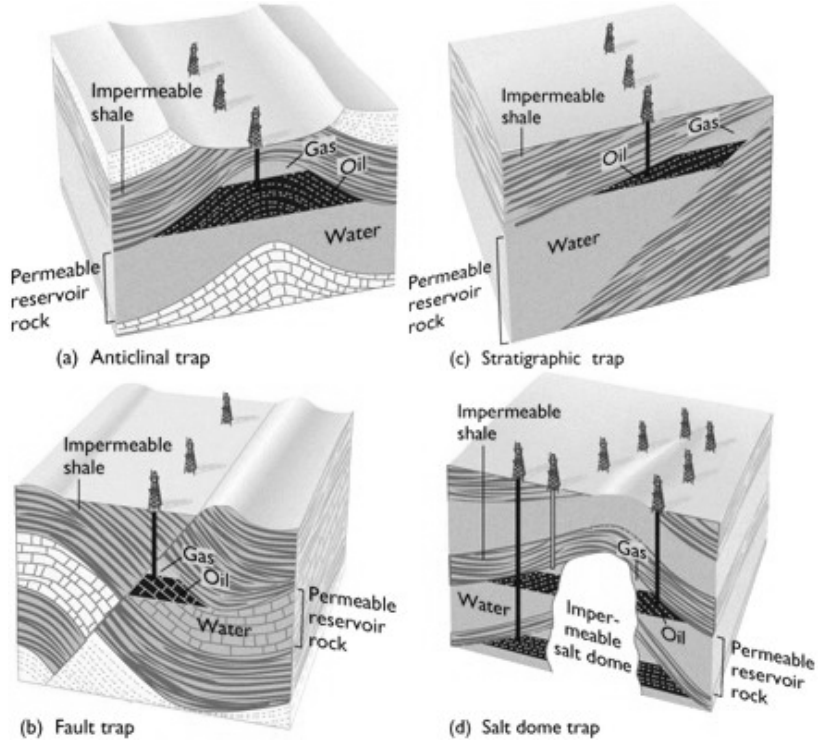


Figura 3.2. Ejemplos de Secciones Geológicas **(Arellano G., Javier)**

Así como existe un proceso geológico directo de exploración, también existe uno indirecto, el cual se basa en diferentes métodos para la elaboración de las bases geológicas. Estos métodos se aprecian en la **figura 3.3**.

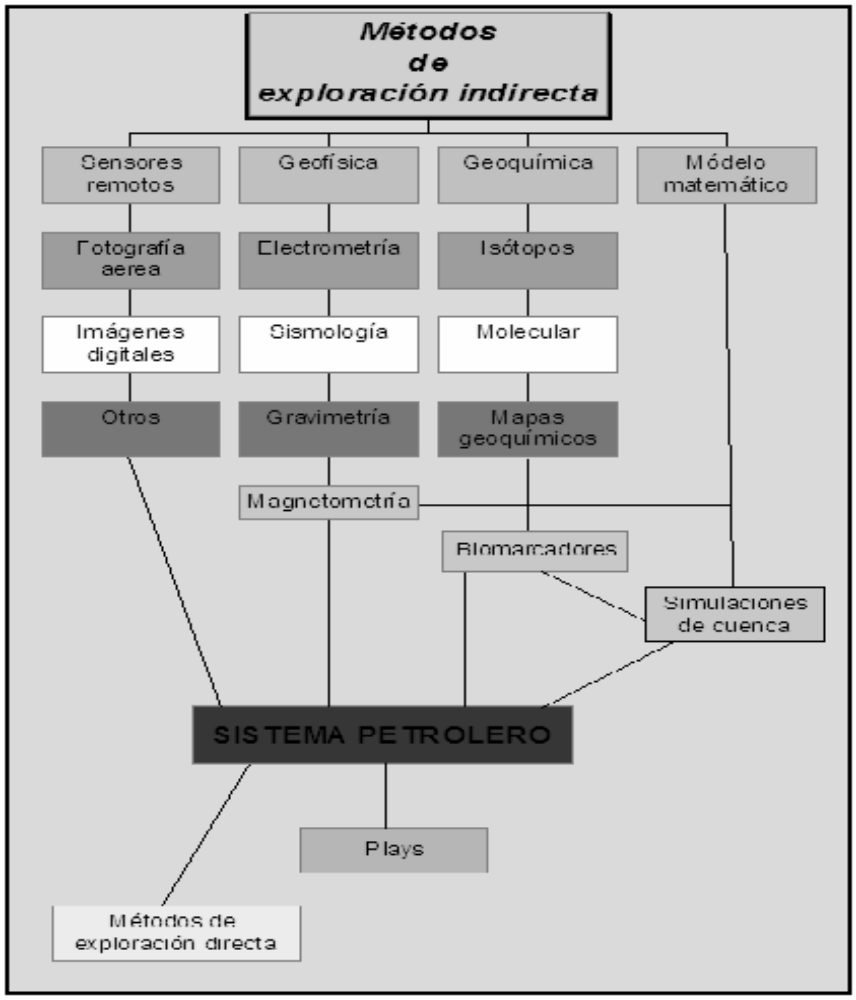


Figura 3.3. Métodos Indirectos de Exploración en el Proceso Geológico **(Arellano G., Javier)**

Ahora bien, nuestro caso de estudio en particular es sobre los yacimientos fracturados; podemos definir una fractura como la ruptura de la roca, que se presenta durante los procesos de deformación de ésta, debida a la perdida de cohesión del material, quedando registrados como planos o superficies de discontinuidad. El sistema de fractura puede favorecer o afectar el flujo de los fluidos dentro del yacimiento; una fractura que no ha sido ocupada por cementación o mineralización de manera total favorece el flujo del aceite hacía

los pozos, sin embargo también favorece el flujo de agua generado por los efectos de la conificación. Cuando una fractura ha sido cementada o mineralizada en su totalidad presenta una barrera impermeable impidiendo de tal forma el flujo de los fluidos y convirtiéndose en una barrera de flujo más (**Arellano G., Javier, 2003)**.

El fracturamiento de la roca no ocurre de manera aleatoria, siempre presenta arreglos orientados. El tamaño y orientación de los bloques de matriz generados por el proceso de fracturación varía dependiendo en gran medida de los esfuerzos que los originaron, así como los esfuerzos presentes en el subsuelo para cuando la fractura es inducida; siendo los bloques más pequeños los que se encuentran en la vecindad de las fallas.

En la ingeniería de yacimientos el estudio geológico debe preceder al estudio ingenieril. Las técnicas geológicas convencionales rara vez proveen suficientes datos para definir el sistema. El ingeniero debe complementar la geología con datos y pruebas de ingeniería para generar la información necesaria en el proceso. Los datos de producción, presión de formación, gradientes de presión y pruebas de presión pueden ser usados para probar la comunicación entre pozos o zonas, comprobar la existencia de fallas y otras barreras y definir al yacimiento en otros aspectos.

3.2.3 Mecánica de Yacimientos.

La mecánica de yacimientos generalmente recibe la mayor atención de los ingenieros en yacimientos. En realidad, muchos ingenieros se especializan en esta área y limitan sus prácticas para evaluar curvas de comportamiento y predecir el desempeño futuro **(Essley, P.L., 1963)**. Algunos de los conceptos que involucra la mecánica de yacimientos, los cuales rigen el comportamiento del sistema roca fluidos, se enuncian y definen abajo.

3.2.3.1 Porosidad de la Roca.

Un concepto muy importante que debemos conocer es la porosidad, este parámetro nos indica el volumen de fluidos que puede almacenar una roca. En el caso de yacimientos

fracturados el concepto de porosidad se maneja para dos sistemas o en dos términos. *Porosidad primaria* que es la que le corresponde al sistema de bloques de matriz y *porosidad secundaria* que es para es sistema de fracturas.

La porosidad nos indica los espacios que se forman entre los granos de la roca y que quedan libres, su manera de interpretarla es en porcentaje. La porosidad es una propiedad de las rocas que se puede cuantificar dividiendo el volumen de espacios que se encuentran en la roca entre el volumen total de la roca **(Prado M., Gustavo, 2004)**:

$$\text{porosidad}\,(\phi) = \frac{\text{volumen de espacios}}{\text{volumen total}}$$

La porosidad primaria se genera cuando el sedimento se deposita inicialmente. Por lo tanto, es una característica inherente a la roca. Su valor depende de varios factores, entre ellos el arreglo y distribución que guardan los granos del sedimento al momento de la depositación y la cementación.

Graton y Fraser en 1935 evaluaron la porosidad de dos configuraciones de empacamiento de granos: una cúbica y una romboédrica, donde los valores obtenidos solamente dependen del empacamiento de los granos y no del radio. Estas configuraciones se aprecian en la **figura 3.4**.

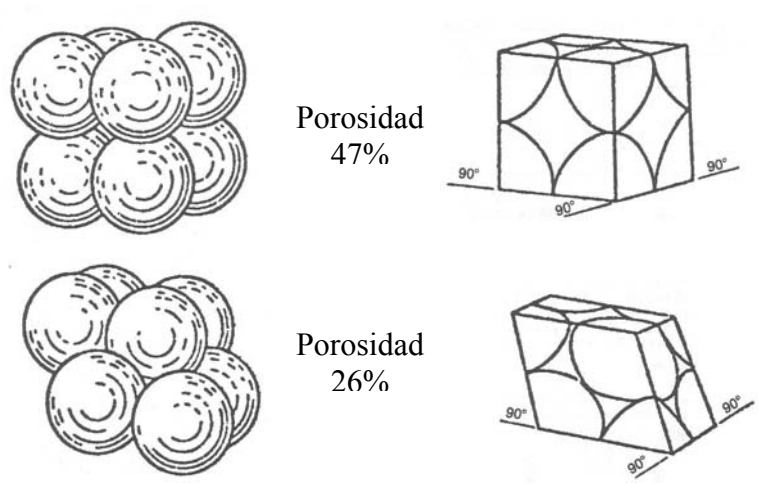


Figura 3.4. Representación de un empacamiento cúbico y romboédrico **(Prado M., Gustavo)**

Estos valores de porosidad son teóricos y debemos recordar que en la realidad no son así, sino serán menores debido a la presencia de cementantes, empacamiento de arcillas, que los granos de sedimento no son esferas perfectas y su tamaño no es constante, entre otros aspectos.

La porosidad secundaria es generada por los procesos geológicos que se presentan después de la depositación de los sedimentos y no tiene relación con la forma de las partículas ni con su empacamiento, en general, la porosidad secundaria se debe a la disolución, recristalización, dolomitización y fracturamiento de la roca ya formada.

La porosidad del sistema de fracturas se puede relacionar con un punto específico de la roca o con volumen total de esta. Para el caso en el que es un punto específico, la porosidad de la fractura resulta un número cercano a 100% (recordemos que se interpreta en porcentaje). Para el caso de relación con el volumen total de la roca, la porosidad resulta un porcentaje muy pequeño. Por esto la porosidad de la fractura es dependiente de la escala ante la cual se relacione.

La determinación de la proporción de la porosidad que corresponde a la matriz y la que está constituida por el sistema de fracturas es muy importante. En algunos casos, más del 50% de la porosidad total corresponde a la secundaria. En un sistema poroso matricial que presente una baja capacidad de flujo, el desplazamiento de los fluidos estará generalmente controlado por las fuerzas capilares presentes. Mientras que en un sistema secundario (porosidad secundaria) imperarán los efectos gravitacionales, dónde las fases se separan con facilidad.

Los yacimientos fracturados siempre deben considerarse como un sistema de dos porosidades. La interacción entre los dos sistemas de porosidad puede afectar considerablemente el comportamiento de un yacimiento, si la comunicación entre una y otra es buena, ambos sistemas de porosidad pueden responder al gradiente de presión generado. De esta manera, la capacidad de almacenamiento y la recuperación de los hidrocarburos en los yacimientos fracturados pueden variar considerablemente.

Cuando la capacidad de almacenamiento en los poros de la matriz es grande, en comparación con la capacidad de almacenamiento de las fracturas, se tienen las mejores condiciones de explotación; aunque se lleguen a presentar problemas en la perforación de los pozos tales como pérdidas de circulación o brotes. En los yacimientos con esta característica los bloques de matriz tienen una permeabilidad vertical que, aunque sea relativamente baja, permite la acción efectiva de la segregación gravitacional de los fluidos en dicha matriz. La interacción entre los fluidos contenidos en los bloques matriciales y los existentes en las fracturas facilitan el desplazamiento del aceite, permitiendo así obtener buenas recuperaciones de manera natural.

También se presenta el caso en el que se tiene la misma capacidad de almacenamiento en la matriz y en las fracturas. Cuando esto se presenta, la matriz es compacta y de baja permeabilidad y las fracturas tienen una permeabilidad muy grande.

El último de los casos es cuando el almacenamiento del sistema es sostenido prácticamente por las fracturas únicamente, esto se presenta cuando la porosidad de los bloques de matriz es casi nula y, además la saturación de agua en la matriz es casi total e inmóvil. Un yacimiento con estas características es aquel que tiene un gasto inicial muy alto pero que declinan de manera drástica en un periodo muy corto de tiempo.

3.2.3.2 Permeabilidad de la Roca.

Al igual que la porosidad, la permeabilidad es una propiedad de la roca y es una medición de la capacidad del medio para transmitir fluidos. Los yacimientos fracturados tienen una permeabilidad primaria y una secundaria, asignándose a los sistemas de matriz y fracturas de la misma manera que la porosidad, es decir, la *permeabilidad primaria* es la que nos indica la capacidad de flujo que tienen los bloques de matriz y la *permeabilidad secundaria* representa la capacidad de flujo del sistema de fracturas en la roca.

A diferencia de la porosidad, definimos otros tipos de permeabilidad en los yacimientos (fracturados o no fracturados); estas otras permeabilidades son **(Prado M., Gustavo, 2004)**:

La *permeabilidad absoluta (k_s)* la cual se refiere a la permeabilidad total de la roca medida con base en el fluido que la satura al 100%. Por ejemplo, en una roca mojada por agua, la permeabilidad absoluta es la permeabilidad al agua cuando ésta satura la roca al 100%.

La *permeabilidad efectiva (k_e)*, es la permeabilidad de un fluido en presencia de otro medida a una saturación específica. La permeabilidad efectiva es generalmente menor que la permeabilidad absoluta y cambia a medida que se modifica la saturación de los fluidos en la roca.

La *permeabilidad relativa (k_r)*, es la permeabilidad medida a una saturación de un fluido específico entre la permeabilidad absoluta de la roca. Es expresada como una fracción. Esta permeabilidad depende de la saturación del fluido de interés.

$$k_{rg} = \frac{k_g}{k}; \text{permeabilidad relativa al gas}$$

$$k_{ro} = \frac{k_o}{k}; \text{permeabilidad relativa al aceite}$$

$$k_{rw} = \frac{k_w}{k}; \text{permeabilidad relativa al agua}$$

Recordando que la permeabilidad primaria de la roca es la permeabilidad que se considera sin tomar en cuenta las fracturas, podemos calcularla utilizando la ecuación de Darcy para fluidos incompresibles.

$$k_m = \frac{q\mu L}{A\Delta p} \qquad (3.1)$$

donde: k: permeabilidad [darcys]

q: gasto [bls/día]

μ: vicosidad [cp]

L: distancia [pies]

A: área de flujo [pies^2]

Δp: diferencial de presión [lb/pg^2]

La presencia de fracturas abiertas y no cementadas incrementa de manera considerable la permeabilidad de la roca. Es posible estimar la permeabilidad de una fractura (permeabilidad secundaria) con la siguiente ecuación **(Prado M., Gustavo, 2004)**:

$$k_f = 54 \times 10^6 w_o^2 \ [\text{darcys}] \qquad (3.2)$$

En dónde w_o es el ancho de la fractura en pulgadas.

La permeabilidad tiene un efecto considerable en el flujo de fluidos en el yacimiento y por consecuencia en la recuperación de aceite. En los yacimientos fracturados la permeabilidad actúa de manera diferente que en yacimientos sin fracturas, ya que los conductos que forman las fracturas pueden favorecer de manera significativa la permeabilidad de la formación. La interacción entre la permeabilidad de la matriz y la permeabilidad de la fractura juega también un papel importante en la facilidad de flujo de los fluidos. Ya que si por ejemplo, tenemos muy buena permeabilidad en la fractura, pero si la mayor cantidad de los hidrocarburos se encuentran en los bloques de la matriz y, no se presenta flujo de la matriz hacia las fracturas, el flujo hacia el pozo será muy reducido o nulo. En cambio si la permeabilidad de la matriz y la de la fractura se relacionan de manera favorable, el flujo de los fluidos a través del medio poroso es mucho mejor y como consecuencia la recuperación de los hidrocarburos será mejor, que es lo que se quiere.

Para efectos de la simulación de yacimientos, las permeabilidades relativas seleccionadas para aplicarse en los modelos matemáticos determinan los resultados de la predicción del comportamiento del yacimiento. Los valores de permeabilidades relativas obtenidas en laboratorio pueden distar mucho de los valores reales de la formación, esto debido a las heterogeneidades de la formación y a que los valores son obtenidos a partir de núcleos que representan solamente una porción muy pequeña de la formación. Es por esta razón que es necesario ajustar los datos de permeabilidad relativa en la simulación de yacimientos para obtener un ajuste histórico que concuerde con el comportamiento pasado del yacimiento **(Prado M., Gustavo, 2004)**.

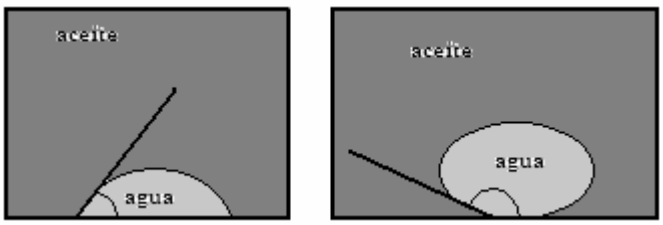
3.2.3.3 Mojabilidad de la Roca.

El concepto de mojabilidad de la roca hace referencia a la "preferencia" que tiene un sólido para hacer contacto con un fluido, el cual es conocido como fase mojante del sistema, más que con cualquier otro.

Las rocas en general, tienden a ser mojadas por agua, por aceite o por una mezcla de agua y aceite, aunque llega a haber casos en los el gas puede ser la fase mojante en algunos minerales como sulfuros.

En los yacimientos petroleros, la mojabilidad gobierna la distribución de los fluidos dentro del espacio poroso, de tal manera que determina en gran medida las saturaciones residuales de los fluidos y la capacidad de fluir de una fase en particular. La mojabilidad de la roca está determinada por la fase que más la imbiba, entendiéndose el término imbibición como el proceso mediante el cual una roca porosa absorbe una fase mojante; la imbibición espontánea se refiere al proceso de una roca de absorber la fase mojante sin la necesidad de una fuerza (presión) que la obligue **(Prado M., Gustavo, 2004)**.

En la **figura 3.5**, se muestra cómo se determina el fenómeno de mojabilidad considerando dos fluidos, agua y aceite, en contacto y sobre una superficie plana.



$\theta < 90°$ $\theta > 90°$

Figura 3.5. Fase mojante y no mojante.

Para determinarlo, se ha convenido, que el ángulo de contacto (θ) es el ángulo medido entre la superficie del sólido y la tangente a la superficie de la fase más densa en un punto de

contacto con la superficie plana, siendo definida la fase mojante en base a los siguientes datos **(Prado M., Gustavo, 2004)**:

1) mojada por agua → para ángulos menores a 75°
2) mojada por aceite → para ángulos mayores a 105°
3) mojabilidad mixta → para ángulos entre 75° y 105°

Los cambios de mojabilidad además de determinar las saturaciones residuales y facilidad de flujo de algún fluido, también afectarán la presión capilar del sistema y la permeabilidad relativa de dichos fluidos.

3.2.3.4 Saturación de Fluidos en la Roca.

La saturación de los fluidos indica la fracción que cada fluido ocupa del espacio poroso de la roca. La saturación es definida por el cociente del volumen del fluido de interés, medido a condiciones del medio poroso, entre el volumen total del espacio poroso, es expresada en fracción o también en porcentaje. La saturación de un fluido no puede exceder el 100% del espacio poroso de la roca.

$$S_g = \frac{\text{vol. de gas en el medio poroso}}{\text{volumen total de poros}}; \qquad \text{saturación de gas}$$

$$S_o = \frac{\text{vol. de aceite en el medio poroso}}{\text{volumen total de poros}}; \qquad \text{saturación de aceite}$$

$$S_w = \frac{\text{vol. de agua en el medio poroso}}{\text{volumen total de poros}}; \qquad \text{saturación de agua}$$

En un sistema roca-fluidos es muy común encontramos las tres fases presentes (gas, aceite y agua). Tomando en cuenta la presencia de estas tres fases podemos establecer la siguiente relación, que cumple con el principio de que el volumen total de fluidos es igual al volumen total de poros interconectados:

$$S_g + S_o + S_w = 1 \tag{3.3}$$

Así como las propiedades de la roca antes mencionadas, la saturación de fluidos también juega un papel importante en la recuperación de hidrocarburos. Un yacimiento con saturación alta de aceite nos dará altas recuperaciones de éste, siempre y cuando existan las relaciones de permeabilidad y movilidad necesarias y adecuadas.

3.2.3.5 Presión Capilar.

Cuando dos fluidos inmiscibles están en contacto dentro de los poros de la roca, se forma una superficie curva entre los dos. La presión en el lado del fluido no-mojante de la interfase (p_{nm}), es mayor que la presión para el lado del fluido mojante (p_m). Esta diferencia de presiones se define como presión capilar (p_c).

$$p_c = p_{nm} - p_m \tag{3.4}$$

La presión capilar en un medio poroso se puede comparar con el aumento de un líquido mojante en un tubo capilar. La capilaridad se define como la elevación o depresión de la superficie de un líquido al estar en contacto con un sólido, como sucede en las paredes internas de un tubo capilar. El peso de la columna de líquido, que ha subido su nivel en las paredes, tomando como referencia el nivel del menisco, es determinado por la **ecuación 3.5**.

$$-\pi R^2 hg(\rho_l - \rho_v) \tag{3.5}$$

donde R = radio del tubo capilar
h = altura del nivel del líquido
$(\rho_l - \rho_v)$ = es la diferencia de densidades entre el líquido y el vapor
g = es la aceleración de la gravedad

La única fuerza responsable del incremento del nivel del fluido mojante en las paredes del tubo capilar es la tensión superficial entre el líquido y el sólido. Las fuerzas atribuidas a la tensión superficial están dadas por la siguiente ecuación:

$$2\pi R\sigma\cos\theta \tag{3.6}$$

donde: σ = tensión superficial

θ = ángulo de contacto entre el líquido y el sólido

Si igualamos las fuerzas debidas al peso del líquido (**ec. 3.5**) y las fuerzas debidas a la tensión superficial (**ec. 3.6**) se tiene:

$$2\pi R\sigma\cos\theta = -\pi R^2 hg\left(\rho_l - \rho_v\right) \tag{3.7}$$

que puede expresarse:

$$\left(\rho_v - \rho_l\right)gh = \frac{2\sigma\cos\theta}{R} \tag{3.8}$$

El término de la **ecuación 3.8** que involucra la diferencia de densidades se puede representar como una diferencia de presión (Δp), a ésta diferencia de presión entre la fase no mojante y la fase mojante, que es lo que se conoce como presión capilar. De tal manera que si sustituimos dicho término por la definición de presión capilar se tiene:

$$p_c = pnm - pm = \frac{2\sigma\cos\theta}{R} \tag{3.9}$$

que es la manera de calcular el valor de la presión capilar en un sistema.

En los yacimientos fracturados, la presión capilar juega un papel muy importante en la producción de los hidrocarburos. Las fuerzas capilares pueden contribuir al proceso de

desplazamiento de un fluido en el medio poroso, como es el caso de la imbibición, o pueden oponerse a este desplazamiento, como es el caso del drene.

El *drene* es el proceso por el cual la fase no-mojante desplaza, del medio poroso, a la fase mojante. Es un proceso forzado (no es espontáneo) pues las fuerzas capilares tienden a retener la fase mojante dentro de la estructura capilar. En general, el aceite se comporta como la fase no mojante en un yacimiento, razón por la cual al principio de la explotación se presenta un desplazamiento por drene.

La *imbibición* es el proceso espontáneo de desplazamiento, con una fase mojante, de la fase no-mojante. Este proceso no requiere aplicación de fuerzas externas al sistema roca-fluidos.

Curvas de Presión Capilar

Durante el proceso de drene se presenta una relación entre la presión capilar y la saturación del fluido, a esta relación se le conoce como curva de drene. Así mismo, durante la imbibición se presenta una relación similar que se conoce como curva de imbibición. Para una distribución normal de poros, la magnitud de la presión capilar para la imbibición es aproximadamente la mitad de la del drene. A esta diferencia en magnitudes para las curvas de drene e imbibición se le conoce como histéresis.

En la **figura 3.6** se aprecia el comportamiento de las curvas de drene e imbibición.

Conforme a la historia regular de llenado de las trampas de hidrocarburos (migración de los hidrocarburos), éstas se encontraban originalmente saturadas al 100 % con agua congénita. Durante el almacenamiento en el espacio poroso, el hidrocarburo desaloja una parte del agua conforme a una curva de drenaje como la curva I en la **figura 3.6**. En este esquema se asume que el agua es la fase mojante y que el hidrocarburo (gas o petróleo) es la fase no-mojante.

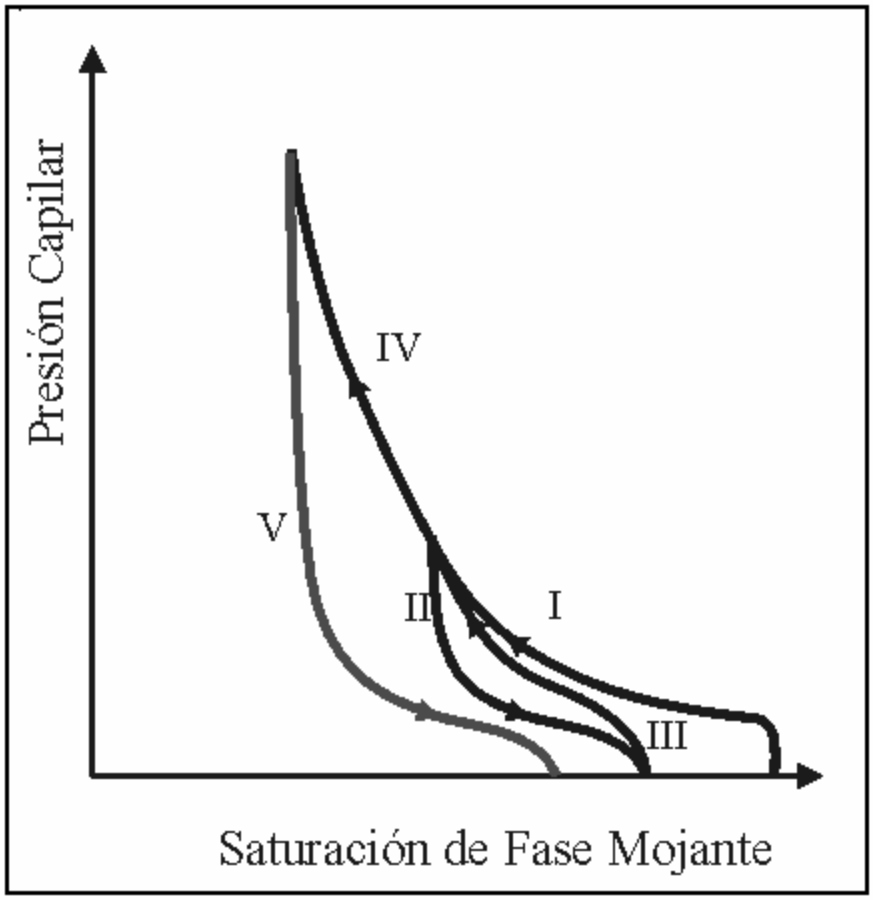


Figura 3.6. Curvas de presión capilar, drene e imbibición **(www.inlab.com.ar/Histeresis.htm)**.

Si el drene se ve interrumpido por alguna razón y comienza un desplazamiento con agua (por pérdida del sello de la trampa o por inyección de agua durante la explotación del yacimiento), la presión capilar del sistema comienza un proceso de imbibición, esquematizado con la curva II en la **fig. 3.6**.

Tal como se observa, el desplazamiento del hidrocarburo no es completo durante la imbibición, pues parte del mismo queda retenido en la estructura porosa bajo la forma de saturación residual de aceite (Sor).

Si posteriormente a este proceso de imbibición, se inicia un nuevo proceso de drenaje, este se desarrolla conforme al comportamiento de la curva III de la **fig. 3.6**. En este punto, las

curvas II y III constituyen un ciclo estable, el cual puede recorrerse una cantidad indefinida de veces sin sufrir alteraciones.

