Procesamiento distribuido y paralelo. Fundamentos y aplicaciones.

Marcelo Naiouf, Armando E. De Giusti, Laura C. De Giusti, Franco Chichizola, Adrian Pousa, Waldo Hasperué, Victoria Sanz, Fabiana Leibovich
Instituto de Investigación en Informática LIDI (III-LIDI) - Facultad de Informática - UNLP
{mnaiouf, degiusti, ldgiusti, francoch, apousa, whasperue, vsanz, fleibovich}@lidi.info.unlp.edu.ar

CONTEXTO

La línea de Investigación que se presenta es parte del Proyecto “Algoritmos Distribuidos y Paralelos. Aplicación a Sistemas Inteligentes y Tratamiento Masivo de Datos” del Instituto de Investigación en Informática LIDI acreditado por la UNLP, y de proyectos apoyados por Cyted, CIC, IBM, Fundación YPF y Telefónica.

RESUMEN

El eje central de esta línea de I/D lo constituye el estudio de los temas de procesamiento paralelo y distribuido, tanto en lo referente a los fundamentos como a las aplicaciones. Esto incluye los problemas de software asociados con la construcción, evaluación y optimización de algoritmos paralelos y distribuidos sobre arquitecturas multiprocesador.

Los temas de interés abarcan: paralelización de algoritmos, paradigmas paralelos, métricas, escalabilidad, balance de carga, y modelos de computación paralela para la predicción y evaluación de performance sobre diferentes clases de arquitecturas de soporte. Tales arquitecturas pueden ser homogéneas o heterogéneas, como multicore, cluster, multicluster y grid.

Se trabaja en la concepción de aplicaciones paralelas numéricas y no numéricas sobre grandes volúmenes de datos que requieren cómputo intensivo, y en el desarrollo de laboratorios remotos para el acceso transparente a recursos de cómputo paralelo.

Palabras clave: *Sistemas paralelos. Algoritmos paralelos y distribuidos. Clusters. Multicluster. Grid. Balance de carga. Evaluación de performance.*

1. INTRODUCCION

El procesamiento paralelo y distribuido se ha convertido en un área de gran desarrollo actual dentro de la Ciencia de la Computación, produciendo profundas transformaciones en las líneas de I/D [1][2][3][4][5][8][9][10][11].

Interesa realizar investigación en la especificación, transformación, optimización y evaluación de algoritmos distribuidos y paralelos. Esto incluye el diseño y desarrollo de sistemas paralelos, la transformación de algoritmos secuenciales en paralelos, y las métricas de evaluación de performance sobre distintas plataformas de soporte (hardware y software). En esta línea de I/D la mayor importancia está dada en los *algoritmos paralelos*, y en los métodos utilizados para su construcción y análisis [6][7][26].

Si bien utilizar múltiples procesadores para resolver problemas, en general de complejidad creciente y para obtener resultados en menor tiempo, resulta un concepto intuitivo, los fundamentos subyacentes a los mecanismos y soportes de paralelización presentan numerosas variantes. Pueden encontrarse diferentes formulaciones paralelas para un problema, y el rendimiento y la eficiencia de cada una dependerá del algoritmo y de la arquitectura utilizada.

1.1. Algoritmos y Sistemas Paralelos

La creación de algoritmos paralelos y distribuidos sobre arquitecturas multiprocesador, o la transformación de un algoritmo secuencial en paralelo, no es un proceso directo. El costo del paralelismo puede ser alto en términos del esfuerzo de programación, ya que debe pensarse en la aplicación de técnicas nuevas reescribiendo totalmente el código secuencial, y las técnicas de debugging y tuning de performance no se extienden de manera directa al mundo paralelo [8][9][10][11].

Un *sistema paralelo* (SP) es la combinación de un algoritmo paralelo y la máquina sobre la cual éste se ejecuta; ambos factores poseen numerosas variantes y de un adecuado “matching” entre ellos depende el éxito de la solución [16][17].

