Técnicas estándares para paralelismo anidado de datos

M. Fabiana Piccoli, A. Marcela Printista[1]

Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950
5700 - San Luis
Argentina
e-mail: \{mpiccoli, mprinti\}@unsl.edu.ar

Resumen

Desarrollar programas paralelos no es una tarea simple [Gar95]. Dependiendo del problema a paralelizar, distintos tipos de paralelismo pueden ser considerados. Los tipos de paralelismo mas comunes son el paralelismo de Datos y el paralelismo de Tareas o de Control [Akl89] [Qui94] [Fos94]. Cada uno de ellos tiene sus ventajas. El paralelismo de Datos es uno de los esfuerzos más exitosos para introducir paralelismo explícito en lenguaje de alto nivel. Este tipo de paralelismo es conveniente particularmente por dos razones: es fácil de programar y en problemas de gran tamaño escala muy bien. Si bien varias implementaciones de lenguajes de datos paralelos existen, la mayoría se centra en la explotación del paralelismo de datos no estructurado, es decir en el paralelismo chato. Por su parte el paralelismo de Tareas es más eficiente y adecuado en problemas con estructura de datos irregulares.
Combinar ambos tipos de paralelismo implica tener un lenguaje con la capacidad de realizar invocaciones paralelas anidadas. Esta capacidad implica la combinación, por un lado de la facilidad de programación sobre un modelo de datos paralelo y por el otro de la eficiencia del modelo de paralelismo de control en la ejecución sobre estructuras de datos irregulares. Este tipo de paralelismo se llama Paralelismo Anidado de Datos [Ble94] [Ble96] y se lo puede considerar una extensión del paralelismo de datos estándar ya que se le suma a éste la capacidad de anidar invocaciones paralelas. Dicho paralelismo, entonces, combina la habilidad de aplicar una función en paralelo sobre cada elemento de una colección de datos y la habilidad de anidar tales llamadas recursivas.

La línea de investigación está centrada en la determinación de metodologías o técnicas para desarrollar programas paralelos que apliquen paralelismo anidado de datos en algún lenguaje paralelo [Gon00a] [Gon00b]. Para lograr establecer una metodología adecuada, es preciso considerar en primer lugar el modelo de computación paralela que se aplica y en segundo lugar el lenguaje paralelo utilizado.

Existen varios modelos de computación paralela, el mas expandido es el modelo *PRAM* [Har94]. Como no existe un modelo de computación paralela universalmente aceptado como es el caso de la programación secuencial, *PRAM* es el modelo de computación paralela más aceptado. El modelo *PRAM* es un modelo abstracto que proyectado sobre una máquina paralela real muestra una pérdida de rendimiento debido a que ignora la jerarquía de memoria de las máquinas paralelas actuales. Su atractivo consiste en el poderoso mecanismo de comunicación que representa la memoria compartida, el sincronismo del modelo y la capacidad de expresar paralelismo anidado de datos.

Considerando a *PRAM* como el modelo de computación paralela, se analizaron diferentes lenguajes orientados específicamente a dicho modelo como son los lenguajes *ll* [Leo95][Leo96] [San96] y *fork95* [For] [Kes95a] [Kes95b] [Kes97a] para máquinas de memoria compartida y se consideraron otros como son *llc y High Performance Fortran (HPF)* [Hig96] [Bra98] [Bra99] para máquinas de memoria distribuida.

El lenguaje *ll* fue especialmente diseñado para el modelo *PRAM*. Este lenguaje asume la existencia de una memoria compartida por todos los procesadores, considerando lecturas y escrituras en la memoria compartida en lugar del pasaje explícito de mensajes. En el análisis realizado al modelo y al lenguaje se puso especial énfasis en la posibilidad de expresar paralelismo anidado de datos y como consecuencia de ello a los conceptos de asignación y activación de procesadores, considerando una propuesta de implementación eficiente para las sentencias que los implementen.

Otro de los lenguajes estudiados es *fork95*. Este lenguaje surge como consecuencia del desarrollo de SB-*PRAM*[Bac97], un proyecto que pretende la construcción de una máquina *PRAM* a fin de revertir una de las acusaciones más serias contra el modelo *PRAM*, la falta de realismo con respecto a las arquitecturas existentes.

