Universidad Nacional Autónoma de México
Instituto de Geofísica

Tesis

"Aplicaciones del Cómputo en Paralelo a la Modelación de Sistemas Terrestres"

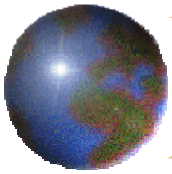
Presenta:

Mat. Antonio Carrillo Ledesma

Para Obtener el Grado de Maestro en Ciencias de la Tierra
(Modelación Matemática y Computacional de Sistemas Terrestres)

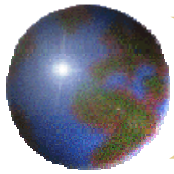
Tutor: Dr. Ismael Herrera Revilla

Agosto del 2006

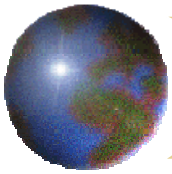


Contenido

- Motivación
- Objetivos
- Conceptos Básicos
  - Ejemplo a Desarrollar
  - Método de Elemento Finito
  - Resolución de Sistemas Lineales
  - Precondicionadores
  - Cómputo en Paralelo
- Infraestructura Utilizada
- Método de Elemento Finito Secuencial
- Método de Subestructuración Secuencial
- Método de Subestructuración Paralelo
- Método de Subestructuración Paralelo Precondicionado
- Análisis de Rendimiento
- Conclusiones
- Trabajo Futuro



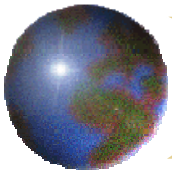
Motivación



Motivación

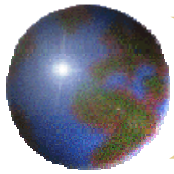
Uno de los grandes retos del área de cómputo científico es poder analizar a priori una serie de consideraciones dictadas por factores externos a la solución del problema de interés, estas consideraciones influirán de manera decisiva en la implementación computacional de la solución numérica, algunas de estas son:

- Número de Procesadores Disponibles.
- Tamaño y Tipo de Partición del Dominio.
- Tiempo de Ejecución Predeterminado.

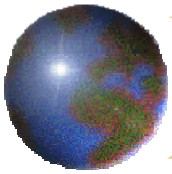


Motivación …

- Los esquemas de discretización tradicionales tipo diferencias finitas y elemento finito generan sistemas algebraicos de ecuaciones lineales y estos pueden ser de gran tamaño.
- Estos algoritmos son secuenciales y su implementación se hace en equipos de cómputo secuencial, teniendo como limitante del problema a tratar el número de nodos n de la discretización ya que el sistema algebraico asociado es de orden n^2.
- En la actualidad es una practica común tomar estos algoritmos secuenciales y mediante directivas de compilación tratar de paralelizarlo. Esta solución no es óptima ya que desperdicia recursos computacionales.

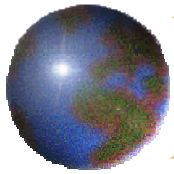


Objetivos



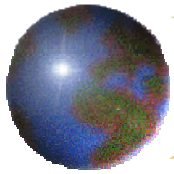
Objetivos Generales

- Mostrar las bases de una metodología que se utiliza para aplicar el cómputo en paralelo a la modelación matemática y computacional de sistemas continuos de forma flexible, escalable y eficiente.
- Mostrar los alcances y limitaciones de la metodología usando como herramientas de evaluación a los métodos de elemento finito secuencial, método de subestructuración secuencial y método de subestructuración paralelo.
- Mostrar los diversos esquemas de optimización aplicables a la metodología.

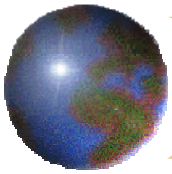


Objetivos Particulares

- Mostrar cómo aplicar la metodología para manejar problemas de gran tamaño (descomposición de malla fina).
- Mostrar cómo descomponer un dominio en un conjunto de subdominios que den una partición en la que el tiempo de cálculo sea mínimo para una configuración de hardware dada.
- Mostrar cuales son las posibles optimizaciones aplicables a una configuración de hardware dada.



Conceptos Básicos

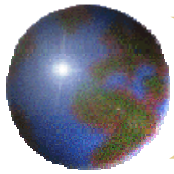


Ejemplo a Desarrollar

Para ejemplificar las ideas básicas de esta metodología, se ha tomado la ecuación de Poisson:

$$\begin{aligned} -\nabla^2 u &= f_\Omega \text{ en } \Omega \\ u &= g_{\partial\Omega} \text{ en } \partial\Omega. \end{aligned}$$

Este es un ejemplo muy sencillo pero gobierna muchos sistemas de la ingeniería y la ciencia, entre ellos el flujo de agua subterránea a través de un acuífero isotrópico, homogéneo bajo condiciones de equilibrio. También es muy usada en la conducción de calor en un sólido bajo condiciones de equilibrio.



Ejemplo a Desarrollar ...

En particular tomemos $\Omega = [-1,1] \times [0,1]$

$$f_\Omega = 2n^2\pi^2 \sin(n\pi x) * \sin(n\pi y)$$

y

$$g_{\partial\Omega} = 0$$

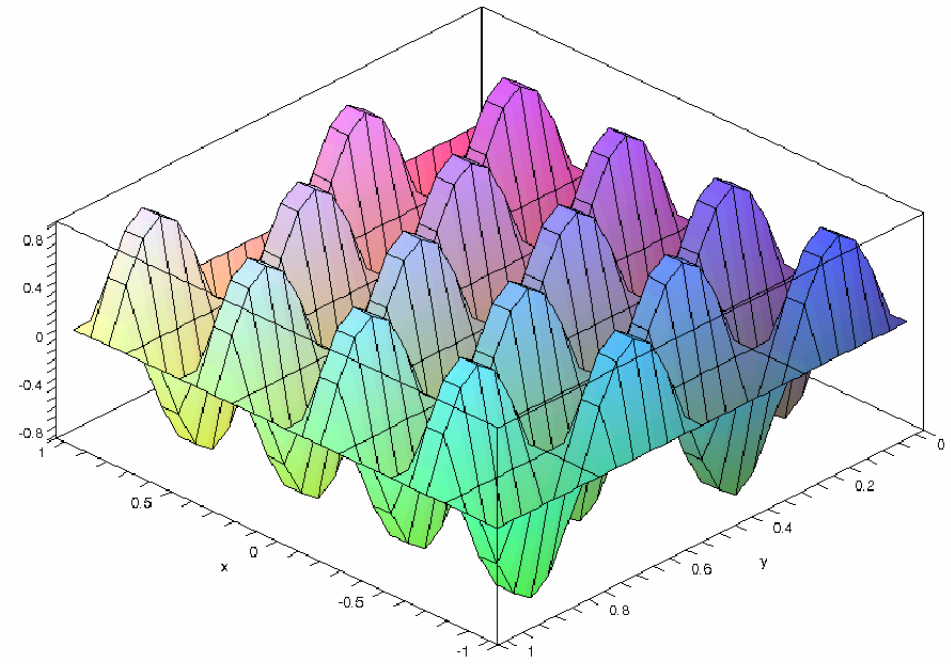
con $n \in \mathbb{N}$.

Cuya solución es

$$u(x,y) = \sin(n\pi x) * \sin(n\pi y).$$

Ejemplo a Desarrollar ...

Si por ejemplo tomamos $n = 4$, la solución se ve como:



Método de Elemento Finito

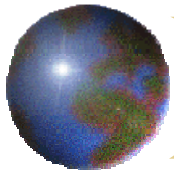
A la ecuación $-\nabla^2 u = f_\Omega$ la multiplicamos por $v \in V = H_0^1(\Omega)$: obtenemos $-v\nabla^2 u = f_\Omega v$, utilizando el teorema de Green se tiene

$$\int_\Omega \nabla v \cdot \nabla u dxdy = \int_\Omega f_\Omega v dxdy - \int_{\partial\Omega} \frac{\partial u}{\partial x} v n_x dx - \int_{\partial\Omega} \frac{\partial u}{\partial y} v n_y dy$$

pero como v se nulifica en la frontera $\partial\Omega$, entonces la ecuación anterior se simplifica a

$$\int_\Omega \nabla v \cdot \nabla u dxdy = \int_\Omega f_\Omega v dxdy$$

para toda $v \in V$.

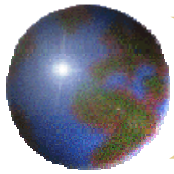


Método de Elemento Finito ...

Definiendo el operador bilineal $a(u,v) = \int_\Omega \nabla v \cdot \nabla u dxdy$

y la funcional lineal $\langle f, v \rangle = \int_\Omega f_\Omega v dxdy$.