Las arquitecturas para procesamiento paralelo y distribuido han evolucionado, y en la actualidad las redes de computadoras constituyen una plataforma de cómputo paralelo muy utilizada por sus ventajas en términos de la relación costo/rendimiento. La noción de sistema distribuido como máquina paralela es común a las denominaciones redes de computadoras, NOW, redes SMP, clusters, multiclusters y grid. En estos casos, se deben identificar las capacidades de procesamiento, interconexión, sincronización y escalabilidad [13].

El concepto de multicluster es una generalización que permite que redes dedicadas a una aplicación paralela se interconecten y puedan cooperar en un algoritmo, compartiendo recursos e incrementando la potencia de cómputo. Esto implica clusters dedicados interconectados, a diferencia de *grid computing* en que cada procesador puede realizar otras tareas independientes del algoritmo, brindando alta disponibilidad de procesamiento y/o de almacenamiento [14][15][16][17].

La caracterización y estudio de rendimiento del sistema de comunicaciones es de particular interés para la predicción y optimización de performance de los algoritmos, así como la homogeneidad o heterogeneidad de los procesadores que componen la arquitectura [18].

Muchos de los problemas algorítmicos en arquitecturas multiprocesador se han visto fuertemente impactados por el surgimiento de las máquinas *multicore* (que integran dos o más núcleos computacionales dentro de un mismo chip). Esto obliga al estudio de performance en sistemas híbridos, en los que se tiene una combinación de memoria compartida y distribuida.

1.2. Métricas del paralelismo

La diversidad de opciones en los SP vuelve complejo el análisis de performance, ya que los ejes sobre los cuales pueden compararse dos sistemas son varios (relacionados con el algoritmo y/o la arquitectura).

Existe un gran número de métricas para evaluar el rendimiento de los SP. Las tradicionales y más conocidas son el tiempo de ejecución paralelo, el speedup y la eficiencia. Otras características de los SP pueden ser analizadas por medio del costo, overhead paralelo, grado de concurrencia, etc. [19]

La *escalabilidad* de un SP permite capturar las características de un algoritmo paralelo y de la arquitectura en la que se lo implementa. Permite testear la performance de un programa paralelo sobre pocos procesadores y predecir su performance en un número mayor, y caracterizar la cantidad de paralelismo inherente en un algoritmo. Una de las principales métricas para medir la escalabilidad es la isoeficiencia.

Al tratar con multiclusters y grid, los problemas clásicos que caracterizan el análisis de los algoritmos paralelos, reaparecen potenciados por las dificultades propias de la interconexión a través de una red que en general es no dedicada. Esto se torna más complejo aún si se considera que cada uno de los nodos puede ser en si mismo una máquina multicore.

1.3 Balance de carga

El *balance de carga* consiste en, dado un conjunto de tareas que comprenden un algoritmo y un conjunto de procesadores donde ejecutarlas, encontrar el mapeo de tareas a procesadores que resulte en que cada una tenga una cantidad de trabajo que demande aproximadamente el mismo tiempo.

Un mapeo que balancea la carga de trabajo de los procesadores incrementa la eficiencia global y reduce el tiempo de ejecución. Este objetivo es particularmente complejo si los procesadores (y las comunicaciones entre ellos) son heterogéneos, y deben tenerse en cuenta las distintas velocidades.

El problema general de asignación es *NP*-completo para un sistema con *n* procesadores, y por lo tanto la tarea de encontrar una asignación de costo mínimo es computacionalmente intratable salvo para sistemas muy pequeños (métodos *óptimos*). Por esto pueden utilizarse enfoques que brindan soluciones subóptimas aceptables, como relajación, desarrollo de soluciones para casos particulares, optimización enumerativa, u optimización aproximada (métodos *heurísticos*) [20].

Si el tiempo de cómputo de una tarea dada puede determinarse “a priori”, se puede realizar el mapeo antes de comenzar la computación (balance de carga *estático*). En muchas aplicaciones la carga de trabajo para una tarea particular puede modificarse en el curso del cómputo, y no puede estimarse de antemano; en estos casos el mapeo debe cambiar durante el cómputo (balance de carga *dinámico*), realizando etapas de balanceo durante la ejecución de la aplicación.

