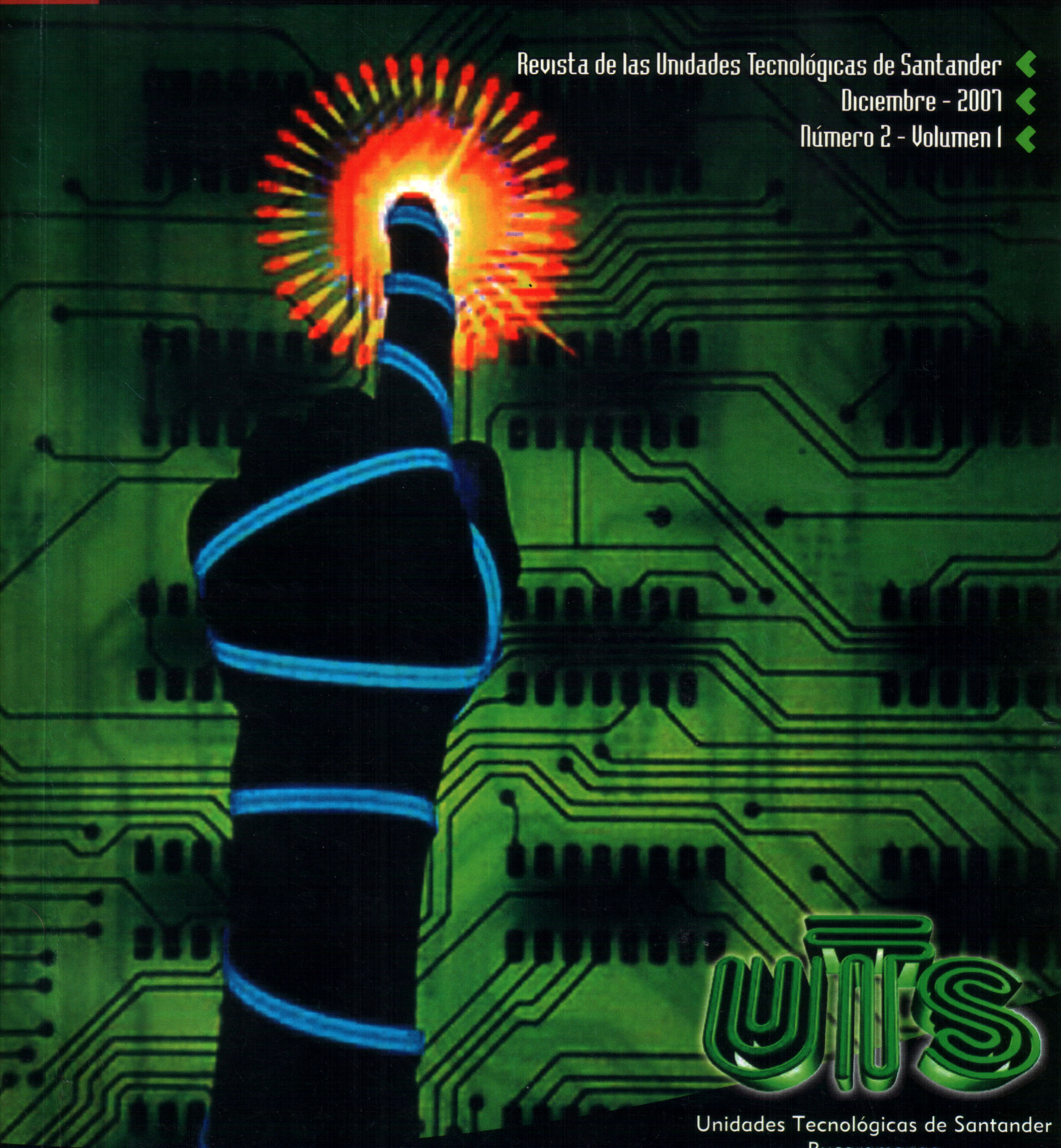
reset-UTS

ISSN 1909-258X

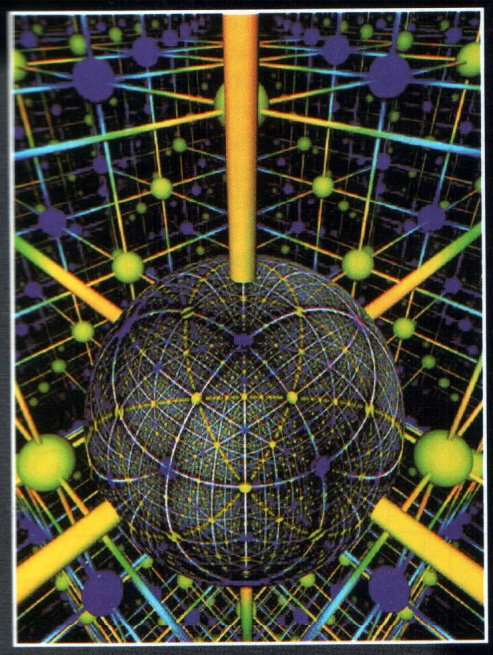
Revista Especializada en Sistemas Informáticos y Electrónicos de Telecomunicaciones



Revista de las Unidades Tecnológicas de Santander
Diciembre - 2007
Número 2 - Volumen 1

UTS

Unidades Tecnológicas de Santander
Bucaramanga



IMPLEMENTACIÓN DE UN CÓDIGO EN PARALELO DE DINÁMICA MOLECULAR PARA EL ESTUDIO DE ZEOLITAS

Cristian Blanco Tirado

Profesor Escuela de Química
Universidad Industrial de Santander
cris@ciencias.uis.edu.co
Bucaramanga. Colombia

Hernando Recaman Chaux

Estudiante de Maestría en Informática
Universidad Industrial de Santander
hrecaman@gmail.com
Bucaramanga. Colombia

RESUMEN. La Dinámica Molecular (DM) es una técnica computacional que monitorea de manera determinística la posición de las partículas de un sistema termodinámico en función del tiempo. Esto se logra mediante la integración de las ecuaciones clásicas de movimiento de Newton. En la DM las fuerzas de interacción entre las partículas se calcula a través de los potenciales de corto (Van Der Waals y enlazantes) y largo alcance (Coulomb). Es bien conocido que el cálculo de la fuerza es la etapa más demandante de recursos computacionales en los cálculos de DM. Este cálculo representa aproximadamente 90% del tiempo que un procesador tardaría en cada ciclo. En este trabajo se presentan los resultados de la implementación de un código de DM en paralelo para el estudio de zeolitas, escrito en C++, que utiliza las librerías MPI y que corre en un "cluster" de computadores del tipo Beowulf. Después de verificar la validez de los resultados mediante la comparación con los obtenidos con el código en serie se procedió a hacer el análisis de