CACIC 2005. XI CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACION

Predicción del cómputo paralelo de una aplicación sobre una colección de clusters geográficamente distribuidos.

Nombre de los autores, afiliación y direcciones

Eduardo Argollo de Oliveira Dias Júnior, Universidad Autónoma de Barcelona (UAB), Arquitectura de Ordenadores y Sistemas Operativos, Dep. Informática, Edifici Q, 08193 Bellatera (Cerdanyola del Vallès), España, eargollo@aows10.uab.es

Adriana Angélica Gaudiani, Universidad Nacional General Sarmiento (UNGS), Instituto de Ciencias, Informática, Módulo 1, Campus Universitario, José M. Gutiérrez 1613, Los Polvorines, Buenos Aires, Argentina, agaudi@ungs.edu.ar.

Dolores Isabel Rexachs, Universidad Autónoma de Barcelona (UAB), Arquitectura de Ordenadores y Sistemas Operativos, Dep. Informática, Edifici Q, 08193 Bellatera (Cerdanyola del Vallès), España, Dolores.Rexachs@uab.es

Emilio Luque, Universidad Autónoma de Barcelona (UAB), Arquitectura de Ordenadores y Sistemas Operativos, Dep. Informática, Edifici Q, 08193 Bellatera (Cerdanyola del Vallès), España, Emilio.Luque@uab.es

Abstract

El propósito de este trabajo es implementar en paralelo una aplicación utilizada para modelizar la transmisión sináptica de neuronas sobre dos cluster geográficamente distribuidos e interconectados por Internet, con el propósito de disminuir el tiempo empleado en su ejecución. Ejecutar a distancia utilizando Internet es una tarea compleja, aunque es posible. Nosotros lo demostramos con un modelo de implementación de dicha aplicación sobre dos clusters distribuidos ubicados en España y Argentina, usando la red Internet como red no dedicada para la interconexión entre ambos. En este trabajo se presenta dicho modelo, con el cual se predice el rendimiento de la aplicación estimando el valor de la granularidad que hace más eficiente el cómputo, debiendo hablar de dos niveles de granularidad, el que se implementará dentro de cada cluster y el que se implementa entre ambos clusters. Los resultados experimentales muestran que se logra más de un 90% de precisión en la predicción y que la mejora en los tiempos de cómputo alcanzan un 83% con la colaboración de ambos clusters.

Términos claves:
Cómputo paralelo, CoHNOW, High Performance Computing, modelo Master-Worker, balanceo de cargas.

Tema del Congreso:
Procesamiento Paralelo.

Predicción del cómputo paralelo de una aplicación sobre una colección de clusters geográficamente distribuidos.

Eduardo Argollo[1], Adriana A. Gaudiani[2], Dolores I. Rexachs[1], Emilio Luque[1]

[1]*Arquitectura de Ordenadores y Sistemas Operativos, Dep. Informática,*
Universidad Autónoma de Barcelona, España
eargollo@aows10.uab.es, *Dolores.Rexachs@uab.es, Emilio.Luque@uab.es*
[2] *Área de Informática, Instituto de Ciencias,*
Universidad Nacional General Sarmiento, Buenos Aires, Argentina
agaudi@ungs.edu.ar

