Evaluación de un Servidor Paralelo de Imágenes usando una Aplicación de Segmentación[1]

C. León, M. Rukoz, R. Surós

Centro de Computación Paralela y Distribuida, Universidad Central de Venezuela.
Apdo. 47002, 1041-A, Los Chaguaramos, Caracas, Venezuela.
Telf. 58 2 6052134 Fax: 58 2 6052131
E-mail: cleon@flynn.ciens.ucv.ve, mrukoz@reacciun.ve, rsuros@reacciun.ve

RESUMEN

En este trabajo se realiza una evaluación de SIMEP: Un Servidor de Imágenes que soporta la ejecución concurrente de MÉtodos Paralelos, tomando como base una aplicación para segmentación de imágenes.

SIMEP es un sistema diseñado con el objetivo de proveer a los desarrolladores de aplicaciones paralelas para procesamiento de imágenes de una herramienta que les permita aislarse de las características de la máquina paralela. Este artículo resume una serie de pruebas llevadas a cabo con la finalidad de evaluar su utilidad, su robustez, su rendimiento en tiempo y la factibilidad de agregar nuevas facilidades.

Adicionalmente, en este trabajo se establecen algunas limitaciones de SIMEP y se comparan los resultados para la aplicación de segmentación de imágenes con los obtenidos en implementaciones realizadas directamente sobre la máquina paralela.

1- INTRODUCCIÓN

En este trabajo se presentan las pruebas llevadas a cabo para evaluar el rendimiento de una herramienta para el desarrollo de aplicaciones paralelas: SIMEP (un Servidor de Imágenes que soporta MÉtodos Paralelos) permite la ejecución concurrente de operaciones paralelas sobre imágenes en un computador Parsytec MC-3 DE [Par92] del Centro de Computación Paralela y Distribuida de la U.C.V.

El objetivo de SIMEP es facilitar la construcción de aplicaciones paralelas para el tratamiento de imágenes; para ello provee un conjunto básico de operaciones implementadas en paralelo, las cuales pueden ser combinadas por el usuario para construir tratamientos más complejos. En [DG87] presentan el uso de este conjunto de operaciones para expresar tratamientos muy frecuentes sobre imágenes, tales como: histograma, reducción de ruido local y dilatación.

Como parte de la evaluación del desempeño de SIMEP se implementó el método del umbral para segmentación de imágenes biomédicas [Riv93]; lo cual nos permite determinar la factibilidad de su uso para el desarrollo de aplicaciones complejas, observar su comportamiento en tiempo de ejecución, además de realizar un análisis comparativo de los resultados con respecto a los obtenidos en otra implementación paralela de este método realizada sobre la misma máquina [Mon96].

Este trabajo está estructurado en cinco (5) secciones. En la sección 2 se describe SIMEP presentando la plataforma de hardware, el conjunto de operaciones que provee, los procesos

[1] Este Trabajo fue financiado por el CONICIT-Venezuela, Proyecto S100710

componentes y las relaciones entre ellos. En la sección 3 se describen los experimentos realizados para evaluar SIMEP a través de una aplicación de segmentación de imágenes. En la sección 4 se muestra el análisis de los resultados de las pruebas. Finalmente, se exponen las conclusiones obtenidas a lo largo del desarrollo de este trabajo.

2.- SIMEP: UN SERVIDOR DE IMÁGENES CON MÉTODOS PARALELOS

SIMEP es una herramienta desarrollada en el Centro de Computación Paralela y Distribuida de la Universidad Central de Venezuela, para un computador PARSYTEC MC 3-DE (ver [Leo95]).

El desarrollo de SIMEP persigue dos objetivos bien relacionados: Por un lado, proveer a los desarrolladores de algoritmos paralelos de una herramienta que les permita aplicar un conjunto de operaciones básicas, ya paralelizadas, sobre imágenes. Dichas operaciones pueden ser combinadas para construir tratamientos paralelos complejos sin necesidad de involucrarse con la arquitectura de la máquina ni con métodos para paralelización de algoritmos. El segundo objetivo, es proveer la facilidad de uso concurrente de imágenes y de operaciones paralelizadas por diferentes usuarios.