rendimiento y eficiencia de la paralelización a través de la evaluación del modelo, el método, la comunicación, la eficiencia y el costo de paralelización, obteniendo datos muy satisfactorios. Por otro lado, se ha utilizado una nueva metodología para el cálculo de las interacciones electrostáticas, que disminuyen sustancialmente los tiempos de ejecución.

Palabras claves. Dinámica Molecular, Zeolitas, Variables Termodinámicas, Programacion en Paralelo, MPI, Cluster Beowulf.

ABSTRACT. Molecular Dynamics (MD) is a computational technique used to determine the position of all particles within a thermodynamic system as a function of time. Particles' evolution in time is achieved by integrating Newton's equations of motion. Interacting forces among particles are computed via the calculation of short range (van der Waals and bonding energy) and long range (Coulomb) potentials. These calculations are the most demanding in terms of computing resources and represent about 90% of the time spent each cycle by a single processor machine. Here we present the results of developing a parallel MD code to study zeolites. This code was written in C++ and it uses the MPI Library to control messages passing. The code runs in a Beowulf cluster of computers. After checking the validity of the results by means of the comparison with the obtained ones with the code in series one proceeded to do the analysis of performance and efficiency of the paralelización across the evaluation of the model, the method, the communication, the efficiency and the cost of paralelización, obtaining very satisfactory information. In addition, we used a new methodology for the calculation of the electrostatic interactions in order to reduce execution times.

