CONSTRUCCION DE UN MULTIPROCESADOR QUE MANEJA LISP
PARTE I: TEORIA DE OPERACION DE LA MAQUINA AHR

Adolfo Guzmán
Luis Lyons
Luis Hugo Peñarrieta
David Rosenblueth
Raul Gómez
Manuel Correa
Dora Gómez
Norma A. de Rosenblueth
Juan López
Isauro Morales
Pablo Martínez

Departmento de Sistemas de Cómputo, IIMAS
Universidad Nacional Autónoma de México

Sumario

Se ha diseñado y construido una computadora de procesamiento en paralelo, de propósitos generales y que usa Lisp como su lenguaje principal. Este diseño surge de la idea de cómo conglomerar varios procesadores iguales (Z-80's) para evaluar o ejecutar simultáneamente un solo programa (escrito en Lisp). La arquitectura es singular porque ella misma es la responsable (y no el programador o el usuario) de la repartición de trabajo entre los procesadores, y de su sincronización.

El lenguaje de programación es Lisp puro. No se necesita especificar orden alguna sobre su ejecución en paralelo; no se necesitan primitivas del lenguaje para sincronizar los diferentes trabajos, ni para administrar recursos. El usuario puede ignorar que varias porciones de su programa se están ejecutando simultáneamente.

Esta computadora, denominada Máquina AHR, muestra que es posible diseñar máquinas que procesen en paralelo, y que sean programadas de tal suerte que el programador no tenga el trabajo de especificar cómo debe distribuirse tal programa entre los varios procesadores.

Otro hallazgo importante es que se derivan grandes beneficios en la construcción de sistemas donde se diseñen tanto las partes de programación como las de hardware, ya que las posibilidades de diseño son más amplias y la construcción se lleva a cabo en un plazo razonable.

Reconocimiento

La computadora AHR ha sido obra de todos los integrantes del Proyecto AHR.

El trabajo aquí descrito ha sido patrocinada parcialmente por el Consejo Nacional de Ciencia y Tecnología (Proyecto 1632; PCAINAL 790250), México.

Resumen Ejecutivo

El Proyecto AHR (Arquitecturas Heterárquicas Reconfigurables) se inició en el IIMAS-UNAM en enero de 1979; su meta es el diseño y construcción de una computadora de propósito general, que procese un lenguaje de alto nivel.

La computadora AHR está formada por varias microcomputadoras (de 1 a 64) que procesan Lisp, y evalúan asíncronamente el o los programas que residen en la parrilla, en tanto que un procesador de entrada y salida relaciona el trabajo de la máquina con el usuario.

El lenguaje que se usa es Lisp puro, sin asignación, sin transferencias de control ni etiquetas, sin operadores y con funciones sin efectos laterales. Desde luego, se permite la recursión. Los argumentos de las funciones se evalúan en paralelo.

Los programas se ejecutan en paralelo, pero el usuario no da ninguna orden especial para que esto suceda, sino que la ejecución de las diferentes partes del programa va siendo realizada conforme se van obteniendo los datos que se necesitan para su ejecución.

El procesador de entrada y salida interactúa con el usuario, haciendo la edición de programas, la carga y el arranque de los mismos. A diferencia de las máquinas convencionales, la máquina AHR no necesita de un sistema operativo tradicional, "programado" en lenguaje de máquina. sino que los mismos circuitos realizan funciones de administración.

Estado del Proyecto

La construcción de la máquina AHR está a punto de terminarse.

Originalidad y relevancia del Proyecto

Es la primer máquina que se construye que procesa Lisp en paralelo, ya sea comercialmente o con fines de investigación. Es un trabajo original en el campo de la Computación. Además, la máquina trata de demostrar y evaluar el poderío del concepto de heterarquía para procesar un mismo trabajo simultáneamente por varios procesadores, y nos enseña (a través de un ejemplo operacional) cómo se resuelven los problemas de sincronización entre procesos, deadlocks, accesos a memorias comunes, colas de trabajos y otros comúnmente asociados a la aglutinación de procesadores que laboran en una tarea común.

Es una implementación en circuitería de un sistema de cómputo idónea para evaluar lenguajes (como Lisp puro) que emplean el cálculo lamda.

Importancia Económica

Como la capacidad de cómputo presente en microprocesadores es mucho más económica que la presente en máquinas grandes (es decir, una suma de dos números de 16 bits sale más barata en una máquina chica que en una grande), resulta importante averiguar (como lo ha hecho el Proyecto AHR) cómo construir una máquina grande a partir de varias pequeñas.