La paralelización de las operaciones básicas se realizó siguiendo el método de particionamiento de dominio, siendo cada imagen distribuida sobre los diferentes nodos particionada de forma equitativa. Las operaciones fueron implementadas de acuerdo al modelo SPMD (Single Program Multiple Data, ver [Dec83]), de forma tal que para aplicar una operación OP_k sobre una imagen Im^j, cada procesador "p" de la máquina paralela ejecuta el mismo código sobre la partición local $Im^j{}_p$ que posee de la imagen.

Para soportar el sistema multiusuario se implementó un mecanismo para el control de concurrencia, que explota la propiedad de conmutatividad a posteriori de las operaciones dentro del método tradicional de bloqueo a dos fases en transacciones anidadas (ver [LR94]). El control de ejecución de las operaciones sobre la máquina paralela, es manejado de acuerdo al modelo Maestro/Esclavo.

2.1.- Conjunto de operaciones sobre imágenes:

Dados un conjunto de coordenadas $X \subset Z^2$ y un conjunto de valores $F = \Re$, decimos que una Imagen es una estructura en dos dimensiones definida por la función:

$$I : X \rightarrow \Re, \qquad I = \{ (x,I(x)): x \in X, I(x) \in \Re \ \}.$$

X es un dominio de forma rectangular y cada elemento (x,I(x)) de una imagen I se denomina un "picture element" ó pixel; donde x es la localización del pixel relativa a un origen fijo e I(x) es el nivel de gris asociado al pixel denotado por un valor en el conjunto $\Re$. En SIMEP, las imágenes se consideran representadas en formato VFF (Visualization File Format), cuyos valores están tabulados dentro del rango [0,255].

En la Tabla 1 se muestran las operaciones sobre imágenes consideradas en SIMEP. La semántica se expresa usando la siguiente notación propuesta en [CFR89]: Sea $(a_i, T.op(a,p,r))$ la invocación por la transacción T de la operación **op** sobre el objeto **a**, a partir de un estado inicial a_i, siendo **p** el parámetro de entrada y **r** el resultado o parámetro de retorno para op. Al ejecutar

$(a_i, T.op(a,p,r))$ se puede observar el efecto producido sobre el objeto a, que es af y el efecto producido sobre T, que viene dado por r; con r = ∅ si la operación no tiene parámetros de retorno.

Sean: a,b,c : Imágenes; $F1 \subset \Re$; $X1 \subset X$; s : lógico; r1 : $\Re$

Operación	Dominio	Rango	Efecto sobre a	Efecto sobre T
Dominio: $(a_i, T.Dom(a,X1))$	Imagen	X	a_i	$\{x \in X : (x, a_i(x))\}$
Rango: $(a_i, T.Ran(a,F1))$	Imagen	$\Re$	a_i	$\{a_i(x) \in \Re : (x, a_i(x))\}$
Modificar: $(a_i, T.Modi(a,b))$	Imagen x Imagen	Imagen	{Ran(a) = Ran(b) ∧Dom(a)=Dom(b)}	∅
Restricción por Coordenadas: $(a_i, T.RPC(a,X1,s))$	Imagen x X	Imagen x Lógico	{(si X1⊂ Dom(a_i) entonces {(x,a_i(x)): x∈X1 } sino a_i }	{(si X1⊂ Dom(a_i) entonces s = verdad sino s = falso)}
Restricción por Valor : $(a_i, T.RPV(a,F1))$	Imagen x $\Re$	Imagen	{ (x,a_i(x)): a_i(x) ∈ F1 }	∅
Extensión : $(a_i, T.Ext(a,b,s))$	Imagen x X	Imagen x Lógico	{si Dom(a_i) ⊂ Dom(b) entonces a_i ∪ {(x,b(x)):x∉ Dom(a_i)} sino a_i}	{ si Dom(a_i) ⊂ Dom(b) entonces s = verdad sino s = falso }
Adición: $(a_i, b_i, T.Add(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{ (x,c(x)) : c(x) = a(x) + b(x) }
Substración: $(a_i, b_i, T.Sub(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{ (x,c(x)) : c(x) = a(x) - b(x) }
Multiplicación: $(a_i, bi, T.Mul(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{ (x,c(x)): c(x) = a(x) * b(x) }
División: $(a_i, bi, T.Div(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{(x,c(x)): c(x) = a(x) / b(x) } b(x)≠0
Máximo: $(a_i, bi, T.MX(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{(x,c(x)): c(x) = max{a(x),b(x)}} max denota máximo sobre $\Re$
Mínimo: $(a_i, bi, T.MN(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{(x,c(x)): c(x) = min {a(x),b(x))} } min denota mínimo sobre $\Re$
Exponente: $(a_i, bi, T.EXP(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{(x,c(x)): c(x) =$a(x)^{b(x)}$} $a(x)^{b(x)}$ definida: a(x)>0 ó (a(x)=0 ó b(x)=0)
Logaritmo : $(a_i, bi, T.LOG(a,b,c))$	Imagen x Imagen	Imagen	a_i , b_i	{(x,c(x)): c(x) = $\text{Log}_{a(x)}^{b(x)}$} $\text{Log}_{a(x)}^{b(x)}$ definida para: a(x) > 0 ∧ B(x) > 0
Producto Puntual : $(a_i, bi, T.PP(a,b,R1))$	Imagen x Imagen	$\Re$	a_i , b_i	$\sum_{x \in X} a(x) * b(x)$
Negación: $(a_i, T.NEG(a,c))$	Imagen	Imagen	a_i	{(x,c(x)): c(x) = - a(x)}
Módulo: $(a_i, T.MOD(a,c))$	Imagen	Imagen	a_i	{(x,c(x)):c(x)=|a(x)|} | | denota módulo en $\Re$

