Mecánica Computacional, Volumen XIX, pp. 397-404
Compilado por: F. Quintana y S. Felicelli
San Carlos de Bariloche, Noviembre de 2000

# ANÁLISIS DE EFICIENCIA DE UN PROGRAMA DE ELEMENTOS FINITOS EN UN CLUSTER BEOWULF

**Andrea M. Yommi, Norberto M. Nigro, Mario A. Storti y Victorio E. Sonzogni**
Centro Internacional de Métodos Computacionales en Ingeniería
CONICET - Universidad Nacional del Litoral
Güemes 3450, 3000 Santa Fe, Argentina

## RESUMEN

En este trabajo se presentan algunos resultados obtenidos al realizar un análisis de eficiencia en un cluster Beowulf del programa de elementos finitos PETSc-FEM, cuyos detalles de programación e implementación se describen en [1]. Las aplicaciones de PETSc-FEM que fueron estudiadas son Laplace 2D y Navier Stokes 2D y 3D. Se tuvieron en cuenta cuestiones relativas al grado de paralelización de los mismos en este tipo de ambiente distribuido, ya sea mediante el incremento del número de procesadores como el aumento del tamaño del problema.

## ABSTRACT

We present an analysis of efficiency on a Beowulf cluster of the finite element program PETSc-FEM. Programming details and implementation of the code are described in [1]. The applications Laplace 2D and Navier Stokes 2D and 3D were studied, regarding their parallel implementation.

## INTRODUCCIÓN

El incremento del tamaño y complejidad de ciertos problemas de simulación numérica en ingeniería han sido superados, en gran parte, mediante la ejecución de los mismos en paralelo. El creciente desarrollo tecnológico ha posibilitado la construcción de redes de PC's a un costo relativamente bajo en relación al costo de las supercomputadoras y con una performance aceptable. Entre las redes de PC's se cuenta con lo que suelen denominarse *"Clusters Beowulf"*, formados por procesadores Pentium o similar, corriendo bajo el sistema operativo Linux [2]. Como se mencionó en [1], el cluster utilizado está formado por 7 computadoras con diferentes características en cuanto al hardware. En general, el trabajo sobre redes heterogéneas implica pérdidas de eficiencia, las cuales pueden reducirse balanceando la carga de trabajo asignada a los procesadores. Por tal motivo fue necesario el estudio previo de cuestiones relativas al cluster, tales como: mflops alcanzados por procesador, velocidad y tiempo de transmisión entre dos nodos cualesquiera y estimación de los *pesos* correspondientes a cada procesador.
El análisis de performance que se describe en este trabajo se centra principalmente en los módulos Laplace y Navier Stokes (incompresible) de la librería PETSc-FEM. El mismo se llevó a cabo subdividiendo el programa principal en etapas. Se midieron los tiempos de ejecución y de cálculo totales y por etapa, para cada problema considerado. Estos datos fueron usados para calcular el speedup y la eficiencia. Los tiempos de comunicación se estimaron teniendo en cuenta los parámetros obtenidos para el modelo de transmisión de información en el cluster. Utilizando rutinas propias de PETSc [3], se estimaron los mflops alcanzados en cada caso.

## CARACTERISTICAS GENERALES DEL PROCESAMIENTO PARALELO

Cada nodo del cluster está formado por un procesador y una memoria local, y los mismos están conectados mediante una red Fast Ethernet. Estas características permiten identificar al cluster como un ambiente de memoria distribuida MIMD que utiliza el modelo de intercambio de mensajes para la comunicación. El diseño de algoritmos eficientes depende de factores tales como el número de procesadores y la capacidad de sus memorias locales, la topología del hardware, la velocidad de cálculo relativa a la velocidad de comunicación, y de propiedades asociadas con la red, tales como la latencia (tiempo mínimo para establecer la conexión entre los procesadores) y el ancho de banda (bandwidth).