Si más adelante se continúa el proceso de drene, se aumentan las fuerzas capilares y la curva capilar queda representada por la curva IV, esquematizada en la **figura 3.6**. Como nos damos cuenta, la curva IV es una continuación perfecta, pero sin solución de continuidad, del comportamiento iniciado con la curva I.

Al haber continuado la curva I, ya no es posible reproducir el camino correspondiente a la curva II. Si se produce un nuevo proceso de imbibición, se origina un nuevo camino, curva V, tal como está indicado. La curva V conduce a un valor de Sor más grande que el correspondiente a la curva II. Esto obedece a que el hidrocarburo contactó una parte más grande del espacio poroso y, por lo tanto, tuvo acceso a ductos capilares no invadidos por el drene inicial; por lo que al retirar el hidrocarburo existen más lugares donde el mismo puede quedar atrapado. En consecuencia, el valor de saturación residual de la fase no mojante, no es una propiedad del medio poroso; sino que también interviene la historia de saturaciones en la magnitud final de esta saturación.

La saturación residual de la fase mojante en condiciones de drene es alcanzada asintoticamente a medida que la presión capilar aumente de manera infinita. En el caso de la imbibición la curva muestra una saturación de la fase no mojante, esta saturación residual ocurre durante el desplazamiento por imbibición en el cual la fase no mojante queda atrapada en espacios porosos pequeños. Este entrampamiento es resultado de la interrupción parcial de la comunicación a través del espacio poroso y como consecuencia varios poros pasan inadvertidos durante la imbibición, por lo que queda una saturación residual de la fase no mojante mayor.

La presión capilar en los yacimientos fracturados se encuentra, en su mayor parte, presente en los bloques correspondientes a la matriz. El desplazamiento de los fluidos en los bloques de matriz estará determinado por las fuerzas capilares que existan entre los fluidos y el espacio poroso que los contenga.

En yacimientos fracturados, el mecanismo de drene ocurre cuando la fase no mojante, presente en las fracturas, desplaza a la fase mojante, presente en la matriz. En este proceso las fuerzas capilares actúan en contra del desplazamiento mientras que las fuerzas gravitacionales actúan a favor. El desplazamiento de fluidos en un yacimiento se ve afectado por las fuerzas capilares que actúan en este. Las fuerzas capilares pueden retener el flujo, pero en algunas ocasiones, bajo ciertas condiciones, pueden favorecerlo. Es decir, las fuerzas capilares contribuyen al desplazamiento de aceite por agua o gas de los bloques de matriz mediante los procesos de imbibición y drene. Dependiendo del comportamiento de la roca y los fluidos, el drene o la imbibición será el efecto que presentará el sistema en la extracción del aceite **(Prado M., Gustavo, 2004)**.

Hay una diferencia entre la ingeniería de yacimientos y la aplicación de la mecánica de yacimientos. La determinación de un mecanismo de producción de un yacimiento y la predicción de su comportamiento futuro no es ingeniería como tal; la ingeniería efectiva, requiere la deducción del probable comportamiento del yacimiento bajo todos los métodos posibles de operación y entonces controlando este suceso obtener el beneficio óptimo. Esto usualmente requiere decisiones de operación antes de que el comportamiento del yacimiento sea claro. Los ingenieros y geólogos no son infalibles, ellos cometen errores; si estas decisiones de operación están precedidas por una prueba sistemática para definir y evaluar el sistema del yacimiento, las oportunidades de deducción exitosa del desempeño futuro del yacimiento y las operaciones de control para obtener un mejor beneficio serán mucho mejores (**Essley, P.L., 1963)**

3.2.4 El Uso de Modelos.

Debido a la complejidad de la mayoría de los yacimientos, es imposible duplicarlos o construir un modelo prototipo verdadero. Todos los modelos con los cuales trabajamos generalmente son sistemas simplificados; tales modelos brindan información valiosa concerniente a la naturaleza general de los sistemas del yacimiento y del flujo de fluidos

dentro de éste. Realmente, es del estudio de estos modelos que obtenemos mucho del conocimiento concerniente a la mecánica del yacimiento **(Essley, P.L., 1963)**.

Podemos realizar la representación del yacimiento por medio de un modelo geológico, recordando que será mucho más sencillo que la estructura real. Para llevar a cabo el modelo geológico, es necesario definir los diferentes modelos que lo pueden representar: *Modelo Estructural*. La construcción de este modelo se refiere principalmente a definir los mapas estructurales de la acumulación de hidrocarburos y la interpretación de los patrones de falla y fracturas que afectan al yacimiento, *Modelo Estratigráfico*. Con este se definen las unidades de flujo en el sistema roca y fracturas y los mapas de distribución de arenas. También se establece la arquitectura interna del yacimiento definiendo mapas estructurales, planos de falla e identificando compartimientos, si es que existiera. Y *Modelo Sedimentológico*. Compuesto de dos partes principales, descripción y clasificación de litofacies y definición del modelo de depositación. Con lo anterior se determina el ambiente sedimentario y define la geometría, distribución y calidad de los depósitos de las unidades de flujo.

Habiendo definido el modelo geológico es necesario elaborar un *modelo petrofísico* que pueda describir las características principales del medio poroso, con el fin de determinar el volumen de hidrocarburos existentes en el yacimiento. Con dicho modelo se definen los parámetros básicos de permeabilidad, porosidad y saturación inicial de agua. Así mismo se definen propiedades de los fluidos,

La ventaja de la gran velocidad en los equipos de cómputo ha permitido la construcción de *modelos matemáticos* para el estudio del flujo multifásico y multidimensional de fluidos. Estos modelos se acercan a duplicar sistemas simples del yacimiento y brindan comprensión adicional concerniente al comportamiento del yacimiento. Como herramientas científicas son soberbias, permitiéndonos no olvidar dos factores significantes **(Essley, P.L., 1963)**: los modelos matemáticos de yacimientos están grandemente simplificados todavía, en comparación con muchos yacimientos, y hasta que podamos definir el sistema del yacimiento de manera exacta es imposible duplicarlo en un modelo. Siempre debemos

recordar que ningún modelo, por bueno que sea, puede darnos una respuesta exacta si la información introducida es errónea.

3.2.5 La Importancia de la Medida del Tiempo.

La optimización requiere la consideración del elemento tiempo; con frecuencia, el *cuándo* hacer algo puede ser aproximadamente tan importante consideración como *qué* hacer. Muchos ingenieros están llegando a ser cada vez más concientes que la medida apropiada del tiempo es una consideración vital en la ingeniería. Las generalizaciones de cómo la medida apropiada del tiempo, para iniciar operaciones en un campo en particular, no son posibles. Una generalización válida en la ingeniería es: el mejor tiempo para aplicar principios de ingeniería de yacimientos es tan tempranamente como sea posible **(Essley, P.L., 1963)**.

3.3 La Ingeniería de Yacimientos. Como Práctica Individual.

La ingeniería de yacimientos es más un arte que una ciencia exacta, aunque tiene una base científica amplia. Mayores factores del yacimiento o fenómenos observados están sujetos a más que una interpretación lógica. **Wyllie (1962)** describe esta peculiaridad de la ingeniería de yacimientos con respecto a la interpretación de pruebas piloto de campo, pero extiende sus observaciones para cubrir todo de ingeniería de yacimientos. Esto es análogo para la condición matemática de tener más incógnitas que ecuaciones y obtener múltiples soluciones. Cuando lo complejo de la geometría del yacimiento, flujo de fluidos multifásicos, gradientes de potencial y la mecánica de yacimientos son considerados, múltiples interpretaciones no deberían causar asombro a ningún ingeniero. Sin embargo con frecuencia también estamos propensos a aceptar como válida la primera interpretación que parece más conveniente para los datos **(Essley, P.L., 1963)**. Esas piezas de información no son propias a incomodarnos o causarnos preguntas sobre nuestra interpretación.

En teoría la ingeniería de yacimientos está basada en amplios principios científicos. En la práctica no es rigurosamente científica. Para empezar tratamos con sistemas que pueden ser

increíblemente complejos e imposibles de definir completamente; para llegar a un diagnóstico de nuestro sistema generalmente contamos con **(Essley, P.L., 1963)**: (1) sólo un poco de los factores físicos, (2) estadísticas de producción, frecuentemente de dudosa puntualidad, (3) muestras representativas de sólo una parte pequeña del yacimiento, (4) técnicas estadísticas de promedios y (5) ecuaciones matemáticas derivadas de asumir que pueden, sólo remotamente, representar las condiciones del yacimiento.

Desde el punto de vista del ingeniero individualista, la ingeniería próspera está limitada a optimizar un sistema desde el tiempo en el que llegó a familiarizarse con ella. Aun desde este limitado punto de vista, la ingeniería efectiva depende de reconocer la naturaleza del yacimiento y su comportamiento. Más ejemplos de una pobre ingeniería individual resultan de una no garantizada confianza en las curvas de comportamiento y los procedimientos de cálculo como herramientas para evaluar el comportamiento del yacimiento; se necesita de un incrementado esfuerzo para definir y evaluar el sistema del yacimiento y aumentar el uso de gráficas que mejoran los esfuerzos individuales de ingeniería.

3.4 La Dificultad de la Ingeniería de Yacimientos.

Los ingenieros de yacimientos tratan con sistemas los cuales no pueden ser examinados físicamente; un conocimiento total del yacimiento no es posible. El trabajo del ingeniero es más complicado por la falta de exactitud de muchos datos; los datos de producción de agua son con frecuencia irreales, las presiones medidas no representan a las presiones estabilizadas y los resultados obtenidos de muestras de fluidos pueden no representar de manera correcta a los fluidos del yacimiento. Consecuentemente no debemos esperar soluciones exactas de nuestros cálculos aun en las ocasiones en que apliquemos ecuaciones muy complejas o modelos muy sofisticados. Esto no quiere decir que nuestros cálculos sean equivocados, sino simplemente que debemos considerar que los elementos que utilizamos provienen de pistas del comportamiento del yacimiento y no de indicadores exactos de este comportamiento **(Essley, P.L., 1963)**.

Más aún, deberíamos cuestionarnos siempre los resultados de nuestros cálculos, si estamos obteniendo la respuesta correcta. Continuamente deberíamos buscar respuestas a las siguientes preguntas **(Essley, P.L., 1963)**: (1) ¿Qué significa la respuesta?, (2) ¿La respuesta ajusta todos los factores?, (3) ¿por qué no es así?, (4) ¿Hay otra interpretaciones posibles de los datos?, (5) ¿Fue correcto lo asumido?, (6) ¿Son los datos reales?, (7) ¿Son necesarios datos adicionales?, (8) ¿Ha habido un estudio geológico adecuado? y (9) ¿Ha sido definido adecuadamente el yacimiento?

Para tener éxito debemos ser innatamente curiosos y científicamente honestos. Debemos cuestionarnos continuamente sobre los resultados y buscar factores adicionales.

3.5 La Experiencia Requerida para la Ingeniería de Yacimientos.

Para tener éxito, el ingeniero debe desarrollar el conocimiento del geólogo sobre sedimentos y condiciones de ambiente; el conocimiento del fisicoquímico sobre las propiedades de los fluidos del yacimiento, el comportamiento de fases, conductividad eléctrica y flujo de fluidos en sistemas porosos y el conocimiento del matemático en el análisis numérico y el uso de computadoras. En suma, debe estar completamente familiarizado con el pasado de producción y las prácticas de terminación en los pozos, incluyendo saber cuales zonas están perforadas en todos los pozos y el desempeño de cada pozo. También debe ser un economista, un contador, un buen negociador y tener conocimientos de leyes e impuestos. Pocos ingenieros desarrollan esta experiencia a profundidad; el ingeniero debe desarrollar un conocimiento trabajando en cada área y saber cuando consultar con especialistas para obtener información adicional. La experiencia del ingeniero en yacimientos debe ser una generalidad y no una especialidad.

CAPÍTULO 4. CONCEPTOS DE PROGRAMACIÓN EN PARALELO.

La industria del petróleo está entre aquellas que son de las primeras en introducir los avances de la computación y donde las nuevas tecnologías son adoptadas completamente y popularmente. La simulación de yacimientos es una de las herramientas más poderosas para el análisis del complicado flujo de fluidos dentro de los yacimientos. Frecuentemente los ingenieros utilizan la simulación para predecir el comportamiento de la producción del yacimiento y el desarrollo de planes de trabajo específicos.

En los pasados años el alto desempeño computacional ha tenido un significante impacto en la evolución de los métodos numéricos predictivos utilizados en la ciencia y la ingeniería. La ingeniería petrolera, en particular, ha visto una mejora significante en las capacidades para la simulación de yacimientos. La complejidad de los modelos de simulación de yacimientos ha llevado a que las computadoras más capaces y rápidas se agoten cada vez más. El actual estado del arte de la arquitectura paralela permite a los modelos de simulación aproximarse más a la realidad mientras se enfatiza en la precisión y eficiencia de los modelos. Muchos modelos han sido investigados en diferentes arquitecturas paralelas y, en general, muestran la gran promesa para el uso de grandes computadoras paralelas para la simulación de yacimientos. A pesar de estos resultados, la simulación de yacimientos ha estado moviéndose lentamente junto con el cómputo en paralelo. Hay muchas razones por las que se presenta este lento avance, p.e. la naturaleza recursiva de las técnicas de solución lineal existentes para el modelado de yacimientos no es realmente adaptable a las arquitecturas paralelas. Por otro lado, el intercambio entre la carga de trabajo y la estructura de datos global aún no se han investigado completamente.

4.1 Arquitectura Paralela y sus Conceptos.

Una computadora paralela puede considerarse como una colección de CPU’s los cuáles trabajan en un mismo escenario para desarrollar una sola tarea. La tarea es subdividida entre los procesadores de tal forma que, si es de manera efectiva, el desempeño total de la computadora en una tarea se escala con el número de CPU’s asignado. La arquitectura de

las computadoras paralelas generalmente ha sido basada en dos conceptos principales: instrucción individual, datos múltiples (SIMD *single instruction, multiple data*) e instrucción múltiple, datos múltiples (MIMD *multiple instruction, multiple data*). La arquitectura SIMD de la 'Illiac IV' representa una de las primeras implementaciones de las arquitecturas paralelas a mediados de 1960. Muchas computadoras han sido basadas en los conceptos SIMD **(Killough, John E., 1993)**. La **figura 4.1** muestra un una configuración SIMD típica, en la cuál un procesador anfitrión o principal distribuye una instrucción hacia procesadores individuales, la cuál continua desempeñando exactamente la misma tarea en cada paso del programa en cada procesador.

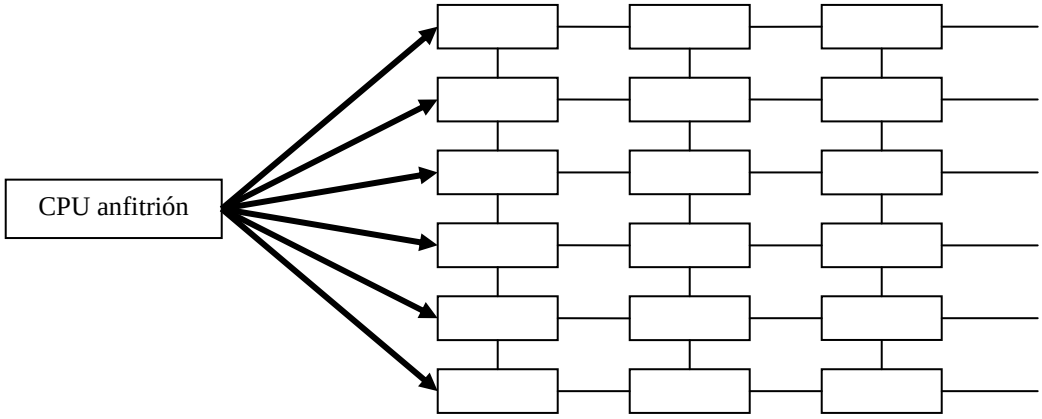


Figura 4.1. Representación de una computadora con arquitectura SIMD. **(Killough, John).**

La arquitectura que se basa en MIMD ha surgido recientemente como la arquitectura dominante. En este caso, cada CPU puede operar independientemente de los otros. La desventaja de esta propuesta es que ocasionalmente la sincronización debería ser ejecutada para garantizar que todos los procesadores están en un punto en la tarea.

La **figura 4.2** muestra un 'árbol de familia' básico para arquitecturas de computadoras paralelas propuesto por **Pancake (1996)**. El *modelo de control* establece cuantas instrucciones diferentes pueden ejecutarse simultáneamente. Como vemos aparecen los términos SIMD y MIMD como la primera parte de la clasificación de computadoras paralelas y, además, en esencia son el único factor distintivo entre éstas. El *modelo de memoria* indica cuantos procesadores pueden acceder directamente a una locación de memoria dada. El *modelo de programación* se refiere a restricciones en el número de

ejecutables que pueden participar en una ejecución paralela. En el modelo MPMD (programas múltiples, datos múltiples) todas las instrucciones que serán llevadas a cabo por los procesadores son combinadas dentro de un solo ejecutable.

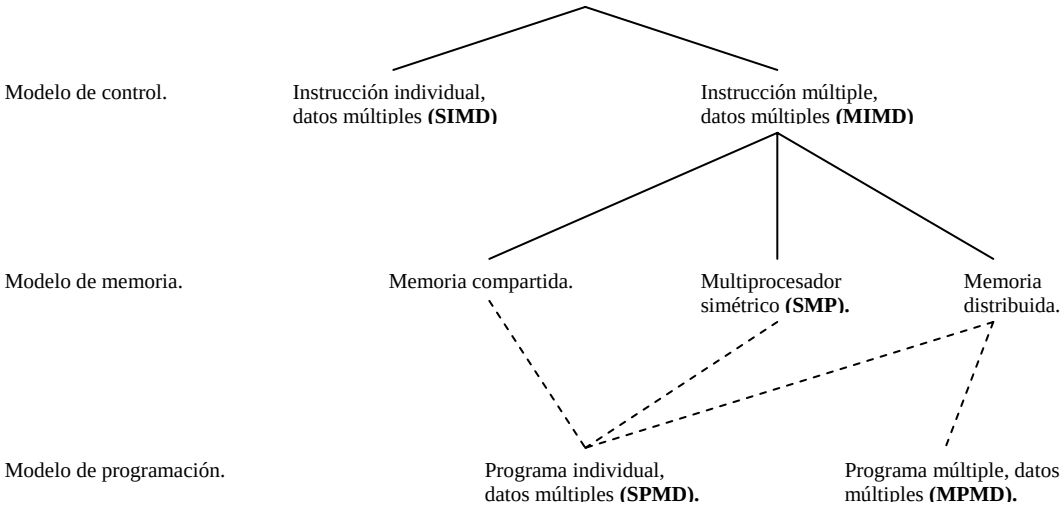


Figura 4.2. 'Genealogía' de sistemas de cómputo paralelos **(Pancake, Ch., 1996).**

En medio de las arquitecturas MIMD están tres configuraciones de memoria básicas: memoria compartida plana, memoria compartida en multi-nivel y memoria distribuida. La memoria compartida plana permite accesar a toda la memoria por cada uno de los CPU's. Una complicada configuración entre los CPU's y la memoria resulta de esta propuesta que, en general, puede ser costosa y no escalarse bien sobre unas pocas decenas de CPU's. Para superar este impedimento en la computación en paralelo masiva (más de 100 CPU's) se han desarrollado dos arquitecturas las cuales tienen la memoria del sistema total dividida, así que existe un estrecho enlace entre un CPU dado y una fracción de memoria del sistema en general. La memoria de multi-nivel depende del concepto de 'cache'. Para la propuesta de cache, software y/o hardware sofisticado mantienen la línea de la locación en curso de los datos, dentro de la memoria global del sistema. Si una pieza de datos referida por una instrucción de un CPU no está dentro de la memoria local, el sistema inicia una transferencia de datos de la locación en curso a la memoria local. Generalmente las

máquinas de memoria compartida están limitadas a alrededor de 16 procesadores debido a las limitaciones del ancho de banda en la memoria **(Killough, John E., 1993)**.

Las máquinas de memoria distribuida pueden contener cientos e incluso miles de procesadores, pero los accesos a la memoria no local pueden imponerse significativamente ante todo. Ya que los simuladores de yacimientos están basados en diferencias finitas o elemento finito, frecuentemente pueden compartir información entre los bloques de la malla. Los algoritmos de solución deben ser seleccionados e implementados con cuidado para minimizar las comunicaciones de los procesadores.

Las computadoras de memoria compartida están compuestas de múltiples unidades de procesamiento central (CPU's) compartiendo una memoria en común. Cada CPU de este tipo de máquinas usualmente es muy rápida y usa procesos de vector. En contraste, en las máquinas de memoria distribuida, cada CPU tiene su propia memoria y se comunica con los otros nodos (CPU's) a través de redes de alta velocidad. El número de CPU's en la máquinas de memoria compartida no puede ser incrementado más allá de un número específico, usualmente 4, 8 o 16; pero en las de memoria distribuida esto no tiene limitantes. Las máquinas de memoria distribuida, algunas veces llamadas procesadores paralelos masivos (MPP's), pueden estar hechas de cientos o incluso miles de CPU's. Cada CPU puede ser de un costo muy bajo y las redes de comunicación y el lenguaje de programación llegar a ser los mayores retos en este tipo de arquitectura. La colección de estaciones de trabajo (clusters) y, más recientemente, la colección de PC's con un switch de alta velocidad provee significantes ahorros y desempeños atractivos.

En las máquinas conocidas como multiprocesadoras simétricas (SMP), la palabra simétrica se refiere al hecho de que cada procesador puede recuperar datos almacenados en cualquier locación de memoria en la misma cantidad de tiempo. Estas máquinas se parecen a las de memoria compartida pero son más lentas y con menor poder de procesador **(Pancake, Ch., 1996)**. Es posible unir SMP's en grandes grupos y darles un poder de procesamiento mayor. La configuración resultante se comporta como una máquina de memoria distribuida, a

excepción de que cada nodo tiene múltiples procesadores compartiendo una memoria común.

El concepto de memoria compartida de ambas configuraciones, memoria plana y multi-nivel, tiene una ventaja sobre otras arquitecturas paralelas en la que la comunicación es ocultada del programador, si esto es requerido. Esto permite el empleo de software en programas seriales o sectorizados que estén en uso. La desventaja para las arquitecturas en multi-nivel es que la eficiencia máxima de la computadora no podrá lograrse a través del uso de comunicaciones automáticas entre hardware y software si es requerido un desempeño de paralelismo masivo.

El concepto de memoria distribuida ha llegado a ser el diseño más aceptado para las computadoras paralelas. Muchos constructores ofrecen estas ventajas, cada uno de esos sistemas ofrece el procesamiento paralelo a través de conectar muchas decenas o algunos cientos de CPU's usando algunos switch o redes interconectadas. El comportamiento de cada CPU varia desde unos pocos 'millones de operaciones de punto flotante por segundo' (MFLOPS) hasta más de 100 MFLOPS. El mejor comportamiento generalmente es logrado en aquel caso en el que se utiliza un código especializado o altamente sectorizado. En promedio, el desempeño de un CPU (o nodo) en programas reales es generalmente alrededor de 10 a 20 MFLOPS **(Killough, John E., 1993)**.

Las computadoras paralelas de memoria distribuida y los clusters son dos herramientas modernas en la industria de la computación y están atrayendo la atención más y más como aquellas utilizadas para resolver problemas prácticos que requieren de grandes cálculos. La simulación numérica de yacimientos requiere de grandes cálculos y tiene relativamente un alto grado de paralelismo y un modesto requerimiento de entrada y salida de datos, así que las computadoras paralelas de memoria distribuida son convenientes para este propósito. Pero la mayoría de los simuladores de yacimientos existentes están basados en computadoras secuénciales y deben ser reestructurados para hacer usadas a su máxima capacidad. La mayor dificultad es el desarrollar un solver de ecuaciones lineales en paralelo robusto y eficiente **(Zhiyuan, M.** y **Fengjiang, J., 1995)**.

La clave en el desempeño para cómputo paralelo en memoria distribuida es la habilidad para comunicar rápidamente información entre procesadores. La información que se tiene en la memoria local de un procesador es requerida por otros procesadores para desarrollar su tarea. El intercambio de mensajes se refiere a la acción de la comunicación de estos datos entre procesadores. Son importantes dos aspectos de comunicación: el retardo de información y el ancho de banda. Para que los datos sean transferidos de un procesador a otro se requiere que el camino de comunicación sea establecido entre los dos. Para el caso de los clusters, aún la longitud física del camino entre los CPU's puede aumentar al retardo de información. Generalmente, en las arquitecturas más comunes la demora es del orden de 5-300 micro-segundos. Una idea simple para reducir el retardo de información es reducir el número de comunicaciones entre procesadores empacando muchos mensajes en un solo envío o paquete. Si esto es realizado, entonces el factor limitante para las comunicaciones puede ser más cerrado, asociado con el ancho de banda interprocesadores o la velocidad a la que los datos son transferidos después de que la liga entre los procesadores ha sido establecida. El ancho de banda de las comunicaciones para las computadoras paralelas varía desde unos 2-400 millones de bytes de datos por segundo **(Killough, John E., 1993)**. El cluster es una forma del procesador en paralelo, este concepto intenta utilizar cada CPU distribuyendo ciclos a través de una organización para desempeñar el cómputo paralelo en medios dispersos. La configuración puede variar extensamente desde clusters homogéneos hasta clusters que involucran muchos tipos diferentes de máquinas. La distancia entre las máquinas también es variable, desde unas cuantos centímetros hasta cientos de metros e incluso miles de kilómetros a través de un continente gracias a las comunicaciones inalámbricas.

La idea básica atrás del comportamiento paralelo es lograr gran granularidad, es decir, si un problema es subdividido en varias tareas paralelas, las tareas deberían ser tan grandes como sea posible reducir la comunicación interprocesadores requerida, a una cantidad insignificante. Desafortunadamente, es igual de importante la idea del balance de carga la cuál requiere que cada una de las tareas sea del mismo tamaño o que haya más tareas que procesadores. Para el caso de unas pocas tareas grandes, por ejemplo, una por procesador, hay una gran dificultad en lograr un mismo tamaño de tarea. El intercambio para la tarea

más pequeña y las demás resulta en reducir la granularidad e incrementar el trabajo general. Obviamente, la clave para lograr un desempeño paralelo eficiente es el intercambio entre el efecto de la granularidad de las tareas y el balance de la carga.

Como resumen de las diferentes arquitecturas de computadoras paralelas se presenta la **tabla 4.1** donde se notan algunas características de estas arquitecturas.

Tabla 4.1. Resumen de Arquitecturas de Computadora s Paralelas
(Pancake, Ch., 1996).

Características.	**SIMD.**	**Memoria Compartida.**	**Memoria Distribuida.**	**Cluster SMP**
Memoria.	Compartida.	Compartida.	Distribuida.	Compartida por los procesadores en un nodo pero distribuida en un cluster.
Secuencia de Instrucción.	Un solo flujo.	Muchos flujos.	Muchos flujos.	Muchos flujos.
Número de Ejecutables.	Un ejecutable.	Un ejecutable (SPMD).	Un ejecutable o varios.	Un ejecutable o varios.
Lo más destacado.	Eficiente y relativamente fácil de programar.	Rápida y con gran memoria.	Versátil y rentable.	Versátil y rentable.
Lo menos destacado.	Se ajustan a pocos problemas.	Cara.	Difícil de usar eficientemente.	Es difícil usar muchos nodos eficientemente.
Mayor obstáculo de Programación.	Usar eficientemente las operaciones con arreglos.	Proteger el acceso a datos compartidos.	Minimizar el costo de comunicación.	Proteger el acceso a datos compartidos (dentro de un nodo).

4.1.1 Balance de Carga.

Dentro de un sistema de memoria distribuida, algunos de los nodos de trabajo pueden ser fuertemente cargados mientras otros tienen muy poco trabajo a desarrollar; esta diferencia en los niveles de carga sugiere que es posible mejorar el desempeño general del sistema por medio del reordenamiento de algunas de las tareas de manera que se optimice la utilización de los recursos disponibles. Esta optimización del trabajo total del sistema se conoce como *balance de carga*. El balance de carga, entonces, trabaja con el problema de determinar un patrón de asignación para las tareas mezcladas entre los procesadores de manera que el total del tiempo de ejecución del trabajo sea minimizado o, en otras palabras, la eficiencia de la paralelización sea maximizada.

