Algoritmo Paralelo de Codificación de Video Basado en H.264/AVC con Balanceo Predictivo de Carga

Carlos Genis-Triana[1], Abelardo Rodríguez-León[2] y Rivera López Reyes[3]

Departamento de Sistemas y Computación. Instituto Tecnológico de Veracruz
Calzada Miguel Ángel de Quevedo 2779, Veracruz, México
[1]carlosgenis@yahoo.com.mx, [2]arleon@itver.edu.mx, [3]rrivera@itver.edu.mx

Abstract. Este artículo muestra el diseño e implementación de un algoritmo paralelo para la codificación de video, el cual se basa en el estándar H264 que incluye paralelización a nivel de GOPs, haciendo una distribución de los mismos en un cluster por medio del protocolo estándar de paso de mensajes MPI. La manera en que se distribuye la carga es mediante la combinación de dos esquemas: inicialmente por preasignación y posteriormente predicción. La idea es que una vez hecha la repartición inicial (preasignación), se procede a determinar la complejidad de los GOPs en tiempo de codificación (predicción), para posteriormente hacer una planificación que permita balancear la codificación, enviando los GOPs más pesados a los procesadores que codificaron previamente GOPs ligeros y viceversa. Se evaluó el algoritmo propuesto, comparándolo contra la versión secuencial del mismo codificador, considerando la tasa de compresión y el aprovechamiento con diferentes números de nodos de procesamiento y con diversos tipos y resoluciones de videos.

1 Introducción

En los últimos años se han propuesto y se han desarrollado una variedad de estándares de compresión de video e imagen (H.26X, MPEG-X, JPEG200), sin embargo, no todos proporcionan las mismas características. Unos obtienen mayor calidad de imagen a costa de un mayor tiempo de procesamiento (tiempo de compresión), por lo que se ha tenido que buscar un equilibrio entre el tiempo de procesamiento y la calidad que se obtiene de las imágenes descomprimidas. Por tanto, si se aumentará la velocidad de compresión se podrían utilizar compresores que ofrecen una calidad mayor. Hasta hace poco los más destacados eran H263 y MPEG4, debido a los grupos que los respaldan. Sin embargo, recientemente surgió uno nuevo, el H264 propuesto por el Joint Video Team (JVT), el cual promete aun más que todos lo antes citados.

Ahora bien, la tarea de comprimir video, demanda sistemas de alto rendimiento para lo cual se tienen las siguientes alternativas: Sistemas fuertemente acoplados, tarjeta codificadora y sistemas débilmente acoplados. Para explotar un sistema multiprocesador (débilmente o fuertemente acoplado) en la compresión del video, se requiere utilizar alguna implementación de algún estándar de compresión, paralelizándolo con técnicas de programación paralela tales como: MPI (Interfaz para el paso de mensajes) desarrollada por el grupo Forum de MPI.

M.A. Cruz-Chávez, J.C. Zavala-Díaz (Eds): CICos 2008, ISBN:978-607-00-0165-9, pp. 112-121, 2008

Este artículo muestra el diseño e implementación de un algoritmo paralelo para la codificación de video, el cual se basa en el estándar H264 que incluye paralelizáción a nivel de GOPs (15 frames), haciendo una distribución de los mismos en un cluster por medio del protocolo estándar de paso de mensajes MPI. La manera en que se distribuye la carga es mediante la combinacion de dos esquemas: inicialmete por preasignación y posteriormente por predicción. La idea es que una vez hecha la repartición inicial (preasignación), se proceda a determinar la complejidad de los GOPs en tiempo de codificación (predicción), para posteriormente hacer una planificación que permita balancear la codificación, enviando los GOPs más pesados a los procesadores que codificarón previamente GOPs ligeros y viceversa. Se evaluó el algoritmo propuesto, comparándolo contra la versión secuencial del mismo codificador, considerando la tasa de compresión y el aprovechamiento con diferentes números de nodos de procesamiento y con diversos tipos y resoluciones de videos. Cabe señalar, que los frames que conforman un GOP se clasifican en base a la técnica de compresión aplicada. El primer frame de cada GOP se denomina I, el cual se comprime utilizando la compresión intracuadros, eliminando sólo la reduncia contenida en un solo frame. El resto de los frames, se comprimen utilizando la compresión intercuadros, denominados de tipo B ó P, dependiendo de cómo se haya aplicado la técnica: Frame P, si se comprime por medio de otro, un I ó P anterior y Frame B, siempre que se comprime a partir de dos frames, un I ó P anterior y un P posterior.

