UN ENTORNO AVANZADO PARA LA SIMULACION DE MULTIPROCESADORES

José M. García
Dpto. de Informática
Universidad de Castilla-La Mancha
02071-ALBACETE (Spain)
utjmgarcia@02ccv1.ucma.es

Resumen

En este artículo presentamos un entorno de programación para multiprocesadores. Este entorno nos permite el desarrollo de programas para un sistema paralelo MIMD con memoria compartida, evaluando el rendimiento que se obtiene sobre la versión secuencial de dicho programa.

Nuestro entorno, desarrollado para máquinas del tipo PC's y estaciones de trabajo, permite la simulación del comportamiento de un multiprocesador. Para programar las diferentes aplicaciones se usa el lenguaje ShaPas, que hemos desarrollado, el cual está basado en el Pascal standard, al que se le han añadido algunas extensiones y modificaciones para permitir una fácil y elegante programación de algoritmos paralelos.

1. INTRODUCCION.

La necesidad de una mayor capacidad de procesamiento en los ordenadores ha conducido al desarrollo de varias arquitecturas paralelas, usualmente del tipo SIMD o MIMD de acuerdo a la clasificación propuesta por Flynn. Los ordenadores del tipo MIMD pueden ser clasificados como máquinas de memoria compartida -multiprocesadores- o de memoria distribuida -multicomputadores-. Aunque los primeros son mas costosos de construir, tienen sin embargo la ventaja de que son mas sencillos de programar.

Desde hace unos años, en el Departamento de Informática de la Universidad de Castilla-La Mancha, hay un grupo de personas que está trabajando en el procesamiento paralelo. Estas investigaciones se han dirigido

al objetivo de paralelizar diversos algoritmos numéricos secuenciales para los diversos modelos de arquitecturas paralelas que en la actualidad nos encontramos. Para conseguir esto, se han desarrollado (y se están desarrollando) diversos entornos de programación que permitan simular el comportamiento de una máquina paralela. Dichos entornos se desarrollan para PC's o estaciones de trabajo que son las máquinas que actualmente se tienen en el Departamento.

Dicha línea de trabajo es común a la seguida en la mayoría de las Universidades y centros de investigación, ya que de esta forma se puede aprender *cómodamente* a programar una máquina paralela, de tal forma que cuando el programador se enfrente con la máquina real tenga una mayor probabilidad de que su aplicación esté bien programada. Ello es debido a que el arte de programar en paralelo es muy diferente de la programación secuencial [1]. Por ello, en los últimos años ha proliferado el uso y desarrollo de entornos y herramientas que hagan mas fácil la tarea de programar en paralelo [2].

Tras habernos centrado durante unos años en los multicomputadores [3,4] debido a su mejor ratio coste/prestaciones, nos estamos fijando últimamente en los multiprocesadores debido a su mayor facilidad en la programación.

En este artículo se presenta el entorno de programación que se ha desarrollado para los multiprocesadores. En dicho entorno un algoritmo paralelo se programa en el lenguaje ShaPas (acrónimo de *Shared Pascal*), un nuevo lenguaje de programación que hemos desarrollado para multiprocesadores basado en el Pascal standard.

2. EL ShaPas.

Muchas discusiones estan habiendo sobre el modelo mas adecuado para programar en paralelo. Por una parte, una escuela opina que lo mejor es dejar esta labor al compilador, ayudado quizás por unas directivas que añada el programador para facilitar esta tarea al compilador. Esta línea de investigación se denomina **paralelización automática**. Por otra parte, y debido a que los resultados que se obtienen no son muy buenos, hay otra escuela que trabaja en desarrollar lenguajes de programación paralelos. Lo habitual es tomar como base un lenguaje secuencial y añadirle una serie de extensiones que permitan el cómodo manejo del paralelismo, aunque también se han desarrollado totalmente nuevos lenguajes de programación, tales como Occam.

En nuestro caso hemos tomado como base el Pascal. Por una parte es un lenguaje muy educativo elegido en muchas universidades para la enseñanza de la programación. Por ello, nos parece que el ShaPas puede ser un lenguaje adecuado para la enseñanza de la programación paralela. Por otra parte, tiene la suficiente potencia y versatilidad como para permitir abordar un gran número de problemas complejos del tipo de algoritmos numéricos.

La concurrencia se permite a nivel de procedimientos y se manifiesta en el programa mediante el empleo de las palabras reservadas **multitask** y **endmultitask**, que comienzan y finalizan los procedimientos que van a ser ejecutados en paralelo. Cuando el compilador encuentra la palabra reservada **endmultitask** suspende la ejecución del programa principal (o procedimiento) y da paso a la ejecución de los procedimientos que se encontraban a partir de la palabra reservada **multitask**. La ejecución en paralelo finaliza cuando lo hacen todos los procedimientos, devolviendose el control al programa principal.

El modelo de programación elegido no se ha restringido al SPMD (único código y multiples datos), pues se permite que en cada procesador se ejecute un proceso diferente al del resto de procesadores. Es decir, no solo permitimos un estilo de programación de paralelismo en los datos sino también de paralelismo en el flujo de control. Asimismo, permitimos que se trabaje con la idea de una zona de memoria privada, ya que un procedimiento puede tener variables locales que podrían no ser accesibles por el resto de procesos (a nivel lógico, ya que a nivel físico todas las variables se localizan en la memoria común de la máquina y por tanto direccionables por cualquier proceso).

Para la sincronización y comunicación de procesos se utilizan las variables compartidas, gracias a la memoria común que posee la máquina. Para el control de los diferentes procesos a una variable común se ha elegido la técnica de los **semáforos**, y se han implementado dos primitivas: wait y signal. Para darle mayor generalidad se han permitido semáforos de orden *n*, y no tan sólo semáforos binarios.

Para la adecuada programación en ShaPas se ha desarrollado un entorno de programación. Dicho entorno, que presenta un interfaz de tipo *amigable*, presenta las características comunes a este tipo de entornos: manejo de ficheros, edición integrada de forma cómoda, compilación del programa con informe de línea y columna del error si lo hay (tanto sintáctico como semántico) y ejecución (de forma simulada) de un programa correcto.

CONCLUSIONES

En este trabajo se presenta un nuevo lenguaje de programación paralelo para multiprocesadores, el ShaPas. Asimismo, se presenta el entorno de programación desarrollado para permitir una cómoda y ágil programación de aplicaciones en paralelo.

Como líneas futuras de trabajo, se quiere completar el entorno con la posibilidad de variar diversos parámetros de un multiprocesador, tal como la red de interconexión, el tamaño y número de las memorias *caché*, el número de procesadores reales, etc. De esta forma, ademas de tener una herramienta que permita la programación de algoritmos en paralelo se quiere evaluar el rendimiento de la máquina.

REFERENCIAS

[1] KARP, A.- Programming for Parallelism, *IEEE Computer*, 20, May 43-57, (1987).

[2] SEGALL, Z. and RUDOLPH, L.- PIE: A Programming and Instrumentation Environment for Parallel Processing, *IEEE Software*, Nov. 22-37, (1985).

[3] GARCIA, J.M.- A new language for multicomputer programming, *SIGPLAN Noticies*, Vol. 27, No. 6, 47-53, (1992).

[4] GARCIA, J.M. and DUATO, J.- An advanced environment for programming transputer networks with dynamic reconfiguration, *Proc. Int. Conf. on Parallel Computing and Transputers Applications '92*, pp. 601-610, Barcelona, September (1992).