Tabla 1: Conjunto de Operaciones

Como se puede observar en la Tabla 1, las operaciones provistas por SIMEP son locales, esto es tratan pixeles en posiciones correspondientes de dos imágenes. Dichas operaciones pueden ser combinadas, en forma arborescente, para construir algoritmos para procesamiento de imágenes como dilatación, segmentación o reducción de ruido local entre otros (ver [DG87]). Una descripción más completa del conjunto de operaciones puede verse en [LR94].

2.2.- Estructura de SIMEP

El computador Parsytec MC-3 DE es de arquitectura MIMD [Dec89] basado en una red de transputers T805 bajo topología física de malla 2D [Par92]. El MC-3 DE sobre el cual se implementó SIMEP dispone de ocho (8) procesadores y tiene asociada una estación Sun Sparc/2 que permite la comunicación entre la red de procesadores y el mundo exterior; dando acceso a facilidades provistas en ambientes Sun. PARIX (PARallel unIX) es el ambiente para administración y programación de Parsytec MC-3 DE, el cual introduce extensiones al sistema

operativo UNIX para cubrir las necesidades de la máquina paralela. Entre las características de este computador, podemos destacar [Par92]:

1- Red de procesadores independientes con memoria local.
2- En cada procesador es posible ejecutar procesos concurrentes individuales (threads), con memoria compartida.
3- Permite la comunicación síncrona y asíncrona [Dec89] entre procesos sobre el mismo ó diferentes procesadores.
4- La comunicación con procesos externos a la Parsytec MC-3, es posible a través de una interfaz socket standard para manejo de comunicación entre procesos sobre redes [Ste90].

SIMEP fue diseñado de acuerdo al modelo Cliente/Servidor [CS93] para el diseño de software. El servicio prestado es la manipulación controlada de imágenes, teniendo a las aplicaciones desarrolladas como sus Clientes. SIMEP es un servidor concurrente (ver [Ste90] y [CS93]), ya que es capaz de mantener múltiples operaciones en ejecución simultáneamente. Sin embargo, los requerimientos de ejecución son analizados en serie debido a la necesidad de manejar información global acerca del estado de las imágenes.

La implementación de SIMEP, se realizó en base a procesos comunicantes ejecutados en forma distribuida sobre la arquitectura física. Algunos procesos residen en el Host y otros en la red de Transputers.

En la Figura 1 se presenta la arquitectura de SIMEP en base a sus procesos componentes, haciendo uso de la siguiente notación:

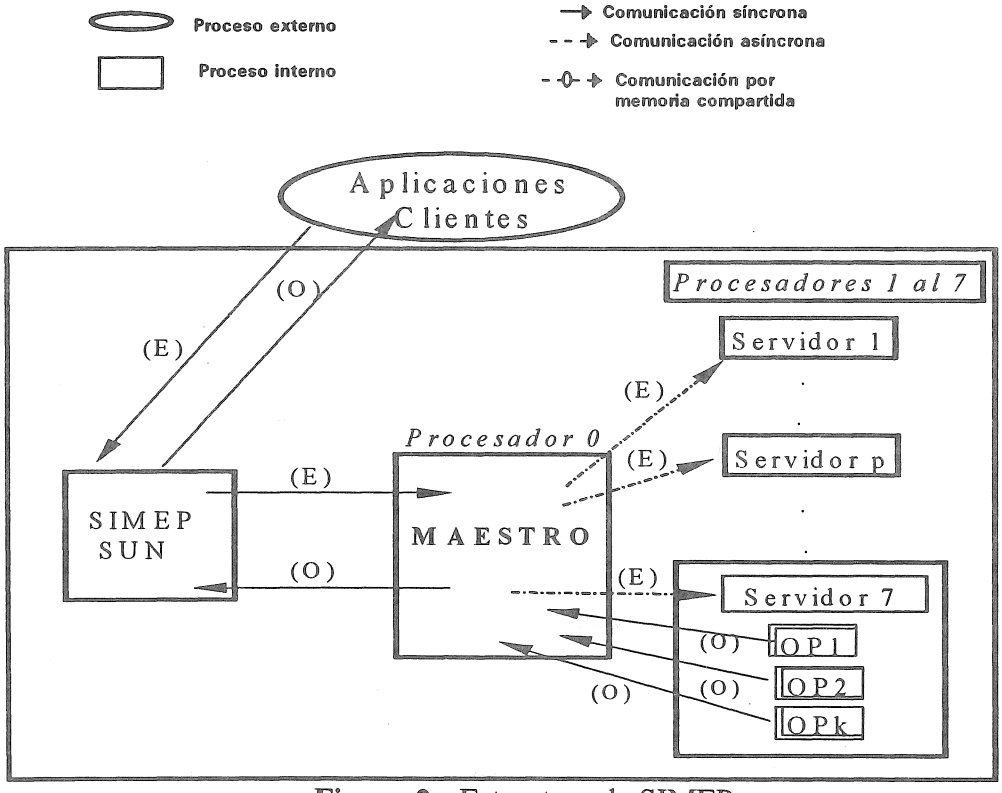


Figura 2: Estructura de SIMEP

En la Figura 2, **E** denota un mensaje para solicitar la ejecución de una operación y **O** un mensaje de finalización. A continuación se describe la función de cada proceso.

Proceso: SIMEP/Sun: Se encarga de interactuar con las aplicaciones clientes y de controlar los accesos concurrentes sobre las imágenes. Es implementado siguiendo el esquema de un servidor iterativo multiservicio basado en Socket [Ste90].

Al recibir mensaje: (E,OP_k,Im^i)

Aplica el método de control de concurrencia, para verificar si es posible ejecutar OP_k considerando las operaciones activas. Si es posible, retransmite el mensaje al proceso Maestro localizado en el procesador 0 de la máquina paralela; sino se encola OP_k.

Al recibir mensaje: (O,OP_k,Im^i)

Envía mensaje (O,OP_k,Im^i) a la aplicación Cliente. Aplica el método de control de concurrencia, a las operaciones en la cola de espera, para verificar si es posible su ejecución. Si fuese posible, envía el mensaje (E,OP_k,Im^i) correspondiente al proceso Maestro; sino mantiene encolada la OP_k.

Proceso Maestro: Reside en el procesador 0 de la máquina paralela, donde se lleva el control de ejecución de las operaciones.

Al recibir mensaje: (E,OP_k,Im^i)

Crea estructuras para el control de ejecución, inicializa Contador-Respuestas para (OP_k,Im^i) en 0 y envía de forma asíncrona el mensaje (E,OP_k,Im^i) a los procesos $Servidor_p$ ($\forall$ p=1..7).

Al recibir mensaje: (O,OP_k,Im^i)

Incrementa en uno el Contador-Respuestas para (OP_k,Im^i). Si la operación terminó en todos los servidores (Contador-Respuestas para (OP_k,Im^i) es 7) envía el mensaje (O,OP_k,Im^i) a SIMEP/Sun.

Proceso Servidor$_p$: Está duplicado en los restantes procesadores de la máquina paralela. Cada $Servidor_p$ (p=1..7) actúa como un encapsulador de las particiones de las imágenes que posee, soportando la ejecución concurrente de los métodos. Esto se muestra en la Figura 2.