El balance estático, en general, es de menor complejidad que el dinámico, pero también menos versátil y escalable. Los métodos dinámicos requieren alguna forma de mantener una visión global y algún mecanismo de análisis para la migración de procesos y/o datos, lo que agrega overhead de cómputo y comunicaciones.

No puede establecerse un método efectivo y que sea eficiente en *todos* los casos. Siempre la elección depende de la aplicación y la plataforma de soporte, y en muchos casos es necesario adaptar o combinar métodos existentes para lograr buena performance [24][27].

1.4 Modelos de representación, predicción y análisis de performance

Es importante referirse a un algoritmo paralelo mencionando el modelo de computación para el que

fue diseñado. Uno de los objetivos en la definición de un modelo de computación es la posibilidad de *predicción de performance* que brinde el mismo, teniendo en cuenta conceptos tales como comunicación, sincronización y arquitectura física. Estos factores impiden que alguno de los modelos existentes pueda ser usarse para *todas* las máquinas paralelas.

Respecto de la representación de las aplicaciones paralelas en arquitecturas distribuidas, existen diferentes modelos basados en grafos para caracterizar el comportamiento de las mismas [21]. Entre los modelos se pueden mencionar el modelo TIG (Grafo de Interacción de Tareas), TPG (Grafo de Precedencia de Tareas) y TTIG (Grafo de Interacción Temporal de Tareas) [22]. Sin embargo, estos modelos consideran que la arquitectura es homogénea, situación que en general no se da en cluster, multicluster y grid, lo que hace necesaria la investigación en esta área. El desarrollo de nuevos modelos de predicción y análisis de performance para estas arquitecturas requiere caracterizar el contexto de comunicaciones entre los procesadores y la asociación entre los algoritmos de aplicación, el paradigma de cómputo paralelo elegido y la arquitectura de soporte.

1.5 Evaluación de performance. Aplicaciones

Es de interés la evaluación de performance de distintas clases de aplicaciones sobre las arquitecturas disponibles. Muchos sistemas paralelos no alcanzan su capacidad teórica, y las causas de esta degradación son muchas y no siempre fáciles de determinar. El análisis permite estudiar el impacto que tienen algunos de estos factores sobre las implementaciones, y adecuar las métricas a las mismas. Interesa estudiar la influencia de las estrategias de distribución de procesos y datos, y la carga (estática o dinámica) asignada a cada procesador sobre el speedup, la eficiencia y la escalabilidad.

Desde el punto de vista de la relación costo/rendimiento, el cómputo paralelo sobre arquitecturas distribuidas ha ganado rápidamente espacio en el campo de las aplicaciones reales dado el bajo costo de los procesadores y estaciones de trabajo estándares junto con su alto rendimiento. Si bien existe una amplia gama de posibilidades estudiadas y publicaciones con todo tipo de aplicaciones resueltas, se acepta que es necesario continuar con la investigación en esta área [13][24].

Entre las aplicaciones de interés se encuentran las numéricas y no numéricas, el tratamiento de imágenes y video, reconocimiento de patrones en secuencias de ADN, bases de datos distribuidas, sistemas inteligentes, data mining, etc.

Por otro lado, interesa el desarrollo de laboratorios remotos para el acceso a recursos de cómputo paralelo. Esto implica software de administración de recursos físicos, comunicaciones y software disponible en clusters, multiclusters y grid, con el objetivo de permitir acceso transparente.

2. LINEAS DE INVESTIGACION y DESARROLLO

- Paralelización de algoritmos secuenciales. Diseño de algoritmos paralelos. Optimización de algoritmos.
- Lenguajes y bibliotecas para procesamiento paralelo y distribuido. Comparación de performance.
- Multithreading en multicore.
- Sistemas paralelos como combinación de software y arquitectura.
- Modelos y paradigmas de computación paralela. Combinación de pasaje de mensajes y memoria compartida en cluster de multicore.
- Modelos de representación y predicción de performance de algoritmos paralelos. Mapping y scheduling de aplicaciones paralelas.
- Métricas del paralelismo. Speedup, eficiencia, rendimiento, isoeficiencia, granularidad, superlinealidad. Escalabilidad de algoritmos paralelos en arquitecturas distribuidas.
- Balance de carga estático y dinámico. Técnicas de balanceo de carga.
- Análisis (teórico y práctico) de los problemas de migración y asignación óptima de procesos y datos a procesadores. Migración dinámica.
- Implementación de soluciones sobre diferentes modelos de arquitectura homogéneas y heterogéneas (multicores, clusters, multiclusters y grid). Ajuste del modelo de software al modelo de hardware, a fin de optimizar el sistema paralelo. Evaluación de performance.
- Laboratorios remotos para el acceso transparente a recursos de cómputo paralelo