Entonces podemos reescribir el problema anterior en su forma variacional, es decir, hay que encontrar $v \in V = H_0^1(\Omega)$ que satisfaga $a(u,v) = \langle f, v \rangle \quad \forall v \in V$.

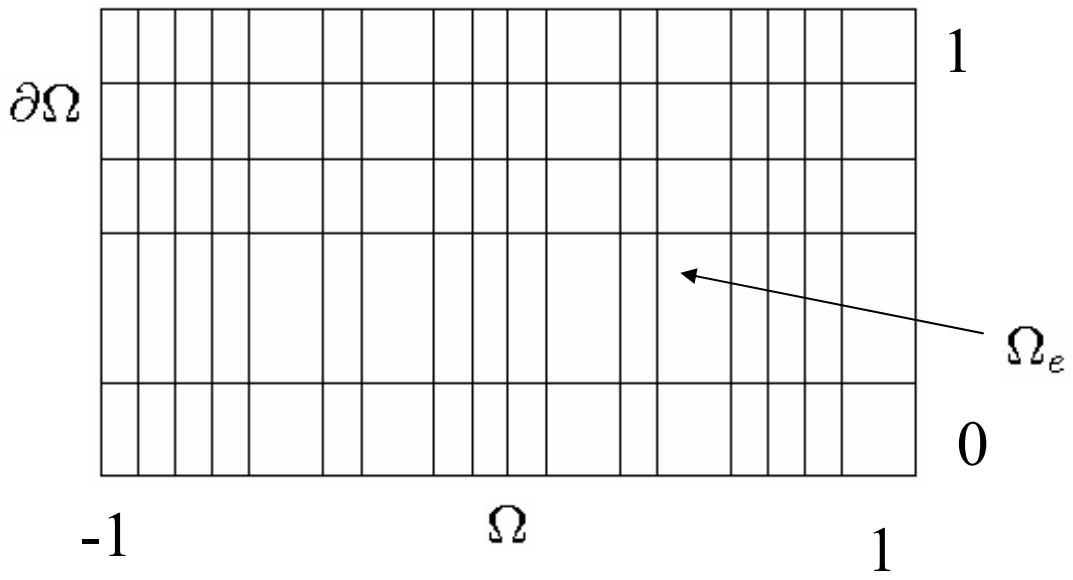


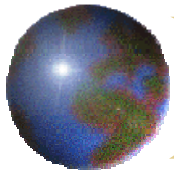
Método de Elemento Finito ...

Sea $\Omega \subset \mathbb{R}^2$ un dominio, el cual es dividido en E elementos Ω_e, tales que $\overline{\Omega} = \bigcup_{e=1}^{E} \overline{\Omega}_e$ donde

- Cada Ω_e es un polígono (rectángulo).
- $\Omega_i \cap \Omega_j = \emptyset$ para cada $i \neq j$.
- El diámetro $Diam(\Omega_e) \leq h$ para cada $e = 1, 2, ..., E$.
- Los vértices de cada Ω_e son llamados nodos, teniendo N de ellos.

Método de Elemento Finito ...





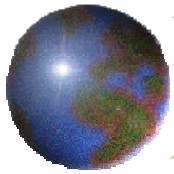
Método de Elemento Finito ...

Si elegimos al conjunto $\mathbb{P}^h[k]$ como el espacio de funciones lineales ϕ_i , definidas por pedazos de grado menor o igual a k, entonces el espacio a trabajar es

$$V^h = Generado\left\{\phi_i \in \mathbb{P}^h[k] \;\middle|\; \phi_i(x) = 0 \text{ en } \partial\Omega\right\}.$$

La solución aproximada de $\int_\Omega \nabla v \cdot \nabla u dxdy = \int_\Omega f_\Omega v dxdy$

queda en términos de $\int_\Omega \nabla \phi_i \cdot \nabla \phi_j dxdy = \int_\Omega f_\Omega \phi_j dxdy$.



Método de Elemento Finito ...

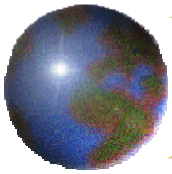
Definimos $K_{ij} \equiv a\left(\phi_i, \phi_j\right) = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dxdy$ y

$$F_j \equiv \langle f, \phi_j \rangle = \int_\Omega f_\Omega \phi_j dxdy \,.$$

Generándose el sistema lineal $\underline{\underline{K}}\underline{u} = \underline{F}$.

Donde la matriz $\underline{\underline{K}} \equiv [K_{ij}]$ es bandada y los vectores son $\underline{u} \equiv (u_1, ..., u_N)$ y $\underline{F} \equiv (F_1, ..., F_N)$.

El número de condicionamiento de la matriz $\underline{\underline{K}}$ para el método de elemento finito es del orden de $\frac{1}{h^2}$.



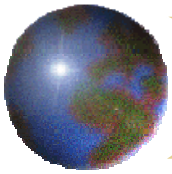
Resolución de Sistemas Lineales

Método Directo

- Eliminación Gausiana
- Eliminación Bandada
- Descomposición LU

Método Iterativo

- Jacobi
- Gauss-Seidel
- Richardson
- Relajación Sucesiva
- Gradiente Conjugado

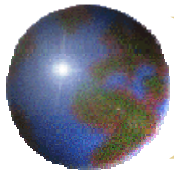


Precondicionadores

¿Qué es un precondicionador?
Es una matriz $\underline{\underline{C}}$ la cual aproxima a la matriz $\underline{\underline{A}}$ del sistema $\underline{\underline{A}}\underline{u} = \underline{b}$ de manera que $\underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{u} = \underline{\underline{C}}^{-1}\underline{b}$ tenga un menor número de condicionamiento que el sistema original $Cond(\underline{\underline{C}}^{-1}\underline{\underline{A}}) < Cond(\underline{\underline{A}})$, dicho de otra forma, un precondicionador es una inversa aproximada de la matriz original $\underline{\underline{C}}^{-1} \simeq \underline{\underline{A}}^{-1}$.

Características de un buen precondicionador:

- Al aplicar el precondicionador debe reducir el número de iteraciones necesarias para resolver el sistema lineal.
- El precondicionador debe ser fácil de calcular.
- El método de gradiente conjugado precondicionado debe de ser fácil de implementar y el tiempo requerido por iteración debe de ser pequeño.



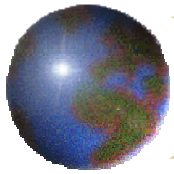
Precondicionadores ...

A posteriori o algrebraicos

- Jacobi
- SSOR
- Factorización incompleta
- Inversa aproximada

A priori o directamente relacionado con el problema que lo origina

- Para el método de subestructuración (problemas elípticos)
  - Derivados de la matriz de rigidez
  - Método de proyecciones



Cómputo en Paralelo

Entenderemos por una arquitectura en paralelo a un conjunto de procesadores interconectados capaces de cooperar en la solución de un problema.

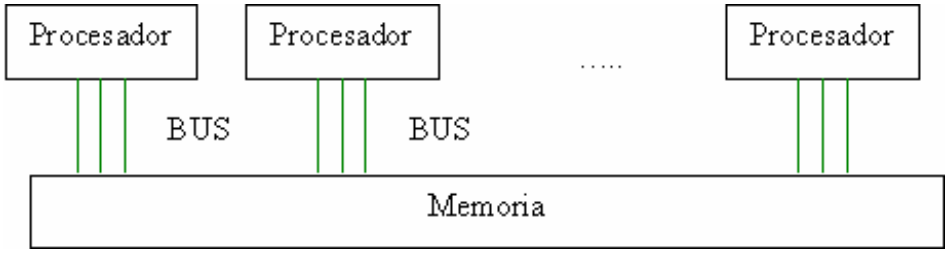
- La computación en paralelo es una técnica que nos permite distribuir una gran carga computacional entre muchos procesadores.

- Y es bien sabido que una de las mayores dificultades del procesamiento en paralelo es la coordinación de las actividades de los diferentes procesadores y el intercambio de información entre los mismos.

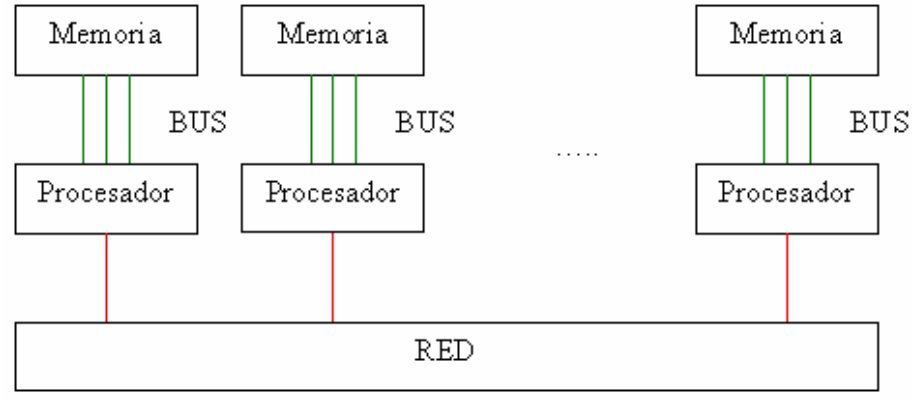
Cómputo en Paralelo ...