Keywords. Molecular Dynamics, Zeolites, Termodynamics Variables, programming in parallel, MPI, Beowulf Cluster.

INTRODUCCIÓN

La Dinámica Molecular (DM) es una técnica computacional ampliamente utilizada en diversas ramas de la ciencia. Su importancia radica básicamente en la posibilidad de determinar la trayectoria de las partículas de un sistema molecular mediante la solución de ecuaciones diferenciales sencillas, i. e. las ecuaciones de movimiento de Newton.

La trayectoria de las partículas permite determinar estadísticamente otras propiedades termodinámicas, por ejemplo temperatura, energía interna, entre otras, definiendo así el estado termodinámico en el cual se encuentra el sistema molecular. Todo cálculo de las trayectorias moleculares involucra la determinación de las fuerzas de interacción intra e intermoleculares, las cuales se pueden descomponer a su vez en fuerzas de corto (Van der Waals) y de largo alcance (Coulomb).

Las fuerzas de corto alcance se pueden calcular de forma directa y convergen rápidamente en función de la distancia (1 / r6) , mientras que las de largo alcance convergen muy lentamente debido a que su dependencia con la distancia es proporcional a (1 / r2). Para solucionar el problema de convergencia en el cálculo de las fuerzas de largo alcance, se han desarrollado metodologías que descomponen el problema en sumas en espacio real y recíproco, sumas de Ewald; estas metodologías aunque convergen rápidamente son costosas computacionalmente.

Es precisamente el cálculo de las fuerzas de largo alcance el factor limitante en el desarrollo de programas de Dinámica Molecular, debido a que el computador gasta la mayor parte del tiempo realizando el cálculo de estas fuerzas. Para eliminar el tiempo de cálculo de las fuerzas de largo alcance, y modelar sistemas moleculares más grandes, se han propuesto soluciones encaminadas a desarrollar metodologías que calculen la fuerza de forma eficiente o a desarrollar los códigos en paralelo.

Sin embargo, no existen en la actualidad sistemas de Dinámica Molecular que utilicen estos dos mecanismos para mejorar la rapidez y eficiencia en el cálculo de trayectorias dinámicas. Para contribuir en la solución del problema, este trabajo desarrolla e implementa un programa de Dinámica Molecular donde se conjuga el cálculo las fuerzas intra e intermoleculares en paralelo junto con la adaptación de metodologías eficientes en el cálculo de las fuerzas de largo alcance.

1. DINÁMICA MOLECULAR.

La técnica de DM se basa en la segunda ley de newton $F_i = m_i \times a_i$ donde Fi es la fuerza ejercida sobre cada átomo i, mi es la masa y ai la aceleración de cada partícula i. Gracias a que la energía potencial es una función local en un sistema molecular, la fuerza ejercida por todas las partículas del sistema sobre una en particular se puede determinar a partir de la energía potencial mediante la expresión:

$$F_i = - \frac{dV_i}{dr_i}$$

Si se conoce la fuerza en cada átomo se puede determinar la aceleración de las partículas en el sistema y por ende la nueva posición. Cuando se integran las ecuaciones de movimiento se obtienen las trayectorias de cada partícula en términos de la variación de las posiciones, velocidades y aceleración con el tiempo. Al conocer las trayectorias de cada átomo el estado termodinámico se puede predecir en el futuro o pasado. Mediante la trayectoria se pueden predecir la temperatura, la presión, energía total y otras variables termodinámicas de interés.[1][2]. En la Figura 1 se observa el diagrama de un programa de DM.