3. RESULTADOS OBTENIDOS/ESPERADOS

- Formar recursos humanos en los temas del Subproyecto, incluyendo tesinas de grado y tesis de postgrado.
- Desarrollar y optimizar algoritmos paralelos sobre los diferentes modelos de arquitectura multiprocesador.
- Realizar la migración de soluciones paralelas en cluster a multicluster y grid.
- Evaluar la eficiencia, rendimiento, speedup y escalabilidad de las soluciones propuestas.

- Mejorar y adecuar las técnicas disponibles para el balance de carga (estático y dinámico).
- Estudiar y proponer optimizaciones para los modelos de predicción/evaluación de performance con diferentes paradigmas de interacción entre procesos, en esquemas multicluster y grid.

En este marco, pueden mencionarse los siguientes resultados:

- Se han utilizado diferentes tipos de arquitecturas homogéneas o heterogéneas:
  - Clusters comunicados en la misma LAN (cada cluster con un nodo principal que maneja las comunicaciones hacia los otros)
  - Clusters en redes diferentes con una conexión directa entre los nodos principales de cada uno vía fibra óptica.
  - Clusters en redes distintas conectadas por Internet en una WAN compartida y por Internet 2 con ancho de banda asegurado.
  - Infraestructura grid experimental en el marco del despliegue del Proyecto CyTEDGrid.
  - Máquinas multicore y cluster de multicore.
- En cuanto a modelos de representación y predicción de performance:
  - Desarrollo de dos modelos (TTIGHa y MPAHA) para representar aplicaciones paralelas de manera más realista considerando la heterogeneidad de los procesadores y de la red de interconexión [23].
  - Desarrollo de tres algoritmos de scheduling (MATEHA, MATEHAIB y AMTHA) [23], que permiten obtener una mejor distribución de procesos en procesadores.
  - Extensión del algoritmo de scheduling AMTHA para realizar la asignación de múltiples aplicaciones paralelas a una misma arquitectura. Dió origen a dos algoritmos de scheduling (SchCE_AMTHA y SchCD_AMTHA).
  - Se está trabajando sobre las modificaciones necesarias a los modelos y algoritmos de scheduling para adecuar sus usos sobre clusters de multicore y grid [27].
- Respecto de los algoritmos implementados, básicamente se trabajó con soluciones paralelas previamente tratadas en clusters:
  - ***N-Queens:*** consiste en ubicar *N* reinas en un tablero de *N*x*N* sin generar un ataque (esto ocurre cuando dos reinas están en la misma fila, columna o diagonal). Para resolver este problema se utilizó un modelo asincrónico, en el cual un porcentaje del trabajo es distribuido inicialmente considerando la potencia de cómputo de cada procesador/cluster. Cuando los procesadores terminan su trabajo le piden más al master. Se estudiaron speedup, eficiencia y balance de carga [24]. Se migró la aplicación a un grid con el objetivo de estudiar el overhead generado por el middleware incorporado. Y se estudia su migración a cluster de multicore.
  - ***N-Puzzle:*** es un problema de optimización discreta en el cual se debe llegar a un tablero "final" a partir de uno "inicial". Para esto deben realizarse intercambios entre una pieza y el lugar libre en el tablero (hueco) [25]. Se utilizó una variante de la heurística clásica de predicción de trabajo a realizar para llegar a una solución (tablero "final") y se demostró que su empleo mejora notoriamente el tiempo del algoritmo secuencial A*. Posteriormente se realizó una solución paralela sobre una arquitectura distribuida para analizar el speedup en función del número de procesadores, la eficiencia y la superlinealidad al escalar el problema. Este es un problema de interés por su aplicación principalmente en el campo de la robótica. Se está trabajando sobre la migración a cluster de multicore y la generalización del problema a multiobjetivos.
  - ***Análisis de Secuencias de ADN:*** el análisis de secuencias de ADN tiene múltiples aplicaciones, una de ellas es la búsqueda de similitud entre una secuencia *test* y las almacenadas en grandes bases de datos. El gran tamaño de las bases de datos, y la complejidad computacional para comparar dos secuencias (Smith-Waterman) hacen necesaria la paralelización del algoritmo [12]. Se diseñaron dos algoritmos de acuerdo a la centralización y/o replicación de la Base de Datos. Estos algoritmos se ejecutaron sobre arquitecturas de tipo cluster y grid, estudiando el speedup y la eficiencia alcanzable, la escalabilidad de las soluciones y el overhead introducido en grid por las comunicaciones y el middleware requerido. Se pretende extender las experiencias sobre GRID utilizando procesadores geográficamente distribuidos sobre diferentes redes ubicadas en Argentina, Brasil y España.
  - ***Extracción de conocimiento en grandes bases de datos utilizando estrategias***