1. Introducción

Una solución de bajo costo para el cómputo paralelo es usar plataformas heterogéneas y distribuidas, con base en las redes de estaciones de trabajo heterogéneas (HNOW).
El impacto de Internet en las comunicaciones y su incremento en el ancho de banda y confiabilidad hacen posible la interconexión de colecciones de clusters geográficamente distribuidos con el objetivo de mejorar el tiempo de cómputo. Estas colecciones de clusters, CoHNOWs, están compuestas por nodos o clusters que pueden estar esparcidos por el mundo y su red de interconexión no es de comportamiento estable. El ancho de banda de las comunicaciones dentro del cluster es de un orden de magnitud mayor que las que ocurren entre los clusters. [4]
Un entorno tan heterogéneo presenta muchas dificultades al intentar predecir el comportamiento del cómputo. A esto se le suma el difícil manejo de Internet como red de intercomunicaciones y su gran cantidad de variables que son casi inmanejables.
El desarrollo de una aplicación en paralelo está fuertemente condicionada por el sistema sobre el cual se va a implementar y por el modelo de programación que se desea utilizar. En este caso, en que el grado de heterogeneidad entre los componentes es tan alto, el modelo de programación que mejor permite manejarlo es el Master-Worker (M/W), modelo que utilizamos al paralelizar esta aplicación. [3]
El objetivo primordial de este trabajo es disminuir el tiempo de ejecución de la aplicación procesando en paralelo sobre dos clusters conectados por Internet, de manera de hacerlo lo más eficiente posible, aprovechando al máximo el poder de cómputo que ofrece este sistema. Para ello elaboramos un modelo de predicción y sintonización del comportamiento de la aplicación durante su ejecución, y comprobamos que es posible ejecutarla a distancia aunque la red de interconexión sea Internet. El desarrollo de este trabajo se basa en la metodología empleada por el grupo de investigación de la Unidad de Arquitectura de Computadoras y Sistemas Operativos, de la Universidad Autónoma de Barcelona, en su investigación sobre la ejecución eficiente de aplicaciones sobre clusters distribuidos en zonas distantes geográficamente, utilizando como benchmark una versión del algoritmo de multiplicación de matrices.[1][2] En nuestro trabajo seguimos los pasos que ellos emplearon en la verificación del modelo.
En las próximas secciones se presenta primeramente, en la Sección 2 el tipo de problema que simula la aplicación utilizada en este trabajo. En la Sección 3 se presenta el entorno de cómputo sobre el que trabajamos.. En la Sección 4 se presenta un modelo analítico de predicción del performance que se podrá alcanzar con el cómputo dentro del cluster primeramente, y la colaboración del cluster remoto posteriormente. En la Sección 5 se presentan resultados experimentales y las conclusiones y continuación de este trabajo en la Sección 6.

2. Problema que simula la Aplicación

Los sistemas complejos son sistemas, generalmente no lineales, presentan comportamientos colectivos que surgen de la dinámica del sistema como un todo. Sus modelizaciones, por medio de simulaciones numéricas hacen de la computadora la herramienta de implementación por excelencia. En ellos intervienen muchos grados de libertad y día a día incrementan su necesidad de capacidad de cómputo. Debido a esto, es de fundamental importancia contar con arquitecturas especializadas en acelerar la velocidad de cómputo de aplicaciones de este tipo. Los clusters de PC's ofrecen una solución rápida a esta necesidad sin grandes inversiones y en particular las colecciones de clusters son una alternativa interesante por facilitar el cómputo intenso.
La resonancia estocástica es un fenómeno que juega un papel fundamental al detectar débiles señales periódicas en presencia de ruido. Este fenómeno es de importancia en neurofisiología, en el estudio de la transmisión sináptica entre neuronas (TSN), siendo las neuronas modeladas por un sistema biestable, con dos estados estables: de disparo y de inactividad. El proceso ya fue simulado mediante un anillo de osciladores armónicos sobreamortiguados con transmisión unidireccional, que operando en un régimen de resonancia estocástica puede comportarse como una unidad de memoria de corto alcance sostenido por ruido. Este trabajo es parte del resultado de las investigaciones que llevan adelante el grupo de Sistema Complejos de la Universidad Nacional General Sarmiento. [5][6][8]
Con esta aplicación se extiende el estudio anterior a una red de osciladores y corresponde al modelo desarrollado por M.Carusela, siendo éste el objeto de estudio de dicha área.

3. Arquitectura del sistema de cómputo

El entorno de cómputo está compuesto por dos clusters ubicados en UNGS, San Miguel, Buenos Aires, Argentina, y en UAB, Bellaterra, Barcelona, España.
La arquitectura del sistema debe adecuarse a los distintos grados de heterogeneidad que éste presenta. Uno de estos niveles es el que interviene en el cómputo propiamente dicho. Está jerárquicamente organizado como Master-Worker, siendo el cluster local o principal, el de Argentina, y el cluster remoto o sub-cluster el de España.
El cluster principal está constituido por un master y sus workers, con diferentes capacidades de cómputo e interconectados por una LAN de 10 Mbps. El sub-cluster contiene un sub-master y sus sub-workers, que también presentan distintas capacidades de cómputo y están interconectados por una LAN de 100 Mbps. En ambos, el sistema operativo es Linux y la librería de comunicación entre workers es MPI. La figura 1 muestra la arquitectura del sistema.