Las arquitecturas hechas con una pluralidad de microprocesadores ofrecen ventajas en cuanto a modularidad, crecimiento incremental, expansión y confiabilidad.

Importancia de su Desarrollo

Parece adecuado saber cómo obtener capacidades considerables de cómputo mediante la aglutinación de computadoras (microcomputadoras) con capacidad reducida. Entonces será posible fabricar (si esto fuere deseable) máquinas pequeñas y con ellas ensamblar otras de tamaño y poderío arbitrario.

Conclusiones Principales

Se conoce ahora cómo ejecutar en paralelo un programa escrito en Lisp, y en esta forma acelerar su ejecución. El usuario no necesita dar ninguna orden especial para que varias porciones de un programa se ejecuten simultáneamente. El programa es idéntico al que se usa en computadoras de un solo procesador.

Se creó, desarrolló y se encontró útil y aparentemente poderoso el concepto de heterarquía, mediante el cual n procesadores colaboran sin intercambios explícitos de información.

Se relegó la interacción con el usuario a un procesador dedicado únicamente para este propósito.

Trabajos Futuros

El Proyecto AHR continuará estudiando la forma de expresar algoritmos en paralelo, que exploten el uso de varias máquinas para un solo trabajo. Es importante saber cómo dar órdenes "sencillas" pero poderosas a un ejército de procesadores. Es también importante diseñar otras arquitecturas donde los procesadores sean lo suficientemente inteligentes para que ellos se encarguen de la sincronización y acceso común a resultados parciales, bajo los lineamientos de los trabajos desarrollados por el Proyecto AHR.

Introducción

Se presentan los principios de diseño y características arquitectónicas principales de la máquina AHR. El sistema está enfocado hacia investigación y enseñanza en Arquitectura de Computadoras, Procesamiento en Paralelo y Sistemas Distribuidos.

Visto por el usuario, AHR es un sistema que usa sólo un lenguaje, conciso y convencional (Lisp).

Internamente, es un sistema de multiprocesadores de propósito general, fuertemente acoplados, sin jerarquía entre ellos.

Cada procesador evalúa una parte diferente del programa, por lo que se obtiene ejecución paralela asíncrona.

La máquina AHR está clasificada como una de instrucciones múltiples y datos múltiples (multiple-instruction multiple-data), y se parece a las máquinas de flujo de datos (data-flow machines).

Qué se desea demostrar

Qué se encontró

* Se desea encontrar la forma de poder atacar en paralelo un solo programa de cómputo. Es decir, que diferentes procesadores ejecuten diferentes partes del programa.

> La máquina AHR propuesta permite que muchos procesadores ejecuten en paralelo uno o varios programas.

* Es importante que el algoritmo (el programa) escrito por el usuario no contenga un número elevado de instrucciones dedicadas a la explicación o comando del paralelismo. También es importante que estos comandos no sean complejos.

Es decir, el programador (el usuario) no debe preocuparse mucho por la ejecución en paralelo, por el número de procesadores, o por su sincronización.

> El usuario no da ninguna orden para manejar el paralelismo. El no percibe ("no sabe") que su programa se está ejecutando en paralelo.

* Deseamos generar una arquitectura de computadora donde la sincronización entre las diferentes partes, la compartición de datos, su actualización oportuna, etc., sea hecha por el sistema, en el sentido del párrafo anterior: el usuario no se preocupa por estas cosas, las que suceden "naturalmente". Esta naturalidad debe estar expresada por reglas simples.

> El usuario no da orden alguna para sincronización.
>
> El sólo obedece las reglas del cálculo lambda: para evaluar una función, hay que tener ya evaluados sus argumentos.
>
> El usuario no se preocupa del transcurso del tiempo. Los postulados no tienen orden de ejecución; el programa no es serial (v.gr.: Fortran) sino aplicativo.

La máquina AHR

Se escogió la integración de varias microcomputadoras, por su bajo precio, para desarrollar nuestro trabajo. Se escogió una máquina de 8 bits: la Z80.

Para simplificar la escritura de los programas, la máquina AHR trabaja como periférico de otra computadora de propósito general. Esta interacciona con los usuarios, permite edición de textos, generación de programas y mantenimiento de archivos en disco. Véase Figura "La máquina AHR como periférico de una máquina convencional". Es decir, la máquina AHR posee dos tipos de procesadores: los de Lisp, que son varios y trabajan en paralelo, y el de E/S, que es único.