### Tiempos de comunicación

El tiempo de transferencia de un mensaje entre dos procesadores viene dado por la función lineal [4]:

$$t_{com} = \alpha + \beta\, n \tag{1}$$

donde $\alpha$ es la latencia o tiempo de *startup*; $\beta$ es el tiempo necesario para transmitir 1 *byte*, y $n$ es la longitud del mensaje. $\theta = 1/\beta$ es el bandwidth. Para clusters Beowulf basados en Fast Ethernet, se han reportados valores de 10 $Mbyte/s$ para el ancho de banda, y valores cercanos a los 200 $\mu s$ para la latencia [2]. Las pruebas realizadas en el cluster permitieron arribar a los valores: $\alpha = 264\ \mu s$ y $\theta = 11\ Mbyte/s = 92\ Mbit/s$ con lo cual, el modelo lineal para la comunicación viene dado por

$$t_{com} = 2.64\ 10^{-4} + 0.0911\ n \tag{2}$$

### Velocidad de cálculo

La velocidad o performance de un procesador se mide generalmente en Mflops. Para evaluar tal velocidad en los nodos del cluster, se utilizó el LINPACK Benchmark [5], el cual consiste en la resolución de un sistema lineal mediante eliminación gaussiana con pivoteo. Los resultados en doble precisión para el problema de orden 1000 × 1000 fueron los siguientes:

- Nodo 1: 33.2 Mflops → P II 350 Mhz
- Nodo 2: 35.5 Mflops → P III 450 Mhz
- Nodos 3 - 7: 38.8 Mflops → P III 500 Mhz

### Balance de carga

Para poder cuantificar las diferencias de velocidades entre los procesadores, se realizaron distintas pruebas en forma secuencial, que permitieron arribar a las siguientes ponderaciones: Nodo 1: $\omega$ = 0.7 - Nodo 2: $\omega$ = 0.95 - Nodos 3-7: $\omega$ = 1. Estos valores son utilizados por PETSc-FEM al particionar la malla de elementos finitos. De esta manera, la carga de trabajo asignada a cada procesador resulta ser proporcional a su velocidad.

### Medidas de eficiencia de programas paralelos

Sea $t_1$ el tiempo de ejecución de un programa usando 1 procesador, y $t_p$ el tiempo de ejecución del mismo programa usando $p$ procesadores. Entonces, una de las medidas que suele usarse para cuantificar la performance de un programa paralelo es el *Speedup*, definido por:

$$S_p = \frac{t_1}{t_p} \tag{3}$$

Otra medida que suele ser de interés es la *Eficiencia*, que es el Speedup relativo al número de procesadores utilizados, y se define por:

$$E_p = \frac{S_p}{p} = \frac{t_1}{p\, t_p} \tag{4}$$

Existen otras formas equivalentes de escribirlas en función de los tiempos de comunicación, coordinación y overhead [6]

## ANÁLISIS DE PERFORMANCE DE PETSc-FEM

Tal como se describe en [1], las aplicaciones constan de dos módulos: la *"rutina del elemento"* y el *"programa principal"*. Este último fue dividido en 4 etapas:

1. *Lectura de la malla*
2. *Perfil de la matriz*
3. *Ensamble de la matriz y del residuo*
4. *Solución del sistema*

Las etapas 2 y 3 son cuasi-perfectamente paralelizables, mientras que la etapa 4 requiere mayor intercambio de información y resulta menos paralelizable. En las aplicaciones, la *lectura de la malla* es realizada por todos los procesadores a la vez, de manera tal que el tiempo transcurrido para esta etapa es el mismo con uno o $p$ procesadores. Dado que se considera un tiempo secuencial, fue excluído del cálculo del Speedup, resultando entonces:

$$S_p = \frac{(T_{global} - T_{ReadMesh})_1}{(T_{global} - T_{ReadMesh})_p} \tag{5}$$

En **Laplace**, la matriz del sistema es simétrica, de modo que se resuelve usando el método iterativo *Gradiente Conjugado* con precondicionamiento Jacobi. En **Navier Stokes**, el perfil de la matriz se obtiene una sola vez, manteniéndose para todas las iteraciones; y el sistema de ecuaciones se resuelve usando GMRES con precondicionamiento Jacobi, tomando un espacio de Krylov de dimensión igual a 50 tanto en 2D como en 3D.
Todos los casos propuestos fueron resueltos en forma secuencial en los Nodos 1 y 3 y en forma paralela con 2, 4 y 7 procesadores, para distintas combinaciones. Por razones de espacio se muestran solo algunos resultados. El Speedup se calculó tomando como tiempo secuencial el obtenido en el procesador más rápido: Nodo 3.