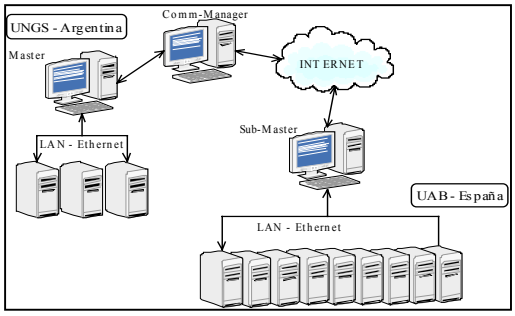

Fig. 1. Arquitectura del sistema de cómputo

Como se explica las comunicaciones también presentan un alto grado de heterogeneidad. La velocidad de las LAN difieren en un orden de magnitud y la conexión entre clusters está constituida por una WAN de comportamiento impredecible.
En cada cluster es adecuado el uso de MPI para las comunicaciones entre procesadores ya que no impide lograr un cómputo eficiente, gracias a la baja latencia y el alto rendimiento del procesamiento, pero esto no es así en la WAN, ya que este sistema falla con el comportamiento de las comunicaciones de Internet. [1]
Para solucionar este problema, las comunicaciones inter-cluster se implementan con un software especialmente diseñado para las comunicaciones a distancias, Communication Manager (CM), el

cual fue desarrollado por E. Argollo en la UAB. Este software cumple la función de administrar las comunicaciones entre el cluster y el sub-cluster monitoreando el comportamiento de Internet.
En ambos clusters se destinó un máquina para actuar como agente de comunicaciones, utilizando el CM. Estas máquinas se comunican con la LAN por medio de MPI y entre ellas con su propia librería de comunicaciones para acceso remoto. [1][7]

4. Modelo del sistema

Luego de tener un buen dominio del funcionamiento de la aplicación e identificadas las variables que determinan su complejidad fue necesario que desarrolláramos una metodología para sintonizarla y paralelizarla buscando lograr el mejor rendimiento posible con un sistema de cómputo altamente heterogéneo, formado por dos clusters geográficamente distribuidos.
A continuación detallamos el modelo que permite predecir el comportamiento de la ejecución y bajo qué condiciones se logra el mejor performance del sistema, dividiendo nuestro estudio dentro de un cluster y entre clusters remotos.

4.1 Predicción Intra-Cluster

Primeramente es importante analizar cómo la aplicación es escrita bajo el modelo M/W y de qué manera se distribuyen los datos y el trabajo para manejar la heterogeneidad del sistema. Los parámetros utilizados y la relación entre estos serán los datos que permitan obtener una correcta predicción de la colaboración que se obtendrá de los workers, para luego ser comprobada con los resultados experimentales.
El modelo de la aplicación trabaja sobre una red de osciladores armónicos sobre los cuales se propaga una señal que proviene de una fuerza aplicada al primero de ellos. Esta señal se propaga de manera unidireccional hacia los otros osciladores de la red siendo alimentada por ruido.
Al comienzo se calculan los valores iniciales de la amplitud en cada oscilador mediante una función random, o sea se calculan los valores iniciales de cada elemento de un arreglo bidimensional, cuya dimensión es de 30 x 30 y cada elemento de la red representa la potencia de cada oscilador.
En cada paso de la propagación se calcula la amplitud de cada oscilador y se guarda en la matriz. Estos valores son integrados una cantidad de veces predeterminada, simulando la evolución de la señal en el tiempo, obteniendo una matriz de resultados con la amplitud resultante en cada oscilador de la red. Este proceso se repite para 100 juegos de condiciones iniciales y se promedian las matrices obtenidas en cada uno. Durante la ejecución el master reparte tareas a cada worker, totalizando unas 100 tareas, cada una con diferentes condiciones de inicio.
La paralelización requirió repartir una cantidad adecuada de tareas para que los workers colaboren en el cómputo final de la manera más eficiente posible, enviando los datos parciales que van obteniendo al Master. Este los recolecta y sigue adelante con el procesamiento secuencial, aunque debe esperar por todos los resultados que le enviarán los workers.
La granularidad dentro de cada cluster está determinada por la cantidad de tareas que el master enviará a cada worker, y la granularidad entre clusters depende de las condiciones de la red de interconexión, y es la cantidad de tareas que el Master enviará a través de su administrador de comunicaciones (CM) al Sub-Master, para que éste reparta a sus workers. Su estimación corresponde a la predicción inter-clusters de la próxima sección.
La cantidad de cómputo que realiza la aplicación está determinada por la complejidad del algoritmo. La función que representa la cantidad de operaciones de punto flotante depende de la dimensión de la red de osciladores, N, de la cantidad de juegos de condiciones iniciales, P, y de la cantidad de pasos de integración, Kt.