La máquina AHR y la de propósito general (llamada "procesador de entrada y salida") comparten memoria común, que utiliza el cargador para meterle los programas y datos a la AHR.

Después de ejecutar, la AHR deja asimismo los resultados en su memoria, de donde serán sacados al impresor o teletipo por el sistema operativo convencional del procesador de entrada/salida.

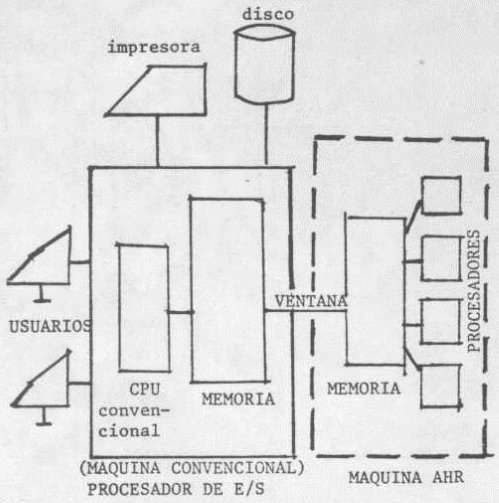

FIGURA "LA MAQUINA AHR COMO PERIFERICO DE UNA MAQUINA CONVENCIONAL"

La máquina AHR y su procesador de entrada y salida (una microcomputadora convencional) se comunican a través de una ventana en sus memorias. De esta forma, la máquina AHR no realiza entrada y salida, ni atiende a usuarios. Nótese la reducción en la programación que esto implica.

El lenguaje

El requerimiento de no pedirle al usuario que especifique el orden de evaluación y la sincronización entre procesadores nos llevó a usar un lenguaje aplicativo, y se escogió Lisp. Un Lisp puro, sin transferencias de control (GOTO), etiquetas, operadores (RPLACA) ni postulados de asignación (SETQ'S).

Las características principales de Lisp son:

* Maneja átomos (elementos o identificadores que no poseen un valor numérico) y listas (árboles o estructuras donde es fácil crear nuevos árboles con elementos en medio).

* Es recursivo, en el sentido de que al definir una función (subrutina), es válido usar lo definido en la definición misma.

* Los argumentos de una función pueden evaluarse en cualquier orden. El requisito es que para poder evaluar la función, ya deben estar evaluados los argumentos.

Las memorias

Para aumentar la rapidez de acceso, se han usado tres memorias independientes, de manera que hasta tres procesadores distintos pueden tener acceso al sistema de memoria, si cada uno accesa una distinta.

Características de la máquina

*COMPUTACION EN PARALELO: La habilidad del artefacto para procesar simultáneamente varias corrientes de instrucciones.

*AGLUTINACION DE VARIAS MAQUINAS PEQUEÑAS EN UNA GRANDE: Diseño basado en nuevos procesadores y circuitería "pequeña", modular.

*LENGUAJE PRINCIPAL DE ALTO NIVEL: El artefacto podrá procesar Lisp como lenguaje principal, sin compilador o intérprete.

*RECONFIGURABILIDAD: Se podrán cambiar automáticamente las funciones de los microprocesadores constituyentes, para que efectúen trabajos distintos de los que inicialmente les fueron asignados, en respuesta dinámica (reconfiguración) a las variaciones de la demanda de proceso, y de los tipos de proceso requeridos.

El sistema es adaptivo a las demandas de los programas que se están ejecutando.

Nota: Las versiones 0 y 1 de la máquina AHR no poseen reconfigurabilidad. El diseño original (AHR-76-1) sí la contempla.

*EXPANDIBILIDAD. El sistema es modularmente expandible. El sistema exhibe una degradación progresiva (graceful degradation) durante las fallas de sus componentes activos.

*EJECUCION SIMULTANEA EN DIFERENTES LENGUAJES. Mediante la codificación de un intérprete de Algol en lenguaje de máquina Z80, varios programas, unos en Lisp y otros en Algol, por ejemplo, podrían estarse ejecutando simultáneamente en una máquina AHR, de haber las suficientes cajas activas.

Nota: Las versiones 0 y 1 de la máquina AHR sólo interpretan Lisp. Otros lenguajes, de ser codificados posteriormente en lenguaje Z80, podrán ejecutarse simultáneamente con Lisp.