## ECUACIÓN DE LAPLACE

Este problema fue resuelto en la cavidad cuadrada $[0,1] \times [0,1]$ para diferentes casos correspondientes a mallas estructuradas de elementos cuadrangulares homogéneos.
En cada caso se especificó el número de elementos por lado $N$ y se fijó la velocidad en un lado del cuadrado: $u(x,0) = 1$. Este problema tiene 2 grados de libertad por nodo: las componentes de la velocidad, y por lo tanto un total de $(N+1)N$. Se consideraron valores de $N$ iguales a 50, 110, 350 y 500, llegando hasta 251.000 grados de libertad.

### Speedup

En las figuras 1 y 2 se muestran las curvas de speedup en función del número de procesadores con y sin balanceo del problema. Se observa que el speedup crece al aumentar el tamaño del problema y también al usar mayor cantidad de procesadores, indicando que el tiempo total disminuye. Al balancear el problema, el tiempo total transcurrido disminuye en promedio un 7.2% respecto al resultado obtenido con ponderaciones unitarias.

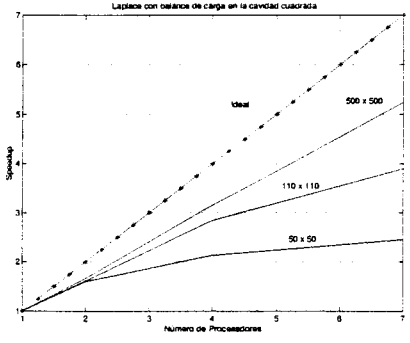


Figura 1: Speedup usando balance de carga

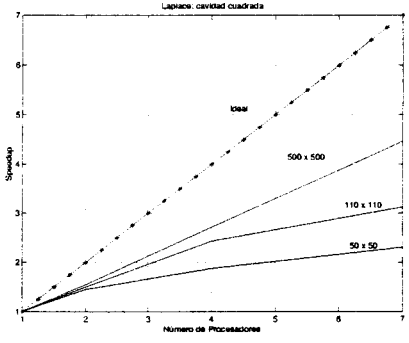


Figura 2: Speedup sin balance de carga

### Tiempos de ejecución y comunicación

El tiempo total transcurrido en función del tamaño del problema se muestra en la figura 3. Aquí se aprecia la ganancia de tiempo que se puede obtener al trabajar en paralelo, con o sin balance de carga. En la figura 4 se observa un diagrama de áreas donde se muestra la evolución del porcentaje del tiempo total que insume cada etapa en función del número de procesadores. Debido a que la lectura de la malla la realizan todos los procesadores, es una etapa que va adquiriendo cierto peso cuando se resuelve el problema en varios procesadores. La misma resulta ser un 15% del tiempo total con 1 procesador, y llega casi al 50% con 7 procesadores. Esto sugiere pensar en una partición de la malla en paralelo, aunque veremos que en problemas transientes, esta etapa deja de tener importancia. Se observa también que la etapa de resolución del sistema se mantiene constante, mientras que las otras disminuyen.

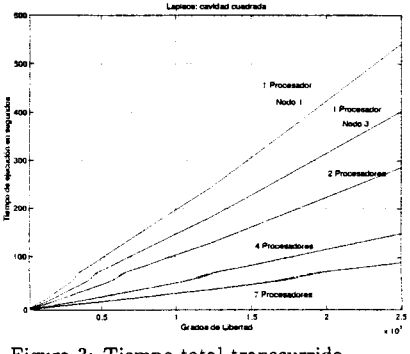


Figura 3: Tiempo total transcurrido

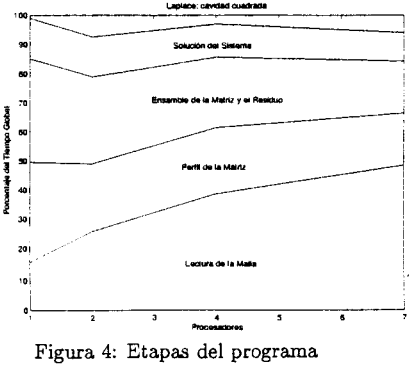


Figura 4: Etapas del programa

El tiempo de cpu pudo calcularse usando una función de PETSc, mientras que el tiempo de comunicación promedio se pudo estimar usando los parámetros de la función lineal dada por la ecuación (2), teniendo en cuenta el número promedio de mensajes enviados por cada procesador ($n_p$) y la longitud promedio de cada uno de ellos ($l_p$), siendo:

$$t_{com} = \alpha\, n_p + \beta\, n_p\, l_p \tag{6}$$

En general, el tiempo de cpu disminuye al utilizar más procesadores, a la vez que aumenta el tiempo de comunicación, aunque en un porcentaje muy pequeño; lo que compensa el tiempo de cpu es el tiempo de coordinación. Debido a las diferencias de velocidades entre los nodos algunos procesadores terminan y quedan ociosos esperando que terminen los otros.