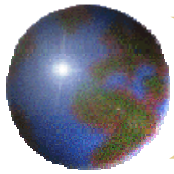
Arquitecturas Paralelas

- Memoria Compartida (multiprocesadores)



- Memoria Distribuida (multicomputadoras)





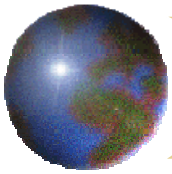
Cómputo en Paralelo ...

Ventajas del uso de Clusters de PC´s

- La construcción y puesta en marcha de un cluster es barata.
- Remplazar componentes defectuosos y escalar el cluster es sencillo.

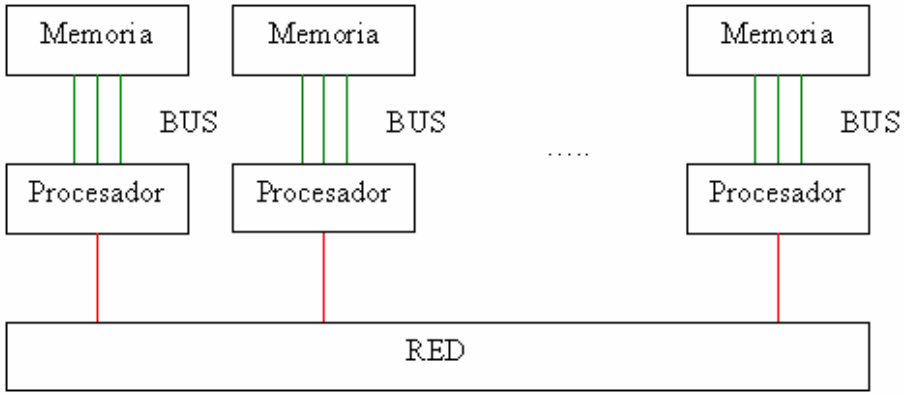
Desventajas del uso de Clusters de PC´s

- No están diseñados para cómputo de alto desempeño.
- Hay poco software para manejo y administración del cluster como un sistema integrado.
- Muchas aplicaciones importantes en multiprocesadores y optimizadas para ciertas arquitecturas no están disponibles para clusters.

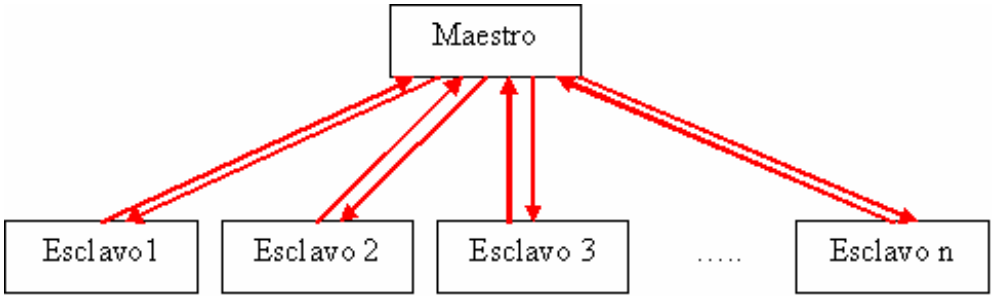


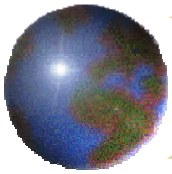
Cómputo en Paralelo ...

- Memoria Distribuida (Cluster)



- Cluster (Bajo Esquema Maestro-Esclavo)





Cómputo en Paralelo ...

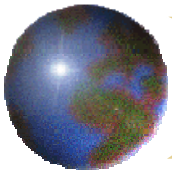
Métricas de desempeño

- Factor de aceleración $s = \frac{T(1)}{T(p)}$

En el caso ideal debería crecer de forma lineal al aumento del número de procesadores.

- Eficiencia $e = \frac{T(1)}{pT(p)} = \frac{s}{p}$

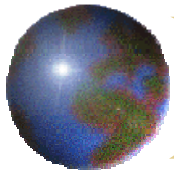
En el caso ideal debería de ser cercana a la unidad cuando el hardware es usado eficientemente.



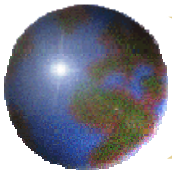
Cómputo en Paralelo ...

- Fracción serial $f = \dfrac{\frac{1}{s} - \frac{1}{p}}{1 - \frac{1}{p}}$

En el caso ideal debería decrecer a cero, un incremento en su valor es aviso de granularidad fina con la correspondiente sobrecarga en los procesos de comunicación.

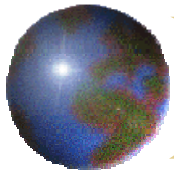


Infraestructura Utilizada

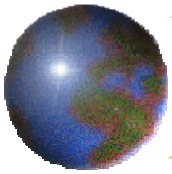


Infraestructura Utilizada

- Computadora Pentium IV HT a 2.8 GHtz con 1 GB de RAM corriendo bajo el sistema operativo Linux Debian Stable.
- Cluster homogéneo de 10 nodos duales Xeon a 2.8 GHtz con 1 GB de RAM por nodo, unidos mediante una red Ethernet de 1 Gb, corriendo bajo el sistema operativo Linux Debian Stable.
- Cluster heterogéneo con el nodo maestro Pentium IV HT a 3.4 GHtz con 1 GB de RAM y 7 nodos esclavos Pentium IV HT a 2.8 GHtz con 0.5 GB de RAM por nodo, unidos mediante una red Ethernet de 100 Mb, corriendo bajo el sistema operativo Linux Debian Stable.

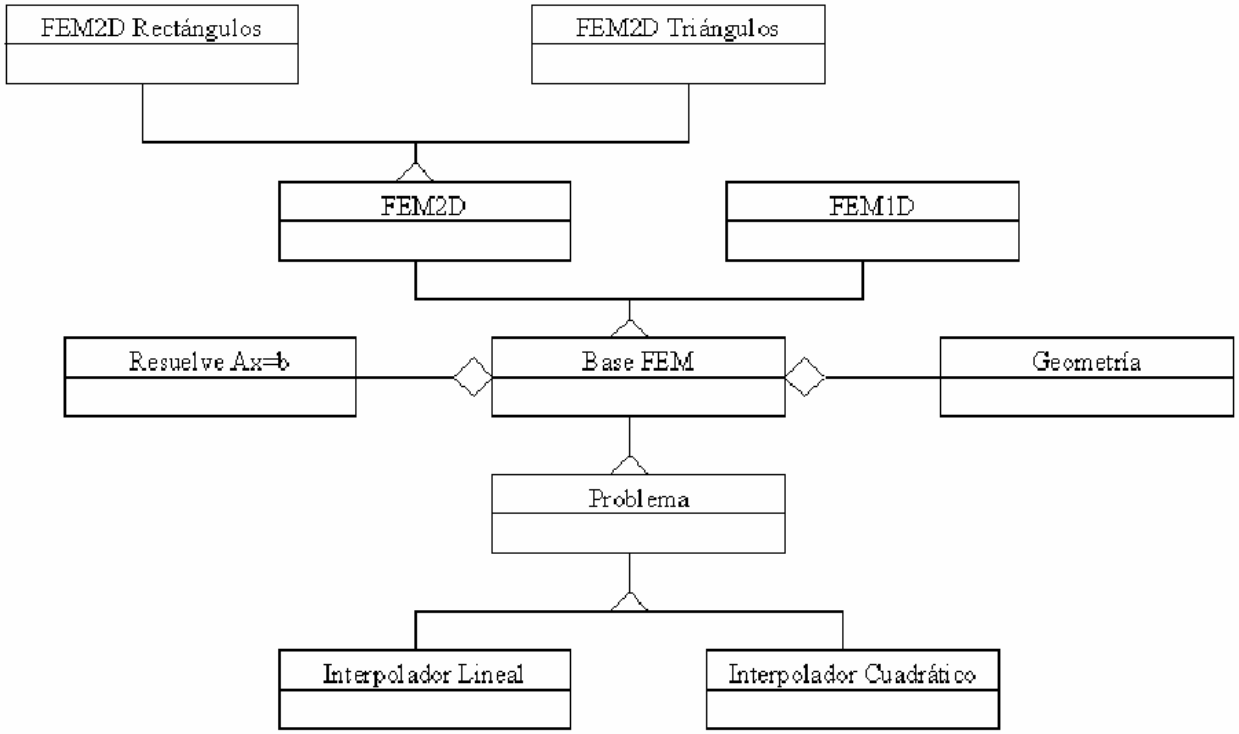


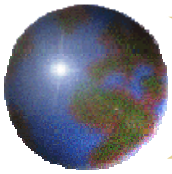
Método de Elemento Finito Secuencial



Método de Elemento Finito Secuencial

Jerarquía de clases para el método de elemento finito:





Método de Elemento Finito ...