*HETERARQUIA. Los procesadores que intervienen en la ejecución de un programa no poseen un orden o jerarquía o subordinación entre ellos.

Teoría de Operación de la Maquina

El usuario, a través de un teletipo o pantalla, usa el procesador de entrada y salida para formar y editar su programa, el que finalmente residirá en disco. Ver figura "La máquina AHR como periférico de una máquina convencional".

Cuando un usuario desea ejecutar su programa, lo baja del disco y lo carga en la memoria pasiva de la máquina AHR, usando el programa cargador, que se ejecuta en el procesador de entrada y salida. Ver figura "Funcionamiento de la máquina AHR". Al dar él el comando de run, el procesador de e/s da una señal a la máquina AHR, que fuerza la ejecución de un nodo inicial (luego se verá en detalle el proceso de ejecución) por algún procesador de Lisp. Como resultado de esta ejecución, más nodos se irán copiando (por los procesadores de Lisp) de la memoria pasiva (donde el programa reside) a la parrilla (donde los programas se evalúan). Finalmente, la ejecución concluye y en la parrilla habrá un apuntador al resultado final, que estará depositado en memoria pasiva. Mientras esto sucede, se usa la memoria de variables para tener actualizados los valores de las variables. El distribuidor administra el tráfico de nodos de la parrilla a los procesadores de Lisp y el FIFO funge como un pizarrón donde se guarda la cola de trabajos pendientes: cola de apuntadores a nodos en la parrilla listos para ser evaluados. El FIFO estará vacío, indicando que ya no hay trabajos pendientes en la parrilla. Entonces el distribuidor interrumpe al procesador de entrada y salida, y le indica (mediante el FIFO-FIFO de memoria pasiva - no mostrado en la figura) dónde está la lista o átomo a imprimir como resultado. El procesador de e/s procede entonces a accesar la memoria pasiva y a imprimir el objeto-dato resultado por el teletipo del usuario correspondiente. Esta impresión puede traslaparse con la ejecución de otro(s) programa(s) en Lisp.

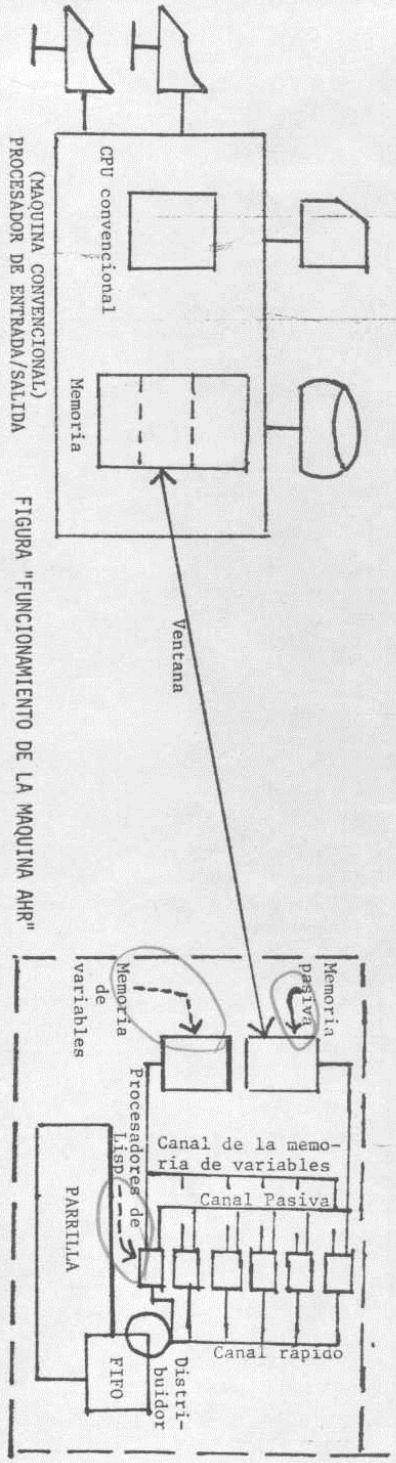

FIGURA "FUNCIONAMIENTO DE LA MAQUINA AHR"

El distribuidor toma nodos listos para ser evaluados (los que halla a través de apuntadores que yacen en el FIFO) y los lanza por el canal rápido, donde algún procesador de Lisp los toma. Lo toma aquél que haya concluido su trabajo previo. Si hay varios procesadores en esta situación, un árbitro que no se muestra escoge un procesador. El procesador de Lisp escogido toma el nuevo nodo, entrega al distribuidor el resultado de la operación que acaba de concluir y se pone a ejecutar la función primitiva que el nuevo nodo contiene. De manera que el distribuidor puede verse como un circuito que entrega trabajo por hacer a los procesadores de Lisp, y recoge de ellos trabajo ya procesado. Ambos tipos de trabajo van a la parrilla.