## ECUACIÓN DE NAVIER STOKES

Este problema fue resuelto en la cavidad cuadrada y en la cavidad cúbica para diferentes casos correspondientes a un *Número de Reynolds* igual a 100. Se consideraron $N$ elementos por lado, teniendo 3 y 4 grados de libertad por nodo respectivamente. Se fijó $u(x,1) = 1$ en el caso 2D y $u(x,y,1) = 1$ para el caso 3D, siendo $u \equiv 0$ en los otros lados. La presión se fijó en un punto. Se propuso un número de Courant igual a 10 y se fijó el paso de tiempo en función del tamaño mínimo del elemento. En el caso 2D se probaron ejemplos de hasta 120.00 grados de libertad, mientras que en el caso 3D se llegó a tener 170.000.

### Speedup

En la figura 5 se muestran las curvas de speedup del caso 2D en función del número de procesadores usando balance de carga. Se observa el mismo comportamiento que con la ecuación de Laplace. Al balancear el problema, el tiempo total transcurrido disminuye respecto al resultado obtenido con ponderaciones unitarias, siendo en promedio del 2.7% con 2 procesadores y llegando al 10.5% con 7 procesadores. En la figura 6 se observan algunas curvas del caso 3D con y sin balance de carga. La disminución promedio de los tiempos totales es del 13% al balancear la carga.

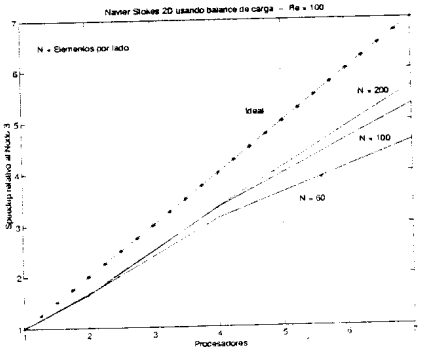


Figura 5: Speedup usando balance de carga - 2D

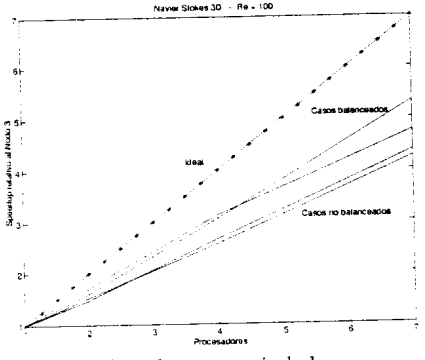


Figura 6: Speedup con y sin balance de carga - 3D

### Tiempos de ejecución y comunicación

En las figuras 7 y 8 se muestran los tiempos por paso temporal, casos 2D y 3D respectivamente, en función de los grados de libertad. Lo que se observaba con Laplace, es lo que se tiene ahora

en cada paso de tiempo, aunque en el caso 3D los tiempos casi se triplican respecto a la versión 2D .