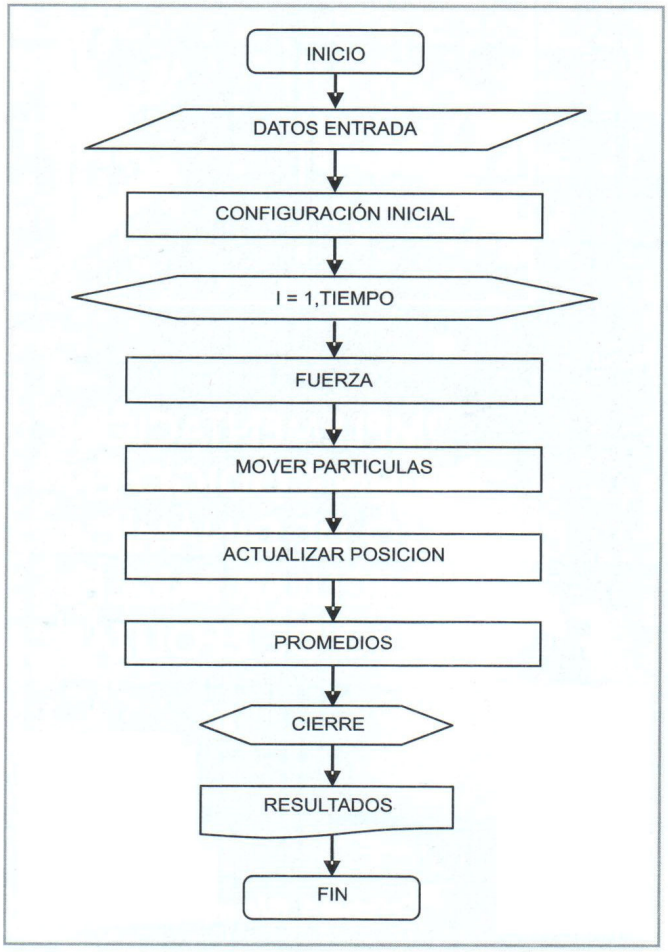


Figura 1. Diagrama de Flujo de un programa de dinámica molecular

En las simulaciones de sistemas zeolíticos se utilizan dos tipos de funciones disponibles para hacer cálculos, los de enlace de valencia y los iónicos. Mientras que los potenciales de enlace de valencia consideran el sistema como una colección de interacciones de corto alcance entre dos, tres y cuatro cuerpos, los potenciales iónicos representan el sistema como un conjunto de cargas que interaccionan vía fuerzas de corto y largo alcance. Las fuerzas intermoleculares entre las zeolitas y los adsorbatos se describen mediante interacciones de Lennard-Jones y términos Coulómbicos, los cuales se obtienen a través de cálculos mecanocuánticos o de datos espectroscópicos, al igual que la descripción de los enlaces, ángulos y torsiones moleculares. En la figura 2 se observan diferentes tipos de interacciones que existen entre las partículas de un sistema molecular.