La determinación de este patrón puede ser hecho de manera estática o dinámica. Por ejemplo, en la simulación las celdas de la malla que tengan cero en volumen de poros y sin requerimientos computacionales pueden ser asignadas estáticamente y permanecer fijos durante toda la simulación. Para esta situación estática, la transferencia de carga es hecha sólo una vez al inicio de la simulación desde el requerimiento del trabajo para que cada porción de la malla pueda ser determinada. El balance de carga dinámico es requerido para situaciones en las cuales la carga de trabajo cambia drásticamente durante el transcurso de una simulación. Esto puede ocurrir, por ejemplo, cuando el comportamiento de fases en un sistema cambia de una sola fase a un sistema de dos fases. El balanceo dinámico utiliza el estado en curso del sistema para ajustar constantemente el nivel de carga de cada nodo. Desafortunadamente la parte asociada con el balanceo dinámico de carga en un sistema de memoria distribuida debido al intercambio de mensajes puede ser sustancial. El balanceo dinámico entonces llega a ser el balance entre la maximización de la utilización de recursos y la minimización del envío de mensajes **(Killough, John E., 1993)**.

Tres estrategias para el balance de la carga de trabajo son reconocidas, estas son: iniciar al remitente, iniciar al receptor y un intercambio periódicamente. Las estrategias de remitente y receptor iniciado han sido estudiadas extensamente en la literatura especializada en ciencias de la computación y son considerados como los procedimientos más atractivos. En el caso del remitente iniciado, los procesadores fuertemente cargados buscan aquellos procesadores que están descargados hacia los cuales puedan transferir algunas de sus tareas en exceso. Lo opuesto ocurre en el caso de receptor iniciado. Los procesadores descargados buscan aquellos nodos congestionados en los cuales puedan adquirir tareas. Finalmente, en el intercambio periódico los nodos intercambian su información de carga para determinar como asignar las tareas en curso.

El balance de carga mantiene una continua dificultad para los modelos de simulación reales, como el número de procesos se aproxima a varios miles, los requerimientos para el balanceo de carga son elevados. Básicamente, el problema se reduce a la eliminación de situaciones en las cuales grandes números de procesadores estén vacíos en un periodo de tiempo significante esperando a que uno o algunos procesadores terminen sus procesos; al

mismo tiempo el envío de mensajes principal debe ser minimizado para mantenerse en niveles razonables.

Wheeler y Smith (1989) direccionan el no balanceo de la carga hacia una malla de forma irregular a través de una redistribución de las celdas activas entre los procesadores. Para situaciones en las que la solución del algoritmo dependa linealmente del número de bloques de malla, esta técnica funciona bien. Por otro lado, para algoritmos que son relacionados no linealmente a las sub-dimensiones asociadas con el procesador, una partición puede exacerbar fuertemente el problema.

Aunque prometedor, el trabajo fuerte puede ser hecho para mejor el balance de carga. Hay dos puntos importantes sin resolver, el número de bloques de malla a ser distribuidos los cuáles rara vez serán un múltiplo perfecto del número de procesadores y, aún si el número de bloques asignado a cada procesador fue igual, el número de iteraciones para cada bloque variará. Esto implica que para llevar a cabo un mejor balance de carga la distribución deba estar basada en los requerimientos de trabajo históricos para cada bloque de la malla. Algunas formas de algoritmos de remitente y receptor iniciados probablemente serán los mejores candidatos para esta solución.

4.1.2 Los PC-clusters y la Simulación de Yacimientos en Paralelo.

La Ingeniería de Yacimientos ha incrementado su dependencia del uso de PC’s para la recolección y el análisis de datos, procesamiento de datos geológicos y construcción de modelos y el análisis y visualización de los resultados de la simulación. Las operaciones numéricas intensivas de la simulación en un ambiente de PC son comunes. El día de hoy, la mayoría o si no es que todos los simuladores comerciales están disponibles en PC’s, esta tendencia de la industria del petróleo nos aleja de las supercomputadoras caras y enormes.

Un PC-cluster es una colección de computadoras de bajo costo (PC’s) conectadas por un switch y funciona como una sola fuente computacional. Aunque la idea de construir un cluster con máquinas comunes y corrientes es simple, el desempeño de un cluster puede ser

completamente impresionante. Esto fue posible porque el núcleo de trabajo computacional (el procesador) fue mejorado rápidamente, más notablemente en los últimos 5 años **(Zhiyuan, M.** y **Fengjiang, J., 1995)**.

4.2 Métrica para un Desempeño Paralelo.

La única buena métrica es comparar el desempeño del programa paralelo con la mejor corrida de un programa serial en sólo un CPU **(Killough, John E., 1993)**. Esto no siempre será posible debido a la falta de facilidades tales como memoria. Un esfuerzo sería el hacer comparación de los resultados con un comportamiento serial a través de la extrapolación. La razón de esta idea es simple, frecuentemente consideramos el comportamiento paralelo como la aceleración, definiendo aceleración como nos indica la ley de Amdahl.

La *Ley de Amdahl* es un simple modelo de comportamiento que muestra que el comportamiento paralelo de un simulador, depende fuertemente de la fracción de los cálculos que son ejecutados en paralelo. Para el procesamiento en paralelo, asumiendo procesador no líder, queda entonces declarada como sigue:

$$S = \frac{1}{1 - f_p + \frac{f_p}{N}} \tag{4.1}$$

Donde *S* es la 'aceleración' relativa al proceso serial, f_p es la fracción de los cálculos que se trabajaron en paralelo quedando definida de la siguiente forma:

$$f_p = \frac{\text{tiempo de código potencialmente paralelo}}{\text{tiempo del código completo}}$$

y *N* es el total del número de procesadores **(Hemanth-Kumar K. y Young, L.C. 1996), (Pancake, Ch., 1996)**.

Para que asimilemos mejor el concepto de f_p en la **figura 4.3** se muestra de manera gráfica.

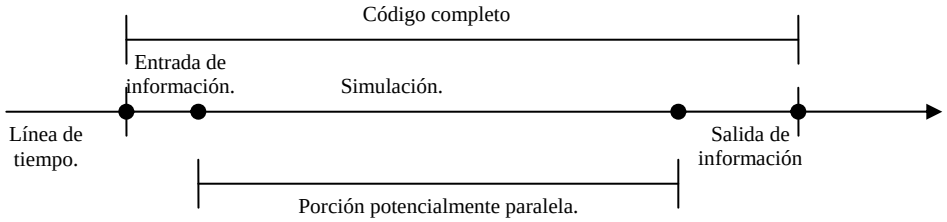


Figura 4.3. Regulación del tiempo de la línea base del programa para estimar el desempeño paralelo: tiempos de código completo vs potencialmente paralelo. Cada punto grueso representa una llamada a la subrutina de tiempo **(Pancake, Ch., 1996).**

Pero ¿La aceleración con respecto a qué? Los algoritmos en paralelo generalmente son menos eficientes que sus complementos seriales; entonces, la comparación del comportamiento paralelo con el mismo código paralelo corriendo en una máquina serial es engañosa. Similarmente, reportar en términos de MFLOPS puede ser también engañoso ya que el algoritmo más ineficiente puede lograr valores de MFLOPS extremadamente altos. Aunque el escalamiento del tamaño del problema frecuentemente guía hacía el mejoramiento del desempeño paralelo, el comportamiento del constante tamaño del problema y grandes números de procesadores puede dar una idea de la ubicación de los cuellos de botella en las comunicaciones, las cuales de otro modo serían disimuladas. En suma, la representación gráfica para un programa en paralelo sin considerar su comportamiento podría disfrazar el hecho que el desempeño paralelo fue verdaderamente tan pobre que utilizando simplemente un proceso serial se lograrían los mismos resultados. En resumen, la meta de la programación en paralelo es obtener resultados más rápidamente o económicamente o para problemas muy grandes que pudieran haber sido logrados razonablemente con una programación convencional.

4.3 Programación en Paralelo.

La programación en paralelo consiste en la división de un problema, presentando como una serie de datos o serie de acciones, entre múltiples elementos de procesamiento que operan

simultáneamente **(Ortega A., J. L. 2003)**. En otras palabras, consiste en la coordinación de recursos computacionales que trabajan simultáneamente para alcanzar un objetivo en común **(Ortega A., J L., 1998), (Cedillo T., U. y Orantes L., R. 2005).**

El potencial del paralelismo descansa en la partición de un problema grande en relación a su complejidad o carga de trabajo. Esta partición es necesaria para dividir este gran problema en subproblemas más sencillos, más fáciles de asimilar y que se puedan trabajar por separado en una forma más práctica. Esto es clave en estos sistemas ya que la partición permite no sólo crear por separado los componentes de software, sino también, ejecutarlos simultáneamente **(Ortega A., J. L. 2003).**

Las ventajas de la programación en paralelo pueden enunciarse en dos principales; estas son: realizar tareas a una escala que para otros sistemas de cómputo sea imposible o no realizable y optimizar el tiempo de ejecución de los programas, especialmente si son robustos **(Cedillo T., U. y Orantes L., R. 2005).**

En teoría, el paralelismo es algo tan sencillo que sólo consiste en repartir un solo problema en muchos procesadores; lo cuál es una buena forma de vencer las restricciones que existen en las máquinas de un solo procesador, y además de brindar soluciones más rápidas, se pueden resolver problemas más grandes y más complejos que inclusive a la mitad de su proceso ya exceden la capacidad de memoria de un solo procesador. Sin embargo en la práctica, realizar el paralelismo implica un gran esfuerzo; pues el programador debe pensar nuevas maneras en su aplicación e inclusive puede presentarse el caso en que tenga que reescribir todo el código serial que ya tenía y sumado a esto, las técnicas para depurar fallas y afinar el desempeño de los programas seriales no siempre son aplicables en la programación en paralelo y cuando lo son, no se realizan con la misma facilidad. Aún más, nada ni nadie nos garantiza que los resultados que se obtengan sean correctos o que los tiempos en la corrida siempre serán menores que los que se tenían en programación serial; por lo que se puede trabajar durante largo tiempo paralelizando una aplicación y al final descubrir que no funciona como esperábamos y que es mejor su desempeño serial.

Existen, sin embargo, algunas precondiciones para el paralelismo que nos ayudan a conocer si es o no viable realizar el esfuerzo de la paralelización; estas precondiciones se enuncian a continuación en función de tres factores que las determinan.

El primer factor se refiere a la frecuencia de uso de la aplicación antes de que se requiera realizar algunos cambios. Si la respuesta es de miles de veces entre revisiones, esta es una aplicación que talvez merece el esfuerzo de mejorar su desempeño; sin embargo, si es una aplicación que necesita ser modificada con gran frecuencia, no permitirá compensar el tiempo empleado en las mejoras.

El segundo factor es el tiempo que actualmente se necesita para ejecutar la aplicación. Por ejemplo, si se necesitan días para obtener resultados, se puede optimizar fuertemente la productividad reduciendo este tiempo a una fracción. Pero si el tiempo que se emplea actualmente es de solo unos minutos, puede que la ganancia de tiempo obtenida sea muy baja contra el esfuerzo requerido para realizar el proceso de la paralelización. Este factor puede ser muy relativo, ya que si, por ejemplo, se desea mejorar un sistema de administración de emergencia en tiempo real, una optimización de segundos puede ser muy buena **(Cedillo T., U. y Orantes L., R. 2005).**

El tercer factor que se tiene se refiere a la satisfacción que se tiene con la complejidad y la resolución de los resultados, es decir, si, por ejemplo, debido a la capacidad de memoria de una máquina serial debemos trabajar con una malla de simulación gruesa en lugar de utilizar una malla fina, el trabajar con computadoras paralelas nos ayuda a vencer en gran medida estas restricciones.

La **figura 4.4** nos muestra un esquema donde se observa como estos tres objetivos determinan las precondiciones de paralelización, así como qué se puede ganar con este procedimiento.

Según la experiencia de algunos científicos e ingenieros, las necesidades deben alcanzar por lo menos un color blanco en el espectro de la **figura 4.4** antes de considerar invertir un

esfuerzo en paralelizar una aplicación. Por otro lado, un solo color negro puede interpretarse como una indicación de que el rendimiento no amerita mucho esfuerzo de paralelización. Además, incluso tres blancos no garantizan que el paralelismo dará buenos resultados, simplemente indica que la aplicación tiene potencial de paralelización **(Pancake, Ch., 1996), (Cedillo T., U. y Orantes L., R. 2005).**

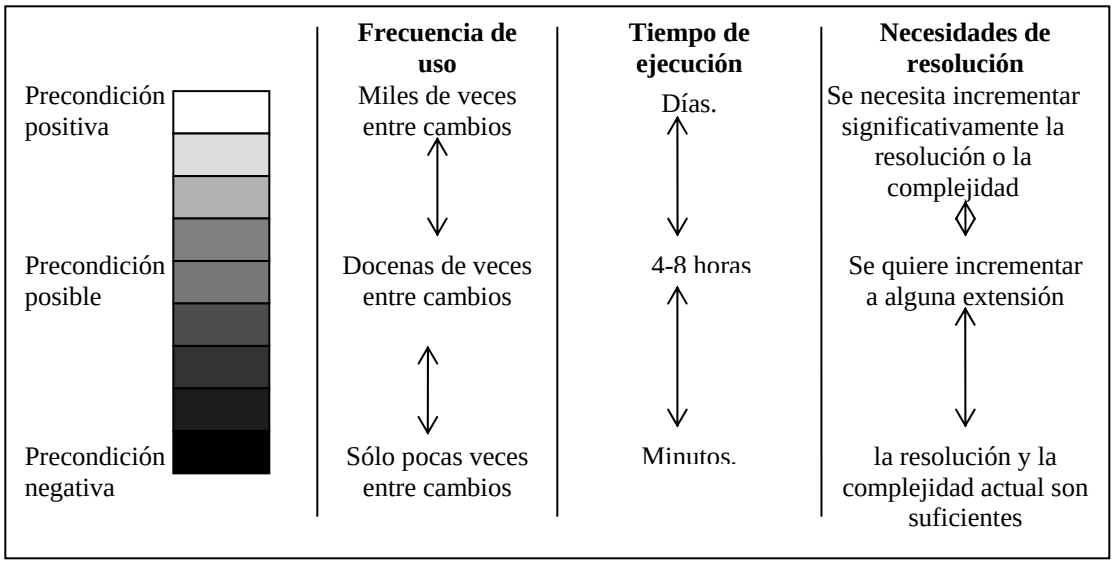


Figura 4.4. Precondiciones para la Paralelización. **(Pancake, Ch., 1996).**

4.3.1 Tipos de Paralelismo.

Para ejemplificar los tipos de paralelismo, consideraremos un problema de mapeo sísmico en las **figuras 4.5** y **4.6**, un problema de un modelo tridimensional de la atmósfera para la **figura 4.7** y un problema de difusión de contaminantes a través del agua subterránea para la **figura 4.8**.

Paralelismo Perfecto. **Figura 4.5.** Los cálculos de cada serie son totalmente independientes y se pueden procesar múltiples series de datos al mismo tiempo, es decir, las imágenes pueden ser procesadas en máquinas independientes que tengan copia de la aplicación.

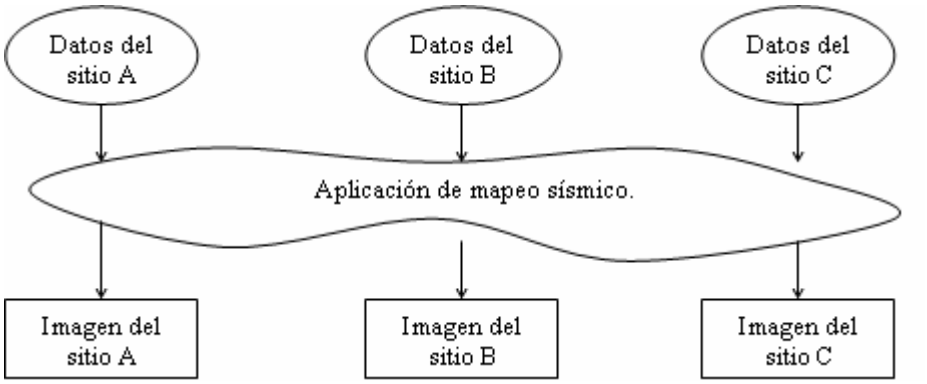


Figura 4.5. Esquema de Paralelismo Perfecto **(Pancake, Ch., 1996).**

Paralelismo de Tubería. **Figura 4.6.** Los datos siguen una secuencia de tubería al pasar de una etapa computacional a otra. En este modelo los resultados son pasados de una sola manera a través de la tubería. La simulación del siguiente paso de tiempo, en nuestro problema de ejemplo, no requiere información de la generación del volumen 3D o de la visualización gráfica. El inicio de todas las secuencias es comenzado conforme los datos de entrada se vuelven disponibles, de tal forma que el mejoramiento del desempeño general dependerá del número de pasos de tiempo que serán procesados una vez que todos los puntos de la tubería están activos. Este tipo de paralelismo implica problemas tales como que, si las etapas no son completamente equivalentes, las más rápidas saturarán a las más lentas, acabando primero. Una manera de solucionar esto puede ser realizando un balance de carga, pero es una tarea muy difícil, por lo que el paralelismo de tubería no es tan sencillo como el paralelismo perfecto.

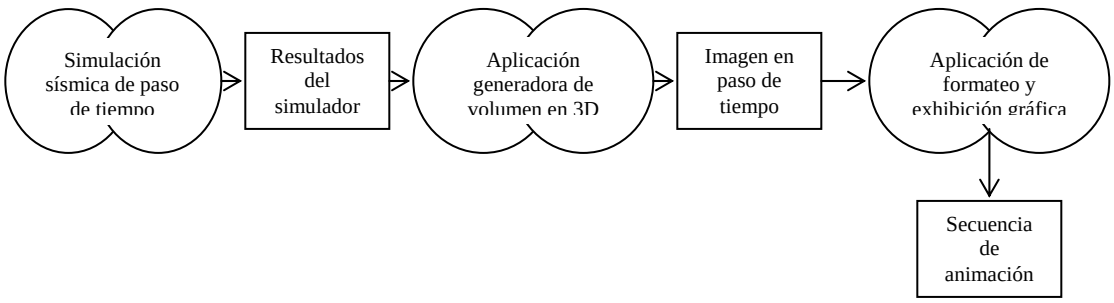


Figura 4.6. Esquema de Paralelismo en Tubería **(Pancake, Ch., 1996).**

Paralelismo Totalmente Sincrónico. **Figura 4.7.** En este tipo, cada cálculo es aplicado simultáneamente a todos los datos. La clave de esto es que los cálculos o las decisiones futuras dependen de los resultados de todos los cálculos anteriores. Por lo regular, no hay tantos procesadores como para trabajar al mismo tiempo todos los grupos de datos, así cada procesador trabaja con más de un grupo de datos. Si estos grupos no son homogéneos, la carga de trabajo variará en cada procesador. Para el ejemplo correspondiente, un disturbio en el estrato que está hasta arriba de la atmósfera modifica primero a los datos de los estratos superiores, ya que son los más cercanos, mientras que los estratos inferiores permanecen sin cambio. Esta variación espacial significa que si cada procesador realiza los cálculos de un grupo de datos de manera horizontal, sólo uno o dos procesadores estarán realizando un trabajo intensivo desde el inicio y la sincronización evitará que los otros procesadores realicen los siguientes cálculos hasta que los procesadores ocupados terminen. Sin embargo, si se aplican cálculos a regiones verticales, el trabajo puede estar uniformemente distribuido al inicio de la corrida, pero cambiara en los siguientes pasos de tiempo cuándo los cálculos difieran en dirección horizontal. En consecuencia, este tipo de paralelismo requiere más esfuerzo del programador que el paralelismo en tubería.

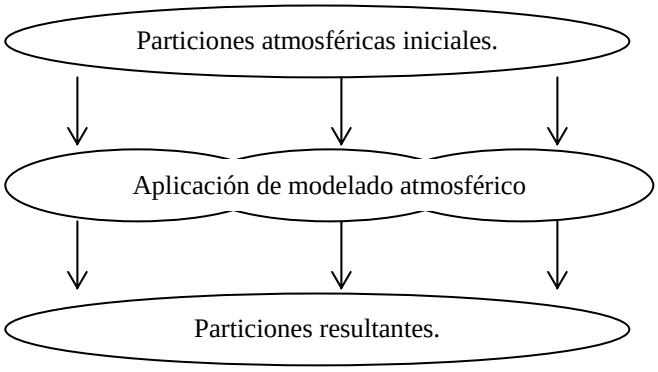


Figura 4.7. Esquema de Paralelismo Totalmente Sincrónico. **(Pancake, Ch., 1996).**

Paralelismo Pobremente Sincrónico. **Figura 4.8.** En este tipo de paralelismo, cuándo cada paso de tiempo termina, los procesadores que ya terminaron su trabajo deben esperar a que los demás terminen para compartir los resultados parciales y comenzar el siguiente paso de

tiempo. La característica clave de este estilo es que cada procesador realiza una parte del problema, intercambiando información intermitentemente. La necesidad de intercambiar información entre los procesadores requiere de pruebas para que un procesador pueda determinar cuándo los datos de los demás procesadores están listos y pueda evitar sobrescribir valores que no se han usado aún. Con este tipo de paralelismo es muy difícil hacer un balance de carga adecuado ya que ésta varía temporal y espacialmente. Es por esto que este tipo de paralelismo es el más difícil. Para el ejemplo propuesto, sólo las particiones de agua cercanas a la fuente de contaminación son afectadas, pero conforme avanza el tiempo los contaminantes se esparcen, formando áreas irregulares de concentración; la carga de trabajo depende de la estructura geofísica y de la cantidad de contaminante, así que variará dramáticamente de una partición a otra, el paralelismo es aplicado dividiendo el trabajo a lo largo de muchos procesadores a cada paso de tiempo; durante los primeros pasos de tiempo cada procesador puede aplicar cálculos a sólo unas cuantas particiones y la duración de los cálculos será breve debido a que las concentraciones son bajas, pero después, conforme las concentraciones van creciendo y afectando otras particiones, un solo procesador puede realizar muchos más cálculos en muchas más particiones a cada paso.

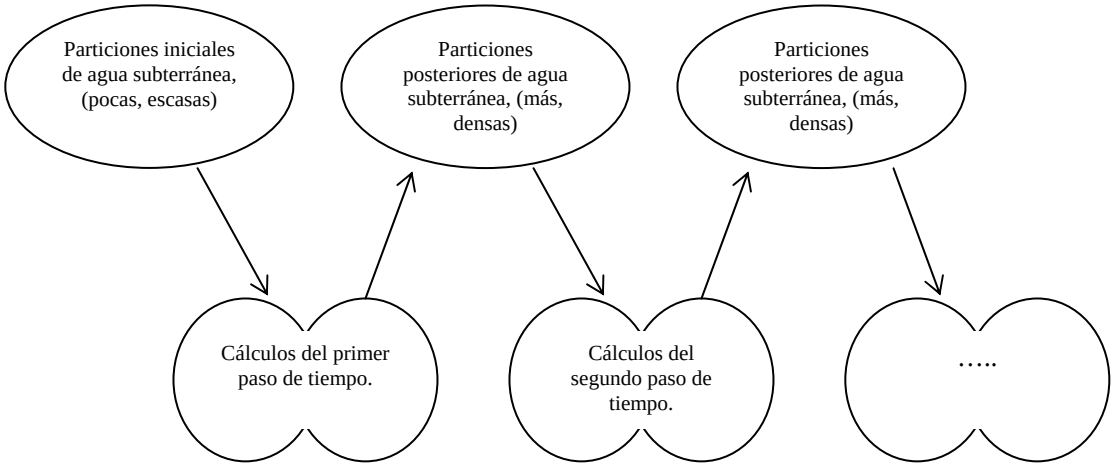


Figura 4.8. Esquema de Paralelismo Pobremente Sincrónico. **(Pancake, Ch., 1996).**

Estos cuatro tipos de paralelismo son los que **Pancake (1996)** propone en su trabajo, pero existen otras propuestas referentes a este mismo tema en la literatura especializada.

4.4 Comunicación entre Procesadores.

Hay tres maneras distintas de generar códigos paralelos; la paralelización semi-automática, la programación paralela de datos y la programación por envío de mensajes de comunicación, que fue la que se utilizó en este trabajo por lo que sólo nos enfocaremos en su explicación.

El envío de mensajes es un modelo de programación en paralelo que se utiliza principalmente en máquinas de memoria distribuida, este modelo puede ser visto como una serie de instrucciones ejecutándose en paralelo sobre diferentes procesadores que intercambian mensajes durante la ejecución para enviar o recibir ciertos valores. Cada procesador posee una serie de datos privados y, si un procesador necesita datos de otro, le tiene que mandar un mensaje de solicitud a este para que se los mande **(Cedillo T., U. y Orantes L., R. 2005)**.

Este modelo de comunicación propone una serie de procesadores que sólo tienen memoria local pero que aún así, son capaces de comunicarse con otros procesadores enviando y recibiendo mensajes entre ellos. Un factor clave en este modelo es que la transferencia de datos de la memoria local de un procesador a la memoria local de otro requiere operaciones que serán realizadas por ambos procesadores. Estas operaciones conocidas como ‘llamadas de comunicación’ son insertadas explícitamente dentro del código fuente.

Debido a los progresos que se consiguieron sobre la programación de aplicaciones bajo el modelo de envío de mensajes, en los 1990´s los fabricantes de máquinas paralelas implementaban su propio ambiente de programación, derivando en un software no portátil. Esto llevó a que se desarrollaran diferentes librerías de envío de mensajes que permiten crear los programas paralelos, implementando así un ambiente de intercambio de mensajes portátil de manera eficiente; por lo que se consideró necesario definir la sintaxis y la semántica de un conjunto de rutinas estándar que serían eficientemente implementadas a una gran variedad de computadoras **(Gordillo R., J. L., 2001).** Dos de las librerías más usadas son la Máquina Virtual Paralela (PVM Parallel Virtual Machine) y la Interfaz de

Envío de Mensajes (MPI Message Passing Interface) las cuáles cumplen con este conjunto de rutinas estándar. En este trabajo se utilizó la librería MPI para generar las comunicaciones en el programa paralelo.

Las principales características y ventajas de MPI son **(Gropp, W., Lusk, E. y Skejellum, A. 2000), (Gordillo R., J. L., 2001), (Cedillo T., U. y Orantes L., R. 2005):**

- No es un lenguaje, sino una librería. Ésta especifica los nombres, secuencias de llamado y resultados de subrutinas que serán llamadas de programas en Fortran, y funciones que serán llamadas de programas en C. los programas que los usuarios escriben en Fortran, C y C++ son compilados con compiladores ordinarios y ligados con la librería MPI.
- Es un estándar, no una implementación en particular. Todos los fabricantes de máquinas ofrecen una implementación de MPI para sus máquinas, las implementaciones, públicas y gratuitas, disponibles deben ser capaces de correr en todas las implementaciones de MPI sin cambio alguno.
- Cumple con el modelo de envío de mensajes.
- Su estructura permite transportar fácilmente códigos existentes a la programación en paralelo y escribir nuevos, sin tener que aprender una nueva serie de conceptos fundamentales.
- Maneja eficientemente los buffers de los mensajes. Los datos son enviados y recibidos por medio de estructuras de datos definidas por el usuario y no usando estructuras definidas internamente por las librerías de comunicación, como ocurría con ambientes paralelos anteriores.
- Permite la programación eficiente, tanto de máquinas paralelas como de grupos de estaciones de trabajo conectadas en red.
- Es totalmente portátil. El código de un programa MPI puede ser compilado y ejecutado en cualquier arquitectura sin necesidad de modificación alguna.
- Está formalmente especificado. Existe un documento oficial que contiene todas las características que debe contener una implementación estándar MPI.

Aunque existen más de 100 instrucciones para el manejo de la librería MPI, es posible crear un programa paralelo manipulando únicamente 6 de estas instrucciones, las cuales son básicas y necesarias (al menos las 4 primeras), pues sin ellas no es posible paralelizar un código. Estas son: *Init* que se refiere a la inicialización del ambiente paralelo, *Finallize* refiriéndose al final del ambiente de paralelismo, *Rank* es referido al rango de trabajo, es decir, cada procesador tiene como nombre un número entero asignado al cual se le llama 'rango' y es de suma importancia para definir las comunicaciones, pues en base a este número se sabe que procesador envía o recibe datos e incluso que procesador realiza tareas y cuál no, *Size* nos indica cuál es el tamaño del ambiente paralelo, es decir, cuantos procesadores son los que están trabajando en el problema en cuestión, *Send* este comando es utilizado para enviar información de un procesador a otro, y *Recv* el cual se utiliza como instrucción de recepción de datos. Cabe mencionar que estos últimos dos comandos son de comunicación puntal, es decir, por cada send existe un recv y cada procesador que utilice un send conoce cuál es el procesador que tiene el recv y viceversa

4.5 Áreas para la Aplicación de la Programación en Paralelo en la Ingeniería Petrolera.

Como mínimo existen tres áreas para la aplicación de la programación en paralelo dentro de la Ingeniería Petrolera. Estas incluyen la Simulación de Yacimientos, la interpretación de registros de pozos y el análisis de las pruebas de pozos.