Si descomponemos el dominio Ω en una partición rectangular de 513x513 nodos, genera 262,144 elementos rectangulares con 263,169 nodos en total, donde 261,121 son desconocidos. El sistema algebraico asociado es de 261,121 x 261,121 resolviendo el sistema mediante diversos métodos numéricos se tiene:

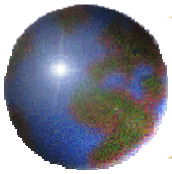
Método Iterativo	Iteraciones	Tiempo Total
Jacobi	865037	115897 seg.
Gauss-Seidel	446932	63311 seg.
Gradiente Conjugado	761	6388 seg.

Método de Elemento Finito ...

Si usamos precondicionadores a posteriori en el método de gradiente conjugado obtenemos:

Precondicionador	Iteraciones	Tiempo Total
Jacobi	760	6388 seg.
SSOR	758	6375 seg.
Factorización Incompleta	745	6373 seg.

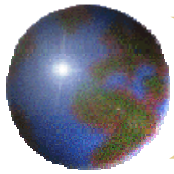
Existen precondicionadores a priori para el método de elemento finito, pero su eficiencia global es pobre con respecto a los métodos de descomposición de dominio.



Método de Elemento Finito ...

Limitaciones del método:

- Al aumentar el número de nodos n, el sistema algebraico crece del orden n^2.
- Un solo equipo de cómputo puede no soportar al sistema algebraico cuando este proviene de una descomposición fina.
- Si se paraleliza, es necesario usar memoria compartida para soportar el sistema algebraico.
- Es un algoritmo secuencial, por ello no es eficiente su paralelización.



Método de Subestructuración Secuencial

Método de Subestructuración Secuencial

El dominio Ω ahora es dividido en $E = n \times m$ subdominios $\Omega_i,\ i = 1, 2, 3, ..., E$ tales que

$$\Omega_i \cap \Omega_j = \varnothing \quad \forall i \neq j \quad \text{y} \quad \overline{\Omega} = \bigcup_{i=1}^{E} \overline{\Omega}_i,$$

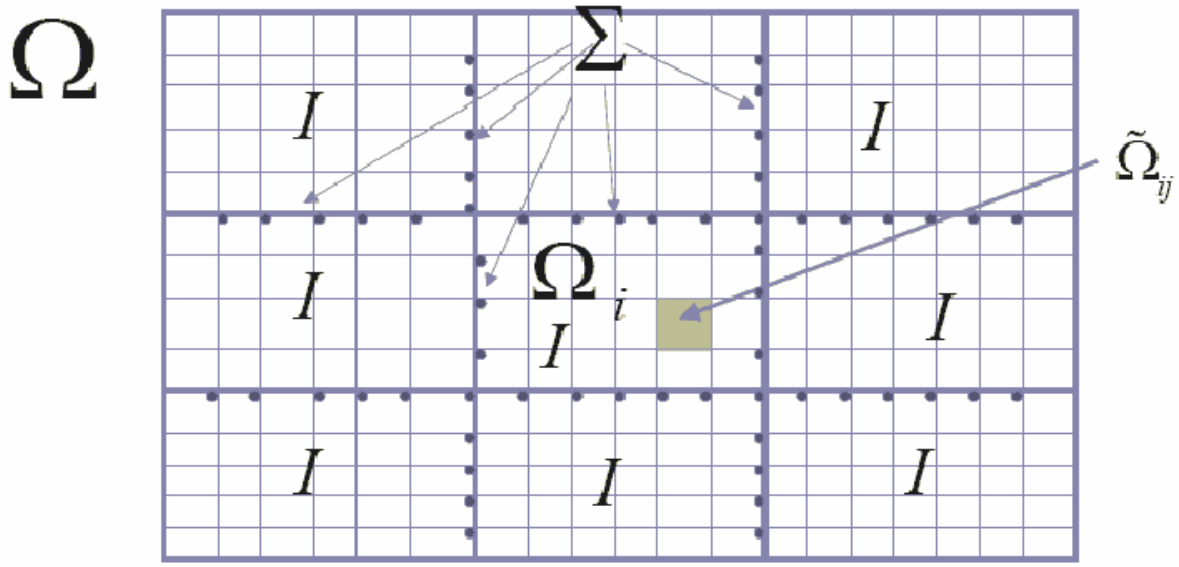
y al conjunto $$\Sigma = \bigcup_{i=1}^{E} \Sigma_i, \quad \text{si } \Sigma_i = \partial\Omega_i \backslash \partial\Omega$$

lo llamaremos la frontera interior de los subdominios.

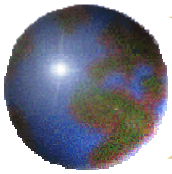
Adicionalmente cada Ω_i es descompuesto en $p \times q$ elementos, obtenemos así la descomposición fina del dominio.

Método de Subestructuración ...

Un ejemplo de una descomposición del dominio se muestra en la siguiente figura:



Descomposición 3x3 y 6x5



Método de Subestructuración ...

Si elegimos al conjunto $\mathbb{P}^h[k]$ como el espacio de funciones lineales ϕ_i, definidas por pedazos en cada Ω_i, de grado menor o igual a k, entonces el espacio a trabajar es

$$V^h = Generado\left\{\phi_i \in \mathbb{P}^h[k] \;\middle|\; \phi_i(x) = 0 \text{ en } \partial\Omega\right\}.$$

La solución aproximada de $\int_\Omega \nabla v \cdot \nabla u dxdy = \int_\Omega f_\Omega v dxdy$

queda en términos de $\int_\Omega \nabla \phi_i \cdot \nabla \phi_j dxdy = \int_\Omega f_\Omega \phi_j dxdy$.

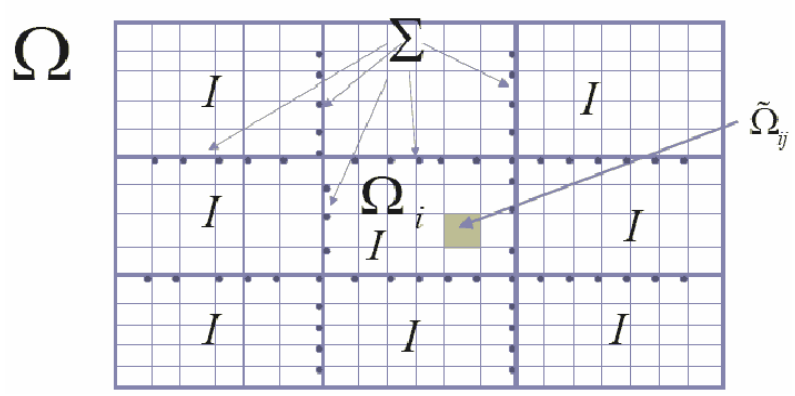
Método de Subestructuración ...

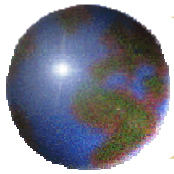
Definiendo ahora para cada subdominio Ω_δ , con $\delta = 1, ..., E$ las matrices

$$\underline{\underline{A}}_\delta^{II} \equiv \left[\left\langle w_\delta^i, w_\delta^j \right\rangle\right] \qquad \underline{\underline{A}}_\delta^{\Sigma\Sigma} \equiv \left[\langle w_\Sigma^\alpha, w_\Sigma^\alpha \rangle\right]$$

y

$$\underline{\underline{A}}_\delta^{I\Sigma} \equiv \left[\langle w_I^i, w_\Sigma^\alpha \rangle\right], \quad \underline{\underline{A}}_\delta^{\Sigma I} \equiv \left[\langle w_\Sigma^\alpha, w_I^i \rangle\right]$$





Método de Subestructuración ...