adaptativas: la tecnología actual posibilita el almacenamiento de enormes volúmenes de información. Es importante encontrar métodos y técnicas de minería de datos que sean capaces de generar conocimiento útil, produciendo resultados que sean de provecho al usuario final. Para esto, se estudian métodos de clustering y clasificación de patrones para lograr asociar respuestas dinámicas con los datos de entrada obtenidos. Dado el gran volumen de información a procesar (que puede estar físicamente distribuida), se requiere analizar la arquitectura y los paradigmas de programación paralela utilizables de modo de minimizar el tiempo de cálculo del proceso adaptativo.

- En cuanto a los laboratorios para acceso remoto a recursos de cómputo paralelo:
  - Se ha desarrollado una aplicación que permite el acceso a los clusters de la Facultad de Informática de manera remota vía WEB (para investigadores y alumnos). Esto implica una capa de software de administración de los recursos y acceso transparente.
  - Se está trabajando en la generalización de la herramienta a clusters que no se encuentran en la misma red

4. FORMACION DE RECURSOS HUMANOS

Existe cooperación con grupos de otras Universidades del país y del exterior.
Dentro de la temática de la línea de I/D se ha concluido una tesis doctoral en 2008 y se espera concluir este año otras 3 tesis que se encuentran en curso y 2 tesis de maestría. También se concluyeron dos Tesina de Grado de Licenciatura y se espera finalizar otra.
Además, se participó en la definición y el dictado de la carrera de Especialización en Cómputo de Altas Prestaciones y Tecnología Grid de la Facultad de Informática de la UNLP.

5. BIBLIOGRAFIA

[1] Basney J., Livny M.. "Deploying a High Throughput Computing Cluster". Buyya R. Ed., High Performance Cluster Computing: Architectures and Systems, Vol. 1, Prentice-Hall, Upper Saddle River, NJ, USA, pp. 116-134, 1999.

[2] Vijay K. Garg. “Elements of Distributed Computing”. Wiley-IEEE Press, 2002.

[3] Liu M. L. “Distributed Computing: Principles and Applications”. Addison Wesley; 1st edition, 2003.

[4] Grama A., Gupta A., Karypis G., Kumar V. "Introduction to Parallel Computing", Pearson Addison Wesley, 2nd Edition, 2003.

[5] Attiya H., Welch J. “Distributed Computing: Fundamentals, Simulations, and Advanced Topics”, (Wiley Series on Parallel and Distributed Computing). Wiley-Interscience; 2nd edition, 2004.

[6] Berman K. A., Pau J. L. “Algorithms: Sequential, Parallel, and Distributed”. Course Technology; 1st edition, 2004.

[7] Wilkinson B., Allen M. “Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers (2nd Edition)”. Prentice Hall, 2004.