En el punto 1 se hace una introducción a la compresión de video y al procesamiento paralelo. En el 2 se hace una descripción detallada del funcionamiento del algoritmo paralelo que se ha imlementado. En el 3 se muestran algunos resultados obtenidos de la evaluación a la que fue sometida dicho algoritmo. Y por último en el 4 se hace mención sobre los trabajos que se podrían realizar más adelante.

2 Descripción del algoritmo paralelo con balanceo predictivo

El algoritmo aquí propuesto, toma como base para su desarrollo un esquema implementado con MPEG4 [6], estableciendo algunas variantes que lo hacen más sofisticado. El objetivo es realizar un *balance de carga* más *dinámico*; es decir, que en tiempo de ejecución se pueda ir determinado la *complejidad* de los *GOPs* con la finalidad de realizar una *distribución* más *eficiente*.

Para este *esquema* de *compresión paralela* no todos lo procesadores involucrados en la tarea tienen la misma función. Existe un *procesador maestro* (PM o P0), cuyo propósito es coordinar y administrar la repartición de los GOPs durante el *proceso de compresión*; el resto de los *procesadores son esclavos*, en este caso particular compresores (PE o PC), que van de P1 a PN - 1. A continuación se detallan la serie de pasos lógicos bajo los cuales se realizo su implementación:

2.1. Distribución por preasignación (inicial)

En la primera tanda de distribución de GOPs realizada por P0 a los PC es por medio del esquema de preasignación, la cual está distribuida a lo largo de toda la secuencia de video; es decir, que para 12 GOPs y 4 nodos se tendría lo mostrado en la **Figura 1**. Así, al nodo 1 se le asigna el primer GOP, al nodo 2 el GOP 3 y así sucesivamente.

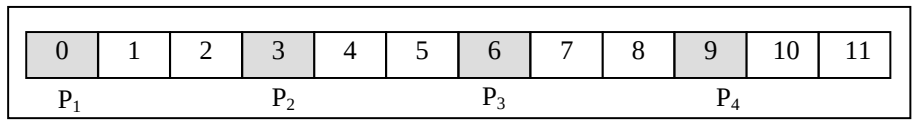

Figura 1 Esquema de inicial de distribución por preasignación

Para esta asignación inicial se toma en consideración cuando el número de procesadores compresores (NPC) no es múltiplo del número de GOPs a comprimir (NGC). Suponiendo que se tienen 5 NPC y 12 NGC, la distribución quedaría como se presenta en la **Figura 2**.

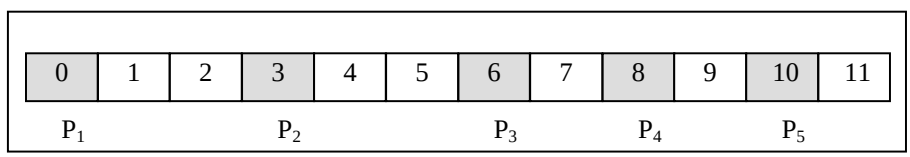

Figura 2. Esquema inicial de distribución por preasignación con NPC no múltiplo de NGC

2.2. Distribución por predicción

Para las posteriores *tandas de repartición* de GOPs se aplica el *esquema por predicción* para lo cual P0 recibe de cada PC las siguientes notificaciones:

- Un *mensaje* del *tiempo estimado* para de comprimir su GOP. Esto se lleva a cabo de la siguiente forma:
  - *Momento de envío*. El i*nstante t* en que se notifica el *tiempo estimado* es cuando el frame comprimido es el 12. Cabe señalar que los frames que conforman cada GOP se enumeran de 0 a 15 y siguiendo el *patrón de compresión paralelo* que se muestra en la **Figura 3**.