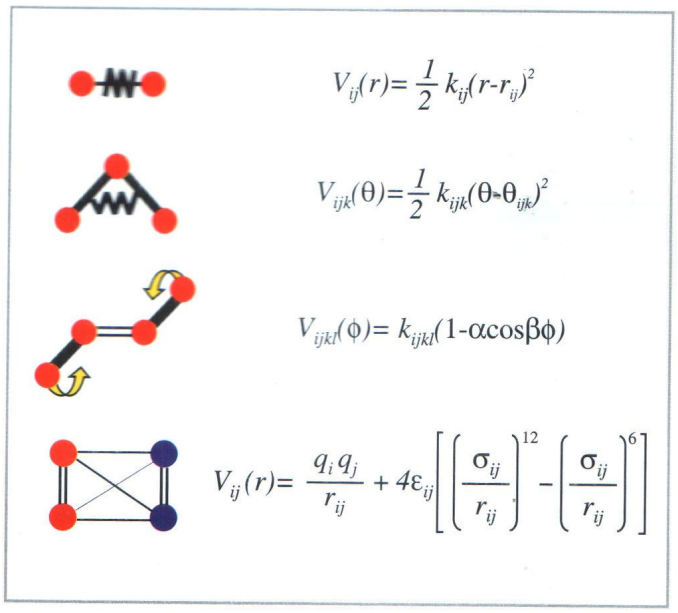


Figura 2. Interacciones intra e intermoleculares de sistemas microscópicos

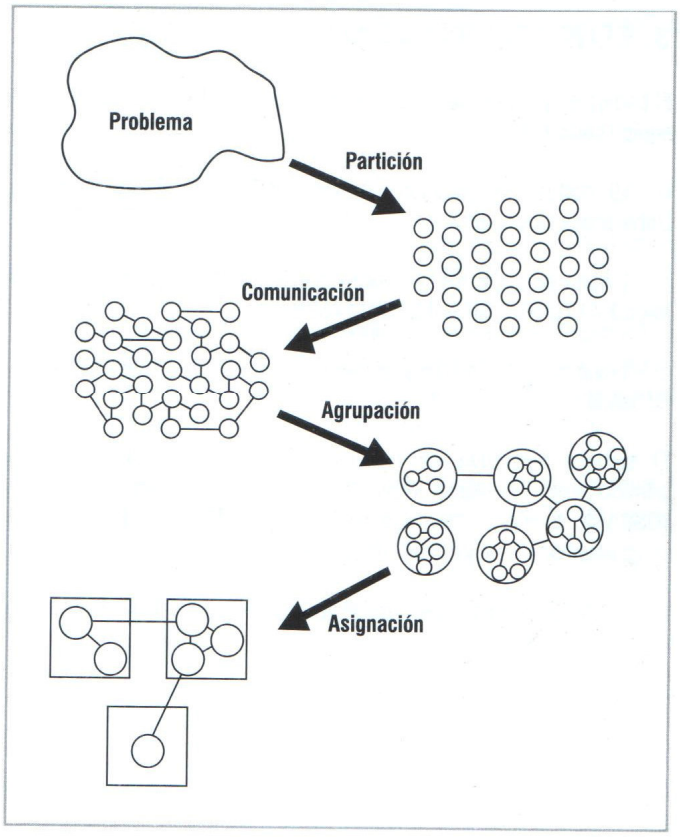
Todas estas interacciones son de corto alcance, convergen muy rápidamente y, como su nombre lo indica, a distancias cortas entre partículas. Finalmente se obtienen las interacciones electrostáticas, las cuales por naturaleza son de largo alcance. Debido a que este potencial es inversamente proporcional a la distancia, esta sumatoria no converge. Para solventar este problema, históricamente se han desarrollado una serie de algoritmos para el cálculo de las energías electrostáticas en materiales periódicos tridimensionales. Entre estos algoritmos está el método de Ewald, [3] el cual, a pesar de haber sido utilizado ampliamente, es costoso computacionalmente debido a su implementación (O(N2)).

2. PROGRAMACIÓN EN PARALELO

El objetivo principal de la programación en paralelo es disminuir el tiempo de ejecución en una aplicación mediante la utilización de métodos de programación que permitan la utilización de sistemas físicos de cómputo que posean varios procesadores[4].

Existen dos paradigmas en la ejecución de programas en paralelo, los de memoria compartida y los de memoria distribuida. En el proyecto se utilizan los clusters del tipo de Beowulf que utilizan memoria distribuida, permitiendo a cada procesador ser autónomo en su memoria[7], también emplea la librería MPI que contiene una serie de bibliotecas ensambladas en los compiladores tradicionales, por ejemplo FORTRAN y C++, para facilitar la comunicación, administración y control de las tareas que cada procesador realiza durante la ejecución del programa[7].

Para crear el código en paralelo se utilizan cuatro etapas fundamentales representadas gráficamente así:



Tomado de [2]

Figura 3. Pasos en la creación de un programa en paralelo

En la primera etapa se subdivide el problema en un gran número de tareas en orden de rendimiento, buscando que la granularidad sea fina y sin incremento del proceso de comunicación. Se analiza el problema y se observa si una solución paralela es buena o no por medio de la identificación de variables, su relación y los ciclos utilizados en ella, la subrutinas de DM paralelizadas que obtuvieron buenos resultados son las del cálculo de fuerzas y energía, aplicando la metodologías de Ewald y Wolf. La técnica de paralelización utilizada es la descomposición funcional, en cual un procesador maestro asigna los trabajos de acuerdo a su tamaño y orden a los procesadores esclavos.