La caracterización de yacimientos requiere el análisis simultáneo de muchas mediciones de los registros de pozos, p.e. acústico, nuclear y de resistividad. Una de las técnicas de interpretación disponible requiere un proceso de inversión en el cuál la herramienta responde a un grupo dado de parámetros del yacimiento, que son simulados y comparados nuevamente a las mediciones del registro actual. Para ajustar los parámetros del yacimiento un juego de datos actuales y uno simulado es obtenido; este proceso normalmente es llevado en programas seriales en nivel por nivel y puede consumir un monto considerable de tiempo en el proceso de cómputo. Para desempeñar muchos niveles en paralelo el

proceso general puede ser acelerado significativamente si cada cálculo es independiente de los otros niveles. Los avances en el análisis de los registros de pozos han guiado al uso de técnicas más complicadas en las cuales arreglos de registros son utilizados; el análisis para cada muestra requiere un complejo proceso de inversión en el cuál muchos niveles son analizados simultáneamente. Sin el uso de la programación en paralelo, el uso de estos procesos son demasiado largos y cansados, afortunadamente estos procesos tuvieron gran granularidad con pocas comunicaciones generales y llegaron a ser eficientes en el ambiente del paralelismo.

El análisis de pruebas de pozos también pone a prueba a la programación en paralelo, aunque las técnicas que se utilizan pueden ser manipuladas con facilidad por supercomputadoras, la programación en paralelo vendrá a ser más común y las técnicas más sofisticadas podrán ser utilizadas; p.e. el juego automático de los datos de pruebas de pozos usando simuladores de pozos individuales, es posible.

Es así como en el futuro, la Simulación de Yacimientos continuará liderando los requerimientos computacionales para las aplicaciones en Ingeniería Petrolera.

4.6 La Simulación de Yacimientos en Paralelo.

Desde las instancias iniciales de la Simulación de Yacimientos, los modelos han continuado agotando las capacidades de las mejores computadoras, se han requeridos grandes modelos numéricos y visuales para adaptarlos a los fenómenos que ocurren en el yacimiento. Empezando a mediados de 1970, la introducción de las supercomputadoras a través de la vectorización cambió completamente la propuesta que había sido tomada para desarrollar modelos numéricos para la simulación. Aunque estas computadoras avanzaron significativamente en la velocidad a la cual los procesos pueden ser hechos, la disponibilidad de la influencia del poder computacional a través de la vectorización guía a una reorganización significante y un replanteamiento de los modelos existentes. El mejoramiento de la tecnología de la computación ha hecho posible atacar los problemas en

la simulación de yacimientos aun aquellos que no se podían manejar hace 10 o 15 años. La necesidad de mejores desempeños para los modelos de simulación de yacimientos petroleros es dominada por la física de los procesos existentes que han sido investigados **(Killough, John E., 1993)**.

Simulaciones completas de campos de aceite negro con grandes números de bloques de malla han sido usadas en la administración de yacimientos. Aun con 50000 bloques, nos permiten sólo uno o dos bloques entre una pareja de pozos adyacentes, así, hay una tendencia continua a incluir demasiados detalles en los modelos de simulación usando (1) mallas muy finas para resolver frentes de flujo, (2) mallas finas para permitir más precisión en la representación de la heterogeneidad y (3) simulaciones composicionales para obtener una representación más detallada de los fluidos y los procesos de recuperación.

El resolver típicamente para frentes de flujo requiere de 10 a 15 bloques entre una pareja de pozos adyacentes. Para muchas simulaciones de campos completos esto puede resultar en más de 1 millón de bloques de malla. La representación más precisa de los fluidos es manejada por los procesos de recuperación mejorada y, debido a que se ha presentado cada día el descubrimiento de yacimientos de gas y condensado, se necesita de simuladores composicionales para la representación de estos fluidos. En muchos casos se utilizan más de 1 millón de celdas de simulación en una simulación composicional, comparada con una simulación de aceite negro de 50000 celdas, un problema composicional de este tamaño representa un incremento del orden de dos a tres veces en los recursos computacionales, p.e. memoria y tiempo de simulación. El procesamiento en paralelo de estos problemas es la manera más viable para acercarse a una solución práctica de estos problemas **(Killough, John E., 1993)**.

El comportamiento de los fluidos multifásicos en el medio poroso está gobernado por una gran variedad de fenómenos complicados; en un problema típico de flujo en el subsuelo, un proceso se identifica en un rango de escalas desde unos milímetros hasta la extensión total del campo, en orden de kilómetros. Los efectos capilares son importantes en escalas

pequeñas y los efectos macroscópicos como el avance del frente de inyección es importante para las escalas grandes.

La pregunta de cómo la escala de laboratorio puede ser extrapolada a la escala de campo está aun abierta. Un esquema numérico/computacional que puede incorporar una descripción de las heterogeneidades locales tan finamente como sea posible, dado el estado del arte de la computación, representa una herramienta muy valiosa en el desarrollo de los esquemas de extrapolación. Muchas áreas son puntualizadas por **Culham (1992)** en las cuales la simulación de yacimientos debe ampliarse para cubrir las necesidades del futuro. Estas áreas son:

- La representación de las heterogeneidades del yacimiento, detalladas, disponibles para la geoestadística y otros datos.
- La inclusión de física detallada, tal como la caracterización compleja de fluidos.
- Mallas finamente detalladas para la precisión numérica.
- Mallas no estructuradas y refinamiento local de malla.

Todas estas áreas guían a un aumento en los requerimientos computacionales para la simulación de yacimientos las cuales pueden ser cubiertas en buena medida por programación en paralelo.

4.6.1 Beneficios de la Simulación de Yacimientos en Paralelo.

Fundamentalmente, el objetivo detrás del esfuerzo de la paralelización es conducir el tiempo empleado en las corridas de las simulaciones a que sea mínimo. Las necesidades y beneficios detrás de tales esfuerzos son numerosos **(Habiballah W.A. et al 2003)**:

- *Simulación con modelos de millones de celdas.* Relativamente la alta resolución de modelos con millones de celdas provee el realismo necesario en el modelado de yacimientos grandes o altamente heterogéneos. Un tamaño de malla de 50 metros es común en el modelado de yacimientos de pequeña a mediana escala. El uso de este mismo tamaño en campos grandes fácilmente resultaría en modelos con más de un millón de celdas. La alta resolución en los bloques de malla también capturará los

detalles de heterogeneidades geológicas, proveyéndonos con un mejor entendimiento u mejor modelado del flujo de fluidos.

- *Complejidad del modelo.* Los modelos composicionales y de doble porosidad doble permeabilidad, por la naturaleza de las formulaciones matemáticas requieren de cálculos numéricos muy extensos. La construcción del Jacobiano y el método de solución (solver) requiere de más tiempo de proceso conforme el número de incógnitas sea mayor.
- *Grandes tiempos de simulación.* Los modelos de simulación para campos viejos también se beneficiarían con la simulación en paralelo. Tales modelos requerirían un tiempo de simulación extenso en el modo serial, pero serían más manejables con simulación paralela.
- *Modelado estocástico.* Las aproximaciones estadísticas para la construcción del modelo geológico resultan en numerosas construcciones de modelos. La simulación en paralelo puede procesar estos modelos con un mínimo o sin escalamiento y proveernos con capacidades para utilizar mejor las herramientas estocásticas y definir y evaluar los riesgos e incertidumbre en las operaciones de campo.
- *Ajuste de historia automático y semiautomático.* La acción de utilizar herramientas para el ajuste de historia depende fuertemente de la velocidad de corrida de una simulación y la evaluación de numerosos modelos. La simulación de yacimientos en paralelo es un complemento que provee los desempeños de simulación necesarios.
- *Mejora al tiempo de respuesta.* Es lo más largo en un estudio de simulación, toma la mayor requerimiento de la fuente y llega a ser lo más costoso. Las corridas de simulación que toman días o semanas para completarse pueden ser mejoradas a una pocas horas si la tecnología de la programación en paralelo es utilizada apropiadamente.
- *Visualización de la simulación en tiempo real.* El acoplamiento de la simulación y la visualización puede ser muy usado y productivo en la administración de los campos. En tales situaciones, los equipos activos pueden investigar el impacto de varias estrategias de producción, también puede localizar los puntos de perforación para los nuevos pozos y planear las trayectorias de los pozos multilaterales. El

desempeño del pozo o del campo puede ser evaluado inmediatamente con tal de que el tiempo de simulación sea reducido al mínimo.

4.6.2 Propuestas para la Simulación de Yacimientos en Paralelo.

La literatura tiene extensos reportes sobre cómo aproximar la paralelización en la simulación de yacimientos. La principal diferencia cae en la metodología seguida en la creación del solver. La solución del solver puede estar clasificada en tres categorías principales: descomposición de dominio, multi-grid y dominio sencillo **(Habiballah W. A. 2003)**.

- Propuesta de *Descomposición de Dominio* (DD). Está basada en la partición de la malla computacional en diferentes subdominios traslapados, cada subdominio es resuelto por separado en diferentes procesadores seguido por la solución del límite y la solución del vértice. La solución es repetida hasta la convergencia y es dependiente de la extensión del traslape.
- Propuesta *Multi-Grid* (MG). Basada en la solución del problema de interés por el trabajo en secuencia de mallas con finuras variables. El error asociado con frecuencias diferentes es reducido en diferentes niveles de malla. La convergencia depende de la relajación, operaciones de restricción, interpolación y la aproximación de mallas bastas. Propuestas combinadas basadas en la DD y MG también han sido empleadas.
- Propuesta de *Dominio Sencillo* (SD). Tiene un fundamento histórico en el procesamiento vectorial y las computadoras con arquitectura de Instrucción Sencilla en Múltiples Datos (SIMD). En esta propuesta, el solver trabaja en un problema sencillo (resuelve una matriz sencilla).

Las propuestas con partición (DD y MG) son muy populares y tienen la ventaja de menor trabajo y menor requerimiento de memoria al compararlas con las propuestas SD. De cualquier forma, las desventajas de DD y MG son: la convergencia depende en el número de subdominios, los patrones de inyección o producción en un subdominio causa convergencia y desbalance de carga entre todos los procesadores, la dificultad en la

predicción de la escalabilidad como cambios en los subdominios puede impactar la convergencia, la optimización de la carga de los procesadores necesita ser hecha automática o manualmente. Las ventajas de las propuestas SD son: simplicidad en los métodos de solución, la predicación del desempeño ya que sólo hay un dominio a resolver, la reproducción de los resultados (el comportamiento de la convergencia no cambia) bajo un basto número de procesadores, buen balance de carga (sólo hay un paso global), la comunicación entre procesadores es regular y finalmente, menores corridas de práctica para los ingenieros de simulación ya que la paralelización es transparente para el usuario y el simulador se comporta como un simulador serial.

CAPÍTULO 5. PARALELIZACIÓN DE UN SIMULADOR NUMÉRICO.

En este capítulo se describe como es que se llevaron a cabo los distintos esquemas de paralelización del simulador utilizado, el cuál es un simulador semi-implícito de aceite negro monofásico y en 3D **(Hernández y Del Valle 2005)** programado en Fortran 90 el cuál brinda resultados confiables pues ya ha sido probado; así como las características que diferencian a cada uno de estos. Este simulador fue adaptado a la doble porosidad, antes de realizar el trabajo en paralelo, validando esta modificación al comparar los resultados obtenidos, de la forma radial, contra los resultados de la ecuación analítica de Warren & Root, que se muestra en el apéndice B obteniendo la gráfica de la **figura 5.1** de esta comparación. La forma cartesiana, que es la que finalmente se utilizó, se consideró correcta cuándo los datos de salida, correspondientes a un procedimiento de simulación con un arreglo de los pozos en forma simétrica (**figura 5.2**), fueron simétricos.

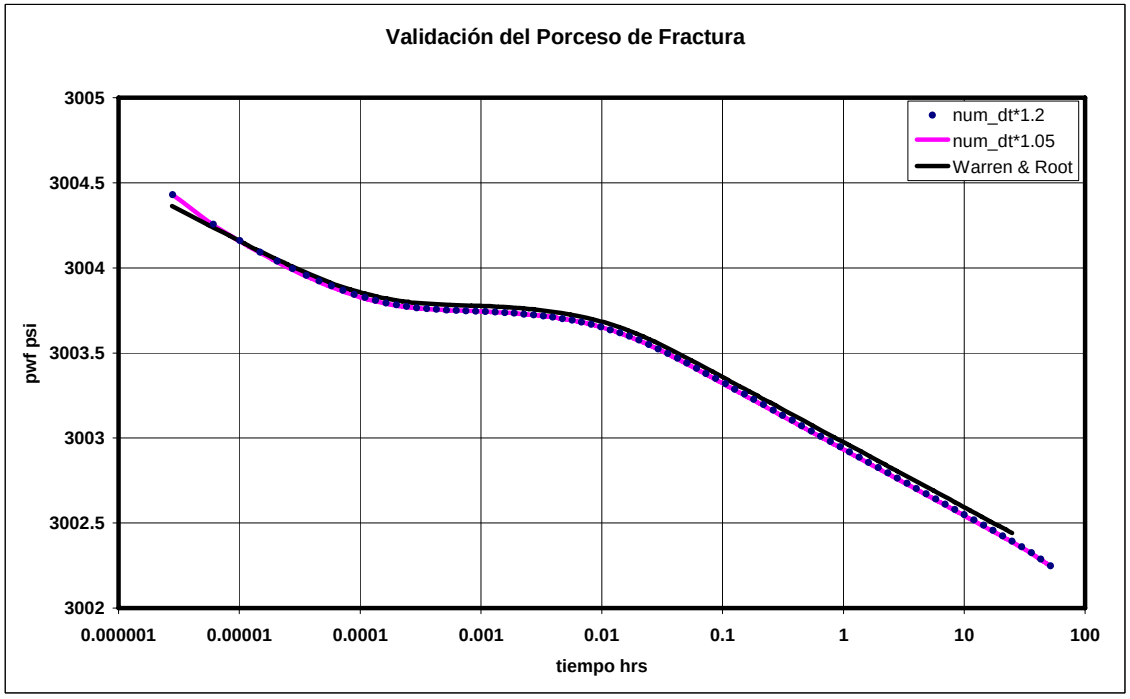


Figura 5.1. Validación del Simulador adaptado a la Doble Porosidad.

En la gráfica se observa el mismo comportamiento clásico de una formación fracturada en las tres líneas; en donde la primera parte es el aporte de la fractura al pozo, después un periodo de transición de la matriz a la fractura y por último el aporte de todo el sistema. Existe una separación entre la línea correspondiente al proceso analítico de

Warren y Root y las correspondientes a nuestro simulador numérico, sin embargo como es menor de 0.1 [psi] se consideró como correcto.

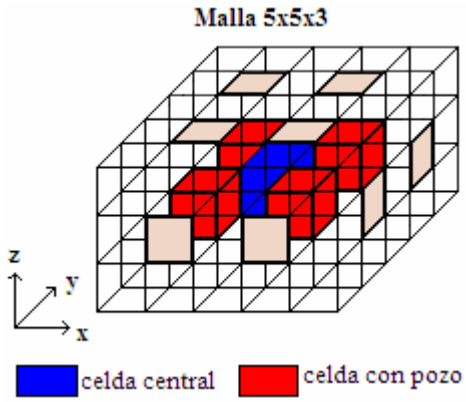


Figura 5.2. Arreglo simétrico de pozos; pozos en (2,2,2), (4,2,2), (2,4,2) y (4,4,2).

Para realizar las pruebas en los esquemas de trabajo en paralelo se utilizó un cluster de memoria distribuida conformado por 16 nodos o servidores con dos procesadores cada uno a 3.0 Ghz con 2 discos duros de 10000 rpm, el cual funciona bajo el sistema operativo Red Hat Empresarial versión 3.0 y con un compilador Portlan Group.

Para la programación en paralelo se utilizó la librería de envío de mensajes MPI de la cuál se habló en el capítulo anterior.

Se probaron dos esquemas de paralelización los cuales fueron:

- Paralelización por diagonales de coeficientes (esquema 1).
- Paralelización por bloques (esquema 2).

Las características de los dos esquemas utilizados se presentan a continuación.

5.1 Paralelización por Diagonales (Coeficientes de la matriz de coeficientes).

El primer esquema que se utilizó para realizar la paralelización del simulador fue la de dividir la creación de la matriz de coeficientes de tal forma que cada coeficiente de la **ecuación 2.12**, fuera calculado por un servidor de manera independiente. Para realizar esta propuesta se utilizaron 6 nodos (uno maestro y cinco esclavos), de tal manera que

el nodo maestro o servidor maestro es el encargado de dar lectura a los datos iniciales y de comunicarlos a los nodos esclavos. Por las características del programa de comunicación utilizado (MPI), los envíos se tienen que realizar en forma de arreglos vectoriales; por lo que el servidor maestro se encarga de crear estos vectores con la información que ya leyó en los datos, mientras que los nodos esclavos o procesadores esclavos deben "descargar" la información de los arreglos vectoriales que han recibido para que puedan trabajar con ella de manera más sencilla y práctica. Cada nodo esclavo, al igual que el maestro, crea la malla de simulación, asigna condiciones iniciales y de frontera y calcula el coeficiente que le ha sido asignado. Los nodos esclavos regresan esta última información al servidor maestro para que este se encargue de calcular los últimos dos coeficientes y solucionar el sistema, para obtener la solución para la presión y continuar el proceso de simulación, tal y como se muestra en la **figura 5.3**. Como se observa en la figura el nodo maestro realiza dos veces la ejecución del ciclo de creación de los coeficientes y creación de la matriz, lo que se traduce en tiempo de corrida y este parámetro es el que se busca disminuir; además de que este esquema está restringido a utilizar tantos procesadores como coeficientes existan, ya que de otra manera se tendría que adaptar ciclos dobles (en el caso en que hubieran menos procesadores disponibles) en los nodos esclavos con lo cual se aumentaría de igual manera el tiempo de proceso. Sería fácil pensar que en lugar de trabajar con igual o menor numero de nodos como coeficientes existan, se trabajara con más procesadores y así no se tendrían que repetir ciclos; esto es cierto sin embargo el inconveniente de esta medida es que los diferentes procesadores no tendrían la misma carga de trabajo (definida en el capítulo anterior) acarreando problemas de diferencias en tiempo entre procesadores e incluso diferencias de precisión en los cálculos. En la **figura 5.4** se muestra como se reparte el trabajo en los 6 servidores utilizados en el esquema de paralelización por coeficientes, apreciándose claramente la diferencia en la carga de trabajo entre los procesadores.

Este esquema de paralelización se pensó y realizó debido a la relativa facilidad que se presenta en el código serial, ya que por la estructura del programa, es sencillo y atractivo hacer que cada servidor calcule un coeficiente y después este coeficiente sea comunicado al nodo maestro; no se tienen que crear vectores auxiliares ni redefinir las variables que ya se tenían en el código serial, del cuál se parte para realizar el proceso de paralelización.

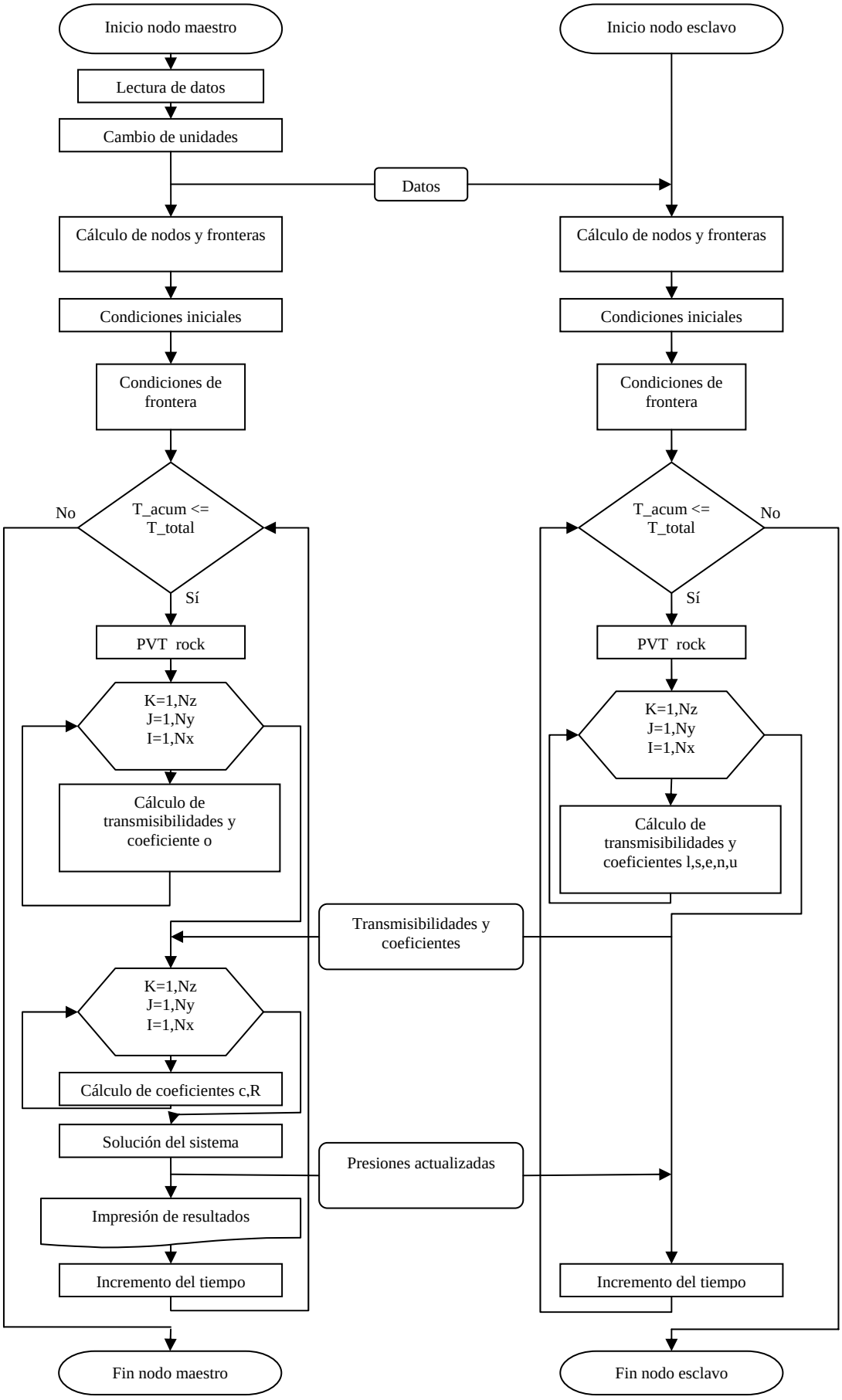


Figura 5.3. Diagrama de flujo para el esquema paralelo por diagonales

Figura 5.4. Matriz de coeficientes para una malla de 3x3x3 para representar el esquema paralelo por diagonales.

Ahora bien, cuándo se observan los resultados correspondientes a esta idea, nos damos cuenta de que no es conveniente este esquema de paralelización debido a los tiempos empleados y a la carga de trabajo mal repartida; por lo que debemos buscar otro que sea más eficiente para que el esfuerzo de paralelizar esta aplicación valga la pena.

5.2 Paralelización por Bloques.

Como el primer esquema de paralelización del simulador no brindo los resultados esperados, se buscó otra manera de realizar la paralelización para que fuera más eficiente, la cuál consistió en partir la matriz de coeficientes en bloques, de tal manera que todos los nodos calculan una parte de todos los coeficientes para formar la matriz. Al igual que en el esquema anterior el nodo maestro es el encargado de hacer la lectura de los datos, crear los arreglos vectoriales para el envío de información y realizar este último; la creación de la malla de simulación, la asignación de condiciones iniciales y de frontera son definidas por cada servidor de manera independiente. La definición del intervalo de trabajo o bloque de cada procesador es obtenida por cada uno de ellos y es independiente uno de otro, sin embargo con un orden bien definido. De esta manera cada nodo se encarga de calcular todos los coeficientes, pero únicamente en el intervalo de elementos que le corresponde; este intervalo queda definido automáticamente en función del número de procesadores con el que se realice el proceso de simulación. Posteriormente cada uno de los procesadores esclavos tiene la tarea de enviar la parte de la matriz de coeficientes que ha definido al procesador maestro, para que este se encargue de ensamblarla en su totalidad, en el orden adecuado, y comenzar con el proceso de solución para la presión e impresión de resultados. La estructura de este programa se muestra en el diagrama de flujo de la **figura 5.5**. Con este esquema de trabajo, se elimina uno de los dos ciclos que realizaba en el esquema anterior el nodo maestro reduciéndose también el tiempo de creación de la matriz de coeficientes; además, con este método no se restringe el número de procesadores a utilizar en 6 (para este estudio), sino que se puede trabajar con el número que se desee, desde 1 hasta Nc, donde Nc es el número de ecuaciones (renglones de la matriz) y si se habla de la carga de trabajo para cada procesador esta será prácticamente la misma para todos los procesadores, sin importar si el número de ellos es pequeño o grande. En la **figura 5.6** se observa como es que en este esquema de bloques la carga de trabajo está mejor

repartida, aprovechando la ventaja que ofrece el cálculo de los coeficientes, donde el coeficiente c_i, por ejemplo, no depende del coeficiente c_{i-1}.

Para este caso, se tuvo que diseñar una subrutina exclusivamente para realizar la partición de la matriz de coeficientes en base al número de renglones que tiene y que cada procesador conozca cuantos elementos de la matriz le corresponde calcular, así como las coordenadas *i,j,k* que le corresponde a cada uno de ellos, para que la creación y comunicación de resultados parciales sea más eficaz. Además de esto, se tuvieron que crear vectores auxiliares para que, por medio de éstos, los procesadores pudieran comunicar sus resultados intermedios al procesador maestro el cual los recibe y continúa con el proceso de la simulación. La ventaja en este esquema es que al momento de definir la longitud de los coeficientes, recordemos que se trabajan como vectores, el único nodo que los define con su longitud total es el nodo maestro, mientras que los otros procesadores únicamente los definen con la longitud que necesita cada uno para su trabajo que es aproximadamente la misma para todos. Esto se traduce en reducción en la memoria ocupada por cada procesador y por consiguiente mayor velocidad en los cálculos realizados. Otro punto a favor es que al momento de calcular los coeficientes, los procesadores no repiten los ciclos *I,J,K* completos, sino que sólo recorren el número de elementos que les corresponde siendo esto el principal factor de ahorro de tiempo.

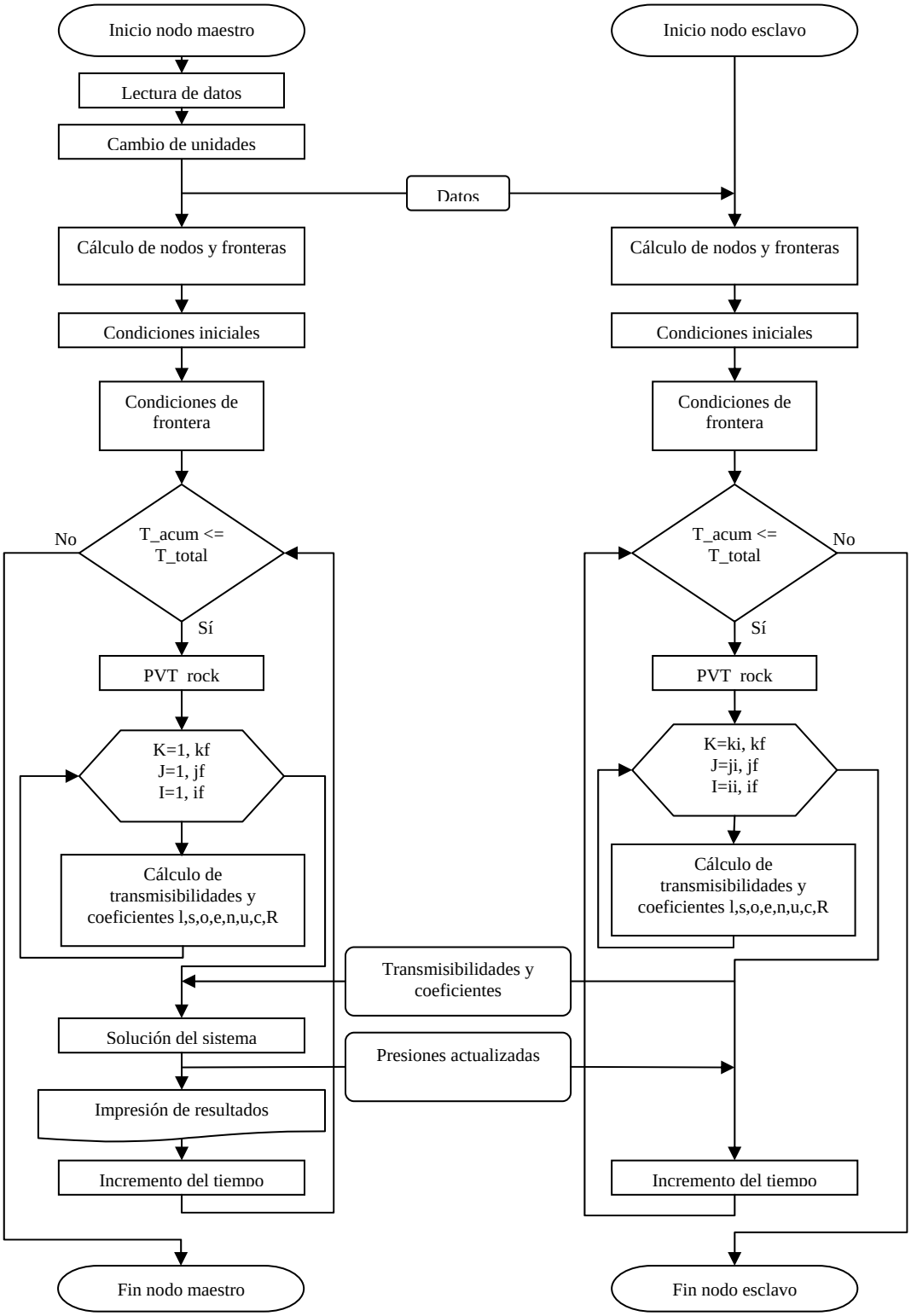


Figura 5.5. Diagrama de flujo para el esquema paralelo por bloques

Figura 5.6. Matriz de coeficientes para una malla de 3x3x3 para representar el esquema paralelo por bloques.