[1] Group supported by the UNSL and ANPCYT (Agencia Nacional para la Promoción de la Ciencia y Tecnología)

Fork95 es un lenguaje paralelo imperativo fuertemente sincrónico, posee sentencias que permiten la programación en modo sincrónico y asincrónico. Es el heredero de *FORK* [Hag92] y además de heredar de éste características tales como el concepto de grupo de procesadores y la diferencia entre espacios de memoria compartidos y privados; incluye otras tales como los tipos de datos puntero y heap. Entre los conceptos que se muestran, se puntualiza la asignación de procesadores.

La existencia de máquinas paralelas reales con memoria distribuida hizo que se analice la portabilidad de los programas *PRAM* a dichas arquitecturas. Ante esta realidad surge el modelo de *Computación Colectiva* [San98] [Gon99b] [Gon00c]. Dicho modelo permite y propone una metodología para la traslación eficiente de algoritmos con paralelismo anidado de datos sobre arquitecturas reales. Para arribar a ello se incorporan los conceptos de hipercubo dinámicos y las relaciones de sociedad entre procesadores de diferentes grupos los cuales determinan las comunicaciones. *Computación Colectiva* no sólo propone una técnica para el desarrollo de algoritmos con paralelismo de datos anidado, sino también se lo puede extender a un modelo de predicción de la performance.
Considerando la metodología propuesta por *Computación Colectiva,* el lenguaje *llc* es la implementación concreta de ella por ejemplo posee funciones de división y de comunicación. Las funciones de División propuesta demuestran ser más eficientes que aquellas definidas en las herramientas estándares como es *MPI* [Sni96].

Otro de los lenguajes considerados es el lenguaje *HPF*, el cual tiene las capacidades de desarrollar algoritmos paralelos portables y de explotar el paralelismo con la mínima intervención del programador. Aprovechando las características de dicho lenguaje y aplicando la metodología propuesta por *Computación Colectiva*, se desarrollaron problemas aplicando paralelismo anidado de datos.

Conclusiones

La metodología propuesta para la implementación eficiente de sentencias anidadas de asignación de procesadores, sincronización y movimiento de datos entre procesadores es adecuada para la traducción de algoritmos *PRAM* a multicomputadoras. Los algoritmos paralelos pueden diseñarse para el modelo *PRAM*, expresarse en un lenguaje orientado a él y luego trasladarse a *llc* o *HPF* sin ninguna consecuencia. Más aún, los resultados computacionales muestran que aplicando la metodología propuesta por *Computación Colectiva* para desarrollar programas con paralelismo anidado de datos en lenguajes como *llc* o *HPF*, estos muestran una eficiencia aceptable y comparable. Se puede entonces decir que *Computación Colectiva* es un compromiso entre el modelo *PRAM* y los estándares actuales de paralelismo.

Bibliografía

[Akl89] Akl, S. G. *The Design and Analysis of Parallel Algorithms.* Prentice-Hall. 1989.

[Bac97] Bach, P., Braun, M., Formella, A., Friedrich, J., Grün, Th., Lichtenau, C. *Building the 4 Processors SB-PRAM Prototype.* Proc. of the Hawaii 30th International Symposium on System Science HICSS-30, 5. pp 14-23. 1997.

[Ble94] Blelloch, G. E., Hardwick J., Sipelstein J, Zagha M and Chatterjee S., *Implementation of a Portable Nested Data-Parallel Language*. Journal of Parallel and Distributed Computing 21. pp. 4-14. 1994.

[Ble96] Blelloch G. - *Programming Parallel Algorithms*. Communications of ACM. Vol. 39, Nº 3. March 1996.

[Bra98] Brandes, T.: ADAPTOR Programmer's Guide (Version 6.0). Technical Documentation, GMD, http://www.gmd.de/SCAI/lab/adaptor/adaptor_home.html. 1998.

[Bra99] Brandes, T.: Exploiting Advanced Task Parallelism in High Performance Fortran via a Task Library . Proceedings of Euro-Par'99 Parallel Processing, Toulouse, France. 1999.

[For] *The PRAM Programming Language Fork 95*. http://www.informatik.unitrier.de/~kessler/fork95/.