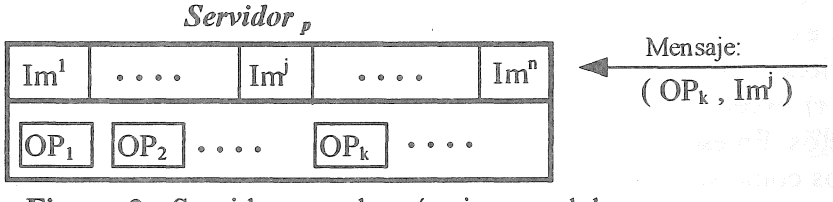


Figura 2: Servidores en la máquina paralela

Al recibir mensaje: (E,OP_k,Im^i)

Se genera un Thread (ó hilo) para ejecutar la operación OP_k sobre la imagen Im^i, de acuerdo al modelo de servidor concurrente [Ste90]. Al finalizar la operación OP_k, el proceso hilo se encarga de enviar el mensaje (O,OP_k,Im^i) con los resultados al proceso Maestro.

Una descripción detallada de SIMEP puede verse en [Leo95].

2.3.- Ejemplo de una aplicación en SIMEP.

En la Figura 3 se presenta un ejemplo de un árbol de composición de operaciones en SIMEP. El símbolo "||" denota ejecución concurrente de los subárboles y ";" denota la ejecución secuencial. De esta manera, se expresa que las secuencias de operaciones: Op1 ; Op2 sobre la imagen 1 y Op1 ; Op2 sobre la imagen 2, pueden ser ejecutadas concurrentemente.

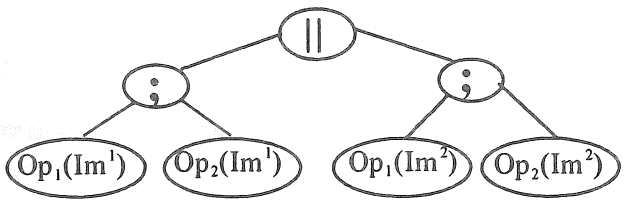


Figura 3: Una aplicación en SIMEP

3.- EVALUACIÓN DE SIMEP, A TRAVÉS DE UNA APLICACIÓN.

La aplicación usada para estudiar el rendimiento de SIMEP es la implementación de una variante del método del umbral para segmentación. La segmentación es un proceso muy usado para extraer una figura o característica de particular interés presente en una imagen [Nie89]. El método de segmentación por umbral genera una imagen en blanco y negro, en donde el objeto de interés es negro y el fondo blanco [DG87]. Una variante de este método en donde sólo el fondo es modificado, es definido como sigue:

Umbral = $(a_i, T.Umb(a,t,b))$ Umb: Imagen x $\Re \rightarrow$ Imagen

Efecto sobre a = a_i Efecto sobre $T = \{(x,b(x)) / b(x = a(x)$ si $a_i(x) \geq t$ y $b(x)=255$ si $a_i(x) < t \}$.

El valor t, denominado umbral, puede ser determinado usando el histograma de valores de gris asociado a una imagen (ver [DG87]). El histograma nos da la distribución de los niveles de gris sobre la imagen sin importar su localización, sino sólo la frecuencia de ocurrencia.

En las imágenes tratadas se estudian tres tipos de objetos (denominados Tipo I, Tipo IIa y Tipo IIb) con el interés de extraer los tipos presentes en una imagen dada. En [Riv93] se presenta un algoritmo para segmentación, que utiliza el porcentaje de pixels oscuros presentes en una imagen para establecer la cantidad de tipos de objetos que ella contiene. Así, se considera que la imagen contiene tres tipos si la cantidad de pixels oscuros (pixels con valor de nivel de gris menor o igual que t) excede el 30% del total de pixels; de lo contrario la imagen presenta sólo dos tipos de objetos. En ese mismo trabajo, el método de segmentación por umbral es implementado en varios pasos como se explica a continuación:

1: Cálculo del histograma de niveles de gris.
2: Determinación del umbral (t) y de la cantidad de tipos de objetos presentes en la imagen.
3: Utilización del umbral para obtener una imagen que contenga los objetos Tipo I.
4: La imagen obtenida en el paso 3 y la imagen original, son utilizadas para obtener la imagen complementaria (la cual contiene los objetos Tipo II).

5: Si la a imagen original contiene tres (3) tipos de objetos, se repiten los pasos 3 y 4 a partir de la imagen complementaria, para obtener los objetos Tipo IIa y Tipo IIb.