En la segunda etapa se define el mejor tipo de comunicación en los procesos paralelizados, en la cual se seleccionó la comunicación Global, Estructurada, Estática y Sincrónica.

Para la tercera etapa se continúa con una revisión de las decisiones tomadas en las dos etapas anteriores con el objetivo de obtener un algoritmo paralelo eficiente.Las tareas y las estructuras de comunicación se evalúan con respecto a los requerimientos de ejecución y los costos de implementación. Si es necesario, las tareas se redefinen para combinarse o agruparse en tareas más grandes para optimizar la ejecución y el desarrollo.

En la etapa final de asignación se especifica en que procesador se ejecutará cada tarea minimizando los costos de comunicación y sincronización, finalizando así el proceso de paralelización[2][4].

3. CONFIGURACIÓN DEL SISTEMA

El código en paralelo se desarrolló en un cluster del tipo Beowulf. Este equipo consta de:

- 19 nodos computacionales, Hewlett-Packard, Proliant DL140, Doble procesador Intel Xeon 3.0 GHz HT, 1GB RAM, DD 80 GB SATA.
- 1 nodo principal, DELL PowerEdge 1850, Doble procesador Intel Xeon 3.4 GHz HT, 2 GB RAM, 2 DD 146GB SCSI.
- 1 switch 3COM 3870 manejable de 24 puertos capa L3, 1 GB ancho de banda.

El sistema operativo es RedHat Enterprise Linux 4, utiliza un administrador de cluster Open Source Cluster Aplication Resource (OSCAR), la interfaz de paso de mensajes MPI y los lenguajes de programación C++ y FORTRAN de Intel.

La evaluación del desempeño de un programa en paralelo se realiza de la siguiente forma:

3.1 MODELO Y MÉTODO

El modelo es un software y el método la simulación de eventos discretos.

3.2 COMUNICACIÓN

En la evaluación del rendimiento de la comunicación en el cluster del sistema se evalúa el ancho de banda y la latencia. Para evaluar estos valores se ejecuta un programa en paralelo que envíe 100 mensajes entre 2 nodos y se cambia el tamaño del mensaje obteniendo la siguiente información promedio:

Tamaño	Ancho de Banda (MB/s)	Latencia (microseg.)
50K	11.440966	190.500000
100K	11.456341	188.000000
1MB	11.456842	128.000000

Tabla 1. Rendimiento en la Comunicación

En la tabla 1 se observa que el ancho de banda se mantiene estable ante el cambio del tamaño del mensaje. Así mismo, se observa que el tiempo de latencia varía acorde al tamaño del mensaje. Se nota que el tiempo de latencia varía acorde al tamaño del mensaje transmitido al formar diferentes colas de transmisión.

3.3 ACELERACIÓN Y EFICIENCIA

Se realizaron un gran número de modelados siendo el más representativo, el realizado con 27678 átomos y 10 iteraciones, generando los siguientes resultados:

Procesadores	Serial(s.)	Paralelo (s.)
13	634	530
19	634	455
25	634	369
32	634	291
35	634	302
37	634	380
40	634	480

Tabla 2. Método de Wolf

Procesadores	Serial(s.)	Paralelo (s.)
13	2538	3000
19	2538	2400
25	2538	2280
32	2538	1790
35	2538	1605
37	2538	1580
40	2538	2612

Tabla 3. Método de Ewald

Para evaluar la información mostrada en las tablas 2 y 3, se determina la aceleración, considerada como la razón del tiempo que tarda el programa secuencial y el tiempo que tarda el correspondiente algoritmo paralelo en ser ejecutado en el mismo computador utilizando p procesadores. El valor resultante en Wolf es:

	P=13	P=19	P=25	P=32	P=35	P=37	P=40
Acel.	1.19	1.39	1.71	2.18	2.10	1.66	1.32

Tabla 4. Aceleración Método de Wolf

Para Ewald se obtiene:

	P=13	P=19	P=25	P=32	P=35	P=37	P=40
Acel.	0.85	1.06	1.11	1.41	1.58	1.61	0.97

Tabla 4. Aceleración Método de Ewald

En los resultados de la muestra se observa que la aceleración aumenta con el incremento en el número de átomos, debido al aumento de la diferencia en tiempo serial Vs tiempos paralelos.