Para conocer cuál era el diseño paralelo más eficiente se realizaron pruebas con las mismas características del yacimiento, fluidos y pozos; tanto para el diseño de programación secuencial como para los dos diseños paralelos. Se llevaron a cabo pruebas con diferentes tamaños de mallas de simulación para comparar los resultados de los comportamientos de los tres esquemas y poder aceptar o descartar lo realizado. Las características generales del yacimiento y sus fluidos y las de los pozos se muestran en las **tablas 5.1** y **5.2** respectivamente.

Tabla 5.1 Datos generales del yacimiento y fluido.

Longitud en X (pie)	: 3445.0
Longitud en Y (pie)	: 3445.0
Longitud en Z (pie)	: 1082.0
Número de nodos en x	: 21
Número de nodos en y	: 21
Número de nodos en Z	: 11
Número de pozos en prod	: 4
Número de pozos en Iny.	: 0
Tiempo Total de Sim(días)	: 50.0
permeabilidad matriz (mD)	: 2.0
Perm fractura en X (mD)	: 250.0
Perm fractura en Y (mD)	: 250.0
Perm fractura en Z (mD)	: 250.0
viscosidad (cp)	: 0.5
Gravedad API	: 40.0
Compres del fluido (1/psi)	: 6.9E-06
Compres de la matriz(1/psi)	: 1.0E-06
Comp de la fractura (1/psi)	: 1.0E-06
Porosidad matriz (-)	: 0.10
Porosidad fractura (-)	: 0.03
FVF Base	: 1.1
factor de forma sigma(1/m2)	: 1.0
Presión base (psi)	: 500.0
Presión inicial (psi)	: 6000.0

Tabla 5.2 Datos generales de los pozos

*xp	* yp	* zp	* rwp	* Qop
1	* 1	* 6	* 0.45	*-10000.0
21	* 21	* 6	* 0.45	*-10000.0
21	* 1	* 6	* 0.45	*-10000.0
1	* 21	* 6	* 0.45	*-10000.0

*xi	* yi	* zi	* rw	* Qoi

Nota: el signo del gasto queda definido bajo la siguiente convención, un gasto negativo representa un gasto de producción, es decir, que sale del yacimiento; y un gasto positivo indica un gasto de inyección, es decir, que entra al yacimiento. Esto es lo mismo en todas las tablas del trabajo que involucren gasto en los pozos.

Para que los estudios fueran congruentes, los distintos tamaños de mallas empleados se adaptaron para que el tamaño de celda fuera el mismo en todas las mallas, modificando los renglones de ‘Número de nodos en X’, ‘Número de nodos en Y’ y ‘Número de nodos en Z’ así como en las longitudes del yacimiento correspondientes de cada dirección, de modo que si se divide la longitud en una dirección entre el número de nodos correspondiente el resultado es el mismo para cada tamaño de malla; manteniendo todos los demás parámetros idénticos. Se hicieron pruebas con mallas 21x21x11, 41x41x11, 61x61x11, 81x81x11 y 101x101x11 para los tres esquemas de programación.

Cabe aclarar que el trabajo realizado únicamente se enfoca a la paralelización de la creación de la matriz de coeficientes y no al método de solución, que es la parte del proceso que más tiempo consume, por lo que cuándo se trabajaba con mallas muy grandes (más de 50,000 celdas), los tiempos totales de simulación se disparaban con valores grandes, por lo que únicamente se plantearon estos cinco tamaños de malla. Sin embargo los resultados correspondientes al trabajo de paralelización de esta tesis son satisfactorios y esto se puede apreciar en el capitulo siguiente.

El esfuerzo de realizar el proceso de paralelización en este simulador es para dar elementos de convencimiento para que este procedimiento se realice en simuladores más complejos, como, por ejemplo, aquellos que se basan en la **ecuación 5.1** que es un simulador de yacimientos multifásicos en tres dimensiones y fracturado.

$$
\begin{aligned}
&\begin{bmatrix} E_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{i-1,jk} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{i-1,jk} + \begin{bmatrix} G_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{i,j-1,k} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{i,j-1,k} + \\
&\begin{bmatrix} C_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{ij,k-1} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{ij,k-1} + \begin{bmatrix} A_{ff} & A_{fm} \\ A_{mf} & A_{mm} \end{bmatrix}^{\nu}_{ijk} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{i,j,k} + \\
&\begin{bmatrix} B_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{ij,k+1} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{ij,k+1} + \begin{bmatrix} F_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{i,j+1,k} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{i,j+1,k} + \\
&\begin{bmatrix} D_{ff} & [0] \\ [0] & [0] \end{bmatrix}^{\nu}_{i+1,jk} \begin{bmatrix} \delta \boldsymbol{U}_f \\ \delta \boldsymbol{U}_m \end{bmatrix}^{\nu+1}_{i+1,jk} = -\begin{bmatrix} R_f \\ R_m \end{bmatrix}^{\nu}_{ijk}
\end{aligned}
\qquad (5.1)
$$

Donde $E_{ff}, G_{ff}, C_{ff}, A_{ff}, A_{fm}, B_{ff}, F_{ff}, D_{ff}, A_{mf}, A_{mm}$ son submatrices de orden (3)x(3) **(Peña Ch., O. y Reséndiz T. J., 2006)**, y las *U* son las variables a solucionar dadas por $\mathbf{U}_{ijk} = \left(p_o, S_g, S_w\right)_{ijk}^T$ para la matriz y la fractura. Cada submatriz de la **ec. 5.1** tiene la siguiente forma.

$$\begin{bmatrix} \frac{\partial F_o}{\partial P_o} & \frac{\partial F_o}{\partial S_g} & \frac{\partial F_o}{\partial S_w} \\ \frac{\partial F_g}{\partial P_o} & \frac{\partial F_g}{\partial S_g} & \frac{\partial F_g}{\partial S_w} \\ \frac{\partial F_w}{\partial P_o} & \frac{\partial F_w}{\partial S_g} & \frac{\partial F_w}{\partial S_w} \end{bmatrix}$$

CAPÍTULO 6. ANÁLISIS DE RESULTADOS.

En este capítulo se presentan los resultados obtenidos de los métodos empleados para la paralelización del simulador utilizado, estos resultados son analizados y comparados entre sí para conformar de manera integrada todas las observaciones que se realizaron así como las ventajas y desventajas que se presentaron de cada esquema propuesto.

Primero se analizará cuál de los dos esquemas de paralelización presentados resulta ser el que mejores resultados brinda, en base al tiempo que emplea la aplicación utilizada para conformar la matriz de coeficientes, ya que este es el parámetro que se busca disminuir principalmente con este tipo de programación. Después se hará un análisis del esquema seleccionado para saber que tan satisfactorio y conveniente resulta ser; este análisis se hace comparando los resultados obtenidos contra los que se obtienen del esquema secuencial.

6.1 Comparación Entre Esquemas de Programación.

Se utilizaron tres esquemas de programación, uno secuencial y dos paralelos. Los dos últimos serán comparados contra el primero, para que se pueda juzgar que tan bien resulta cada uno de los esquemas alternos propuestos.

En la **figura 6.1** se presenta una gráfica en la cuál se muestra el comportamiento de los tres esquemas. Esta gráfica fue realizada con propiedades de yacimiento, fluidos y tiempo total de simulación idénticos para cada corrida. Únicamente se realizaron 16 iteraciones o pasos de tiempo por cada corrida en cada tamaño de malla; este número de iteraciones es suficiente para observar la tendencia que se presentará en un número mayor de las mismas, ya que se toma el promedio del tiempo de creación del jacobiano para cada tamaño de malla. Los esquemas paralelos fueron realizados con seis procesadores debido a que cuando se realizó la paralelización por diagonales este es el número obligado a ser utilizado, por los motivos que se explican en el apartado referente a este procedimiento en el capítulo anterior; y para que se tuviera igualdad de condiciones el procedimiento de paralelización por bloques fue realizado con este mismo número de procesadores.

Observando en la gráfica la línea correspondiente al proceso secuencial nos damos cuenta que se aprecia una tendencia lineal en su comportamiento, lo cuál es aceptable, pues para las condiciones de simulación empleadas sólo se modificó el tamaño de malla manteniendo los demás parámetros iguales.

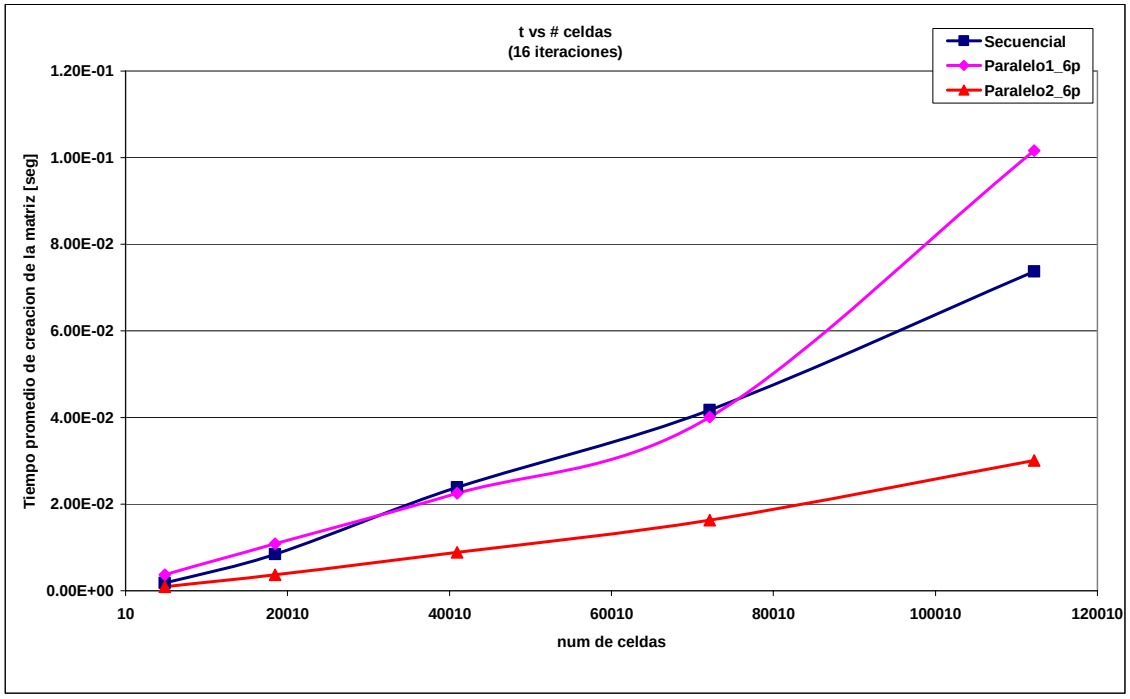


Figura 6.1. Comparación de Esquemas de Programación.

El comportamiento que presenta la línea “Paralelo1_6p”, correspondiente al proceso de paralelización por diagonales, merece gran atención. Como se observa el inicio de la línea (menos de 40,000 celdas de simulación) los valores de tiempo están por encima de los que se presentan en la línea “Secuencial” esto se debe a que en el diseño de este esquema los procesadores esclavos recorren un ciclo completo de Nx*Ny*Nz mientras que el procesador maestro recorre dos veces este ciclo, como se aprecia en el diagrama de flujo de la **figura 5.4**, y si además de esto recordamos que en un proceso paralelo existen comunicaciones entre procesadores (como fue explicado en el capítulo 5) y, por la estructura de este procedimiento, las comunicaciones empleadas a pesar de que son pocas, son muy grandes ya que cada procesador envía un vector de Nx*Ny*Nz elementos lo cuál se traduce en tiempo de envío. A pesar de todo esto, la diferencia no es escandalosa y es razonable, incluso se aprecia que el valor del tiempo comienza a estar por debajo de los valores en el proceso secuencial, lo que está cumpliendo con lo que se esperaba. Sin embargo, si se piensa trabajar con mallas de simulación de este

tamaño o menores, no vale la pena realizar el esfuerzo de paralelizar el simulador y mucho menos de hacer la inversión en el equipo de cómputo ya que los resultados son prácticamente los mismo. En la parte intermedia (40000-80000 celdas), la distribución del trabajo refleja ganancia en el tiempo a pesar del doble ciclo; sin embargo no es una ganancia muy buena por lo que también habría que analizar si vale la pena hacer la inversión para trabajar en paralelo con mallas de este tamaño. Por último, la parte final de esta línea el tiempo crece demasiado; esto se debe a los mismos motivos que se explican arriba. Para este caso, este tiempo de proceso se incrementa de manera tan drástica debido a la duplicidad del ciclo de la creación de la matriz de coeficientes en el nodo maestro y a que el tiempo de comunicaciones influye directamente en el proceso de creación de esta matriz, pues los coeficientes c y R no pueden ser conformados si no se tiene la información de los otros seis coeficientes (exclusivamente en este proceso de paralelización por diagonales), algo que en el esquema secuencial no se presenta ya que en el secuencial sólo se realiza una vez el ciclo de creación y no se emplea tiempo esperando los resultados de los demás coeficientes para iniciar nuevamente un ciclo largo.

Las ventajas que se presentan en la utilización de este método sólo son para el programador debido a que la manera en que se realiza la paralelización es muy sencilla y sin complicaciones, pues la estructura original del programa facilita este esquema. Sin embargo, a pesar de lo fácil que pueda ser la paralelización los resultados no son los que se desean, cayendo en el apartado de que no siempre lo más sencillo es lo mejor. Por otro lado, las desventajas son muchas ya que no se presenta un trabajo paralelo efectivo, la carga de trabajo entre los procesadores está muy desbalanceada, todos los procesadores asignan memoria que nunca ocupan y, como ya se mencionó antes, recorren un ciclo demasiado largo sin que sea necesario.

Considerando las líneas “Secuencial” y Paralelo2_6p” de la figura 6.1, se puede observar que su comportamiento es similar pero los tiempos que se registran en la segunda línea son menores durante todo el proceso que los de la primera, lo que indica que este esquema de paralelización arrojó los resultados que se esperaban cumpliendo con el objetivo del trabajo; notamos además que entre mayor sea el número de celdas de simulación mayor es el ahorro de tiempo que se tiene cuándo se utiliza el programa en paralelo y que en este caso, las comunicaciones entre procesadores no repercuten de

manera importante en el tiempo del proceso. Además de esto, si recordamos el diagrama de flujo que corresponde a este procedimiento de paralelización, **figura 5.6**, no existe duplicidad en el ciclo de conformación de la matriz de coeficientes y por la estructura del código, el procesador maestro no tiene que esperar a que los nodos esclavos terminen y así se desarrolla un trabajo paralelo eficiente.

Las ventajas que se pueden mencionar en este proceso son que no se limita el uso de procesadores exclusivamente a seis, la carga de trabajo en los procesadores es aproximadamente la misma en todos los que se empleen, los procesadores no asignan memoria a las variables de manera innecesaria ya que sólo asignan la que ocupan y los ciclos de desarrollo que recorren es mucho más corto que el que se tenía originalmente. Las desventajas que se tienen son exclusivas del programador ya que la manera de realizar la paralelización no es tan directa y sencilla como se deseara, se tienen que emplear procedimientos auxiliares para realizar la repartición de trabajo, pero cuándo este inconveniente es superado los resultados obtenidos son buenos. Para este caso el resultado fue satisfactorio pero hay que realizar más pruebas para comprobar que efectivamente se da por bueno el procedimiento propuesto.

6.2 Análisis del Esquema Seleccionado.

La gráfica de la **figura 6.2** muestra el comportamiento del esquema paralelo elegido (paralelización por bloques) contra el esquema secuencial. Se presenta la línea del desempeño secuencial así como las correspondientes al desempeño paralelo con 2, 4, 6, 8 y 10 procesadores.

Para obtener esta gráfica también se respetaron las condiciones de simulación durante todos los procesos, siendo el tamaño del yacimiento y la cantidad de celdas lo único que se varió, esto con el fin de que, sin importar el número de celdas, éstas siempre fueran del mismo tamaño para que la comparación de resultados se realice de la mejor manera. En las **tablas 6.1** y **6.2** se muestran las propiedades empleadas para la simulación con una malla de 21x21x11 y una malla de 101x101x11 respectivamente; nótese el cambio en los renglones correspondientes a ‘Longitud en X’, ‘Longitud en Y’ y ‘Longitud en Z’ y si estos valores se dividen entre los números de nodos correspondientes, arrojan el mismo valor.

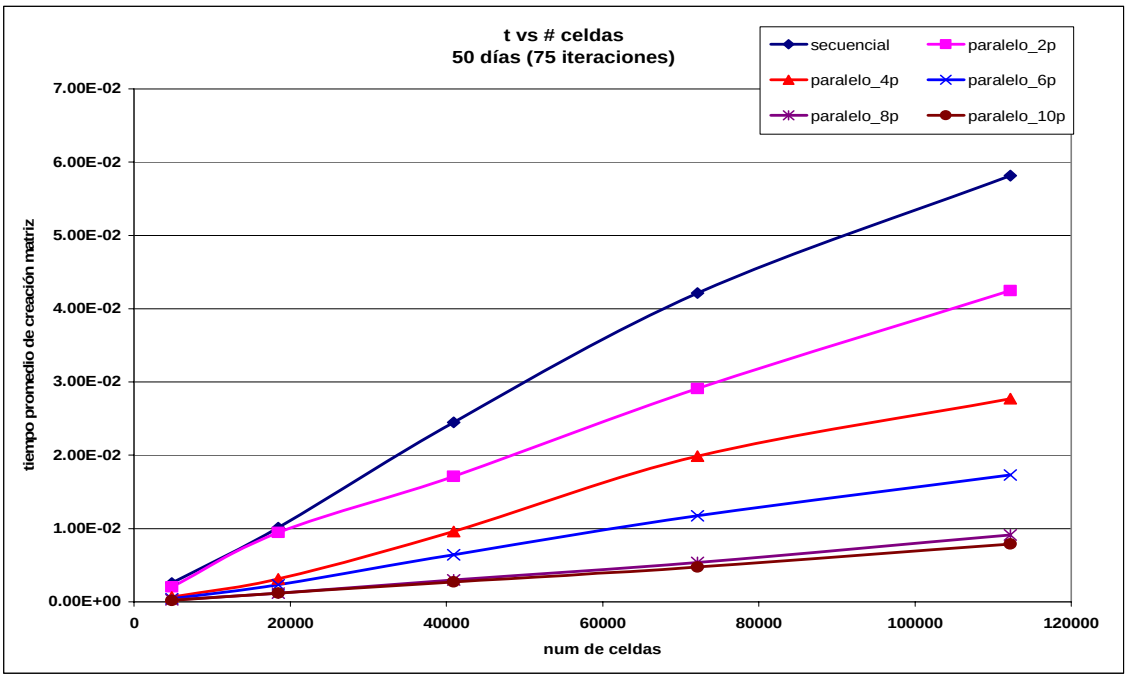


Figura 6.2. Comportamiento del tiempo promedio de cada iteración con respecto al tamaño de malla empleado, para diferente número de procesadores.

Tabla 6.1. Datos empleados para comparar esquemas de programación. Malla 21x21x11.

Longitud en X (pie)	: 3445.0
Longitud en Y (pie)	: 3445.0
Longitud en Z (pie)	: 1082.0
Numero de nodos en x	: 21
Numero de nodos en y	: 21
Numero de nodos en Z	: 11
Numero de pozos en prod	: 4
Numero de pozos en Iny.	: 0
Tiempo Total de Sim(dias)	: 50.0
permeabilidad matriz (mD)	: 2.0
perm_fractura en X (mD)	: 250.000
perm_fractura en Y (mD)	: 250.000
perm_fractura en Z (mD)	: 250.000
viscosidad (cp)	: 0.50
Gravedad API	: 40.0
comp fluido (psi-1)	: 6.9E-06
Comp matriz (psi-1)	: 1.0E-06
Comp fractura (psi-1)	: 1.0E-06
porosidad_matriz (-)	: 0.10
porosidad_fractura (-)	: 0.03
FVF Base (Pi)	: 1.1
factor de forma sigma(1/m2)	: 1.0
Presion base (psi)	: 500.0
Presion inicial (psi)	: 6000.0

Como se observa en la gráfica de la figura, entre mayor sea el tamaño de la malla mayor es el tiempo que se emplea en cada iteración y, por otro lado, entre mayor sea el número de procesadores menor es el tiempo que se emplea en la creación de la matriz de coeficientes notándose un decremento importante en los valores por cada incremento de procesadores; sin embargo después de utilizar 8 procesadores el resultado ya no varía de manera importante, pues se aprecia que la diferencia entre la línea correspondiente a 10 procesadores ya no está muy distante de la que corresponde a 8 procesadores. Este comportamiento, a pesar de que en este trabajo se toma como correcto, no puede generalizarse como una regla para todos los trabajos de paralelización; en este estudio se presentó de esta manera. Sin embargo deja planteada la idea de que al desarrollar un trabajo paralelo llegará un momento en que, sin importar cuánto se aumente el número de procesadores, la ganancia en tiempo de una cantidad a otra ya no será relevante.

Tabla 6.2. Datos empleados para comparar esquemas de programación. Malla 101x101x11

Longitud en X	(pie)	: 16568.0
Longitud en Y	(pie)	: 16568.0
Longitud en Z	(pie)	: 1082.0
Numero de nodos en x		: 101
Numero de nodos en y		: 101
Numero de nodos en Z		: 11
Numero de pozos en prod		: 4
Numero de pozos en Iny.		: 0
Tiempo Total de Sim(dias)		: 50.0
permeabilidad matriz	(mD)	: 2.0
perm_fractura en X	(mD)	: 250.0
perm_fractura en Y	(mD)	: 250.0
perm_fractura en Z	(mD)	: 250.0
viscosidad	(cp)	: 0.5
Gravedad API		: 40.0
comp fluido	(psi-1)	: 6.9E-06
Comp matriz	(psi-1)	: 1.0E-06
Comp fractura	(psi-1)	: 1.0E-06
porosidad_matriz	(-)	: 0.10
porosidad_fractura	(-)	: 0.03
FVF Base (Pi)		: 1.1
factor de forma sigma(1/m2)		: 1.0
Presion base (psi)		: 500.0
Presion inicial (psi)		: 6000.0

Todo esto se puede notar mejor en la gráfica de la **figura 6.3**, donde se muestra cómo se comporta el tiempo de creación de la matriz contra el número de procesadores, para cada tamaño de malla utilizado. Si observamos con atención, los valores de tiempo promedio de cada iteración disminuyen conforme incrementa el número de

procesadores empleado, pero al llegar a 8 procesadores el comportamiento de las líneas (tamaño de malla) tiende a ser horizontal, que nos indica lo que se mencionó en el párrafo anterior.

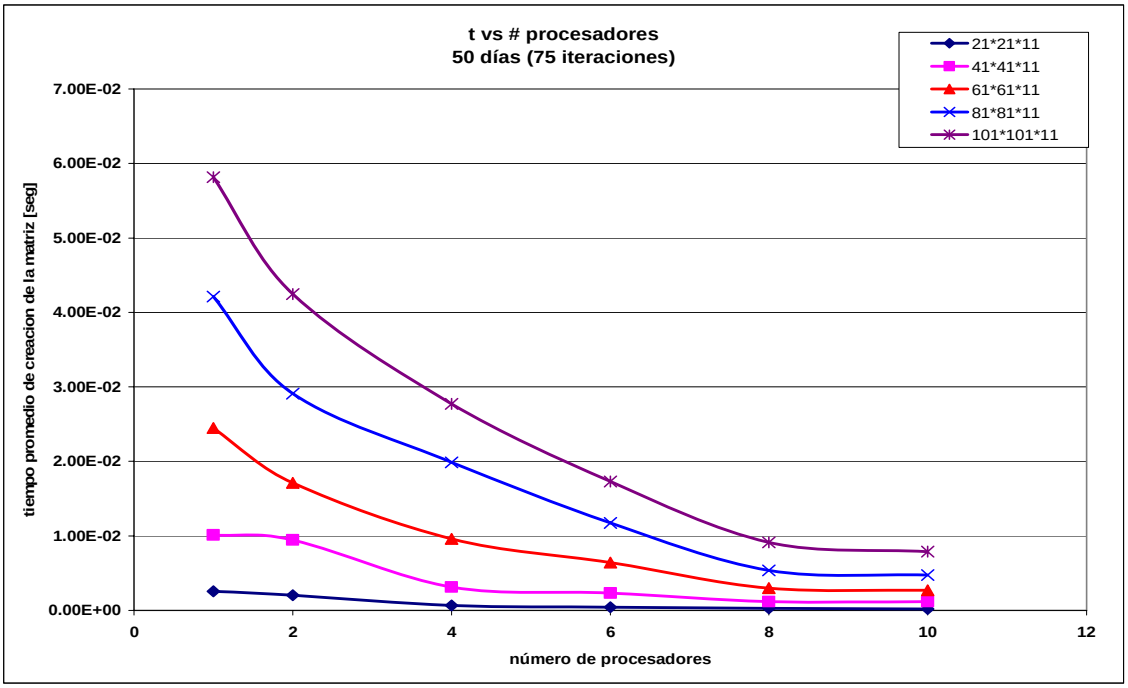


Figura 6.3. Comportamiento del tiempo promedio de cada iteración con respecto al número de procesadores empleado, para cada tamaño de malla utilizado.

Con base a los resultados mostrados y el análisis realizado se puede aceptar que el procedimiento de *Paralelización por Bloques* es bueno, dándonos pauta a continuar con el estudio de que tan efectivo resulta este tipo de programa sobre uno secuencial, en lo concerniente al tema de estudio.

6.2.1. Análisis de la Efectividad del Esquema Seleccionado.

En este apartado se realizará el estudio de que tan efectivo resulta el esfuerzo de paralelizar la parte de la creación de la matriz de coeficientes en un proceso de simulación numérica.

Para iniciar con este análisis hay que observar la **figura 6.4**. Aquí se muestra una gráfica en la que se presenta el tiempo acumulado de todas las iteraciones realizadas

para la creación de la matriz contra el número de celdas de la simulación para diferente número de procesadores.

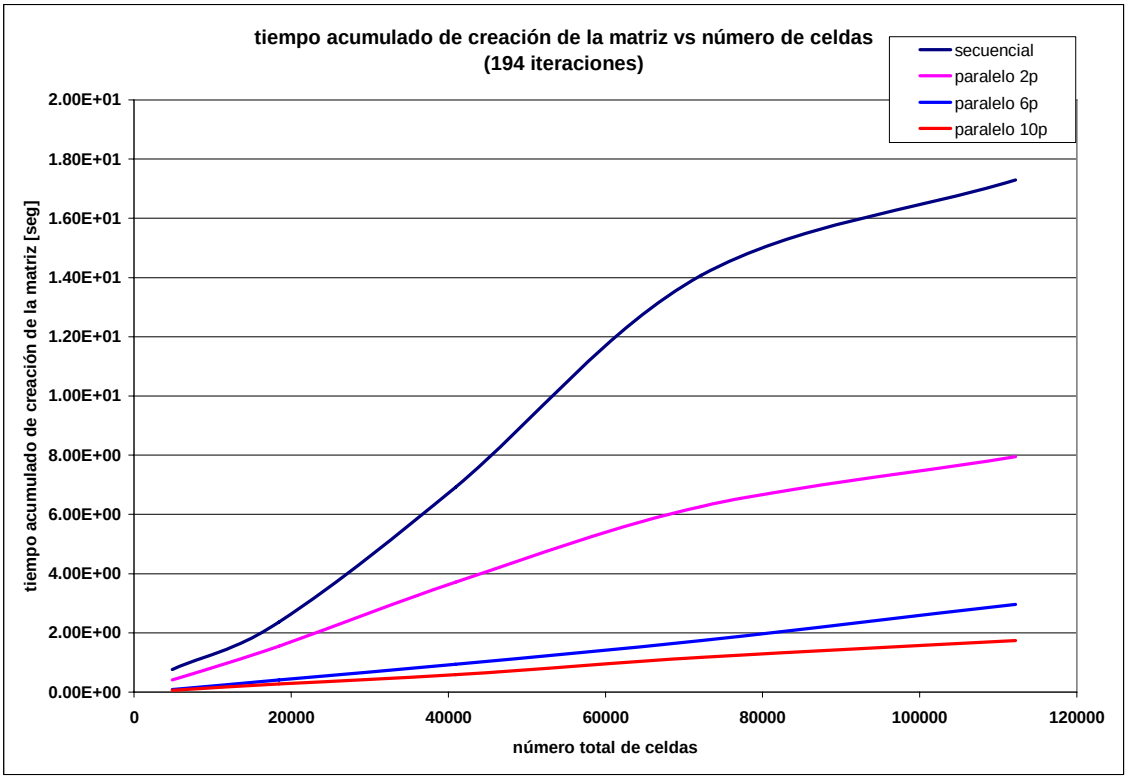


Figura 6.4. Comportamiento del tiempo acumulado de creación de la matriz con respecto al tamaño de malla, para diferente número de procesadores.