Este método de segmentación fue implementado de forma secuencial sobre una Sun Sparc/2 ([ver [Riv93]) y de forma paralela sobre la Parsytec MC-3 DE (ver [Mon96]).

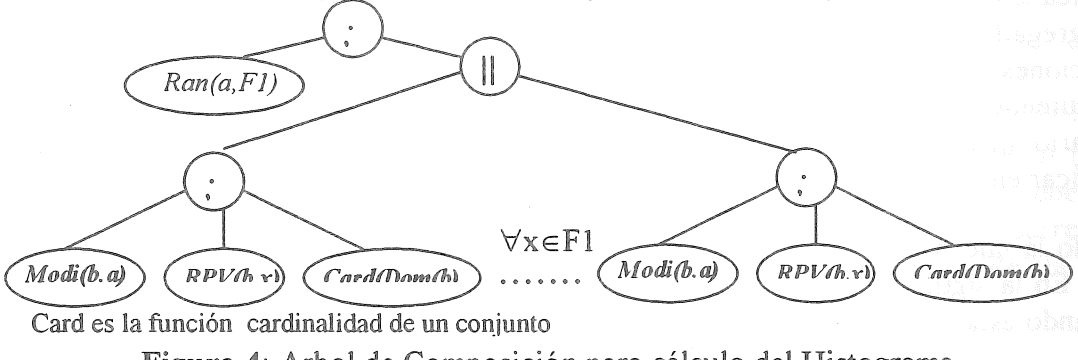
La implementación descrita en [Mon96] fue realizada usando directamente las herramientas para programación provistas en la máquina paralela y siguiendo el modelo de partición de dominio para la paralelización y de Maestro/Esclavo para el control. Los pasos 1 y 2 son realizados localmente en cada proceso esclavo y enviando luego los umbrales locales al maestro donde se calcula el global, que después es enviado a cada esclavo para realizar la segmentación local.

Las características del trabajo de [Mon96] que nos permite situarnos en el mismo escenario, lo hizo ideal para tomarlo como patrón de referencia y punto de comparación para esta evaluación de SIMEP.

El método de segmentación por umbral ya descrito, puede ser expresado a través de las operaciones básicas provistas por SIMEP (ver Tabla 1). En particular, el cálculo del histograma de niveles de gris (paso 1) y la segmentación de la imagen (pasos 4 y 5) son los procesos que pueden ser desarrollados como una composición de las operaciones de SIMEP, mientras que la determinación del valor del umbral (t) y de la cantidad de objetos presentes debe ser realizado por la aplicación cliente. Es importante destacar que dentro del proceso de segmentación en si es posible solapar la ejecución de las operaciones básicas que lo componen explotando la facilidad de ejecución concurrente de operaciones paralelas provistas por SIMEP.

A continuación, se describe la implementación en SIMEP del método de segmentación por umbral, a través de los árboles de composición de operaciones predefinidas que corresponden a cada paso:

Paso 1: Calcular el histograma de niveles de gris, calculado en paralelo.



Card es la función cardinalidad de un conjunto

Figura 4: Arbol de Composición para cálculo del Histograma

Paso 2: En base al histograma, determinar el umbral (t) y la cantidad de objetos presentes en la imagen. Este paso es realizado por la aplicación Cliente, en base a los resultados dados por SIMEP para el paso anterior. Si la imagen posee tres tipos de objetos, se calcula el segundo umbral (t1) simultáneamente con el paso 3.

Paso 3: Realizar la segmentación de la imagen original (denotada a en el árbol) para obtener los objetos Tipo I y Tipo II. Nótese la ejecución concurrente en SIMEP de las operaciones componentes.

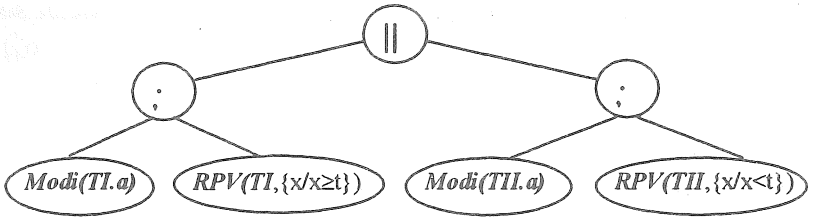


Figura 5: Segmentación

Paso 4: Si la imagen contiene tres tipos de objetos, se realiza la segmentación de la imagen Tipo II para obtener los objetos Tipo IIa y Tipo IIb,