P es la variable que interesa en nuestro estudio, ya que es la determinante de la granularidad. Su valor es el que indica la cantidad de tareas que el Master debe repartir por vez a cada worker, o al Sub-Master, a medida que cada uno finaliza hasta llegar a completar las 100 tareas totales de todo el cómputo.

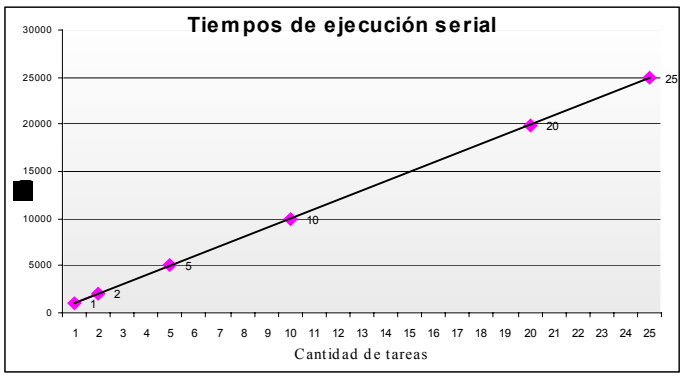

Fig. 2. Tiempos de ejecución serial en función de la granularidad

Los valores que fijamos para esta aplicación, en función de los requerimientos de sus usuarios, son N= 30, K_t= 2^{17} y cte= 600. Sus usuarios realizan sus ejecuciones variando otros parámetros inherentes al modelo, los cuales les permiten obtener resultados para análisis a posteriori sobre el problema físico. Estos parámetros no intervienen en el cálculo de la complejidad.

La expresión de la complejidad es de la forma:

$$F(P) = Cte * P$$

por lo tanto, el cómputo crece como O(P).

Este comportamiento se comprueba en la figura 2, que muestra los datos experimentales de las corridas en modo serial, variando el valor de P. Los datos del gráfico corresponden a uno de los nodos del cluster, aunque el comportamiento es idéntico en cualquiera de ellos.

Una de las medidas del trabajo que realiza el cluster está dada por las prestaciones límite que se pueden lograr (*ClusterPerfLimite)*. Para calcularlo analizamos la cantidad de cómputo y de comunicaciones que realiza durante el proceso. El tiempo de cómputo para realizar una tarea de granularidad P (*CompTime*), está determinado por el cociente entre la cantidad de operaciones de punto flotante, (*Operaciones*) y la suma del performance que cada worker del cluster obtiene para una tarea de esa medida (*LocalClusterPerf* (1)).

La carga de comunicaciones está dada por la cantidad de bytes que transmite cada worker al master al finalizar con una tarea. El worker envía una matriz de NxNx600 float, donde N es la dimensión de la red. Esta cantidad es constante, independientemente de la cantidad de tareas que realice.

El tiempo empleado en transmitir esta cantidad de datos (*CommTime*) resulta del cociente entre la cantidad de datos a transmitir (*Comm)*, y el throughput (*LanThPut)* de la red. (2)

El máximo performance del sistema será posible con el mínimo tiempo ocioso posible. Para lograrlo es necesario que el tiempo de comunicaciones sea menor o igual al tiempo de cómputo, en caso contrario la red se satura (3).

$$CompTime = \frac{Operaciones}{LocalClusterPerf} \quad (1)$$

$$CommTime = \frac{Comm}{LanThPut} \quad (2)$$

$$CompTime \geq CommTime \quad (3)$$

Según las ecuaciones (1) y (2), es posible encontrar el límite de prestaciones de la red para una tarea de grano P (*ClusterPerfLimit* (4)). El rendimiento esperado está limitado por el cómputo (*LocalClusterPerf)* y las comunicaciones (*ClusterPerfLimit)*, por lo tanto, se espera que esté acotado por las limitaciones propias del cluster, y por las limitaciones que surgen al exigir que el tiempo de comunicaciones no supere el de cómputo. El valor del performance esperado (*ClusterExpectedPerf*) nos permite estimar las prestaciones del cluster local para un valor determinado de la granularidad (5).

$$ClusterPerfLimit \leq \frac{P \cdot Operaciones \cdot LanThPut}{Comm} \quad (4)$$

$$ClusterExpectedPerf\ (P) = Min(LocalClusterPerf\ (P), ClusterPerfLimit) \quad (5)$$

El análisis fue hecho midiendo el rendimiento en tareas por segundo, en lugar de utilizar la cantidad de operaciones de punto flotante del algoritmo, de esta manera aseguramos más exactitud en los cálculos manteniendo la validez del modelo.