Podríamos definir (no se construyen estas matrices globales) las matrices $\underline{\underline{A}}^{II}$, $\underline{\underline{A}}^{\Sigma I}$, $\underline{\underline{A}}^{\Sigma I}$ y $\underline{\underline{A}}^{\Sigma\Sigma}$ donde

$$\underline{\underline{A}}^{II} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{II} & & & \\ & \underline{\underline{A}}_2^{II} & & \\ & & \ddots & \\ & & & \underline{\underline{A}}_E^{II} \end{bmatrix} \qquad \underline{\underline{A}}^{I\Sigma} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{I\Sigma} \\ \underline{\underline{A}}_2^{I\Sigma} \\ \vdots \\ \underline{\underline{A}}_E^{I\Sigma} \end{bmatrix}$$

$$\underline{\underline{A}}^{\Sigma I} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{\Sigma I} & \underline{\underline{A}}_2^{\Sigma I} & \cdots & \underline{\underline{A}}_E^{\Sigma I} \end{bmatrix} \qquad \underline{\underline{A}}^{\Sigma\Sigma} \equiv \left[\sum_{i=1}^{E} \underline{\underline{A}}_i^{\Sigma\Sigma} \right]$$

Método de Subestructuración ...

Definiendo a $\underline{u}=(\underline{u_I},\underline{u_\Sigma})$ como $u_I=(u_1,...,_{N_I})$ y $u_\Sigma=(u_1,...,_{N_\Sigma})$.
entonces podemos generar el sistema:

$$\begin{aligned}\underline{\underline{A}}^{II}\underline{u_I}+\underline{\underline{A}}^{I\Sigma}\underline{u_\Sigma} &= \underline{b_I}\\ \underline{\underline{A}}^{\Sigma I}\underline{u_I}+\underline{\underline{A}}^{\Sigma\Sigma}\underline{u_\Sigma} &= \underline{b_\Sigma}\end{aligned}$$

el cual es equivalente al sistema algebraico generado por el método de elemento finito.
Despejando $\underline{u_I}$ del sistema anterior, tenemos el sistema lineal

$$\left(\underline{\underline{A}}^{\Sigma\Sigma}-\underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{\underline{A}}^{I\Sigma}\right)\underline{u_\Sigma}=\underline{b_\Sigma}-\underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{b_I}$$

a $\underline{\underline{S}}=\underline{\underline{A}}^{\Sigma\Sigma}-\underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{\underline{A}}^{I\Sigma}$ comúnmente se le llama el complemento de Schur.

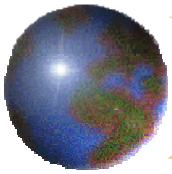
Método de Subestructuración ...

Pero como nosotros tenemos definidas de forma local en cada Ω_i a las matrices $\underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y $\underline{\underline{A}}_i^{II}$, entonces el complemento de Schur local queda definido por $\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I} \left(\underline{\underline{A}}_i^{II}\right)^{-1} \underline{\underline{A}}_i^{I\Sigma}$.

Generándose el sistema virtual cuyas incógnitas son los nodos de la frontera interior

$$\left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right] \underline{u_\Sigma} = \left[\sum_{i=1}^{E} \underline{b}_i\right]$$

donde $\underline{b}_i = \underline{\underline{A}}_i^{\Sigma I} \left(\underline{\underline{A}}_i^{II}\right)^{-1} \underline{b_I}_i$.



Método de Subestructuración ...

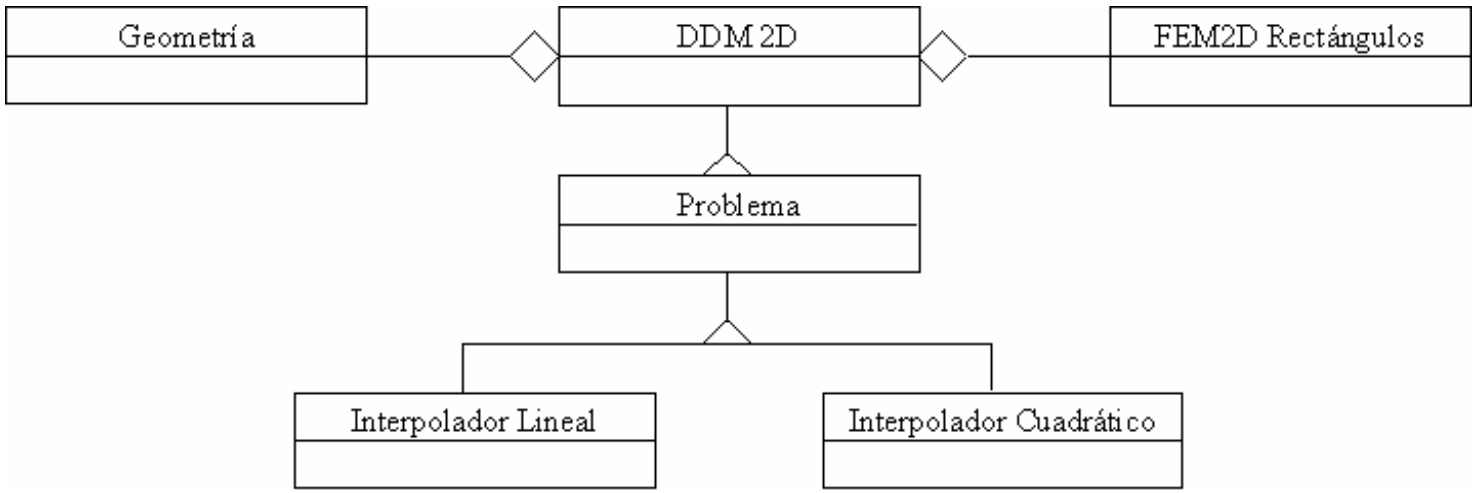
Usando el método de gradiente conjugado, resolvemos los nodos de la frontera interior $\underline{u_\Sigma}$.
Para ello, es necesario hacer por cada iteración del método una multiplicación de un vector de la dimensión de $\underline{u_\Sigma}$ con todos los $\underline{\underline{S}}_i$, esto se logra pasando sólo las entradas correspondientes al subdominio y así poderlo multiplicar por $\underline{\underline{S}}_i$.

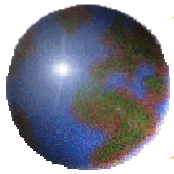
El número de condicionamiento de la matriz del método de subestruturación es del orden de $\frac{1}{h}$.

Una vez conocido $\underline{u_\Sigma}$ se calcula $\underline{u_I}_i = \left(\underline{\underline{A}}_i^{II}\right)^{-1}\left(\underline{b_I}_i - \underline{\underline{A}}_i^{\Sigma I}\underline{u_\Sigma}_i\right)$ resolviendo con esto los nodos interiores.

Método de Subestructuración ...

Jerarquía de clases para el método de subestructuración secuencial:





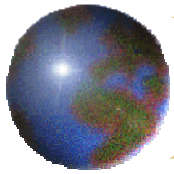
Método de Subestructuración ...

Para resolver el mismo dominio Ω del ejemplo anterior (FEM cuyo tiempo de ejecución es de 6,388 seg), podemos construir algunas de las siguientes descomposiciones:

Descomposición	Nodos Interiores	Subdominios	Elementos Subdominio	Total Nodos Subdominio	Nodos Desconocidos Subdominio
2x2 y 256x256	260100	4	65536	66049	65025
4x4 y 128x128	258064	16	16384	16641	16129
8x8 y 64x64	254016	64	4096	4225	3969
16x16 y 32x32	246016	256	1024	1089	961
32x32 y 16x16	230400	1024	256	289	225

Obteniendo los siguientes resultados:

Partición	Nodos Frontera Interior	Iteraciones	Tiempo Total
2x2 y 256x256	1021	139	5708 seg.
4x4 y 128x128	3057	159	2934 seg.
8x8 y 64x64	7105	204	1729 seg.
16x16 y 32x32	15105	264	1077 seg.
32x32 y 16x16	30721	325	1128 seg



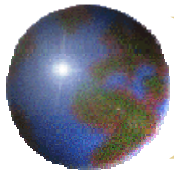
Método de Subestructuración ...

Limitaciones del método:

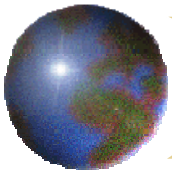
- Al aumentar el número de subdominios, el sistema algebraico crece, pero son locales a cada subdominio.
- Un solo equipo de cómputo puede no soportar a los sistemas algebraicos cuando este proviene de una descomposición fina.

Ventajas del método:

- Si se paraleliza, no es necesario usar memoria compartida, sólo se requiere memoria distribuida para soportar el sistema algebraico.
- Es un algoritmo paralelo, por ello es eficiente su paralelización.



Método de Subestructuración Paralelo

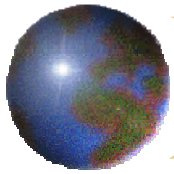


Método de Subestructuración Paralelo

La computación en paralelo es una técnica que nos permite distribuir una gran carga computacional entre muchos procesadores.

La implementación de este método se desarrolló para usar el esquema maestro-esclavo mediante el paso de mensajes vía MPI.