Otro indicador es la aceleración en términos de porcentaje de código paralelizado, calculado a través de la ley de amhdal para la cual siendo T el tiempo del algoritmo secuencial, si f es la fracción de operaciones del algoritmo que hay que ejecutar secuencialmente ($0 < f < 1$), entonces la aceleración máxima obtenible ejecutando el programa en p procesadores es:

$$A_p = \frac{T}{T(f((1-f)/p))} = \frac{1}{f((1-f)/p)}$$

Proces.	Ewald (seg)	Wolf (seg)	Diferencia (veces)
17	5.1	0.8	6.3
21	3.8	0.8	4.75
25	3.7	0.8	4.62
27	3.7	1	3.7
31	3.7	1	3.7
33	5.2	1.3	4

Tabla 6. Comparación Ewald y Wolf

Cuando f -> 0, entonces Ap= p. Este límite nunca es obtenible debido a que inevitablemente parte del algoritmo es secuencial y la comunicación y sincronización de procesos consume tiempo. Aplicando la ecuación se obtiene para Wolf el 56% y para Ewald el 40%.

La eficiencia en términos de porcentaje de código paralelizable y número de procesadores utilizados es la aceleración dividida por el número de procesadores p. La eficiencia resultante en el método de Wolf es 0.068 y con Ewald 0.043.

3.4 RAPIDEZ ENTRE EWALD Y WOLF

Para obtener este parámetro se compara la muestra representativa paralelizada entre Ewald y Wolf con 2160 átomos.

Para las iteraciones centrales se produce:

Proces.	Ewald (seg)	Wolf (seg)	Diferencia (veces)
17	5.1	0.8	6.3
21	3.8	0.8	4.75
25	3.7	0.8	4.62
27	3.7	1	3.7
31	3.7	1	3.7
33	5.2	1.3	4

Tabla 6. Comparación Ewald y Wolf

La diferencia promedio en las iteraciones paralelas centrales es de 4.5 veces más rápida en Wolf con respecto a Ewald. En el programa en serie Ewald necesita de un tiempo de 5.3 seg. y en Wolf 1.3 seg., produciendo una diferencia de 4.07 veces.

4. CONCLUSIONES

La paralelización del código que calcula las fuerzas permite la duplicación de la aceleración con respecto al código serie.

En la codificación de un programa en paralelo es necesario crear el código en serie para comparar si el tiempo que dicho proceso produce es menor que la solución en serie.

Los mejores resultados en tiempos de respuesta para el programa en paralelo se obtienen cuando el dominio es muy grande y el número de procesadores es determinado por la cantidad de mensajes que se producen durante su ejecución.

El rendimiento del método de Wolf es superior en 4 veces con respecto al método de Ewald.

5. REFERENCIAS

[1] D.C. Rapaport, The Art of Molecular Dynamics Simulation, USA: Cambridge University Press, 2002, 1240 p.

[2] M. Cook and W. C. Conner, How big are the pores of zeolites? , in Procedings of the 12th International Zeolite Conference, edited by M.M. J. Treacy, B.K. Marcus, M. E. Bisher, and J. B. Higgins, pags 409-414, Material Research Society, Warrendale, PA, 1999.

[3] C. Blanco, Modeling Equilibrium and nonequilibrium dinamics in Zeolites and zeolites-guest systems, 2004, Trabajo de grado Chem., UMASS, Area de Química.

[4] O. Plata, Modelos y Arquitecturas Escalables, Universidad de Málaga, 2002.

[5] C. J. Hernandez, Especialización en Computación de Alto Rendimiento, Universidad Industrial de Santander, Agosto de 2006.

[6] I. Llorente, Computación Científica sobre Sistemas Multiprocesador, Universidad Complutense de Madrid, 2001.