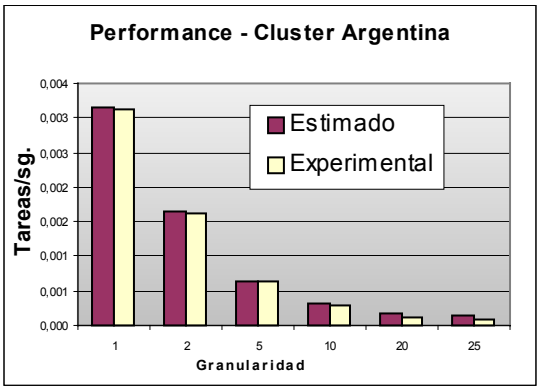

Fig. 3 Performance – Cluster Argentina

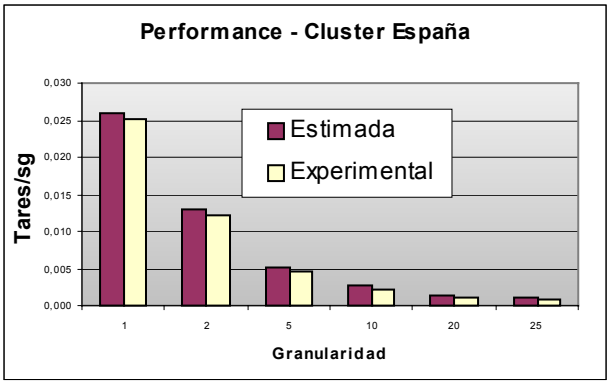

Fig. 4 Performance – Cluster España

Según los resultados obtenidos en las experiencias, una granularidad de 1 tarea es la más adecuada, debido a que es el valor que mejor permite balancear la carga. En las Figuras 3 y 4 se muestra el performance logrado en cada cluster. Al aumentar la granularidad el costo del desbalance de carga afecta en gran medida el rendimiento general obtenido. No es el uso de la red la que está limitando el cómputo sino las características propias del hardware de los workers quienes determinan las máximas prestaciones posibles, y éste es el límite que determina el performance esperado por el cluster local, en la ecuación 5. Esto se debe a que la aplicación realiza una gran cantidad de cómputo, que crece como O(P), y luego de cada tarea envía una matriz de resultados cuya dimensión no depende de la variable considerada en la granularidad, siendo siempre una cantidad fija de bytes. Como la tasa de transferencia de la red, *LanThPut*, es también constante, el valor del performance límite (*ClusterPerfLimit)* es constante en función de P y de varios órdenes de magnitud más grande que el performance local.

En la Tabla 1 se muestran los valores estimados por el modelo y los obtenidos experimentalmente en cada cluster independientemente uno del otro. Las ejecuciones se realizaron sobre un total de 100 tareas a repartir por el master, con 4 workers en el cluster de Argentina y 10 workers en el cluster de España.

		Performance (tareas/sg)					
	Grano	**1**	**2**	**5**	**10**	**20**	**25**
Argentina	LocalClusterPerf	0,00316	0,00164	0,00065	0,00032	0,00017	0,00015
	ClusterPerfLimit	0,47059	0,47059	0,47059	0,47059	0,47059	0,47059
	Prestaciones logradas	0,00311	0,00161	0,00062	0,00028	0,00013	0,00009
	Aproximación	98%	98%	96%	85%	75%	60%
España	LocalClusterPerf	0,02583	0,01291	0,00517	0,00258	0,00129	0,00103
	ClusterPerfLimit	4,53881	4,53881	4,53881	4,53881	4,53881	4,53881
	Prestaciones logradas	0,02503	0,01211	0,00456	0,00215	0,00102	0,00071
	Aproximación	97%	94%	88%	83%	79%	69%

Tabla 1. Predicción y ejecución sobre los clusters de Argentina y España.