La ejecución del árbol de cálculo del histograma originó un elevado volumen de mensajes entre los procesos componentes de SIMEP debido a que: Por cada requerimiento de ejecutar una operación básica se emite un mensaje desde la aplicación cliente al proceso SIMEP/Sun, que una vez verificado el control de concurrencia se retransmite al proceso Maestro. El Maestro genera un mensaje para cada proceso Servidor, donde finalmente se inicia un Thread para cada operación la cual inicia el flujo de comunicación en sentido inverso. Considerando 255 valores diferentes de gris, se llega a tener hasta ese número de operaciones ejecutándose concurrentemente en cada procesador.

En vista que el cálculo del histograma en base a las operaciones básicas de SIMEP resultó sumamente lento, alrededor de 20 minutos, y en algunos casos bloqueaba el sistema completamente, se decidió hacer más eficiente este cálculo implementándolo como una operación básica, enriqueciendo así el conjunto de operaciones iniciales. La operación Histograma no modifica las imágenes, esto hace que pertenezca al conjunto de operaciones de consulta y pueda ser agregado sin alterar las tablas de conmutatividad determinadas para el conjunto de operaciones ni el mecanismo para el control de accesos concurrentes (ver [LR94]). Fue incluida en siguiendo el mismo modelo SPMD como el resto de las operaciones, para lo cual sólo fue necesario desarrollar el código correspondiente a un procesador y enlazarlo a SIMEP, sin modificar en modo alguno su estructura.

Usando la nueva operación Histograma el árbol de composición del paso 1 se reduce a un sólo nodo. En la siguiente sección se presentan los resultados obtenidos para el método del umbral, utilizando esta nueva operación.

4.- ANÁLISIS DE RESULTADOS.

Para las pruebas del método de segmentación por umbral sobre SIMEP, se utilizaron 8 imágenes de tamaño 512 x 512 pixels. Esta mismas fueron usadas en [Mon96]. La Tabla 2 muestra para cada imagen: el tiempo secuencial y el tiempo usando 8 procesadores reportados en [Mon96], así como el tiempo de ejecución sobre SIMEP.

Imagen	Tiempo Secuencial	Tiempo Paralelo (8 Procesadores)	
		[MON 96]	SIMEP
1	5,36	2,06	1,699
2	5,54	2,07	1,517
3	5,47	2,05	1,514
4	5,30	2,04	1,668
5	3,57	1,52	1,431
6	3,60	1,52	1,513
7	3,57	1,52	1,535
8	3,53	1,51	1,460

Tabla 2: Tiempos de procesamiento

El tiempo de ejecución en SIMEP corresponde al promedio de realizar 10 pruebas con la imagen. Las primeras cuatro (4) imágenes contienen los tres tipos de objetos y las restantes presentan sólo dos, por ello las diferencias en tiempo. Para la imágenes con tres tipos de objetos la desviación standard es de 0,17 seg., mientras que para las que presentan dos tipos de objetos es de 0,053 seg.

La Figura 6 muestra, en diagrama de barras, la comparación gráfica entre los tiempos para la segmentación por umbral obtenidos en este trabajo y los reportados en [Mon96] usando 8 procesadores de la Parsytec MC-3 DE. Los tiempos están expresados en segundos.