Esta gráfica esta hecha con base a los datos que se muestran en la **tabla 6.3**.

Tabla 6.3. Registro de los tiempos para diferentes mallas y número de procesadores.

	número de procesadores	número total de celdas				
		21x21x11	41x41x11	61x61x11	81x81x11	101x101x11
		4,851	18,491	40,931	72,171	112,211
tiempo de creación acumulado [seg], a 1460 días, 194 iteraciones	1	7.54E-01	2.37E+00	6.91E+00	1.41E+01	1.73E+01
	2	4.03E-01	1.55E+00	3.71E+00	6.27E+00	7.95E+00
	6	7.91E-02	4.10E-01	9.44E-01	1.73E+00	2.96E+00
	10	5.74E-02	2.78E-01	5.88E-01	1.16E+00	1.74E+00

Si se toman los valores correspondientes a una malla de 112,211 celdas y se analiza cuál es la ganancia de tiempo cuándo se utilizan 10 procesadores contra la utilización de 1 procesador se aprecia que es de 17.3–1.74=15.56 [segundos] para 194 iteraciones.

Tal vez así no se vea la ventaja de realizar el esfuerzo para paralelizar una aplicación, pero veamos una predicción de un proceso de simulación de condiciones reales (tamaño de malla de un millón de celdas por lo menos y cien mil iteraciones totales), en base a estos resultados. Para esto necesitamos la gráfica de la **figura 6.5**, que es la misma que la de la **figura 6.4** sólo que en escala logarítmica, en donde se ha trazado una línea de tendencia que se ha extrapolado hasta un tamaño de malla de 1,000,000 de celdas; y la **tabla 6.4** donde se cuantifican los valores de interés hacia la predicción de 100,000 iteraciones. Los valores que se pronostican para las condiciones mencionadas son: 200 [seg] para 1 procesador y 20 [seg] para 10 procesadores (haciendo la extrapolación del tiempo acumulado de 194 iteraciones), obteniendo un ahorro de *0.0043 [hrs]* para el estudio de tamaño de malla=112,211 de celdas y 194 iteraciones; y de *25.7731 [hrs]* para el tamaño de malla=1,000,000 de celdas y 100,000 iteraciones.

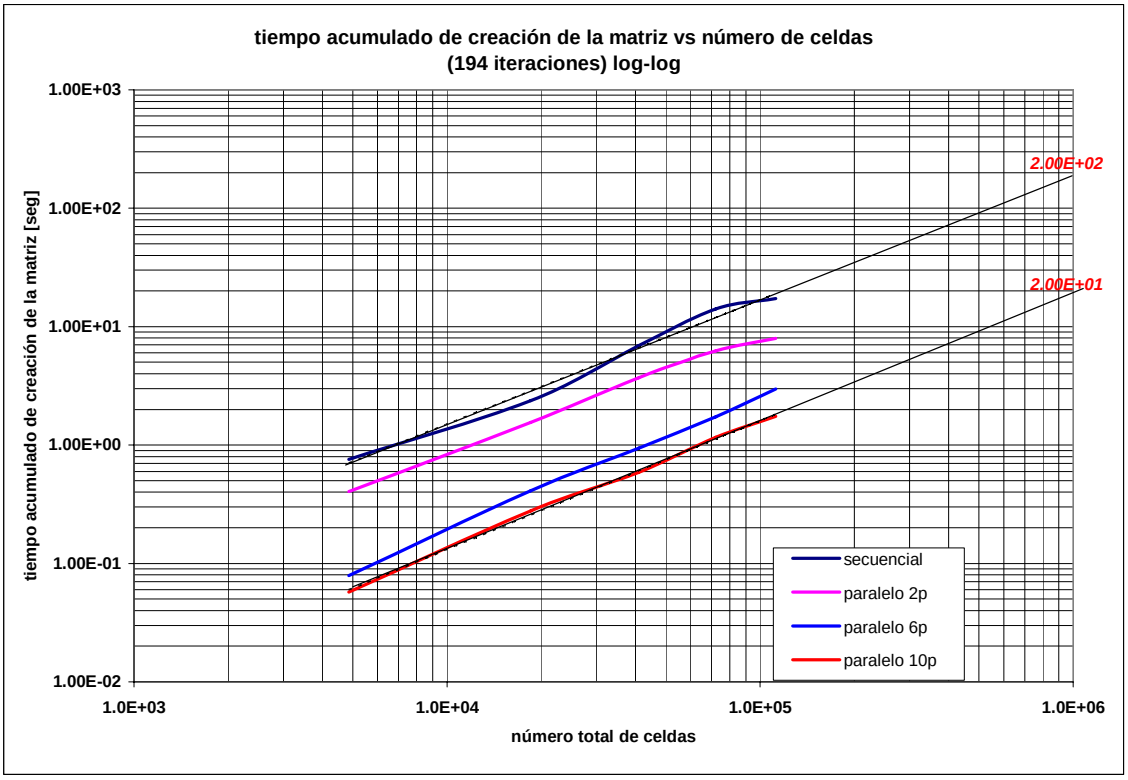


Figura 6.5. Tendencia del tiempo de creación de matriz de coeficientes con respecto al número de celdas, para diferente número de procesadores.

Tabla 6.4. Predicción del ahorro de tiempo con un esquema paralelo

	número de procesadores	reales		predecidos	
		núm celdas 112,211	núm de iteraciones	núm celdas 1,000,000	núm de iteraciones
tiempo registrado [seg]	1	17.3	**194**	200	**100,000**
	10	1.74		20	
diferencia [seg]		15.56		180	
tiempo ahorrado, promedio por iteración [seg]		0.080206186		0.927835052	
tiempo ahorrado [seg]		15.56		92783.50515	
tiempo ahorrado [min]		0.259333333		1546.391753	
tiempo ahorrado [hrs]		0.004322222		25.77319588	

Los resultados que se ven en la **tabla 6.4** ya son más convincentes, pues al hablar de un ahorro de más de un día en una simulación a cualquier ingeniero le agradaría la idea; además este ahorro de tiempo es considerando las condiciones de simulación de este estudio únicamente, es decir, un simulador semi-implícito de aceite negro monofásico en el cuál no se varia ningún parámetro más que la presión, y si se complica más el simulador considero que los resultados serán aún más atractivos. Es por esto que este trabajo tiene la finalidad de servir cómo base para realizar la paralelización de un simulador más complejo, p.e. multifásico, fracturado y tridimensional; tal y como se explica en el capítulo anterior donde se demuestra cómo es que cambia la conformación de los elementos de la matriz, pues en este caso son escalares y en el simulador multifásico son matrices de 3x3. Entonces si aquí los resultados ya son interesantes, se debe considerar realizar el esfuerzo de paralelizar simuladores con las características del jacobiano citadas.

Todos estos resultados y comparaciones han sido para la parte de la matriz de coeficientes (jacobiano), los cuáles nos han dejado satisfechos pues es el objetivo del trabajo. Sin embargo, no se puede dejar de lado los resultados totales de la simulación, en lo que respecta al comportamiento de la presión del yacimiento, que es lo que importa y de nada serviría ahorrar días y días en un proceso si al final arroja resultados incorrectos que no se puedan emplear en la ingeniería. Es por esto que se presenta el siguiente apartado donde se comparan los resultados de la simulación para corroborar

que el proceso paralelo arroja los resultados correctos. Nuevamente es comparado con el proceso secuencial del simulador base.

Se debe recordar también que en este trabajo no se modificó la estructura básica del simulador, es decir, la parte correspondiente a la lectura de datos de entrada, generación de nodos y fronteras, así como lo más importante que es el procedimiento de solución continua trabajando de manera secuencial y por un solo procesador (que para el caso de los nodos y fronteras todos los procesadores lo realizan, pero de manera secuencial). Con esto como antecedente se incrementa la idea de que cuando se paralelice por completo un simulador de yacimientos la ganancia en tiempo total de las corridas y la capacidad para el manejo de información disminuirá y aumentara, respectivamente, de manera considerable para un óptimo proceso de Simulación de Yacimientos.

6.3 Comparación de los Resultados de la Simulación.

Como ya se anticipó, aquí se mostrará si el procedimiento paralelo es completamente confiable al comparar los resultados finales de la simulación contra los que se tienen del procedimiento secuencial. En esta parte ya no se manejan tiempos de proceso, sólo se mostrarán presiones, lo que es el objetivo de este simulador en específico.

Las **figuras 6.6** y **6.7** muestran graficas de P_{wf} contra el logaritmo del *tiempo de simulación*, con base a las propiedades de las **tablas 6.5** y **6.6** para la primera y de las **tablas 6.7** y **6.8** para la segunda.

En la gráfica de la **figura 6.6** se presenta la P_{wf} del pozo 1 para los casos de corridas en esquema secuencial y paralelo para 2, 6 y 10 procesadores; como se ve, los valores son los mismos durante todo el tiempo de la simulación, el cuál fue de un año, pues las líneas correspondientes a cada prueba se empalman completamente.

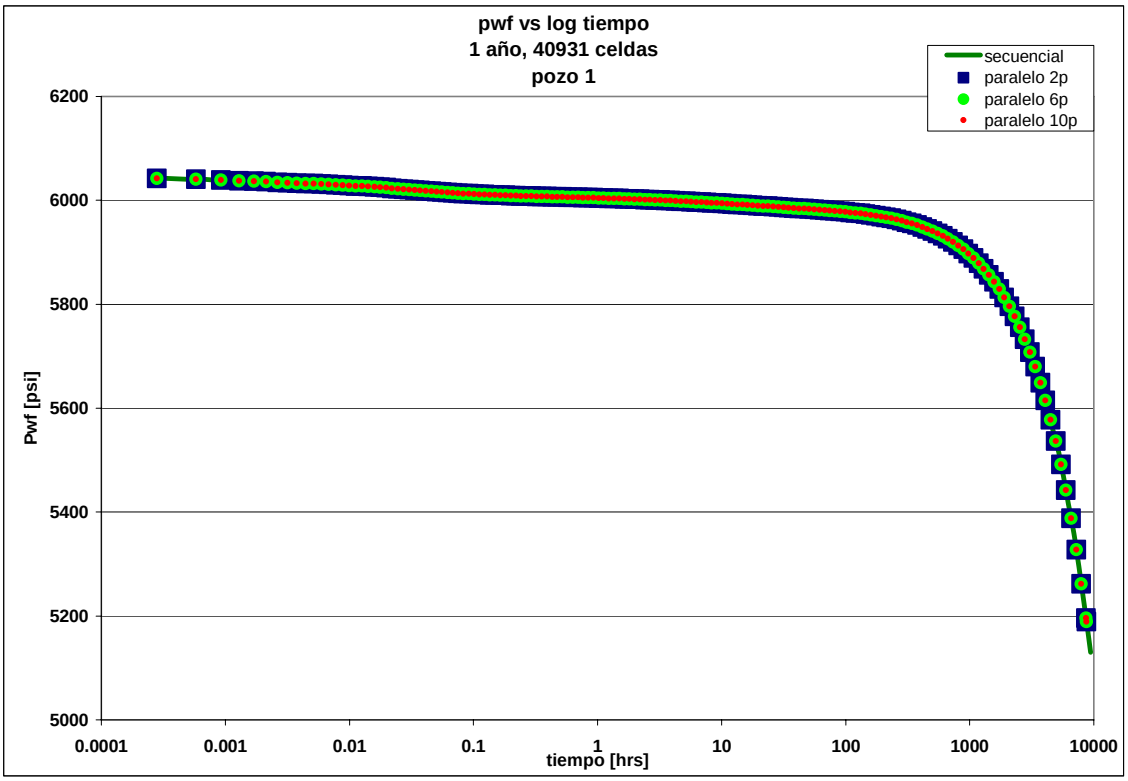


Figura 6.6. Pwf contra tiempo en una simulación de 1 año, para diferente número de procesadores.

Tabla 6.5. Datos empleados para comparar resultados finales de la simulación. Características del Yacimiento y fluidos. Malla 61x61x11.

Longitud en X	(pie)	: 6560.0
Longitud en Y	(pie)	: 9840.0
Longitud en Z	(pie)	: 820.0
Numero de nodos en x		: 61
Numero de nodos en y		: 61
Numero de nodos en Z		: 11
Numero de pozos en prod		: 2
Numero de pozos en Iny.		: 0
Tiempo Total de Sim(dias)		: 365.0
permeabilidad matriz	(mD)	: 2.0
perm_fractura en X	(mD)	: 250.0
perm_fractura en Y	(mD)	: 250.0
perm_fractura en Z	(mD)	: 250.0
viscosidad	(cp)	: 0.5
Gravedad API		: 40.0
comp fluido	(psi-1)	: 6.9E-06
Comp matriz	(psi-1)	: 1.0E-06
Comp fractura	(psi-1)	: 1.0E-06
porosidad_matriz	(-)	: 0.10
porosidad_fractura	(-)	: 0.03
FVF Base (Pi)		: 1.1
factor de forma sigma(1/m2)		: 1.0
Presion base (psi)		: 500.0
Presion inicial (psi)		: 6000.0

Tabla 6.6. Datos de los pozos para malla 61x61x11.

```
*xp    * yp    * zp    * rwp     * Qop
 4     * 4     *  4    * 0.45    *-10000.0
  58   *  58   *   8   * 0.45    *-10000.0
*
*xi    * yi    * zi    * rwi     * Qoi
```

En la gráfica de la **figura 6.7** también ocurre la superposición de las líneas correspondientes a la P_{wf} del pozo en análisis (pozo 2 de la **tabla 6.8**) a lo largo de todo el tiempo de simulación que para este caso fue de 4 años.

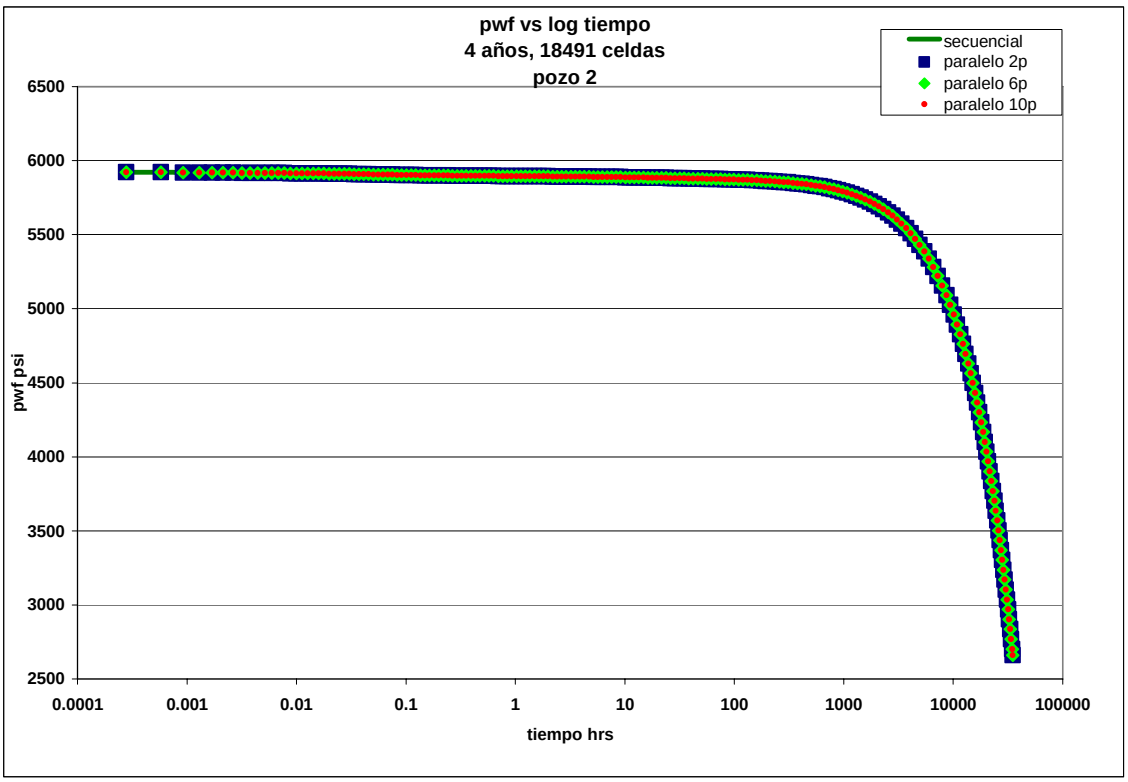


Figura 6.7. Pwf contra tiempo en una simulación de 4 años, para diferente número de procesadores.

Por último en la **figura 6.8** se muestra una gráfica de la presión de fractura de la celda central del yacimiento, en una malla de 21x21x11; es decir, la celda ubicada en las coordenadas (11,11,6), notándose el mismo fenómeno que en las dos gráficas anteriores.

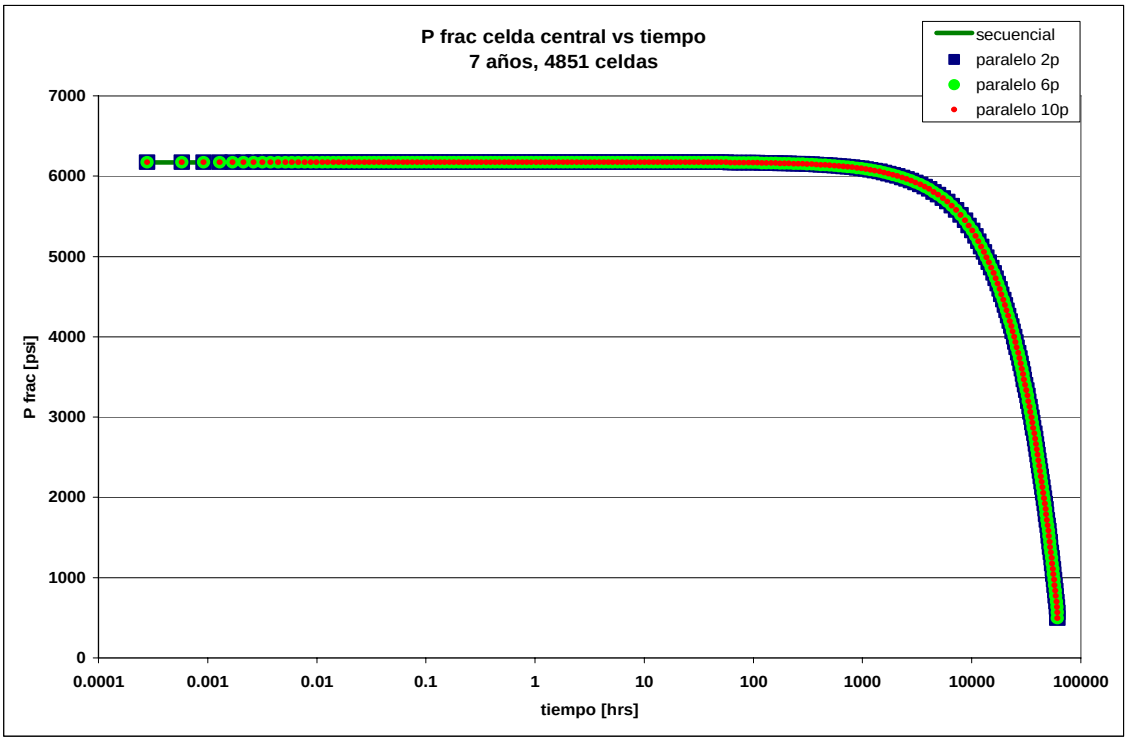


Figura 6.8. Presión de fractura de la celda central contra tiempo en una simulación de 7 años, para diferente número de procesadores.

Tabla 6.7. Datos empleados para comparar resultados finales de la simulación. Características del Yacimiento y fluidos. Malla 41x41x11.

Longitud en X	(pie)	: 6560.0
Longitud en Y	(pie)	: 9840.0
Longitud en Z	(pie)	: 820.0
Numero de nodos en x		: 41
Numero de nodos en y		: 41
Numero de nodos en Z		: 11
Numero de pozos en prod		: 2
Numero de pozos en Iny.		: 0
Tiempo Total de Sim(dias)		: 1460.0
permeabilidad matriz	(mD)	: 2.0
perm_fractura en X	(mD)	: 250.0
perm_fractura en Y	(mD)	: 250.0
perm_fractura en Z	(mD)	: 250.0
viscosidad	(cp)	: 0.5
Gravedad API		: 40.0
comp fluido	(psi-1)	: 6.9E-06
Comp matriz	(psi-1)	: 1.0E-06
Comp fractura	(psi-1)	: 1.0E-06
porosidad_matriz	(-)	: 0.10
porosidad_fractura	(-)	: 0.03
FVF Base (Pi)		: 1.1
factor de forma sigma(1/m2)		: 1.0
Presión base (psi)		: 500.0
Presión inicial (psi)		: 6000.0

Tabla 6.8. Datos de los pozos para malla 41x41x11.

```
*xp    * yp    * zp    * rwp     * Qop
 4     * 4     *  4    * 0.45    *-10000.0
  38   *  38   *   8   * 0.45    *-10000.0
*
*xi    * yi    * zi    * rwi     * Qoi
```

Las imágenes de la **figura 6.9** muestran una visualización del yacimiento para una malla 21x21x11, con base en los datos de las **tablas 6.9** y **6.10**.

Como se muestra en la figura, la visualización de la malla de simulación es una secuencia de tiempo donde, con ayuda del código de colores, se aprecia el decremento de la presión en el yacimiento. Esta simulación fue hecha con el simulador en estructura paralela.

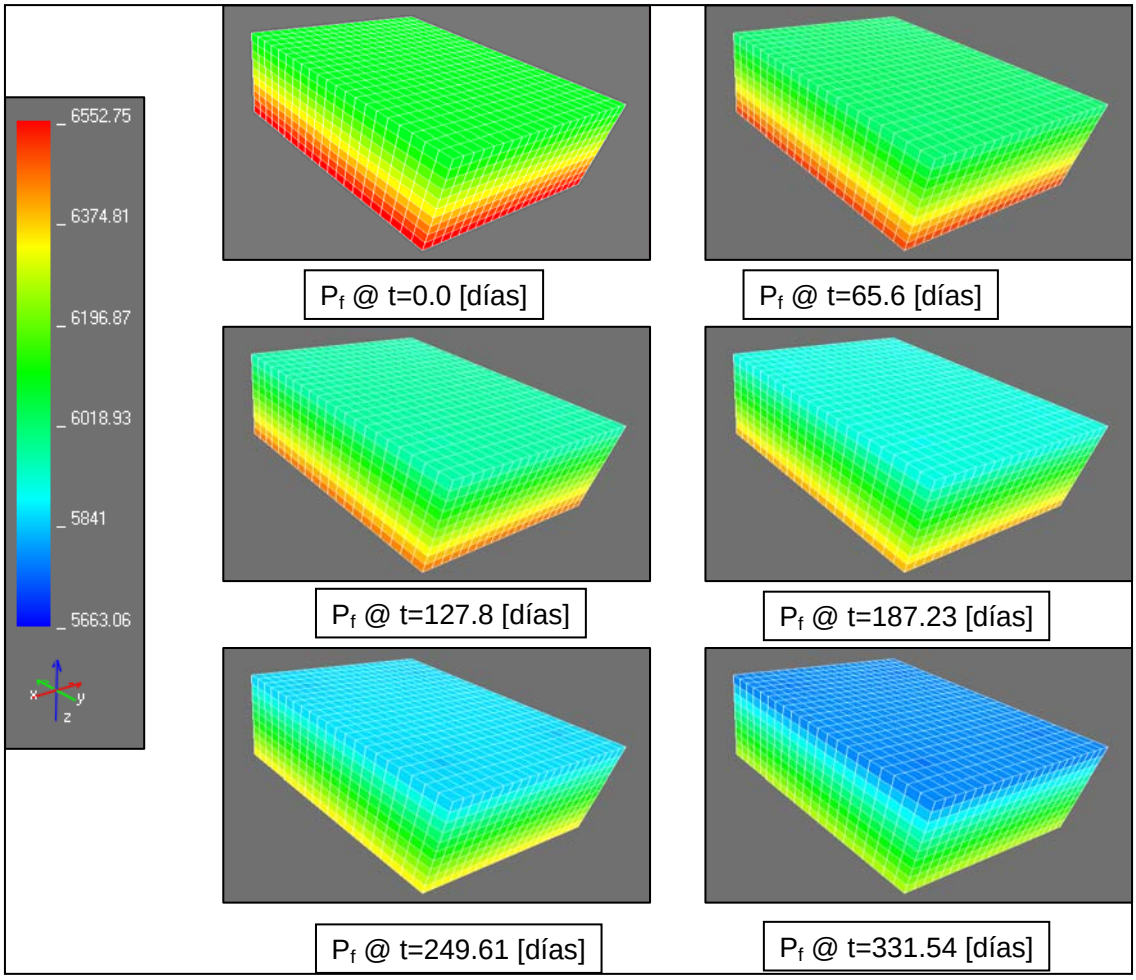


Figura 6.9. Visualización de una malla 21x21x11 en una simulación de 365 días.

Tabla 6.9. Datos empleados para la simulación de la figura 6.9. Características del Yacimiento y fluidos. Malla 21x21x11.

```
Longitud en  X     (pie)      : 15560.0
Longitud en  Y     (pie)      : 20840.0
Longitud en  Z     (pie)      : 1620.0
Numero de nodos en x          : 21
Numero de nodos en y          : 21
Numero de nodos en Z          : 11
Numero de pozos en prod       : 4
Numero de pozos en Iny.       : 0
Tiempo Total de Sim(dias)     : 365.0
permeabilidad matriz  (mD)    : 2.0
perm_fractura en X    (mD)    : 250.0
perm_fractura en Y    (mD)    : 250.0
perm_fractura en Z    (mD)    : 250.0
viscosidad            (cp)    : 0.5
Gravedad API                  : 40.0
comp fluido        (psi-1)    : 6.9E-06
Comp matriz        (psi-1)    : 1.0E-06
Comp fractura      (psi-1)    : 1.0E-06
porosidad_matriz       (-)    : 0.10
porosidad_fractura     (-)    : 0.03
FVF Base (Pi)                 : 1.1
factor de forma sigma(1/m2)   : 1.0
Presión base (psi)            : 500.0
Presión inicial (psi)         : 6000.0
```

Tabla 6.10. Datos de los pozos para malla 21x21x11.

```
*xp   * yp   * zp   * rwp    * Qop
  4   *  4   * 10   * 0.45   *-10000.0
 18   * 18   * 10   * 0.45   *-10000.0
  4   * 18   * 10   * 0.45   *-10000.0
 18   *  4   * 10   * 0.45   *-10000.0
*
*xi   * yi   * zi   * rwi    * Qoi
```

CAPÍTULO 7. CONCLUSIONES Y RECOMENDACIONES.

Conclusiones

Las conclusiones de este trabajo son las siguientes:

- La paralelización por bloques es más eficiente que la paralelización por diagonales.

- El proceso de paralelización es una alternativa muy eficaz para la simulación de yacimientos a escala de campo, ya que brinda un camino alterno para mejorar la velocidad de procesamiento de cómputo.

- La programación en paralelo además de optimizar el tiempo de cómputo, es útil para solucionar problemas grandes donde el consumo de memoria de los equipos de cómputo representa una desventaja.

- Para el procedimiento propuesto se tiene un mejoramiento en la reducción de tiempo cercano a 1/10. Aunque esto puede mejorar, es necesario considerar desde el inicio la lógica del código bajo ambiente multiproceso.

- La formulación presentada puede ser aplicada fácilmente a simuladores más sofisticados.

Recomendaciones

Como recomendaciones se proponen las siguientes:

- Aplicar formulación de paralelización por bloques en la construcción de matrices Jacobianas de simuladores más complejos.