[8] Akl S. “Parallel Computing. Models and Methods”. Prentice Hall. 1997.

[9] Andrews G. “Foundations of Multithreaded, Parallel and Distributed Programming”, Addison Wesley, 2000

[10] Leopold C. "Parallel and Distributed Computing. A survey of Models, Paradigms, and Approaches", Wiley Series on Parallel and Distributed Computing. Albert Zomaya Series Editor, 2001

[11] Jordan H. F., Alaghband G., Jordan H. E. "Fundamentals of Parallel Computing", Prentice Hall, 2002¨

[12] Chichizola F., Naiouf R. M., De Giusti L. C., Rodríguez I., De Giusti A. E. “Parallel Processing ADN Sequences on Multicluter and Grid Architectures. Software Overhead”. Proceedings del IX Workshop de Procesamiento Distribuido y Paralelo. La Rioja, Argentina. ISBN: 978-987-24611-0-2. Octubre de 2008.

[13] Tinetti F. “Cómputo Paralelo en Redes Locales de Computadoras”. Tesis Doctoral. Universidad Autónoma de Balencia. 2004.

[14] IEEE Task Force on Cluster Computing (www.ieeetfcc.org)

[15] Foster I., Kesselman C. “The Grid 2: Blueprint for a New Computing Infrastructure”. (The Morgan Kaufmann Series in Computer Architecture and Design). Morgan Kaufmann; 2nd edition, 2003.

[16] Juhasz Z., Kacsuk P., Kranzlmuller D. (Eds), “Distributed and Parallel Systems: Cluster and Grid Computing”. (The Intl Series in Engineering and Computer Science). Springer; 1st edition, 2004

[17] Silva V. “Grid Computing For Developers” (Programming Series). Charles River Media; 1st edition, 2005.

[18] Kurmann C., Rauch F., Stricker M. "Cost/ Performance Tradeoffs in Network Interconnects for Clusters of Commodity PCs". Technical Report 391, Swiss Federal Institute of Technology Zurich, Institute for Computer Systems, January 2003

[19] Sun X-H . "Scalability versus Execution Time in Scalable Systems". Journal of Parallel and Distributed Computing, Number 12, 2002, pp 173-192

[20] Naiouf R. M. “Procesamiento paralelo. Balance dinámico de carga en algoritmos de sorting”. Tesis doctoral. Universidad Nacional de La Plata, 2004.

[21] Kalinov A., Klimov S.. “Optimal Mapping of a Parallel Application Processes onto Heterogeneous Platform”. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS’05), April 2005

[22] Roig C., Ripoll A. Guirado F. “A New Task Graph Model for Mapping Message Passing Applications”. IEEE Transactions on Parallel and Distributed Systems, vol 18 (12), pp.740-1753. Diciembre 2007.

[23] De Giusti L. “Mapping sobre Arquitecturas Heterogéneas”. Tesis Doctoral, Universidad Nacional de La Plata (2008).

[24] Naiouf R. M., De Giusti L. C., Chichizola F., De Giusti A. E. “Dynamic Load Balancing on Non-homogeneous Clusters”. G.Min et al. (Eds.): ISPA 2006 Ws, LNCS 4331, pags. 65-73, 2006. Springer – Verlag. Berlin Heidelber 2006.

[25] Sanz V., De Giusti A. E., Chichizola F., Naiouf R. M., De Giusti L. C. “Parallel Processing Puzzle N^2-1 on Cluster Architectures. Performance Analysis”. 30th International Conference on Information Technology Interfaces, 2008. ITI 2008. 23-26 Page(s):879 – 884. June 2008.

[26] Qiu X., Fox G. G., Yuan H., Bae S., Chrysanthakopoulos G., Nielsen H. F. “Performance of Multicore Systems on Parallel Data Clustering with Deterministic Annealing”. LNCS 4331, pags. 407-416. ISBN 978-3-540-69383-3. Springer Berlin / Heidelber 2008.

[27] Yi Liu, Xin Zhang, He Li, Depei Qian. “Allocating Tasks in Multi-core Processor based Parallel Systems”. Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing – Workshops. IEEE Computer Society. 2007.