Y es bien sabido que una de las mayores dificultades del procesamiento en paralelo es la coordinación de las actividades de los diferentes procesadores y el intercambio de información entre los mismos.

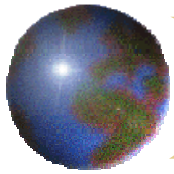


Método de Subestructuración ...

Para resolver el mismo dominio Ω del ejemplo anterior (513x513 nodos), con una descomposición de 4x4 subdominios y cada subdominio descompuesto en 128x128.

El balanceo de cargas si se trabaja con P procesadores se obtiene al descomponer el dominio en $n \times m$ subdominios tal que $(P-1) \mid (n*m)$.

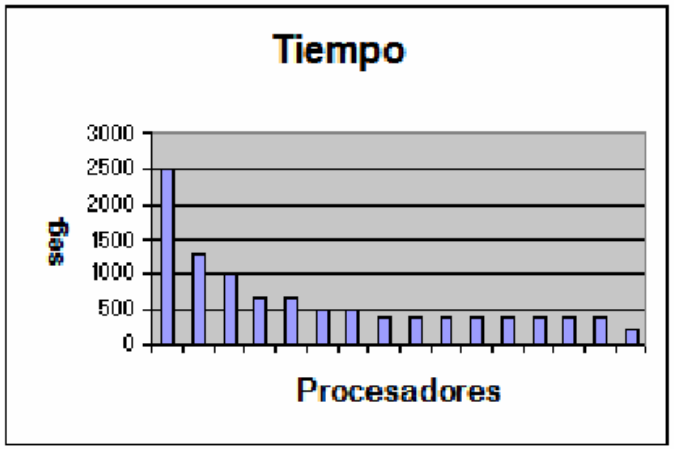
Obteniendo los siguientes resultados:

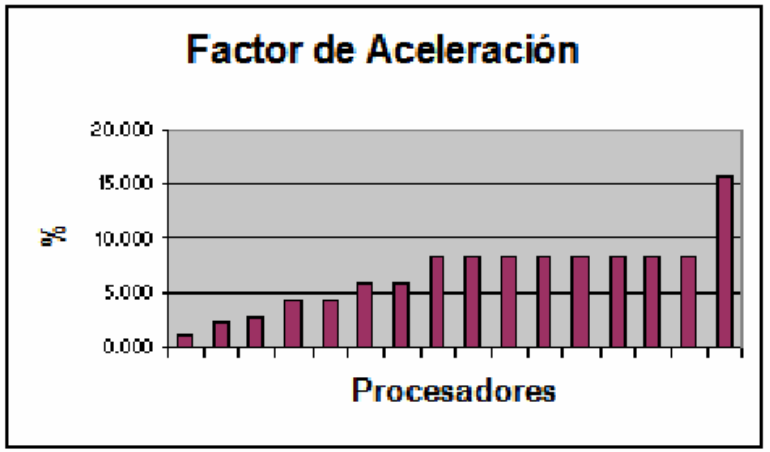


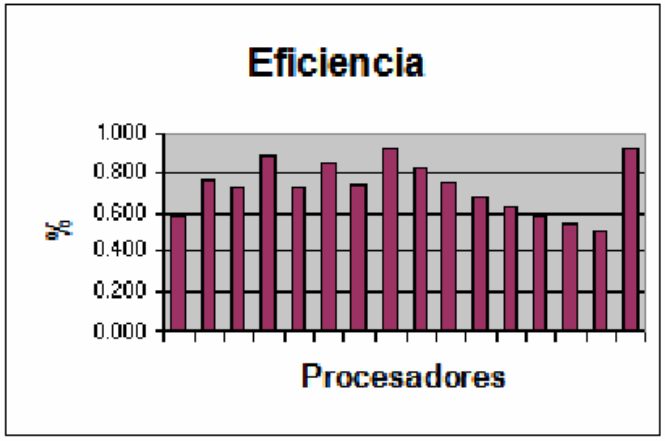
Método de Subestructuración ...

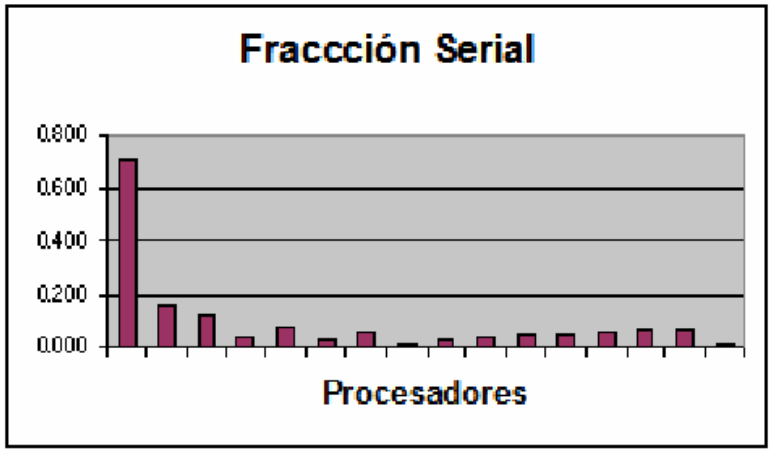
Procesadores	Tiempo	Factor de Aceleración	Eficiencia	Fracción Serial
1	2943 seg.			
2	2505 seg.	1.17485	0.58742	0.70234
3	1295 seg.	2.27258	0.75752	0.16004
4	1007 seg.	2.92254	0.73063	0.12289
5	671 seg.	4.38599	0.87719	0.03499
6	671 seg.	4.38599	0.73099	0.07359
7	497seg.	5.92152	0.84593	0.03035
8	497 seg.	5.92152	0.74019	0.05014
9	359 seg.	8.19777	0.91086	0.01223
10	359 seg.	8.19777	0.81977	0.02442
11	359 seg.	8.19777	0.74525	0.03441
12	359 seg.	8.19777	0.68314	0.04216
13	359 seg.	8.19777	0.63059	0.04881
14	359 seg.	8.19777	0.58555	0.05444
15	359 seg.	8.19777	0.54651	0.05926
16	359 seg.	8.19777	0.51236	0.06344
17	188 seg	15.65425	0.92083	0.00537

Método de Subestructuración ...

