Programación Paralela con Esqueletos: fundamentos e implementación

F. Saez, M. Printista

LÍNEA DE INVESTIGACIÓN EN PARALELISMO

LIDIC- UNSL

Ejército de los Andes 950, San Luis, Argentina.

e-mail: {**bfsaez@unsl.edu.ar, mprinti@unsl.edu.ar**}

Resumen: Los sistemas de programación paralela con esqueletos han sido desarrollados para simplificar la tarea de desarrollo de software paralelo. Esta forma de programación permite reducir los tiempos de diseño, codificación, testeo y mantenimiento.

Dentro de la línea de investigación dedicada a metodologías de programación paralela se tiene como objetivo el desarrollo de una librería que permita no sólo desarrollar soluciones paralelas a problemas científicos comunes (regulares), sino que además tenga la flexibilidad suficiente para resolver problemas complejos (irregulares).

Keywords: Programación Paralela Estructurada, Esqueletos, Conceptos de Diseño

1 Introducción

La principal motivación de la programación enfocada en esqueletos es abstraer un patrón común de paralelismo y ofrecer al programador una herramienta de alto nivel que acelere el desarrollo de software paralelo. Esta metodología de programación reduce los tiempos de diseño, pruebas y de codificación, a cambio de un mínimo conocimiento de los principales constructores de esta metodología.

Entre los sistemas representativos basados en esqueletos podemos nombrar al sistema P3L de la Universidad de Pisa [11], su análogo comercial SKIECL de QSW Ltd. [10], la librería eSkel desarrollada por Cole en la Universidad de Edinburgo [3], y más recientemente el entorno de programación paralela en java completamente basado en esqueletos Lithium y los entornos ASSIST [8] y Muskel [9] orientados al uso de grid. Todos estos sistemas proveen al usuario con un número fijo de esqueletos de alto nivel, que pueden ser configurados para solucionar una aplicación particular en forma paralela.

2 Programación con esqueletos

Un esqueleto es una herramienta genérica que permite solucionar un problema concreto, a partir de la instanciación de una aplicación específica. Desde estos programas esqueletos pueden ser derivados modelos que ilustran como el paradigma resuelve problemas específicos [2]. El paradigma de programación tiene como premisa, que los programas paralelos están compuestos de componentes paralelas que implementan un patrón (Ej. Divide y vencerás, pipeline, Automata Celular, etc.) y componentes secuenciales específicas de una aplicación (Ej. La división binaria en Quicksort, el producto escalar en la multiplicación de matrices, etc). En esta sociedad entre el programador y el esqueleto, el programador codifica las componentes secuenciales de un problema específico, y los aspectos de comunicación y sincronización entre procesadores son solucionados por el esqueleto.

Dentro del marco de la programación paralela, y específicamente el desarrollo de sistemas de programación esqueletal, nuestro interés esta enfocado en los siguientes aspectos:

1. Desarrollar e implementar una librería de constructores de alto nivel que permita cubrir una amplia gama de soluciones.

2. Ampliar el catálogo existente de aplicaciones modelos que pueden resolverse con la librería.

3. Estudiar e implementar las características fundamentales que debe poseer un sistema de programación esqueletal para ampliar el soporte a la solución de problemas algoritmicamente complejos.

4. Definir una terminología estandar que permita una clarificación de los conceptos fundamentales en el diseño de esqueletos.

5. Considerar no tan solo la performance de un esqueleto, sino también la usabilidad como métricas importante en el diseño de los mismos.

6. Trabajar en la búsqueda de un modelo que permita obtener predicciones acerca del comportamiento de los programas.

Con el propósito de formalizar las propiedades fundamentales de diseño que debe satisfacer un sistema de programación con esqueletos, nos adherimos a la terminología expuesta por Cole [1], la cual es clara y simplifica la comprensión de los conceptos.

Entre los conceptos fundamentales de diseño que se han tenido en cuenta en nuestros esqueletos, se pueden citar el $Anidamiento$ (Nesting), el ***Modo de interacción*** y el $Spread$.

El anidamiento de esqueletos es una propiedad importante que debe satisfacer un sistema esqueletal para poder combinar e integrar las distintas facilidades formando así estructuras no convencionales que permitan resolver algoritmos más complejos. También permite que el sistema simplifique la expresión de algoritmos paralelos que operan sobre estructuras de datos irregulares, ya que permite expresiones directas de conceptos como "En paralelo, por cada vértice en un grafo, buscar su vecino mínimo" o "En paralelo, por cada fila en una matriz, sumar sus elementos". En ambos casos las acciones internas (Buscar el vecino mínimo o sumar la fila) también pueden ser realizadas en paralelo.

El modo de interacción es otro aspecto que debe tenerse en cuenta si lo que se requiere es flexibilidad. la estructura de un esqueleto también debería permitir que se produzcan interacciones no contempladas por el patrón y de esa manera atender necesidades excepcionales de la aplicación.