Cómo se ejecuta un programa que ya está en memoria pasiva

Le ejecución de un programa comienza cuando, después de ser copiado de disco a la memoria pasiva por el cargador de la máquina convencional, ésta interrumpe a la máquina AHR, dándole asimismo el apuntador a la dirección del FIFO-FIFO.

El FIFO-FIFO nos indica la dirección en memoria pasiva donde está un apuntador a la expresión cuya ejecución se demanda.

Por ejemplo, después de haber cargado la definición de la función "factorial" en memoria pasiva, el usuario escribe (FACTORIAL 4) indicando que quiere aplicar la función factorial al argumento 4.

El apuntador a "FACTORIAL" apunta a ese átomo, y en su lista de propiedades aparece, entre otras cosas, su definición como función, la que ya previamente cargó el usuario.

La máquina AHR responde mediante su distribuidor, que fuerza un nodo EVAL con esa dirección como argumento.

Este nodo es expelido por el distribuidor a través de su canal rápido a alguno de los procesadores de Lisp. Uno de ellos lo recibirá y lo comienza a evaluar.

(Mientras tanto, el distribuidor y todo el resto de la máquina AHR están listos para evaluar otros programas que vengan del procesador de e/s).

La evaluación del procesador que tomó el nodo EVAL originará el copiado de la función factorial de memoria pasiva a parrilla (más detalles en la Comunicación AHR-79-2). Parte de ese copiado lo realiza el procesador que tomó el nodo EVAL, y parte es diferido a otros procesadores. Veamos esto en detalle.

El procesador de Lisp que tiene a su cargo la ejecución del nodo EVAL consulta la definición de factorial, observa que es una lambda (lambda (n) ...) y regresa como resultado de su observación.

((lambda (n) (if (eq n 0)) ...) 4).

Es decir, pide que se escriba eso en la parrilla.

¿Cómo se regresa este resultado? El procesador de Lisp que tiene consigo el nodo EVAL, cuando ya sabe qué debe poner en la parrilla, interrumpe al distribuidor y le pide "intercambio": el procesador de Lisp levanta la mano señalando: "listo para aceptar nuevo trabajo y para que me recojan mi resultado". El distribuidor le recoge su resultado, metiéndolo a la parrilla (¿dónde? el resultado lleva indicadores a este efecto) e insertándole un nuevo nodo al procesador de Lisp, nodo que es nuevo trabajo a efectuar. Probablemente ahora no haya nuevo trabajo a efectuar, si es que no han llegado más tareas del procesador de entrada y salida.

Al ir a depositar este "resultado", que es aún parcial, ya que le falta procesarse, a la parrilla, el distribuidor toma nota de los nodos que ya están listos para ser evaluados (en realidad, quien "toma nota" es el procesador de Lisp, pero aquí se simplifica la explicación), y mete al FIFO apuntadores a estos nodos. Y deposita el resultado, el cual contiene esos nodos listos para ser evaluados, en parrilla. En seguida el distribuidor distribuye trabajo nuevamente, sacándolo del FIFO y expeliéndolo hacia algún procesador de Lisp.

Mientras cada nodo que va al procesador de Lisp sea convertido en un resultado con uno o más nodos (en promedio) listos para ser evaluados, el proceso continúa, el árbol de evaluación en la parrilla se expande, porque regresan a ella más nodos listos para la evaluación que los que salieron, el contenido del FIFO crece y la evaluación prosigue.

Cuando cada nodo procesado por algún procesador de Lisp regrese resultados finales (constantes: átomos o listas), es decir sin contener nodos listos para ser evaluados, el FIFO se acorta, el árbol de evaluación en parrilla se encoge y la evaluación se acerca a su fin.

Eventualmente, el FIFO está vacío y todos los procesadores de Lisp han terminado: están pidiendo más trabajo. La evaluación ha concluido.

Conclusiones y Bibliografía

Estas se encuentran al final del Artículo "Construcción de un Multiprocesador que Maneja Lisp. Parte II: Arquitectura de la Máquina AHR" (en estas memorias), el que es continuación de éste.