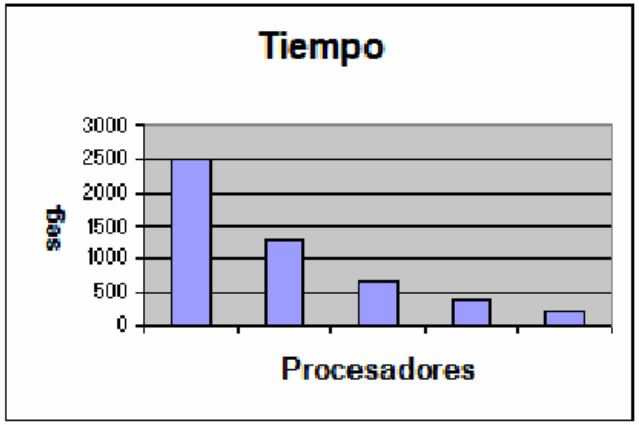


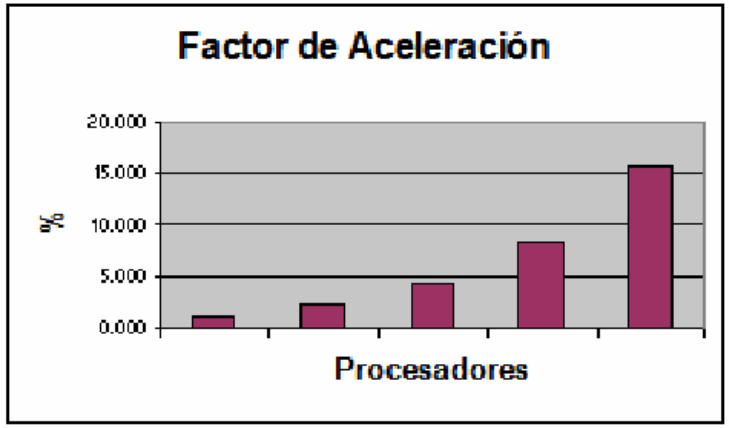


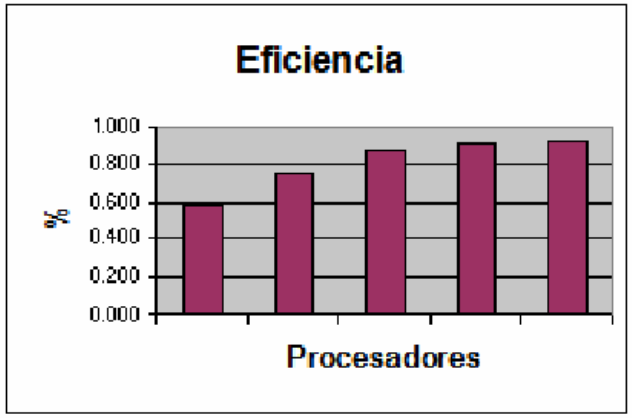


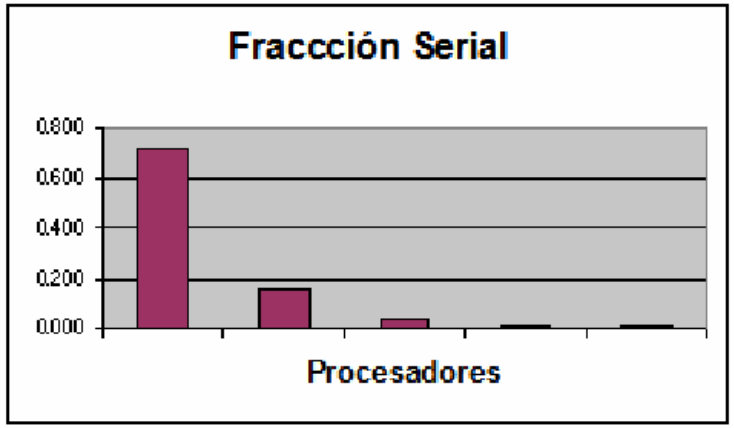


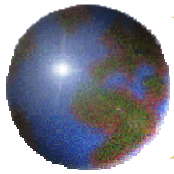
Método de Subestructuración ...



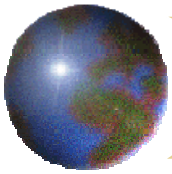








Método de Subestructuración Paralelo Precondicionado

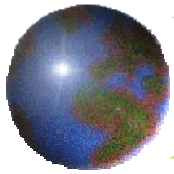


Método de Subestructuración Paralelo Precondicionado

Usando el mismo programa que para el método de subestructuración paralelo, implementamos el método de subestructuración paralelo precondicionado haciendo uso de un precondicionador a Priori en el Método de Gradiente Conjugado.

El precondicionador usado es el derivado de la matriz de rigidez, este es implementado en el método de gradiente conjugado.

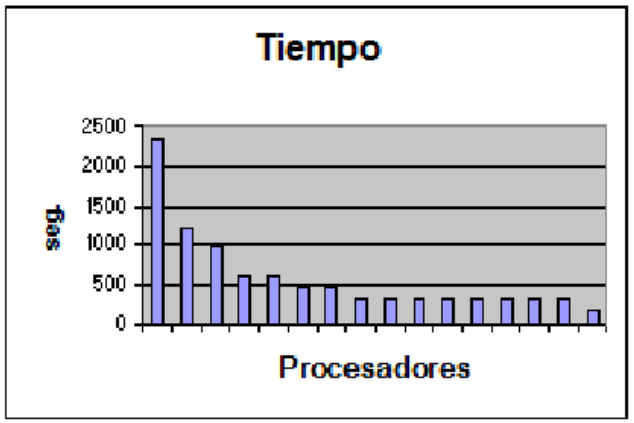
Tomando la misma descomposición del dominio (4x4 subdominios y cada subdominio en 20x20) y usando el precondicionador se disminuyó aproximadamente 50 % el número de iteraciones (de 91 a 47), obteniendo los siguientes resultados:

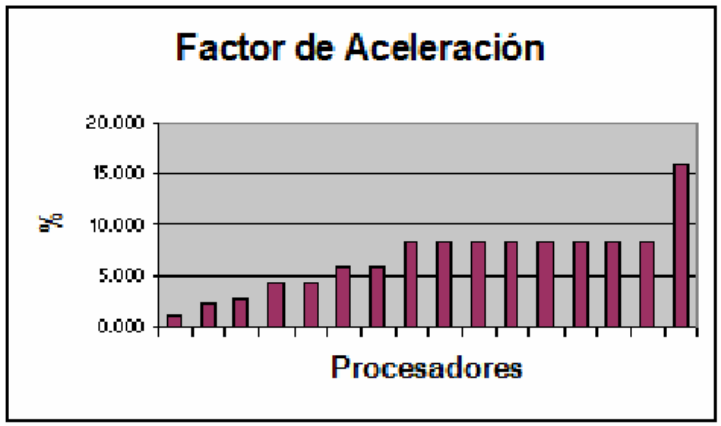


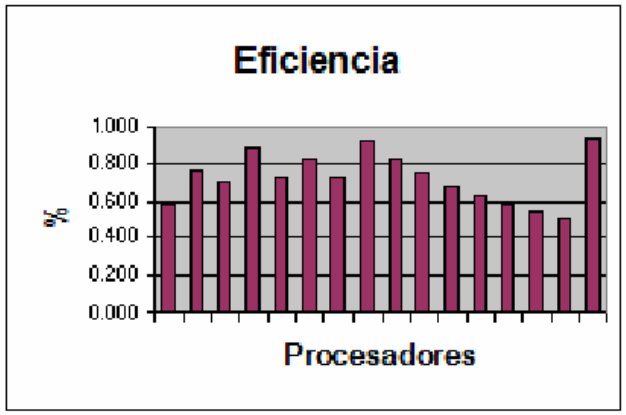
Método de Subestructuración ...

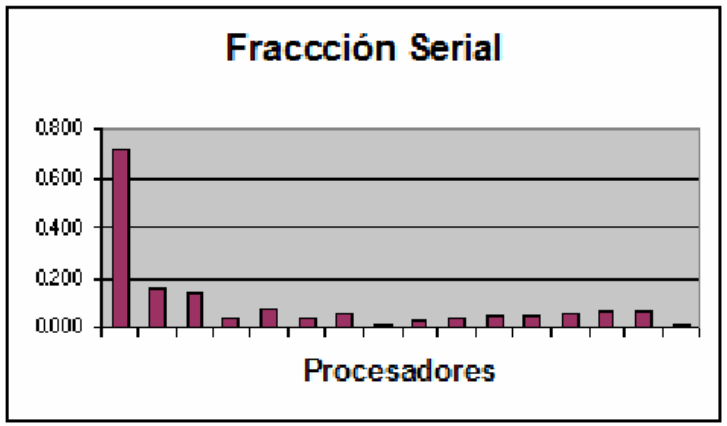
Procesadores	Tiempo	Factor de Aceleración	Eficiencia	Fracción Serial
1	2740 seg.			
2	2340 seg.	1.17094	0.58547	0.70802
3	1204 seg.	2.27574	0.75858	0.15912
4	974 seg.	2.81314	0.70328	0.14063
5	626 seg.	4.37699	0.87539	0.03558
6	626 seg.	4.37699	0.72949	0.07416
7	475 seg.	5.76842	0.82406	0.03558
8	475 seg.	5.76842	0.72105	0.05526
9	334 seg.	8.20359	0.91151	0.01213
10	334 seg.	8.20359	0.82035	0.02433
11	334 seg.	8.20359	0.74578	0.03408
12	334 seg.	8.20359	0.68363	0.04207
13	334 seg.	8.20359	0.63104	0.04872
14	334 seg.	8.20359	0.58597	0.05435
15	334 seg.	8.20359	0.54690	0.05917
16	334 seg.	8.20359	0.51272	0.06335
17	173 seg.	15.83815	0.93165	0.00458

Método de Subestructuración ...