En sistemas de programación paralela como MPI, el programador necesita especificar el tipo, la cantidad de datos y un buffer para la transferencia de datos (para envío o recepción). Esta forma de especificar los datos es insuficiente a la hora de tener un dato distribuido entre un conjunto de procesos y que se considere lógicamente completo. Esta carencia de MPI y otros sistemas es solucionada a través del ***Spread***, el cual especifica como las distintas contribuciones de datos que los distintos procesos que conforman el esqueleto, deberan ser tratadas (como por ejemplo, como datos completos o como una parte de una "variable distribuida").

La implementación de la librería de esqueletos se desarrolla en MPI. Esta interfase de programación nos permite, como implementadores de esqueletos, controlar la descompocisión de tareas y la administración de la comunicación y sincronización entre procesos. La opción de trabajar con MPI surge de su calidad de

estándar, ya que es un requisito fundamental lograr implementar una herramienta que sea adoptada rápidamente por los programadores.

La librería cuenta con un prototipo para el patrón Divide y Vencerás y distintas versiones del mismo; como son los Hipercubos Binarios y las cláusulas paralelas $forall$ [6, 4, 7, 5]. Para el diseño de los mismos se han analizado aspectos fundamentales como lo son la usabilidad, aplicabilidad y performance. Para poder obtener medidas de la ejecución de estas implementaciones, se han desarrollado un amplio conjunto de aplicaciones modelos que cubren las distintas opciones a las que se pueden adaptar el esqueleto. Entre las aplicaciones desarrolladas podemos nombrar algoritmos de ordenamiento, suma de vectores, transformada rápida de Fourier, búsqueda de componentes conexas, etc.

3 Conclusiones y Trabajos Futuros

En este trabajo, se comentaron algunos de los conceptos de diseño que deben soportar los sistemas esqueletales si se desea obtener un entorno de programación flexible que ayude al programador a resolver una amplia gama de problemas irregulares y complejos en forma paralela.

Como trabajos a cumplir en forma próxima se encuentran la incorporación de nuevos esqueletos a la librería, la ampliación del catalogo con problemas complejos del mundo real, el estudio y análisis de características avanzadas (balance de carga, tolerancia a fallos, portabilidad, etc.) y la busqueda de un modelo unificado que permita obtener predicciones acerca del comportamiento de los programas.

Cada uno de estos puntos deben ser desarrollados y mejorados si queremos que un sistema de programación esqueletal alcance los objetivos de programabilidad, portabilidad, performance y lo más importante, que demuestre que se puede obtener beneficios con su uso.

Agradecimientos

Los autores deseamos agradecer a la Universidad Nacional de San Luis, la ANPCYT y el CONICET por su contínuo soporte en el desarrollo de nuestras investigaciones.

Bibliografía

[1] M. I. Cole. A. Benoit. *Two Fundamental Concepts in Skelletal Parallel Programming*. ICCS 2005, LNCS 3515, 2005.

[2] M. I. Cole. *Algorithmic Skeletons: Structured Management of Parallel Computation. Research Monographs in Parallel and Distributed Computing.* Pitman, London, UK., 1989.

[3] M. I. Cole. *eSkel: The edinburgh Skeleton library Version 2.0*. Draft API reference manual. Internal Paper, School of Informatics, University of Edinburgh, 2003.

[4] C. Rodriguez León F. Piccoli, M. Printista. *Dynamic Hypercubic Parallel Computations*. Proceeding (466) Parallel and Distributed Computing and Systems, 2005.

[5] M. Printista F.D. Saez. *Programación Paralela Esqueletal*. XIII Congreso Argentino de Ciencias de la Computación, Corrientes and Resistencia, Argentina, October 2007.

[6] M. Printista F.D. Saez, R. Gallard. *Paradigms of Parallel Programming*. Workshop de Investigadores en Ciencias de la Computación (WICC 2003), Tandil, Argentina, May 2003.

[7] M. Printista J.G. Zanabria, F. Piccoli. *Hypercubic Comunications in MPI*. Tesis submited for UNSL, 2005.

[8] Danelutto M. Aldinucci M. *Algorithmic skeletons meeting grids*. Parallel Computing Volume 32 , Issue 7 , Elsevier Science B. V. publisher Amsterdam, The Netherlands, 2006.

[9] Danelutto M. Aldinucci M. Dazzi P. *Muskel: an expandable skeleton environment*. 2007.

[10] B. Bacci M. Danelutto S. Pelagatti and M. Vanneschi. *SkIE: A Heterogeneous Environment for HPC Applications.* ., 1999.

[11] Danelutto M. Vanneschi M. Bacci B. Pelagatti S. Orlando S. *P3L: A structured high level programming language and its structured support.* Concurrency: Practice and Experience, 7(3):225-255, May 1995., 1995.