De la tabla se deduce que el rendimiento esperado (ClusterExpectedPerf) coincide en muy buen porcentaje con los valores del rendimiento local (LocalClusterPerf) para bajas granularidades. Cuando el grano aumenta la diferencia se incrementa. Esto se debe a que aumenta el desbalance de la carga. Nuevamente se comprueba que la granularidad de 1 es la que mejor balancea la carga, teniendo también en cuenta que las comunicaciones no limitan el cómputo.

	Tiempo en comunicaciones (sg)					
	1	**2**	**5**	**10**	**20**	**25**
Argentina	212,50	106,25	42,50	21,25	10,63	8,50
España	21,25	10,63	4,25	2,13	1,06	0,85

Tabla 2. Tiempo empleado por cada cluster en transmitir los resultados según la granularidad.

En la Tabla 2 se muestra los tiempos empleados en transmitir las matrices resultados según la granularidad utilizada en el cómputo *(CommTime)*, en cada cluster. El valor de la velocidad de transmisión de la red de comunicación es de 8 Mbps en Argentina y 80 Mbps en España. Estos valores establecen un límite inferior para los tiempos de cómputo y no podrán mejorarse aunque contemos con todos los recursos necesarios a nuestra disposición. Son los tiempos mínimos que debe esperar el sistema y que deben ser menores que el tiempo de cómputo para mantener la eficiencia del sistema según se predice con el modelo (5).

4.2 Predicción Inter-Clusters

Del mismo modo que modelamos el performance estimado dentro de cada cluster, el proceso de cómputo colaborativo entre ambos clusters debe asegurar que el cluster remoto contribuya con sus mejores prestaciones. Esto requiere predecir la cantidad de carga que el cluster remoto repartirá a sus workers en función del comportamiento de las comunicaciones entre ambos.
La carga de trabajo que el master enviará al sub-master debe ser la adecuada para minimizar el tiempo ocioso de los workers de este último, de esta forma podemos lograr el performance óptimo de todo el CoHNOW.
Al igual que sucede en el análisis intra-cluster, el tiempo de computo debe ser mayor o igual que el tiempo de comunicaciones, para lograr la mejor contribución posible del cluster remoto al procesamiento. O sea, debemos asegurar que el performance de la red (*NetPerfLimit*) sea mayor que la capacidad del cluster (*ContribPerfLimit (6)*). El performance obtenido por el CoHNOW se estabilizará en el mínimo valor que resulte entre el performance esperado del cluster, (*ClusterExpectedPerf)*, obtenido del modelo intra-cluster ya analizado, y el performance con el que contribuye el cluster remoto *(ContribPerfLimit)*, el cual crece hasta alcanzar el máximo valor,.

$$NetPerfLimit \geq ContribPerfLimi\ t \quad (6)$$

$$NetPerfLimit = \frac{P \cdot NetThPut}{Comm} \quad (7)$$

$$P \geq \frac{ContribPerfLimit * Comm}{NetThPut} \quad (8)$$

De esta forma se obtiene el valor de la granularidad en función del valor que tome la tasa de transferencia de la red (8). Este valor es el que asegura el mejor rendimiento con la colaboración del cluster remoto en el cómputo y en función del estado de la red de interconexión.
A pesar que la tasa de transferencia cluster—sub-cluster o sub-cluster—cluster es impredecible, resulta fácil medir su promedio mediante la ejecución de un programa especialmente diseñado para medir las comunicaciones. Este valor (*NetThPut*) es calculado dinámicamente, de esta manera el CM ajusta los valores de carga durante la ejecución de la aplicación para mantener o mejorar el nivel de colaboración que presta el sub-master.
En definitiva el performance de colaboración que un sub-cluster puede aportar es:

$$ExpectedContribPerf = Min(ClusterExpectedPerf, ContribPerfLimit) \quad (9)$$

5. Experiencias

La ejecución de la aplicación con la colaboración de ambos clusters verifica los resultados que surgen del modelo.
El performance usado como referencia es el obtenido con la aplicación TSN, corrida en modo master-worker en cada cluster, de modo local e independiente uno del otro. Este resultado está representado en la tabla 3. como el benchmark con el que medimos el performance de los clusters de Argentina y España, en tareas por segundo, comparando la potencia de cómputo que cada uno es capaz de lograr localmente.
La ejecución en el CoHNOW se hizo repartiendo 100 tareas totales con una granularidad de 1 tarea en el cluster Master de Argentina, y una granularidad de 6 tareas que el CM envía al Sub-Master de España. El cluster de España repartirá las tareas a sus sub-Workers con granularidad 1. La granularidad del Master se obtiene de la predicción intra-cluster, según las estimaciones supuestas en 4.1, y la utilizada con el Sub-Master surge de los valores obtenidos con el software de comunicaciones y depende del estado de Internet en el momento que se hizo la experiencia y de las ecuaciones que surgen del modelo Inter-clusters (9).
Según la Tabla 3, el performance pico logrado por cada cluster es un poco menor que el benchmark ya que debemos tener en cuenta el overhead de cómputo que el software de comunicaciones CM le agrega al procesamiento. Los valores del performance contribuido disminuyen aún más debido al desbalance de la carga del sistema, aunque se logra una buena aproximación al benchmark, alrededor de un 80%.
Los resultados más significativos son los obtenidos en la estabilización del performance total, o sea el que alcanza cada cluster en su etapa de estabilización del cómputo en el CoHNOW, y según se puede ver son muy cercanos al del benchmark, más del 90%. Se logra un excelente aprovechamiento del sistema.

	Performance (tareas / sg)			
	Benchmark	Pico en Paralelo	Contribuida	Estabilización
Argentina	0,00266	0,00266	0,00216	0,0024
		100%	81%	90%
España	0,02457	0,02355	0,01880	0,0200
		96%	77%	81%

Tabla 3. Valores del performance obtenido al ejecutar el Benchmark, y su comparación con la ejecución final entre ambos clusters.

En la figura 5 se muestra el performance logrado en cada cluster y en el CoHNOW durante la ejecución, viendo cómo el cómputo tiende a un valor de estabilización gracias al trabajo del CM.
El cluster Master realizó 10 tareas y el Sub-Master colaboró con 90 de las 100. Como mostramos en la Tabla 3. el poder de cómputo del cluster de España es casi de un orden de magnitud mayor que el de Argentina, según los valores es 9,24 veces superior, y la experiencia comprueba que su colaboración fue del 90% del cómputo total.