[Fos94] Foster I., *Designing and Building Parallel Programs*. Addison-Wesley. 1994.

[Gar95] García F., *Programación en Paralelo y Técnicas Algorítmicas.* Ph Degree. D.E.I.O.C., La Laguna University. 1995.

[Gon00a] Gonzalez, J.A., Leon, C., Piccoli, M.F., Printista, M, Roda, J.L., Rodriguez, C., Sande, F. *Toward Standard Nested Parallelism*. 7mo Euro PVM/MPI Users´Groups Meeting: Recent Advances in Parallel Virtual Machine and Message Passing Interface. Springer. Balatonfured, Hungria. Pp. 96-103. Septiembre 2000.

[Gon00b] González J., León C., Piccoli F., Printista M., Roda J., Rodríguez C., Sande F. *Supporting Nested Parallelism*. VI Congreso Argentino de Ciencias de la Computación en la Universidad de Tierra de Fuego. Octubre de 2000. Usuhaia. Argentina. En prensa.

[Gon00d] Gonzalez, J.A., Leon, C., Piccoli, M.F., Printista, M, Roda, J.L., Rodriguez, C., Sande, F. *The Collective Computing Model*. Journal of Computer Science and Technology, Special Issue : Concurrent, Parallel and Distributed Processing, PGNAUT, ISTEC. Argentina. Octubre 2000. http ://journal.info.unlp.edu.ar/~journal/HOME.HTML

[Gon99b] Gonzalez, J.A., Leon, C., Piccoli, M.F., Printista, M, Roda, J.L., Rodriguez, C., Sande, F. *Collective Computing*. V Congreso Argentino de Ciencias de la Computación en la Universidad del Centro. Tandil, Argentina. Octubre 1999.

[Hag92] Hagerup, T. Schmidt and Seidl, H., *FORK: A High-Level Language for PRAMs*, Future Generation Computer Systems 8. pp.379-393. 1992.

[Har94] Harris, T.J., *A Survey of PRAM Simulation Techniques*. ACM Computing Surveys, Vol. 26, No.2. pp. 187-206. June 1994.

[Hig96] High Performance Fortran Forum - *High Performance Fortran Language Specification version 2.0.* URL www.crpc.rice.edu/HPFF/hpf2/. 1996

[Kes95a] Kessler, C., Seidl, H. *Fork95 Language and Compiler for SB-PRAM.* Proc. 8th Workshop on Compilers for Parallel Computers. Pp 28-40. 1995.

[Kes95b] Kessler, C., Seidl, H. *Integrating Synchronous and Asynchronous Paradigms: The Fork95 Parallel Programming Language*. Proceedings of MPPM-95 Conference on Massively Parallel Programming Models. IEEE CS Press. Berlin. 1995.

[Kes97a] Kessler, C., Seidl, H. T*he Fork95 Parallel Programming Language: Design, Implementation, Application.* Int. Journal of Parallel Programming. Vol 25, Nº1. Pp 17-50. 1997.

[Leo95] León, C., Sande, F, Rodríguez, C., García, F. - *A PRAM Oriented Programming*. Proc. Of the Euromicro Workshop on Parallel and Distributed Processing. IEEE CS Press. Pp 182-191.1995.

[Leo96] León C., *Diseño e Implemantación de Lenguajes Orientados al Modelo PRAM*. Tesis Doctoral. D.E.I.O.C., La Laguna University, 1996.

[Qui94] Quinn M. - *Parallel Computing. Theory and Practice*. Second Edition. McGraw-Hill, Inc. 1994.

[San96] Sande, F., Garcia, F. León, C., Rodríguez, C. - *The ll Parallel Programming Systems.* IEEE Transactions on Education, Vol. 39, Nº4. Pp 457-464. 1996.

[San98] Sande, F. - *El Modelo de Computación Colectiva: Una Metodología Eficiente para la Ampliación del Modelo de Librería de Paso de Mensajes con Paralelismo de Datos Anidado*. PhD thesis. Universidad de La Laguna, 1998.

[Sni96] Snir, M., Otto, S., Huss-Lederman, S., Walker, D., Dongarra, J. - *MPI: The complete Reference*. Cambridge, MA: MIT Press, 1996.