- Investigar otros procedimientos para optimizar tanto el tiempo como la memoria utilizada.

- Considerar la paralelización de otras partes del programa como la correspondiente al proceso de solución de la matriz Jacobiana.

NOMENCLATURA.

A	área	[L^2]
B_o	factor de volumen del aceite	[bl/bl]
b_o	inverso del factor de volumen del aceite	[bl/bl]
c_r	compresibilidad de la roca	[lb/pg^2]$^{-1}$
c_f	compresibilidad de la formación	[lb/pg^2]$^{-1}$
D	profundidad	[pie]
DD	descomposición de dominio	
g	gravedad	[L/T^2]
k	permeabilidad absoluta	[mD]
k_e	permeabilidad efectiva	[mD]
k_{rg}	permeabilidad relativa a la fase gas	[fracción]
k_{ro}	permeabilidad relativa a la fase aceite	[fracción]
k_{rw}	permeabilidad relativa a la fase agua	[fracción]
L_x	longitud en la dirección x	[pie]
L_y	longitud en la dirección y	[pie]
L_z	longitud en la dirección z	[pie]
Nc	tamaño de la matriz de coeficientes	
p	presión	[lb/pg^2]
Pc	presión capilar	[lb/pg^2]
Pm	presión en el lado del fluido mojante	[lb/pg^2]
Pnm	presión en el lado del fluido no mojante	[lb/pg^2]
p_{ref}	presión de referencia	[lb/pg^2]
p_{wf}	presión de fondo fluyendo	[lb/pg^2]
PP	programación en paralelo	
q	gasto volumétrico	[bl/día]
$\tilde{q}_s$	gasto volumétrico @ CS por unidad de volumen de roca	[L^3/tL^3]
r	dirección radial o coordenada en la dirección r	[pie]
r_w	radio de pozo	[pie]
θ	ángulo de contacto	
ρ	densidad	[M/L^3]
Δp	diferencial de presión	[lb/pg^2]
σ	factor de forma	[1/m^2]
γ	peso específico	[lb/pie^3]

ϕ	porosidad	[fracción]
μ	viscosidad	[cp]
S_g	saturación de la fase gas	[fracción]
S_o	saturación de la fase aceite	[fracción]
S_w	saturación de la fase agua	[fracción]
SNY	simulación numérica de yacimientos	
Δt	intervalo de tiempo	[día]
t	tiempo	[día]
V_p	volumen poroso	[pie^3]
V_r	volumen de roca	[pie^3]
W, Y	auxiliar para el manejo de doble porosidad	
x	dirección *x* o coordenada en la longitud *x*	
y	dirección y o coordenada en la longitud y	
z	dirección *z* o coordenada en la longitud *z*	
xp	coordenada x del pozo productor	
yp	coordenada y del pozo productor	
zp	coordenada z del pozo productor	

Subíndices.

i	relativo a dirección *x* o radial
i+1/2	fronteras de la celda *i*
j	relativo a dirección *y*
k	relativo a dirección *z*
f	fractura
m	matriz.
mf	matriz-fractura.
n	índice de nivel de tiempo
o	fase aceite
r	dirección radial, roca
t	tiempo
x	dirección x
y	dirección y
z	dirección z

REFERENCIAS.

Arana O., Víctor. H., Apuntes de clase de Simulación Matemática de Yacimientos, Facultad de Ingeniería, UNAM. Semestre 2005-II.

Arellano G., Javier, Apuntes de clase de Geología de Yacimientos de Fluidos, Facultad de Ingeniería, UNAM. Semestre 2003.II.

Cabrera F., Eduardo César, Gordillo R., José L. y Solís G., Omar Alejandro,
"Programación de Aplicaciones Numéricas Paralelas", V Semana de Supercómputo,

Calhoun, J.C., "A Definition of Petroleum Engineering", JPT (Julio 1963).

Cedillo T., Uriel y Orantes L., Rodrigo, "Descomposición del dominio con programación en paralelo aplicada a la simulación numérica de yacimientos", Tesis Licenciatura (Licenciatura en Ingeniería Petrolera)-UNAM, Facultad de Ingeniería, México: El autor 2005.

Culham, W. et al., "Improved Financial and Operational Forecasting with Large-Scale Reservoir Models", Cray Channels, 1992, pp 26-31.
DGSCA – UNAM, México, D.F.

Dogru, Ali H., "Megacell Reservoir Simulation", SPE Distinguished Author Series, mayo 2000, 54-60 pp.

Dogru, Ali H., Sunaidi, H.A., Habiballah, W.A., Fung, L., Al-Zamel, N., Shin, D., "A Massively Parallel Reservoir Simulator for Large Scale Reservoir", SPE.

Elkins, L., "Some Important Precepts in the Practice of Reservoir Engineering", SPE 1963.

Essley, P.L., "What is Reservoir Engineering?", JPT (Enero 1965), 19-25pp.

González-Escribano, A., Llanos, D., Orden, D. y Palop, B., "Parallelization Alternatives and their Performance for Convex Hull Problem", Applied Mathematical Modelling #30, 2006, 563-577 pp.

Gordillo R., José L., "Taller de Envío de Mensajes (MPI)", Clusters de Alto Rendimiento, DGSCA-UNAM, México D.F., junio 2001.

Gropp, W., Lusk, E. y Skjellum, A., "Using MPI", The MIT Press Cambridge, Massachusetts, Londres Inglaterra, 2000.

Habiballah, W.A., Hayder, M.E., Khan, M.S., Issa, K.M. Zahrani, S.H. Shaikh, R.A. Uwaiyedh, A.H. Tyraskis, T.P. y **Baddourah, M.A.,** "Parallel Reservoir Simulation Utilizing PC-Clusters in Massive Reservoir Simulation Models", SPE, 2003.

Hemanth-Kumar, K. y **Young, L.C.,** "Parallel Reservoir Simulator Computations", SPE Computer Applications, Abril 1996, 50-53 pp.

Henn, N., Quintard, M., Bourbiaux, B. y Sakthikumar, S., "Modelling of Conductive Faults with a Multiscale Approach", Oil & Gas Science and Technology – Rev. IFP, Vol. 59 (2004), No. 2, pp. 197-214

Hernández R., Orlando y Del Valle M., Paul, "Simulador Numérico de Aceite con interfaz de visualización para uso académico", Tesis Licenciatura (Licenciatura en Ingeniería Petrolera)- UNAM, Facultad de Ingeniería, 4-5 pp., México: El autor 2005.

Killough, John E., "Will Parallel Computing Ever Be Practical?", SPE, 1993.

Nolen, J. S. y **Stanat, P. L.,** "Reservoir Simulation on Vector Processing Computers". SPE, 1981.

Odeh, A.S.: "Reservoir Simulation - What Is It? ", *J. Pet. Tech.,* (Nov. 1969) 1383-88 pp.

Ortega A., Jorge L. y Roberts, Graham, "Architectural Patterns for Parallel Programming", Tesis, Department of Computer Science, University of London, Londres Inglaterra, Septiembre 1998.

Ortega A., Jorge L., "The Shared Resource Pattern, An Activity Parallelism Architectural Pattern for Parallel Programming", Departamento de Matemáticas, Facultad de Ciencias, UNAM, México D.F. 2003.

Pancake, Cherri M., "Is Parallelism for You?", Oregon State University, IEEE Computational Science & Engineering, 1996.

Wheeler, J. A. y **Smith, R. A.,** "Reservoir Simulation on a Hypercube", SPE, 1989.

Wyllie, M.R.J., "Reservoir Mechanics- Stylized Myth or Potential Science?", JPT (Junio 1962).

Zhiyuan, M. y **Fengjiang, J.,** "Simulation of Black oil Reservoir on Distributed Memory Parallel Computers and Workstation Cluster", SPE, 1995.

APENDICE A. ECUACIÓN GENERAL DE FLUJO DE FLUIDOS EN MEDIOS POROSOS.

Para obtener la ecuación de flujo monofásico en medios porosos considere un volumen elemental representativo del medio poroso, en donde existe flujo unidimensional. Con base en el principio de la conservación de la materia, se puede establecer el siguiente balance de masa en el elemento.

La masa deberá conservarse a través de todo el sistema.

Considérese la conservación de la masa en coordenadas rectangulares y sólo en la dirección X.

Ritmo de entrada de masa del aceite (o) al volumen de control =

$$\left(A\rho_o v_{ox}\right)\Big|_x \qquad \text{A.1}$$

Ritmo de salida de masa del aceite (o) del volumen recontrol =

$$\left(A\rho_o v_{ox}\right)\Big|_{x+\Delta x} \qquad \text{A.2}$$

Donde *A* es el área transversal del elemento expuesta al flujo, ρ_o es la densidad volumétrica del aceite y v_{ox} es la velocidad macroscópica del aceite en la dirección x.

El ritmo de acumulación de masa en el volumen de control está dado por el ritmo de cambio de la masa del aceite.

Ritmo de acumulación de masa en el volumen de control =

$$A\Delta x \frac{\partial}{\partial t}\left(\phi S_o \rho_o\right) \qquad \text{A.3}$$

El ritmo de producción/inyección de masa de aceite del volumen de control, definiendo $\hat{q}_o$ = gasto volumétrico del aceite a condiciones de yacimiento por unidad de volumen de roca es:

Ritmo de producción/inyección del aceite =

$$A\Delta x\left(\rho_o \hat{q}_o\right) \qquad \text{A.4}$$

Sustituyendo las ecuaciones A.1 a A.4 en el postulado de conservación de la materia, dividiendo la expresión resultante entre el volumen de control, $A\Delta x$, rearreglando y tomando límites cuándo $\Delta x \to 0$, se llega a

$$-\frac{\partial}{\partial x}\left(\rho_o v_{ox}\right)+\rho_o \hat{q}_o=\frac{\partial}{\partial t}\left(\phi\rho_o S_o\right) \qquad \text{A.5}$$

La ecuación A.5 puede extenderse al caso de flujo multidimensional. En notación del operador nabla ∇ estas ecuaciones pueden escribirse en forma más general como:

$$-\nabla\cdot\left(\rho_o v_o\right)+\rho_o \hat{q}_o=\frac{\partial}{\partial t}\left[\phi\left(\rho_o S_o\right)\right] \qquad \text{A.6}$$

Para flujo laminar en el yacimiento, podemos expresar la velocidad del aceite mediante la ecuación de Darcy

$$v_o=-\frac{kk_{ro}}{\mu_o}\left(\nabla p_o-\gamma_o\nabla D\right) \qquad \text{A.7}$$

Sustituyendo la ecuación (A.7) en la ecuación (A.6) se tiene lo siguiente

$$\nabla\cdot\left[\frac{kk_{ro}\rho_o}{\mu_o}\left(\nabla p_o-\gamma_o\nabla D\right)\right]+\hat{q}_o=\frac{\partial}{\partial t}\left[\phi\left(\rho_o S_o\right)\right] \qquad \text{A.8}$$

Como $k_{ro}=1$ y $S_o=1$ (debido a que es una sola fase), tenemos entonces

$$\nabla\cdot\left[\frac{k\rho_o}{\mu_o}\left(\nabla p_o-\gamma_o\nabla D\right)\right]+\hat{q}_o=\frac{\partial}{\partial t}\left[\phi\rho_o\right] \qquad \text{A.9}$$

Recordando que el operador nabla ∇ se define, en el sistema de coordenadas cartesianas, como

$$\nabla=\frac{\partial}{\partial x}i+\frac{\partial}{\partial y}j+\frac{\partial}{\partial z}k$$

Ahora, considerando que es flujo e una sola dirección, tenemos que

$$\frac{\partial}{\partial x}\left[\frac{k\rho}{\mu}\frac{\partial p}{\partial x}\right] \pm \hat{q}_o = \frac{\partial}{\partial t}(\phi\rho) \qquad \text{A.10}$$

Que es la ecuación que gobierna el flujo de una sola fase en una dimensión en forma horizontal asumiendo que aplica la ley de Darcy.

Si aplicamos la derivación de un producto de funciones $\frac{\partial(uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$ entonces

$$\frac{k\rho}{\mu}\frac{\partial}{\partial x}\frac{\partial p}{\partial x} + \frac{k}{\mu}\frac{\partial p}{\partial x}\frac{\partial \rho}{\partial x} + \hat{q}_o = \left[\phi\frac{\partial p}{\partial t} + \rho\frac{\partial \phi}{\partial t}\right]$$

Agrupando y aplicando la regla de la cadena

$$\frac{k\rho}{\mu}\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right) + \frac{k}{\mu}\frac{\partial p}{\partial x}\frac{\partial \rho}{\partial p}\frac{\partial p}{\partial x} + \tilde{q}_o = \left[\phi\frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} + \rho\frac{\partial \phi}{\partial p}\frac{\partial p}{\partial t}\right]$$

$$\frac{k\rho}{\mu}\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right) + \frac{k}{\mu}\frac{\partial \rho}{\partial p}\left(\frac{\partial p}{\partial x}\right)^2 + \tilde{q}_o = \left[\phi\rho\left(\frac{1}{\rho}\frac{\partial \rho}{\partial p} + \frac{1}{\phi}\frac{\partial \phi}{\partial p}\right)\frac{\partial p}{\partial t}\right]$$

Como el gradiente de presión es muy pequeño se puede eliminar y, $c_r = \frac{1}{\phi}\frac{\partial \phi}{\partial p}$

$c_f = \frac{1}{\rho}\frac{\partial \rho}{\partial p}$ y $c_t = c_r + c_f$ entonces

$$\frac{k\rho}{\mu}\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right) + \hat{q}_o = \left[\phi\rho c_t\frac{\partial p}{\partial t}\right] \qquad \text{A.11}$$

Dividiendo por la densidad a condiciones estándar

$$\frac{k\rho}{\mu\rho_s}\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right)+\frac{\hat{q}_o}{\rho_s}=\left[\phi\frac{\rho}{\rho_s}c_t\frac{\partial p}{\partial t}\right]$$

Si $B=\dfrac{\rho_s}{\rho}$ tenemos que

$$\frac{k}{\mu B}\frac{\partial}{\partial x}\left(\frac{\partial p}{\partial x}\right)+\hat{q}_s=\frac{\phi c_t}{B}\frac{\partial p}{\partial t} \qquad \text{A.12}$$

Donde $\hat{q}_s$ es el gasto volumétrico a condiciones estándar por unidad de volumen de roca

$$\tilde{q}_s\left[\frac{L^3}{tL^3}\right]=\frac{\hat{q}_o\left[\dfrac{m}{tL^3}\right]}{\rho_s\left[\dfrac{m}{L^3}\right]}$$

Si k y μ son constantes, se puede expresar de la siguiente manera.

$$\frac{\partial^2 p}{\partial x^2}=\frac{\mu}{k}\left[\phi c_t\frac{\partial p}{\partial t}-\tilde{q}_s\right] \qquad \text{A.13}$$

APÉNDICE B. MODELO DE WARREN Y ROOT.

El modelo de *Warren y Root*, presenta el yacimiento naturalmente fracturado como un sistema idealizado formado por paralelepípedos rectangulares idénticos separados por un a red ortogonal de fracturas. Se considera que el flujo hacia el pozo existe solo en la red de fracturas, mientras que la matriz alimenta continuamente al sistema de fracturas.

En un yacimiento de doble porosidad, una matriz porosa de baja conductividad se encuentra adyacente a un medio de alta conductividad, las fracturas. En este modelo los fluidos fluyen hacia el pozo a través de las fracturas, mientras que los bloques de matriz las alimentan continuamente, **figura B.1**.

Los efectos de doble porosidad están descritos en términos de dos parámetros adimensionales, λ y ω, que relacionan las propiedades de la matriz con las propiedades de la fractura. λ depende de la relación de transmisibilidades matriz-fractura, mientras que ω relaciona la capacidad de almacenamiento de la fractura con la capacidad de almacenamiento del sistema completo. λ y ω estan definidas por las ecuaciones (B.1) y (B.2).

$$\lambda = \alpha \frac{k_m}{k_f} r_w^2 \qquad \text{(B.1)}$$

$$\omega = \frac{\phi_f c_{tf}}{\phi_f c_{tf} + \phi_m c_{tm}} \qquad \text{(B.2)}$$

α es un factor de forma que depende de la geometría de flujo entre la matriz y las fracturas.

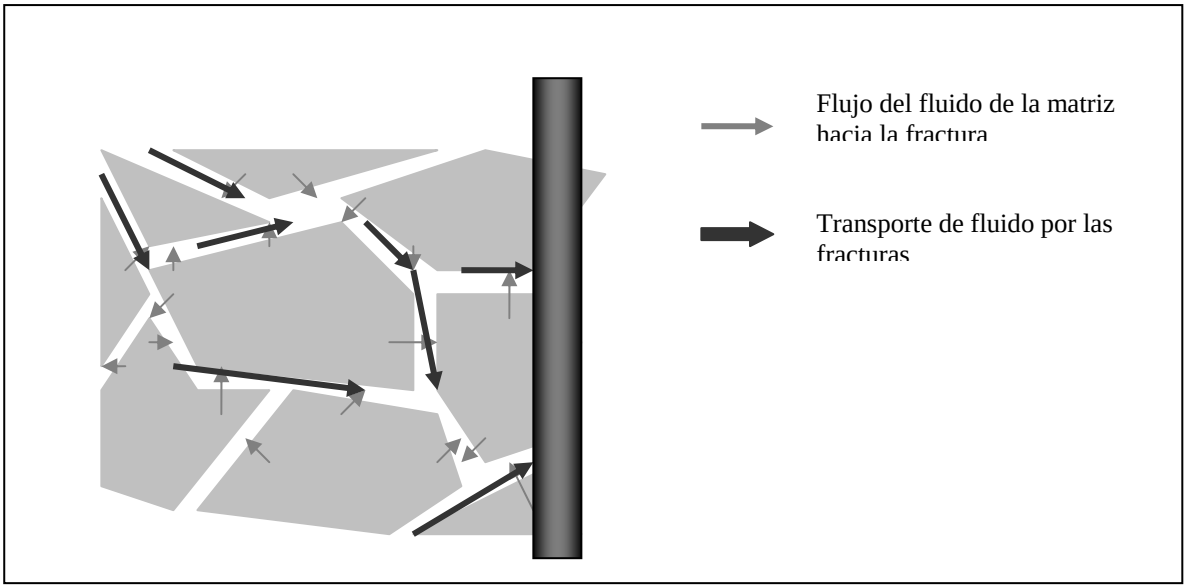


Figura B.1. Esquema del flujo de los fluidos en un yacimiento naturalmente fracturado.

Las expresiones en forma adimensional que describen el flujo en medio de doble porosidad obtenidas por Warren y Root fueron:

$$\frac{\delta^2 P_{fD}}{\delta r_D^2} + \frac{1}{r_D}\frac{\delta P_{fD}}{\delta r_D} = (1-\omega)\frac{\delta P_{mD}}{\delta t_D} + \omega\frac{\delta P_{fD}}{\delta t_D} \qquad \text{(B.3)}$$

para el medio continuo, es decir la fractura, y

$$(1-\omega)\frac{\delta P_{mD}}{\delta t_D} = \lambda\left(P_{fD} - P_{mD}\right) \qquad \text{(B.4)}$$

para el medio discontinuo, es decir, la matriz.

La solución de estas ecuaciones para un yacimiento infinito y para un yacimiento finito con un pozo productor a gasto constante es

$$P_{wD} = \frac{1}{2}\left[Ln\, t_D + 0.80908 + E_i\left(\frac{-\lambda t_D}{\omega(1-\omega)}\right) - E_i\left(\frac{-\lambda t_D}{(1-\omega)}\right)\right] + S \qquad \text{(B.5)}$$

para un yacimiento infinito, y

$$P_{wD} = \left(\frac{2}{r_D^2 - 1}\right)\left[\frac{1}{4} + t_D + \frac{(1-\omega)^2}{\lambda}\left(1 - e^{\frac{-\lambda tD}{\omega(1-\omega)}}\right)\right] - \frac{3r_D^4 - 4r_D^4 Ln\, r_D - 2r_D^2 - 1}{4\left(r_D^2 - 1\right)^2} \qquad \text{(B.6)}$$

para una yacimiento finito, donde r_D y t_D son el radio y tiempo adimensionales, definidos por

$$r_D = \frac{r}{r_w} \qquad \text{(B.7)}$$

$$t_D = \frac{0.0002637 k_f t}{\left[(\phi c_t)_f + (\phi c_t)_m\right]\mu\, r_w^2} \qquad \text{(B.8)}$$

En la ecuación (B.5), el factor *S* de representa el factor de daño y el termino $E_i(x)$ representa la función Integral Exponencial.

La presión real puede ser obtenida de la definición de presión adimensional

$$P_D = \frac{k_f h\left[P_i - P(r,t)\right]}{141.2 qB\mu} \qquad \text{(B.9)}$$

Una gráfica semilog de la presión Vs. tiempo revela la presencia de dos líneas rectas paralelas en lugar de una, como lo obtenido generalmente para el caso de un yacimiento convencional, **figura B.2**. Dado que el sistema de fracturas es de alta transmisibilidad y esta directamente conectado al pozo, este responde primero; la matriz no esta conectada de

forma directa al pozo y es de baja transmisibilidad, por lo tanto responde después. La separación entre las dos rectas depende de la capacidad de almacenamiento relativa, **ω**, mientras que el periodo de transición de la presión de la primera a la segunda línea será una función la relación de transmisibilidades, **λ**.

Los valores de **ω** pueden ser menores o iguales a uno. El caso especial de **ω** = 1 ocurre cuando la porosidad de la matriz es cero, en este caso se tiene un comportamiento como el de un yacimiento convencional (de una porosidad). En yacimientos naturalmente fracturados, ϕ_f es generalmente muy pequeña, sin embargo, la alta compresibilidad de la fractura, c_{tf}, resulta, frecuentemente, en valores menores a 0.1 para **ω**. Los valores de **λ** habitualmente se encuentran en el rango de 10^{-3} a 10^{-10}. Valores mayores a 10^{-3} para **λ** significan que la heterogeneidad del yacimiento es insuficiente para que los efectos de doble porosidad sean de importancia, y nuevamente se tiene un comportamiento de un yacimiento convencional.

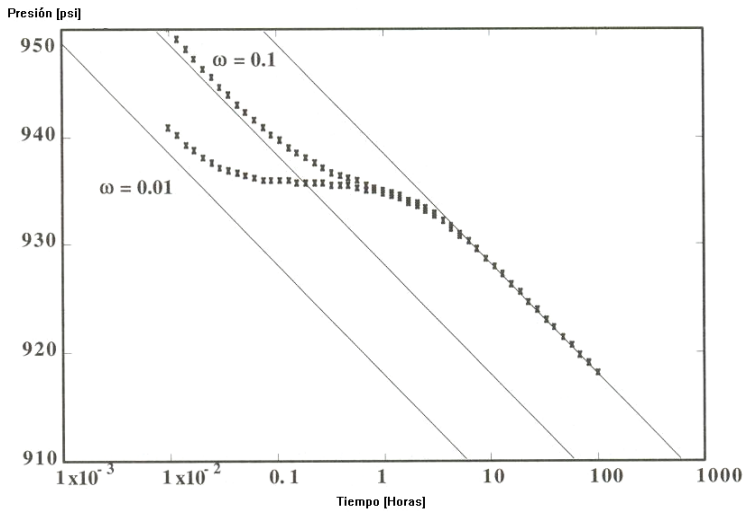


Figura.B.2. Comportamiento de la presión en un yacimiento naturalmente fracturado.

APÉNDICE C. SUBRUTINA DEL SIMULADOR EN PARALELO.

Módulo Principal.

```
   PROGRAM aaa3D
!    Yacimiento 3D(x,y,z). Solución numérica
!    Last change:  LFA   5 Jul 2006    6:44 pm

     USE input_data
     USE GMRES_module

     IMPLICIT NONE
     Integer          :: i,j,k,m, stat, STATUS
     CHARACTER(LEN=10) :: time,date
     REAL(KIND=dp)::tol_lu, toler,dtmax, dtinc

     INCLUDE "mpif.h"   ! incluimos las variables necesarias en mpi

     REAL(KIND=dp), ALLOCATABLE, DIMENSION(:)     :: RARRAY, RARRAY_PRO, RARRAY_INJ

     INTEGER, ALLOCATABLE, DIMENSION(:)      ::  IARRAY, IARRAY_PRO, IARRAY_INJ

     call MPI_INIT(IERR)
     CALL MPI_COMM_RANK(MPI_COMM_WORLD,RANGO,IERR)
     CALL MPI_COMM_SIZE(MPI_COMM_WORLD,NP,IERR)

     ALLOCATE (IARRAY(5))
     ALLOCATE (RARRAY(19))

  IF (RANGO.EQ.0) THEN
     PRINT*," "
     PRINT*,"          Any question or request"
     PRINT*,"          contact"
     PRINT*,"          Victor H Arana  at"
     PRINT*,"          vharana@yahoo.com"
     PRINT*,"          UNAM, Mexico City"

! Primero leera las variables y despues las dimencionará

     CALL Reading

     !Some variables
     ne=1
     Nc=ne*nx*ny*nz
     !This vector is used by NSPIV
     max=Nc*Nc !I need to figure it out

      ! Dx y Dy are constants. The distans between  node 1 and node Nx,Ny
      ! is Lx and Ly, respectivaly.

      RARRAY(1) = Lx
      RARRAY(2) = Ly
      RARRAY(3) = Lz
      RARRAY(4) = Tiempo_t
      RARRAY(5) = Perm_mat
      RARRAY(6) = Permx_frac
      RARRAY(7) = Permy_frac
      RARRAY(8) = Permz_frac
      RARRAY(9) = vis
      RARRAY(10) = ro
      RARRAY(11) = cfluid
      RARRAY(12) = cmat
      RARRAY(13) = cfrac
      RARRAY(14) = poro
      RARRAY(15) = poro_frac
      RARRAY(16) = FVF
      RARRAY(17) = shape_factor
      RARRAY(18) = prs_ref
      RARRAY(19) = prs_ini

      IARRAY(1) = Nx
      IARRAY(2) = Ny
      IARRAY(3) = Nz
      IARRAY(4) = wpro
      IARRAY(5) = winj
  END IF

   CALL MPI_BCAST(IARRAY, 5, MPI_INTEGER, 0, MPI_COMM_WORLD, ierr)
```

```
 CALL MPI_BCAST(RARRAY, 19, MPI_DOUBLE_PRECISION, 0, MPI_COMM_WORLD, ierr)

IF (RANGO.NE.0) THEN
     Lx = RARRAY(1)
     Ly = RARRAY(2)
     Lz = RARRAY(3)
     tiempo_t = RARRAY(4)
     perm_mat = RARRAY(5)
     permx_frac = RARRAY(6)
     permy_frac = RARRAY(7)
     permz_frac = RARRAY(8)
     vis = RARRAY(9)
     ro = RARRAY(10)
     cfluid = RARRAY(11)
     cmat = RARRAY(12)
     cfrac = RARRAY(13)
     poro = RARRAY(14)
     poro_frac = RARRAY(15)
     FVF = RARRAY(16)
     shape_factor = RARRAY(17)
     prs_ref = RARRAY(18)
     prs_ini = RARRAY(19)

     Nx = IARRAY(1)
     Ny = IARRAY(2)
     Nz = IARRAY(3)
     wpro = IARRAY(4)
     winj = IARRAY(5)
END IF

 ALLOCATE(CELDA(5))
  Inc = INT( (Nx*Ny*Nz)/Np )
  m = 0
  DO k=1,Nz
     DO j=1, Ny
        DO i=1, Nx
           m = m+1
           IF (m.EQ.Inc*rango+1) THEN
              CELDA(1) = i
              CELDA(2) = j
              CELDA(3) = k
              CELDA(4) = m
              CELDA(5) = Inc*(rango+1)
              exit
           END IF
        END DO
        IF (m.EQ.Inc*rango+1) EXIT
     END DO
     IF (m.EQ.Inc*rango+1) EXIT
  END DO

   If (Rango.EQ.Np-1) celda(5) = Nx*Ny*Nz

   m2 = celda(5)
   m1 = celda(4)

 ALLOCATE(m_v(2))

 if (rango==(np-1)) then
   m_v(2) = m2
   m_v(1) = m1
 end if

  call mpi_bcast(m_v, 2, mpi_integer, np-1, mpi_comm_world, ierr)

 mf2 = m_v(2)
 mf1 = m_v(1)

 if (rango==0) ALLOCATE(coef_aux(mf2-mf1+4))

 ALLOCATE (RARRAY_PRO(4*WPRO))
 ALLOCATE (RARRAY_INJ(4*WINJ))
 ALLOCATE (IARRAY_PRO(3*WPRO))
 ALLOCATE (IARRAY_INJ(3*WINJ))

IF (RANGO.EQ.0) THEN
     DO i=1, wpro
        IARRAY_PRO(i*3-2) = xp(i)
        IARRAY_PRO(i*3-1) = yp(i)
        IARRAY_PRO(i*3) = zp(i)
```

```
      END DO

      DO i=1, WINJ
         IARRAY_INJ(i*3-2) = xi(i)
         IARRAY_INJ(i*3-1) = yi(i)
         IARRAY_INJ(i*3) = zi(i)
      END DO

      DO i=1, WPRO
         RARRAY_PRO(i*4-3) = rwp(i)
         RARRAY_PRO(i*4-2) = qop(i)
         RARRAY_PRO(i*4-1) = tin_p(i)
         RARRAY_PRO(i*4) = tfi_p(i)

      END DO

      DO i=1, WINJ
         RARRAY_INJ(i*4-3) = rwi(i)
         RARRAY_INJ(i*4-2) = qoi(i)
         RARRAY_INJ(i*4-1) = tin_i(i)
         RARRAY_INJ(i*4) = tfi_i(i)
      END DO
   END IF
   CALL MPI_BCAST(IARRAY_PRO,SIZE(IARRAY_PRO),MPI_INTEGER,0,MPI_COMM_WORLD,ierr)
     CALL MPI_BCAST(IARRAY_INJ,SIZE(IARRAY_INJ),MPI_INTEGER,0,MPI_COMM_WORLD,ierr)
   CALL MPI_BCAST(RARRAY_PRO,SIZE(RARRAY_PRO),MPI_DOUBLE_PRECISION,0,
                           MPI_COMM_WORLD, ierr)
    CALL MPI_BCAST(RARRAY_INJ, SIZE(RARRAY_INJ), MPI_DOUBLE_PRECISION, 0,
                           MPI_COMM_WORLD, ierr)

   IF (RANGO.NE.0) THEN
      include "all_arrays.inc"

      DO i=1, wpro
         xp(i) = IARRAY_PRO(i*3-2)
         yp(i) = IARRAY_PRO(i*3-1)
         zp(i) = IARRAY_PRO(i*3)
      END DO

      DO i=1, WINJ
         xi(i) = IARRAY_INJ(i*3-2)
         yi(i) = IARRAY_INJ(i*3-1)
         zi(i) = IARRAY_INJ(i*3)
      END DO

      DO i=1, WPRO
         rwp(i) = RARRAY_PRO(i*4-3)
         qop(i) = RARRAY_PRO(i*4-2)
         tin_p(i) = RARRAY_PRO(i*4-1)
         tfi_p(i) = RARRAY_PRO(i*4)
      END DO

      DO i=1, WINJ
         rwi(i) = RARRAY_INJ(i*4-3)
         qoi(i) = RARRAY_INJ(i*4-2)
         tin_i(i) = RARRAY_INJ(i*4-1)
         tfi_i(i) = RARRAY_INJ(i*4)
      END DO
   END IF

      ! Setting Dx,Dy
      CALL nodos

      ! Initial DT
      DT=1.0     !Seconds
      dtmax=30.0d0*86400.0d0

      !Initial Conditions
      CALL CIniciales

      !Boundary Conditions
      CALL CFrontera

      ! Accumulative Time
      T_acum=0.0D0
      TOLER=1.0d-11
     TOL_LU=1.0d-2
      Timp = 110.0*sec_day
       NIMP  = 1
      nts=0
```

```
      dtinc=1.1

   DO WHILE (T_acum < Tiempo_t)
!     DO WHILE (nts < 10)
       nts=nts+1
       call Coeficients

! Accumulatibe Time
        T_acum=T_acum + DT  ! En segundos

    IF (RANGO.EQ.0) THEN
      CALL Cpu_time(tt1)
!     call NSPIV(Nc,IA,JA,ARED,B,MAX,X,IERR,ne)
      call SOLIN(Nc,IA,JA,Ared,B,X,iord,riord,NTS,1,TOL_LU,TOLER)
!     Call GMRES(12, X, Nc, IA, JA, ARED, B, TOLER)
      CALL Cpu_time(tt2)

        m =0
       DO k=1,nz
        DO j=1,ny
         DO i=1,nx
          m=m+1
           Prs_new_f(i,j,k)=X(m)
           prs_new_m(i,j,k)=(W(m)*prs_new_f(i,j,k)+Y(m)*prs_old_m(i,j,k))/   &
                      (W(m)+Y(m))
         END DO
        END DO
       END DO

        call results

    WRITE(23,'(5x,i5,20(3x,e14.5))')nts, (vector_tiempo(i),i=0,np-1)!'(5x,i5,7(3x,e14.5))'
    WRITE(24,'(5x,i5,7(3x,e14.5))')nts, T_acum/sec_day, DT/sec_day, tt2-tt1
   END IF

     call mpi_bcast(Prs_new_f, SIZE(Prs_new_f), mpi_double_precision, 0, mpi_comm_world, ierr)
     call mpi_bcast(Prs_new_m, SIZE(Prs_new_m), mpi_double_precision, 0, mpi_comm_world, ierr)

        !New DT
       DT = DT*DTinc

       IF (DT.GT.DTMAX) DT = DTMAX
       IF (T_acum+Dt .GT. Tiempo_t) DT = Tiempo_t-T_acum

        !New Dt, Now P @ n+1 becomes P @ n
       do k=1,nz
        do j=1,ny
          do i=1,nx
           prs_old_f(i,j,k)=prs_new_f(i,j,k)
           prs_old_m(i,j,k)=prs_new_m(i,j,k)
          end do
        end do
       end do

       !Next DT y T_acum
 END DO !Do while

    call mpi_barrier(mpi_comm_world, ierr)
    CALL MPI_FINALIZE(IERR)
     END program aaa3D
```