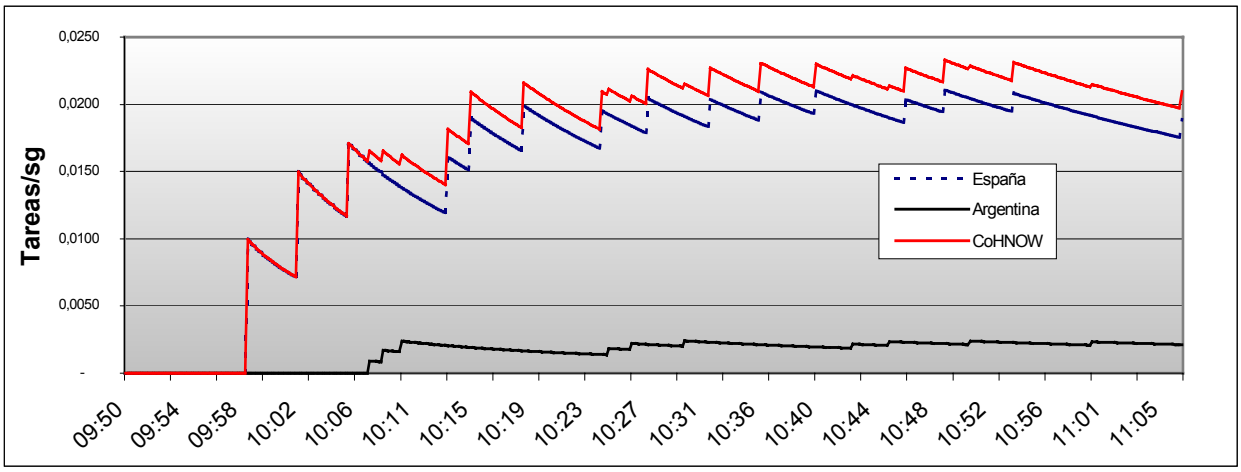

Fig. 5. Comportamiento del cómputo durante una ejecución en el CoHNOW

El performance que se logra alcanzar con el CoHNOW en el momento de estabilización del sistema es de 0,0224 tareas/sg. Teniendo en cuenta que éste finalizó con las 100 tareas al terminar el cómputo, nos permite predecir que el cómputo total será de 4586,16 sg. Experimentalmente obtuvimos un tiempo total de 4621,19 sg. (medido en el master). El error es del 1%.
El mejor tiempo de ejecución de la aplicación TSN medido en el cluster de Argentina, en modo M/W, con 4 workers, y utilizando granularidad 1 para minimizar el desbalance de cargas, fue de 32128 sg. El tiempo total de ejecución en el CoHNOW utilizando los parámetros que se deducen del modelo presentado fue de 4621 sg. Se logró una significativa reducción del tiempo total de cómputo del 83%.

6. Conclusiones

La aplicación utilizada en este trabajo, caracterizada por su gran cantidad de cómputo y por comunicaciones que no crecen con la medida del problema, se convirtió en un caso de prueba interesante para comprobar la capacidad de predicción del modelo y predecir el comportamiento del cómputo sobre clusters geográficamente distribuidos comunicados con una red impredecible como es Internet. Su ejecución permitió ajustar parámetros del software de comunicaciones logrando obtener la máxima colaboración entre clusters.

Por otro lado, y para satisfacción de los usuarios de la aplicación, esta forma de ejecutarla brindó una posibilidad de cómputo que, con recursos baratos y accesibles, mejora el performance obtenido con el CoHNOW en un 83% respecto a los mejores tiempos que podían lograr en el cluster local.[1]
Esta aplicación es fácilmente escalable, por lo cual extenderemos el estudio de su performance a clusters y CoHNOW´s con más potencia de cómputo para verificar los resultados del modelo, pero modificando las variables determinantes de la granularidad en esta nueva etapa como ser el tamaño de la red.
Esta aplicación será utilizada con intensidad por sus usuarios y consta de varios módulos de procesamiento, por lo cual en el futuro pretendemos que sea la base del diseño de un framework orientado a facilitarles el proceso de cómputo, de manera que los recursos disponibles sean transparentes, y les facilite el uso de clusters remotos que estén disponibles para colaborar en la ejecución. Esto será posible gracias al software de comunicaciones CM que de manera dinámica irá adaptando la carga del cómputo en el CoHNOW sacando el mejor provecho de Internet como la red de interconexión entre clusters.

Bibliografía

[1] E. Argollo, D. Rexachs and E. Luque, "Efficient Execution of Scientific Computation on Long-distance Geographically Distributed Clusters", PARA'04 State-of-the-Art in Scientific Computing June 20-23, 2004.

[2] O. Beaumont, F. Rastello and Y. Robert. "Matrix Multiplication on Heterogeneous Platforms", IEEE Trans. On Parallel and Distributed Systems, vol. 12, No. 10, October 2001.

[3] O. Beaumont, A. Legrand, and Y. Robert, "The Master-Slave Paradigm weigh Heterogeneous Processors", IEEE Trans. On Parallel and Distributed Systems, vol.14, No. 9, September 2003.

[4] R. Buyya. High Performance Cluster Computing: Architectures and Systems, volume 1. Prentice Hall PTR. 1999.

[5] M. Carusela, R. Perrazo, L. Romanelli. "Stochastic resonant memory storage device", Phys. Rev. E 64 031101, 2001.

[6] M. Carusela, R. Perazzo, L. Romanelli, "Information transmission and storage sustained by noise". Elsevier Science B.V. Physica D. 177-183, August 2002.

[7]A. Furtado, J. Souza, A. Rebouças, D. Rexachs y E. Luque, "Architectures for an Efficient Application Execution in a Collection of HNOWS", In: D. Kranzlmüller et al. (Eds.): Euro PVM/MPI 2002, LNCS 2474, pp. 450-460, 2002.

[8] R.Perazzo, L.Romanelli, R. Deza, "Fault tolerance in noise-enhanced propagation". Phys.Rev. E 61 R 3287, 2000.

[1] Estos resultados fueron presentados en un trabajo anterior: D. Persano, A. Gaudiani, D. Rexachs, E. Luque. "*Implementación en paralelo de un modelo de transmisión sináptica sobre un cluster heterogéneo*" (pp. 366--373)- CACIC 2003.