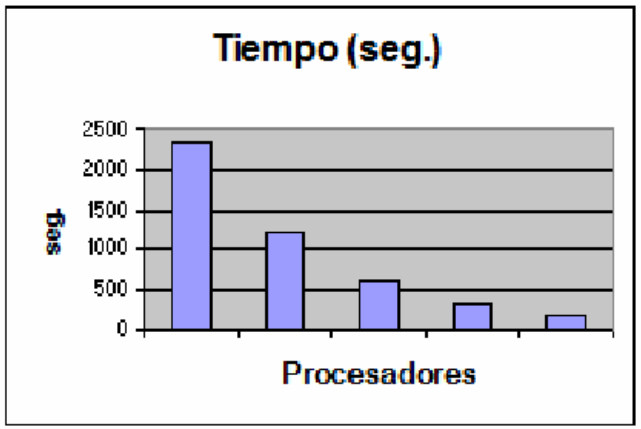


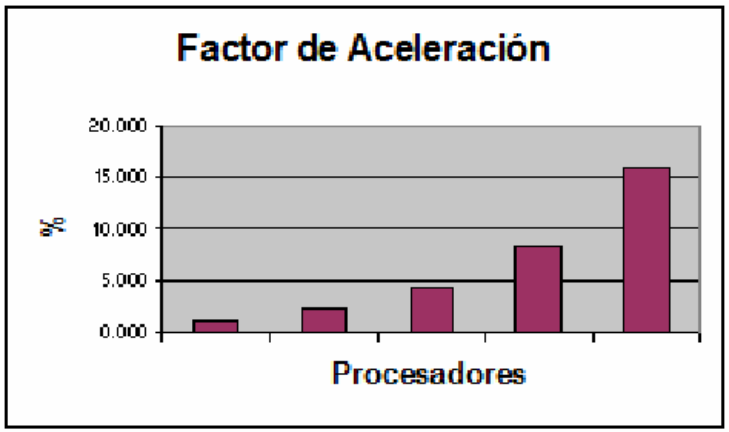


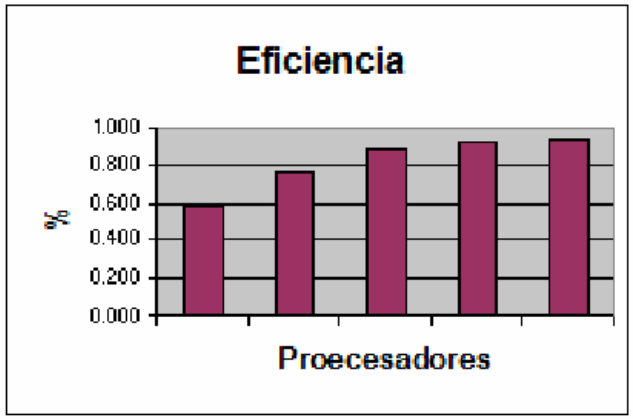


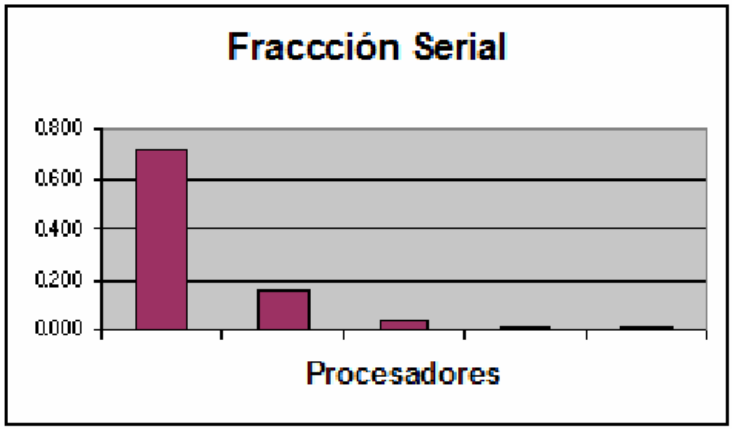


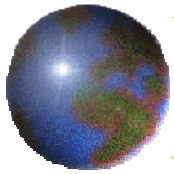
Método de Subestructuración ...



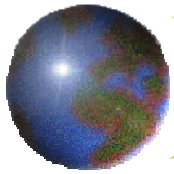






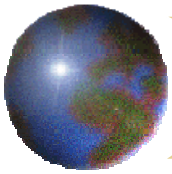


Análisis de Rendimiento



Análisis de Rendimiento

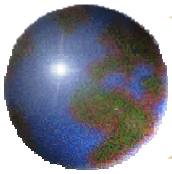
- Afectación del rendimiento al aumentar el número de subdominios en la descomposición.
- Descomposición óptima para una configuración de hardware paralelo dada.
- Consideraciones para aumentar el rendimiento.



Análisis de Rendimiento ...

Afectación del rendimiento al aumentar el número de subdominios en la descomposición es debido a que:

- En cada subdominio se deberán generar $\underline{\underline{A}}_i^{II}, \underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y calcular $\left(\underline{\underline{A}}_i^{II}\right)^{-1}$.
- Si el número de subdominios es pequeño, el tamaño de las matrices será grande. El costo computacional de su generación será alto.
- Si por el contrario el número de subdominios es grande, el tamaño de las matrices es pequeño, pero el esquema maestro-esclavo se degradará inexorablemente al aumentar el número de subdominios por el incremento de trabajo en el nodo maestro.



Análisis de Rendimiento ...

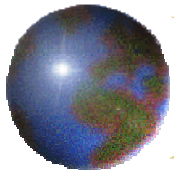
Descomposición óptima para una configuración de hardware paralelo dada. Deseamos resolver el mismo dominio Ω pero ahora usando 3,4,5 y 6 procesadores

- Cluster homogéneo

Partición	Tiempo en 3 Procesadores	Tiempo en 4 Procesadores	Tiempo en 5 Procesadores	Tiempo en 6 Procesadores
2×2 y 256×256	2576 seg.	2084 seg.	1338 seg.	—
4×4 y 128×128	1324 seg.	1071 seg.	688 seg.	688 seg.
8×8 y 64×64	779 seg.	630 seg.	405 seg.	405 seg.
16×16 y 32×32	485 seg.	391 seg.	251 seg.	251 seg.

- Cluster heterogéneo

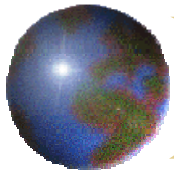
Partición	Tiempo en 3 Procesadores	Tiempo en 4 Procesadores	Tiempo en 5 Procesadores	Tiempo en 6 Procesadores
2×2 y 256×256	2342 seg.	1895 seg.	1217 seg.	—
4×4 y 128×128	1204 seg.	974 seg.	626 seg.	626 seg.
8×8 y 64×64	709 seg.	573 seg.	369 seg.	369 seg.
16×16 y 32×32	441 seg.	356 seg.	229 seg.	229 seg.



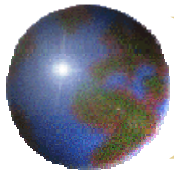
Análisis de Rendimiento ...

Consideraciones para aumentar el rendimiento

- Balance de cargas de trabajo homogéneo.
- Usar el mejor precondicionador a priori para ese problema.
- Usar bibliotecas que optimizan las operaciones en el manejo de los elementos de las matrices (densas y/o bandadas).
- Usar al momento de compilar los códigos, directivas de optimización.
- Implementar otras estrategias de paralelización
  - Al generar las matrices.
  - Al realizar los cálculos requeridos entre las matrices y el vector.
  - A nivel del compilador.



Conclusiones

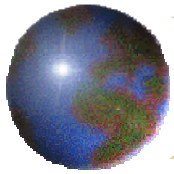


Conclusiones

En el presente trabajo, se mostró para el método de descomposición de dominio de subestructuración paralelo precondicionado:

- Los alcances y limitaciones.
- Los diversos esquemas de optimización.
- La forma de generar una descomposición del dominio que sea óptima para una configuración de hardware paralelo dado.

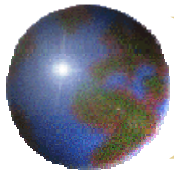
Conjuntando los métodos de descomposición de dominio, la programación orientada a objetos y esquemas de paralelización que usan el paso de mensajes, es posible construir aplicaciones que coadyuven a la solución de problemas en dos o más dimensiones concomitantes en ciencia e ingeniería, los cuales pueden ser de tamaño considerable.



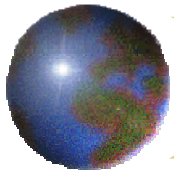
Conclusiones ...

Las aplicaciones desarrolladas bajo este paradigma serán eficientes, flexibles y escalables; a la vez que son abiertas a nuevas tecnologías y desarrollos computacionales y al ser implantados en clusters, permiten una codificación ordenada y robusta, dando con ello una alta eficiencia en la adaptación del código a nuevos requerimientos, como en la ejecución del mismo.

De forma tal que esta metodología permite tener a disposición de quien lo requiera, una gama de herramientas flexibles y escalables para coadyuvar de forma eficiente y adaptable a la solución de problemas en medios continuos de forma sistemática.



Trabajo Futuro

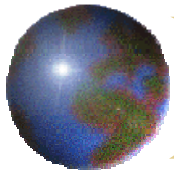


Trabajo Futuro

Trabajar en otros métodos de descomposición de dominio como Trefftz-Herrera, FETI, Galerkin Discontinuo, Multigrid, entre otros. Permitiendo tener un grupo de herramientas que pueden ser usadas en múltiples problemas escogiendo la que ofrezca mayores ventajas computacionales para un problema dado.

Construcción de precondicionadores a priori para los diversos métodos. Con la visión de que la construcción de estos sean óptimos.

Trabajar en problemas elípticos, parabólicos e hiperbólicos, tanto lineales y no lineales. Permitiendo así, atacar una gran gama de problemas en medios continuos.



Gracias…

Nota: Los programas desarrollados en el presente trabajo pueden descargarse de la página Web:

http://www.mmc.igeofcu.unam.mx/