Módulo de la Construcción de Coeficientes.

```
    MODULE input_data
!     Entrada de Datos
!     Last change:  LFA   5 Jul 2006    6:49 pm

      USE Constants
      USE my_library

      IMPLICIT NONE

      ! Allocating arrays
      Include "Arrays.inc"

      REAL(KIND=dp) :: mt_1,mt_2, mt_3,mt_t1, mt_t2, mt_t3, tt1, tt2
      CHARACTER(LEN=23) :: OPF
      CONTAINS
```

```
!_____________________________________________________
      SUBROUTINE Reading

      integer          :: i,j,k,STATUS
!
      OPEN(UNIT=2,FILE ='3Dxy0100-4.dat',STATUS='unknown')
      OPEN(UNIT=3,FILE ='Wells0100.dat',STATUS='unknown')
      READ(2,100)Lx,Ly,Lz,nx,ny,nz,wpro,winj,Tiempo_t,perm_mat,permx_frac,permy_frac,permz_frac,
                 Vis,API,Cfluid,Cmat,Cfrac,poro,poro_frac,FVF,shape_factor,Prs_ref,Prs_ini

  100 FORMAT(30x    ,/,          &   ! salta un renglon
           30x,F12.3,/,          &   ! Long in X          (real)
           30x,F12.3,/,          &   ! Long in Y          (real)
           30x,F12.3,/,          &   ! Long in Z          (real)
           30x,I4  ,/,           &   ! Numero de nodos X   (integer)
           30x,I4  ,/,           &   ! Numero de nodos Y  (integer)
           30x,I4  ,/,           &   ! Numero de nodos Z   (integer)
           30x,I4  ,/,           &   ! Numero de pozos Pro (integer)
           30x,I4  ,/,           &   ! Numero de pozos iny (integer)
           30x,F12.3,/,          &   ! Tiempo total       (real)
           30x,F12.3,/,          &   ! Permeabilidad matriz  (real)
           30x,F12.3,/,          &   ! Perm_fractura en x  (real)
           30x,F12.3,/,          &   ! Perm_fractura en Y  (real)
           30x,F12.3,/,          &   ! Perm_fractura en Z  (real)
           30x,F12.3,/,          &   ! Viscosidad         (real)
           30x,F12.3,/,          &   ! API               (real)
           30x,E12.3,/,          &   ! Compr fluido       (cientifica)
           30x,E12.3,/,          &   ! Compr matriz       (cientifica)
           30x,E12.3,/,          &   ! Compr fractura     (cientifica)
           30x,F12.3,/,          &   ! Porosidad          (real)
           30x,F12.3,/,          &   ! Poro_fractura      (real)
           30x,F12.3,/,          &   ! FVF,               (real)
           30x,F12.3,/,          &   ! Factor de forma sigma, (real)
           30x,F12.3,/,          &   ! Presion de ref     (real)
           30x,F12.3 )               ! Presion inicial    (real)

      Write(*,100)Lx,Ly,Lz,nx,ny,nz,wpro,winj,Tiempo_t,perm_mat,  &
                  permx_frac,permy_frac,permz_frac,           &
                  vis,API,Cfluid,Cmat,Cfrac,poro,poro_frac,FVF,shape_factor,Prs_ref,Prs_ini

      WRITE(OPF,"(A,I4.4,A,I2.2,A)") "3Dxy_par",Nx-1,"-",Np,".res"
      OPEN(UNIT=12,FILE=OPF,STATUS='unknown')

      WRITE(OPF,"(A,I4.4,A,I2.2,A)") "3DPwf_par",Nx-1,"-",Np,".res"
      OPEN(UNIT=22,FILE=OPF,STATUS='unknown')

      WRITE(OPF,"(A,I4.4,A,I2.2,A)") "tiempo_par",Nx-1,"-",Np,".res"
      OPEN(UNIT=23,FILE=OPF,STATUS='unknown')

      WRITE(OPF,"(A,I4.4,A,I2.2,A)") "tiempo2_par",Nx-1,"-",Np,".res"
      OPEN(UNIT=24,FILE=OPF,STATUS='unknown')

      WRITE(23,*)"       tiempos"
      WRITE(23,*)" iteracion        t_tot_de_creacion"

      !Setting dimensions
      include "all_arrays.inc"

      ! Producing wells
        READ(3,*)
        READ(3,*)
      DO i=1,wpro
        READ(3,102)xp(i),yp(i),zp(i),rwp(i),qop(i),Tin_p(i),Tfi_p(i)
      END DO
      ! Injecting wells
        READ(3,*)
        READ(3,*)
      DO i=1,winj
        READ(3,102)xi(i),yi(i),zi(i),rwi(i),qoi(i),Tin_i(i),Tfi_i(i)
      END DO

  102 FORMAT(3(I6,1x),4(F8.2,1x))

      !Change of units to SI
      call Cambio_unidades

      END subroutine Reading
!_____________________________________________          _____________
      SUBROUTINE nodos
```

```
      DX=LX/NX
      DY=LY/NY
      DZ=LZ/NZ
      Vr=DX*DY*DZ

     END SUBROUTINE Nodos
!_____________________________________________________________
     SUBROUTINE CIniciales

      Integer       :: i,j,k,m

      m=0
      DO k=Nz,1,-1
      !m=m+1 ! CAMBIO REALIZADO: JUEVES 24 FEB 2005
       DO j=1,Ny
        DO i=1,Nx
          depth(i,j,k)=0.0D0
          if (k==Nz)then
            depth(i,j,k)= DZ/2.0D0
          else
            depth(i,j,k)= (DZ/2.0D0)+m*DZ
          endif
        END DO
       END DO
      m=m+1   ! CAMBIO REALIZADO: JUEVES 24 FEB 2005
      END DO

      DO k=Nz,1,-1
       DO j=1,Ny
        DO i=1,Nx
         prs_old_m(i,j,k)= Prs_ini+ro*depth(i,j,k)
         prs_old_f(i,j,k)= Prs_ini+ro*depth(i,j,k)
         Km(i,j,k)=perm_mat
         Kfx(i,j,k)=permx_frac
         Kfy(i,j,k)=permy_frac
         Kfz(i,j,k)=permz_frac
         bo(i,j,k)=(1.0/FVF) ! Be carefully with b
         fi_m(i,j,k)=poro
         fi_f(i,j,k)=poro_frac
         Vo(i,j,k)=Vis

        END DO
       END DO
      END DO
     END SUBROUTINE CIniciales
!_____________________________________________________________
     SUBROUTINE CFrontera

      Integer       :: i,j,k,m

      DO k=1,Nz
       DO j=1,NY
        do i=1,Nx
         Qo(i,j,k)=0.0D0
          !Boundary Internal conditions

          DO m=1,wpro
            IF(i==xp(m).and.j==yp(m).and.k==zp(m)  )then
              Qo(i,j,k)= Qop(m)
            END if
          END DO

          DO m=1,winj
            IF(i==xi(m).and.j==yi(m).and.k==zi(m)  )  Qo(i,j,k)= Qoi(m)
          END DO

        enddo
       END DO
      END DO

     END SUBROUTINE CFrontera
!_____________________________________________________________
     SUBROUTINE PVT_Rock
       Integer       :: i,j,k

     DO k=1,Nz
       DO J=1,Ny
         DO i=1,Nx
        fi_m(i,j,k) =poro*(1.0+Cmat*(Prs_old_m(i,j,k)-Prs_ref))
        Dfi_m(i,j,k)=poro*Cmat
```

```
      bo(i,j,k) =(1.0/FVF)*(1.0+Cfluid*(Prs_old_m(i,j,k)-Prs_ref))
      Dbo(i,j,k)=bo(i,j,k)*Cfluid
      fi_f(i,j,k) =poro_frac*(1.0+Cfrac*(Prs_old_f(i,j,k)-Prs_ref))
      Dfi_f(i,j,k)=poro_frac*Cfrac
    END DO                                                                    !X
  END DO                                                                      !Y
END DO                                                                        !Z
    END SUBROUTINE PVT_Rock
!_____________________________________________________
    SUBROUTINE Coeficients

    integer      :: i,j,k,i1,j1,k1,m,n,row,jrow,col,eq,STATUS,source
    REAL(KIND=dp) :: aux01,Auxp,aux25,aux26,Toy1,Toy2,Tox1,Tox2,Area Toz1,Toz2,etq_o,
                     etq_e,etq_s,etq_n,etq_l,etq_u,etq_c,etq_B,etq_w, etq_y

   include "mpif.h"
     INTEGER, DIMENSION (MPI_STATUS_SIZE) :: status_2

    etq_o = 1000
    etq_e = 1001
    etq_s = 1002
    etq_n = 1003
    etq_l = 1004
    etq_u = 1005
    etq_c = 1006
    etq_B = 1007
    etq_w = 1008
    etq_y = 1009

    call PVT_Rock
    call cpu_time(mt_1)
    call cpu_time(mt_t1)

    !Zeros
    Coef_U=0.0D0
    Coef_L=0.0D0
    Coef_s=0.0D0
    Coef_o=0.0D0
    Coef_c=0.0D0
    Coef_e=0.0D0
    Coef_n=0.0D0
    Coef_R=0.0D0

    m=0
    k1 = Celda(3)
    j1 = Celda(2)
    i1 = Celda(1)
    m1 = Celda(4)
    m2 = Celda(5)

    DO k=k1,Nz
      DO J=j1,Ny
        DO i=i1,Nx

         !Contador de celdas
          m=m+1

      ! X - Direction
      ! Evaluación corriente arriba para propiedades

       IF(I /= 1) THEN
        Area=DY*DZ
        auxp=prs_old_f(i,j,k)-prs_old_f(i-1,j,k)
           IF(auxp > 0.0D0)Then
             TOX1=(Area/DX)*kfx(i,j,k)*bo(i,j,k)/Vo(i,j,k)
           else
             TOX1=(Area/DX)*kfx(i-1,j,k)*bo(i-1,j,k)/Vo(i-1,j,k)
           endif
       ELSE
        TOX1=0.0D0
       ENDIF

      ! Evaluación corriente arriba para propiedades

       IF(I /= NX) THEN
         Area=DY*DZ
         auxp=prs_old_f(i+1,j,k)-prs_old_f(i,j,k)
            IF(auxp > 0.0D0)Then
              TOX2=(Area/DX)*Kfx(i+1,j,k)*bo(i+1,j,k)/Vo(i+1,j,k)
            ELSE
```

```
        TOX2=(Area/DX)*Kfx(i,j,k)*bo(i,j,k)/Vo(i,j,k)
      ENDIF
  ELSE
    TOX2=0.0D0
  ENDIF

  ! Y - Direction
  ! Evaluación corriente arriba para propiedades

  IF(J /= 1) THEN
    Area=DX*DZ
    auxp=prs_old_f(i,j,k)-prs_old_f(i,j-1,k)
      IF(auxp > 0.0D0)Then
       TOY1=(Area/DY)*Kfy(i,j,k)*bo(i,j,k)/Vo(i,j,k)
      ELSE
       TOY1=(Area/DY)*Kfy(i,j-1,k)*bo(i,j-1,k)/Vo(i,j-1,k)
      ENDIF
  ELSE
    TOY1=0.0D0
  ENDIF

  ! Evaluación corriente arriba para propiedades

  IF(J /= NY) THEN
    Area=DX*DZ
    auxp=prs_old_f(i,j+1,k)-prs_old_f(i,j,k)
      IF(auxp > 0.0D0)Then
       TOY2=(Area/DY)*Kfy(i,j+1,k)*bo(i,j+1,k)/Vo(i,j+1,k)
      ELSE
       TOY2=(Area/DY)*Kfy(i,j,k)*bo(i,j,k)/Vo(i,j,k)
      ENDIF
  ELSE
    TOY2=0.0D0
  ENDIF

  ! Z - Direction

  IF(k /= 1) THEN
   Area=DX*DY
   auxp=prs_old_f(i,j,k)-prs_old_f(i,j,k-1)+ro*(depth(i,j,k)-depth(i,j,k-1))
     IF(auxp > 0.0D0)Then
       TOZ1=(Area/DZ)*kfz(i,j,k)*bo(i,j,k)/Vo(i,j,k)
     else
       TOZ1=(Area/DZ)*kfz(i,j,k-1)*bo(I,j,k-1)/Vo(i,j,k-1)
     endif
  ELSE
   TOZ1=0.0D0
  ENDIF

 ! Evaluación corriente arriba para propiedades

  IF(k /= NZ) THEN
    Area=DX*DY
    auxp=prs_old_f(i,j,k+1)-prs_old_f(i,j,k)+ro*(depth(i,j,k+1)-depth(i,j,k))
      IF(auxp > 0.0D0)Then
        TOZ2=(Area/DZ)*Kfz(i,j,k+1)*bo(I,j,k+1)/Vo(I,j,k+1)
      ELSE
        TOZ2=(Area/DZ)*Kfz(i,j,k)*bo(i,j,k)/Vo(i,j,k)
      ENDIF
  ELSE
       TOZ2=0.0D0
  ENDIF

  Coef_o(m) = TOX1
  Coef_e(m) = tox2
  coef_s(m) = toy1
  coef_n(m) = toy2
  coef_l(m) = toz1
  coef_u(m) = toz2

  Aux01 = (Vr/Dt)*(fi_f(i,j,k)*Dbo(i,j,k)+bo(i,j,k)*Dfi_f(i,j,k))

!---------here is the key

  W(m) = shape_factor*Vr*km(i,j,k)*bo(i,j,k)/Vo(i,j,k)
  Y(m) = Vr*bo(i,j,k)*fi_m(i,j,k)*(Cmat+cfluid)/dt
  aux25 = (W(m)**2/(Y(m)+W(m)))-W(m)
  aux26 = W(m)*Y(m)/(Y(m)+W(m))

  Coef_c(m)= -(toz1+tox1+toy1+aux01+toy2+tox2+toz2-aux25)!*P i,j
```

```
      Coef_R(m)= - aux01*prs_old_f(i,j,k)-qo(i,j,k)-aux26*prs_old_m(i,j,k)

      !termino de gravedad
      Coef_R(m)=Coef_R(m)+toz2*ro*(depth(i,j,k+1)-depth(i,j,k))-toz1*ro*(depth(i,j,k)-depth(i,j,k-1))

      B(m)=Coef_R(m)

      IF (m.EQ.CELDA(5)-CELDA(4)+1) EXIT
    END DO                                                                    !X
    i1 = 1
    IF (m.EQ.CELDA(5)-CELDA(4)+1) EXIT
  END DO                                                                  !Y
  j1 = 1
  IF (m.EQ.CELDA(5)-CELDA(4)+1) EXIT
 END DO                                                                    !Z

 call cpu_time(mt_2)
 mt_3=mt_2-mt_1

 if (rango.ne.0) then
   coef_o(SIZE(coef_o)  ) = mt_3
   coef_o(SIZE(coef_o)-1) = REAL(m2)
   coef_o(SIZE(coef_o)-2) = REAL(m1)

  CALL MPI_rsend(Coef_o, SIZE(coef_o), mpi_double_precision, 0, etq_o, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_e, SIZE(coef_e), mpi_double_precision, 0, etq_e, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_s, SIZE(coef_s), mpi_double_precision, 0, etq_s, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_n, SIZE(coef_n), mpi_double_precision, 0, etq_n, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_l, SIZE(coef_l), mpi_double_precision, 0, etq_l, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_u, SIZE(coef_u), mpi_double_precision, 0, etq_u, mpi_comm_world, ierr)
  CALL MPI_rsend(Coef_c, SIZE(coef_c), mpi_double_precision, 0, etq_c, mpi_comm_world, ierr)
  CALL MPI_rsend(B, SIZE(B), mpi_double_precision, 0, etq_B, mpi_comm_world, ierr)

  call mpi_rsend(w,  SIZE(w),  mpi_double_precision, 0, etq_w, mpi_comm_world, ierr)
  call mpi_rsend(y,  SIZE(y),  mpi_double_precision, 0, etq_y, mpi_comm_world, ierr)
end if

  call cpu_time(mt_t2)
  mt_3=mt_2-mt_1
  mt_t3=mt_t2-mt_t1

!   Now, I am able to get the array A(l) and solve it by using NSPIV
 if  (rango ==0) then
   vector_tiempo(0)=mt_3
   do i=1,np-1
      call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_o, mpi_comm_world, status_2, ierr)
      m1 = INT(coef_aux(SIZE(coef_aux)-2))
      m2 = INT(coef_aux(SIZE(coef_aux)-1))
      mt_3 = coef_aux(SIZE(coef_aux))
      coef_o(m1:m2) = coef_aux(:(m2-m1+1))

  call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_e, mpi_comm_world, status_2,ierr)
      coef_e(m1:m2) = coef_aux(:(m2-m1+1))

   call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_s, mpi_comm_world,status_2, ierr)
      coef_s(m1:m2) = coef_aux(:(m2-m1+1))

    call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_n, mpi_comm_world, status_2,ierr)
      coef_n(m1:m2) = coef_aux(:(m2-m1+1))

   call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_l, mpi_comm_world, status_2,ierr)
      coef_l(m1:m2) = coef_aux(:(m2-m1+1))

     call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_u, mpi_comm_world, status_2,ierr)
      coef_u(m1:m2) = coef_aux(:(m2-m1+1))

     call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_c, mpi_comm_world, status_2,ierr)
      coef_c(m1:m2) = coef_aux(:(m2-m1+1))

     call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_B, mpi_comm_world, status_2,ierr)
      B(m1:m2) = coef_aux(:(m2-m1+1))

     call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_w, mpi_comm_world, status_2,ierr)
      w(m1:m2) = coef_aux(:(m2-m1+1))

      call mpi_recv(Coef_aux, SIZE(coef_aux), mpi_double_precision, i, etq_y, mpi_comm_world, status_2,ierr)
      y(m1:m2) = coef_aux(:(m2-m1+1))
      vector_tiempo(i)=mt_3
   end do
```

```
    call pointers(ia,ja,nx,ny,nz,ne,coef_e,coef_o,coef_s,coef_n,coef_L,coef_U,coef_c,ARED)
 END IF
END SUBROUTINE Coeficients
!_____________________________________________________
     SUBROUTINE Cambio_unidades

     INTEGER :: i

!    Internationa sytem of units is used
     Vis     = Vis*cp_pa_s                !From cp to Pa*s
     perm_mat=perm_mat*mD_m2          !From mD to m2
     permx_frac  = permx_frac*mD_m2          !From mD to m2
     permy_frac  = permy_frac*mD_m2          !From mD to m2
     permz_frac  = permz_frac*mD_m2          !From mD to m2
     Cmat  = Cmat/pa_psi                   !From psi-1 to Pa-1
     Cfrac  = Cfrac/pa_psi                 !From psi-1 to Pa-1
     Cfluid= Cfluid/pa_psi              !From psi-1 to Pa-1
     Tiempo_t= Tiempo_t*sec_day                !From days to sec
     Prs_ref = Prs_ref*pa_psi           !From psi to Pa
     Prs_ini = Prs_ini*pa_psi           !From psi to Pa
     Lx      = Lx*ft_m                  !From ft to m
     Ly      = Ly*ft_m                  !From ft to m
     Lz      = Lz*ft_m                  !From ft to m
     Ro      = 141.5/(API+131.5)
     Ro      = Ro*1000.0*9.80

     DO i=1,wpro
       rwp(i)=rwp(i)*ft_m            !From ft to m
       qop(i)=qop(i)/(sec_day*bl_m3)  !From Bl/dia to m3/sec
       Tin_p(i)=Tin_p(i)*sec_day      !From days to sec
       Tfi_p(i)=Tfi_p(i)*sec_day      !From days to sec
     END DO

     DO i=1,winj
       rwi(i)=rwi(i)*ft_m            !From ft to m
       qoi(i)=qoi(i)/(sec_day*bl_m3)    !From Bl/dia to m3/sec
       Tin_i(i)=Tin_i(i)*sec_day      !From days to sec
       Tfi_i(i)=Tfi_i(i)*sec_day      !From days to sec
     END DO
 END SUBROUTINE Cambio_unidades
!_____________________________________________________
   SUBROUTINE Results

     INTEGER  :: i,j,k,m
     REAL(KIND=dp) :: WI,roo,aux1,aux2, TAUX

           arra(:,:,:,1) = prs_new_f(:,:,:)/pa_psi
           arra(:,:,:,2) = prs_new_m(:,:,:)/pa_psi

       WRITE( *,130) NIMP, NTS, T_acum/sec_day, Dt/sec_day
       TAUX = REAL(NIMP)*TIMP

       IF (NTS.EQ.1) THEN
         arra(:,:,:,1) = prs_old_f(:,:,:)/pa_psi
         arra(:,:,:,2) = prs_old_m(:,:,:)/pa_psi
       END IF
       if (t_acum.ge.taux) then
          NIMP = NIMP+1
       END IF

    130 FORMAT (2x, 2(I3,1x), f12.5, 1x, E10.4)

    do m=1,wpro
      !Peacemen´s Model
      aux1=SQRT(kfy(xp(m),yp(m),zp(m))/kfx(xp(m),yp(m),zp(m)))*Dx*Dx+SQRT(kfx(xp(m),yp(m),zp(m))/kfy(
        xp(m),yp(m),zp(m)))*Dy*Dy
      aux2=SQRT(SQRT(kfy(xp(m),yp(m),zp(m))/kfx(xp(m),yp(m),zp(m))))+SQRT(SQRT(kfx(xp(m),yp(m),zp(
        m))/kfy(xp(m),yp(m),zp(m))))
  roo=0.28d0*SQRT(aux1)/aux2

      WI=2.0D0 *Pi*SQRT(kfx(xp(m),yp(m),zp(m))*kfy(xp(m),yp(m),zp(m)))*Dz/(LOG(rwp(m)/roo))
       pwf(m)=prs_new_f(xp(m),yp(m),zp(m))-(Qo(xp(m),yp(m),zp(m))*Vo(xp(m),yp(m),zp(m))/WI)
    END DO
      WRITE(22,311)T_acum*24.0/sec_day,(pwf(m)/pa_psi,m=1,wpro), &
               prs_old_f(INT(Nx/2),INT(Ny/2),INT(Nz/2))/pa_psi, &
               prs_old_m(INT(Nx/2),INT(Ny/2),INT(Nz/2))/pa_psi
     311 FORMAT(2x,F12.5,2x,10(F13.5,2x))
   END SUBROUTINE Results
!_____________________________________________________________________ SUBROUTINE
PROWRITE(arra,unt_no,nxx,nyy,nzz,nss, tv, tempo)
```

```
 USE Constants
 REAL(kind=dp),    INTENT(IN), DIMENSION(:,:,:,:)                    :: arra
 REAL(kind=dp),    INTENT(IN)                                 :: tempo
 INTEGER,          INTENT(IN)                                         :: nxx,nyy,nzz,nss,unt_no, tv
 INTEGER                                                              :: i,j,k,m

  WRITE(unt_no,1012)tv,tempo/86400.0d0
    do m=1,nss
      write(unt_no,'(/,1x,A4,I5,)')'$M=',m
      do k=1,nzz
        write(unt_no,'(/,1x,A4,I5,/)')'$Z=',k
        do i=1,nxx
          write(unt_no,1021,ADVANCE='no')(arra(i,j,k,m),j=1,nyy-1)
          write(unt_no,'(1x,f10.4)')arra(i,nyy,k,m)
        end do
      end do
    end do

1012 FORMAT(//,1x,'$P= ',i3,f12.5)
1021 FORMAT(1x,600(F10.4,1x,','))
   return
  END SUBROUTINE PROWRITE
 END MODULE input_data
```