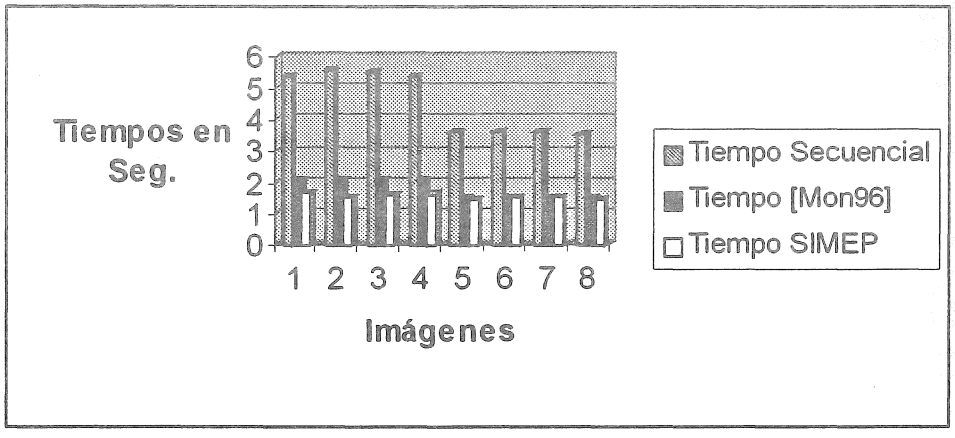


Figura 6: Comparación de tiempos

Se puede observar que para las imágenes con tres (3) tipos de objetos, los resultados obtenidos con SIMEP son significativamente mejores que los reportados en [Mon96]. Esto se debe a que en esta aplicación se explota la facilidad provista por SIMEP para ejecutar concurrentemente múltiples operaciones básicas implementadas en paralelo y la ejecución de la segunda segmentación (objetos Tipo II) se inicia de inmediato al finalizar la anterior.

5.- CONCLUSIONES

En este trabajo se mostró que es posible, para un programador, combinar las operaciones básicas provistas en SIMEP con el fin de construir tratamientos más complejos, aliviándose la tarea de desarrollar aplicaciones paralelas para tratamiento de imágenes digitales.

Así mismo, se mostró que el conjunto de operaciones definido en [LR94] puede ser enriquecido, sin gran dificultad, agregando nuevas operaciones con lo cual se incrementa la utilización y la eficiencia de SIMEP en aplicaciones de tratamiento de imágenes.

La facilidad de SIMEP de permitir la ejecución concurrente de operaciones paralelizadas, puede reducir el tiempo de ejecución global de una aplicación. Este es el caso de la segmentación por el método del umbral, para la que se obtuvo tiempos mejores o iguales a los reportados en trabajos previos que implementan directamente el método sobre la máquina paralela.

Actualmente, estamos evaluando la herramienta de forma más amplia, a través de la generación aleatoria de árboles de ejecución que representen cualquier estructura de una aplicación. Esto nos permitirá mejorar el rendimiento y establecer con exactitud las limitaciones de SIMEP.

6.- REFERENCIAS BIBLIOGRÁFICAS

[CFR89] CART Michéle,FERRIE Jean,RICHY Hélène. "Contrôle de l'Exécution de Transactions Concurrentes". Technique et Science Informatique, vol.8., No. 3, pag. 225-240. Junio 1989.

[CS93] COMMER Douglas,STEVENS David. Internerworking with TCP/IP Client-Server Programming and Applications BSD Socket Version. Volume III. Prentice Hall 1993.

[Dec89] DECEGAMA Angel. The Technology of Parallel Processing. Parallel Processing Architecture. Volume 1. Prentice Hall. 1989.

[DG87] DOUGHERTY Eduard,GIARDINA Charles. Matrix Structured Image Processing. Prentice Hall. 1987.

[LR94] LEÓN Claudia,RUKOZ Marta. "Control de Concurrencia en un Servidor de Imágenes Distribuido". Memorias de la XX Conferencia Latinoamericana de Informática, PANEL'94, pag. 1363-1374. Ciudad de México. Septiembre 1994.

[LR95] LEÓN Claudia,RUKOZ Marta. "Sistema Distribuido para Tratamiento de Imágenes". Memorias de la XXI Conferencia Latinoamericana de Informática, PANEL'95. Canela, Brasil. Julio 1995.

[Leo95] LEÓN Claudia. "SIMEP: Un Servidor de Imágenes que soporta Métodos Paralelos". Trabajo de Grado Magister Scientiarum. U.C.V. Octubre 1995.

[Mon96] MONZÓN Marisela,ORTIZ Mercedes,RIVAS Miriam,RUKOZ Marta. "Paralelización de tres métodos de Segmentación de Imágenes Biomédicas". Métodos Numéricos y Simulación en Ingeniería. Cerrolaza, Gajardo y Brebbia Editores. 1996.

[Nie89] Niemann H. "Pattern Analysis and Understanding". Second Edition Spriger-Verlay. 1989.

[Par92] PARIX Documentation. Release 1.1. Parsytec Computer GmbH. Alemania. 1992.

[Riv93] RIVAS Miriam. "Tratamiento de imágenes provenientes de estudios histoquímicos del músculo estriado". Trabajo de Grado Magister Scientiarum. U.C.V. 1993.

[Ste90] STEVENS Richard. UNIX Network Programming. Prentice Hall. 1990.