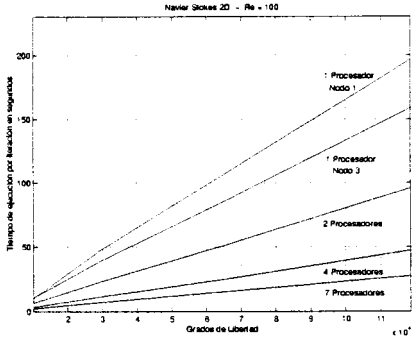


Figura 7: Tiempo por iteración - 2D

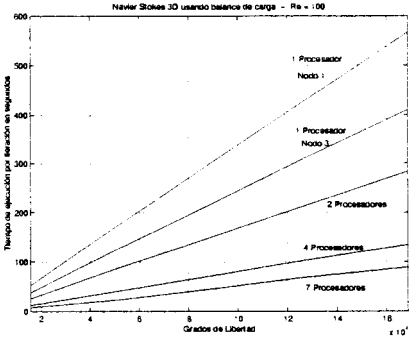


Figura 8: Tiempo por iteración

En las figuras 9 y 10 se observan diagramas de áreas con la evolución del porcentaje del tiempo total que insume cada etapa en función del número de procesadores. Las etapas: lectura de la malla y obtención del perfil resultan despreciables por lo que no se consideran. En el caso 2D la etapa de resolución insume casi un 70% del tiempo total, mientras que en el caso 3D apenas llega a ser del 18% para igual dimensión del espacio de Krylov. Esto explica porque los Mflops que se obtienen de los profiling de PETSc son bajos en el caso 3D y aceptables en 2D. Casi la totalidad de las operaciones que se realizan se concentran en la etapa de resolución, y PETSc obtiene los Mflops dividiento las operaciones realizadas por el tiempo total transcurrido. Por ejemplo, en el caso 2D con 1 procesador se alcanzan los 32 Mflops en el Nodo 1 y 39 Mflops en el Nodo 3. Con 2 procesadores se llega a 68 Mflops, similar a LINPACK. Con 4 procesadores se llega a los 140 Mflops usando la red homogénea formada por los nodos 3, 4, 5 y 6; y con 7 procesadores se obtienen 213 y 193 Mflops con y sin balance de carga respectivamente.

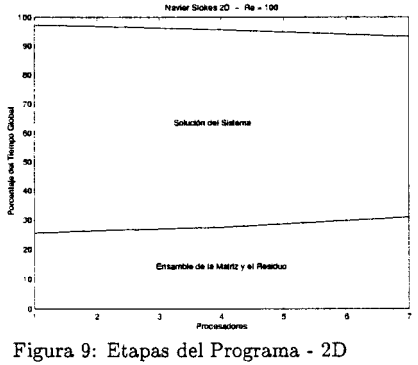


Figura 9: Etapas del Programa - 2D

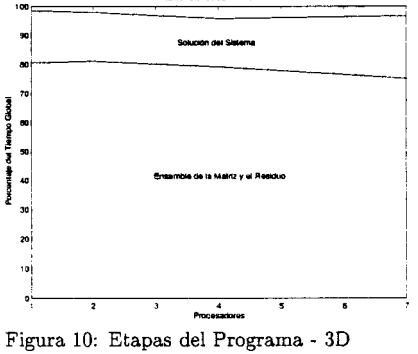


Figura 10: Etapas del Programa - 3D

El análisis de los tiempos de cpu y comunicación para Navier Stokes tanto en 2D como en 3D resultó ser similar al realizado con la ecuación de Laplace.

## AGRADECIMIENTOS

Este trabajo ha recibido soporte financiero del *Consejo Nacional de Investigaciones Científicas y Técnicas* (CONICET, Argentina), *Banco Interamericano de Desarrollo* (BID) y *Universidad*

*Nacional del Litoral* a través de los proyectos CONICET PIP 198/98, *"Germen-CFD*, SECyT FONCyT PICT 51 *"Germen"* y CAI+D-UNL-94-004-024.

## REFERENCIAS

[1] **N. Nigro, V. Sonzogni, M. Storti y A. Yommi.** *Implementación de un programa de elementos finitos de propósito general, multi-física para uso en entornos de cálculo distribuido*, a ser presentado en el ENIEF'2000, Noviembre 20-24, Bariloche, Argentina.

[2] **T. L. Sterling, J. Salmon, D. Becker y D. F. Savarese.** *How to Build a Beowulf*, The MIT Press, Massachusetts, Institute of Technology, 1999.

[3] **S. Balay, W. Gropp, L. C. McInnes y B. Smith.** *PETSc 2.0 Users Manual*, Argonne National Laboratory. Mathematics and Computer Science UC-405, 1997.

[4] **J. J. Dongarra, T. Dunigam**, *Message - Passing Performance of Various Computers*, Rep. University of Tennessee and Oak Ridge National Laboratory, 1997 - *www.netlib.org/utk/papers/commperf.ps.*

[5] **J. J. Dongarra**, *Performance of Various Computers Using Standard Linear Equations Software*, 1998. *(http://www.netlib.org/benchmark/).*

[6] **C. Succi, F. Papetti,** *An Introduction to parallel computational Fluid Dynamic*, 1996.