PC – Comprime GOP N (16 Frames)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I	B	B	**P**	B	B	**P**	B	B	**P**	B	B	**P**	B	B	**P**

Figura 3 Patrón de compresión paralelo

  - *Cálculo del tiempo estimado de compresión*. Se obtiene a partir de 3 cálculos hasta el instante t:
    - Obtención de los *tiempos promedios* de cada tipo de frame comprimido:

$$TPC_{FI} = \sum_{i=0}^{0} TC_{FI}[i] / NC_{FI} \quad (1)$$

$$TPC_{FB} = \sum_{i=0}^{7} TC_{FB}[i] / NC_{FB} \quad (2)$$

$$TPC_{FP} = \sum_{i=0}^{3} TC_{FP}[i] / NC_{FP} \quad (3)$$

donde:

TPC_FI - Tiempo promedio de compresión de frames I
TPC_FB - Tiempo promedio de compresión de frames B
TPC_FP - Tiempo promedio de compresión de frames
TC_FI - Tiempo de compresión de frames I
TC_FB - Tiempo de compresión de frames B
TC_FP - Tiempo de compresión de frames P
NC_FI – Número comprimido de frames I
NC_FB – Número comprimido de frames B
NC_FP – Número comprimido de frames P

- Cálculo del tiempo estimado de cada tipo de frame comprimido:

$$TEC_{FI} = TPC_{FI}[NFI_{SC}] \quad (4)$$

$$TEC_{FB} = TPC_{FB}[NFB_{SC}] \quad (5)$$

$$TEC_{FP} = TPC_{FP}[NFP_{SC}] \quad (6)$$

donde:

TEC_FI - Tiempo estimado de compresión de frames I
TEC_FB - Tiempo estimado de compresión de frames B
TEC_FP - Tiempo estimado de compresión de frames P
NFI_SC – Número de frames I sin comprimir
NFB_SC – Número de frames B sin comprimir
NFP_SC – Número de frames P sin comprimir

- Cálculo del tiempo estimado:

$$TEC = TEC_{FI} + TEC_{FB} + TEC_{FP} \quad (7)$$

donde:

TEC - Tiempo estimado de compresión de frames

- Un *mensaje* de *terminación* de compresión para una *nueva asignación* de GOP. Esto realiza de la siguiente manera:
  - *Momento d*e envío. Cuando un PC se encuentra en el frame 15, entonces se asume que ya comprimió su GOP.
  - *Determinación de la asignación de GOP*. P0 verifica entre aquellos PC que hayan realizado su notificación del *tiempo estimado* y en base a éstos identifica el que tenga el GOP más complejo; es decir, el que tenga un mayor tiempo y que no haya sido asignado. Una vez hecho esto, el P0 le asigna al PC que terminó, el GOP más próximo al GOP más complejo, en base a la hipótesis: “GOPs consecutivos tiene un nivel de complejidad similar”.

2.3. Integración de GOPs en un solo fichero

A fin de obtener el mismo *patrón de compresión* del *codec secuencial*, cada GOP comprimido es almacenado por el PC y unido por P0 dentro de un mismo archivo en el orden que les corresponde en la secuencia de video, omitiendo el último frame P de cada GOP de tal forma que se tengan 15 frames registrados por cada uno, como se muestra en la **Figura 4**.

P0 – Almacena GOP 0 (15 Frames)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
I	B	B	**P**	B	B	**P**	B	B	**P**	B	B	**P**	B	B	~~**P**~~

P0 – Almacena GOP 1 (15 Frames)

15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
I	B	B	**P**	B	B	**P**	B	B	**P**	B	B	**P**	B	B	~~**P**~~

Figura 4. Patrón de compresión paralelo almacenado

Los pasos descritos en los puntos 2.2 y 2.3 se repiten hasta que no haya más GOPs por asignar. En la **Figura 5** se puede observar el esquema funcional del nuevo codec basado en un esquema por predicción.

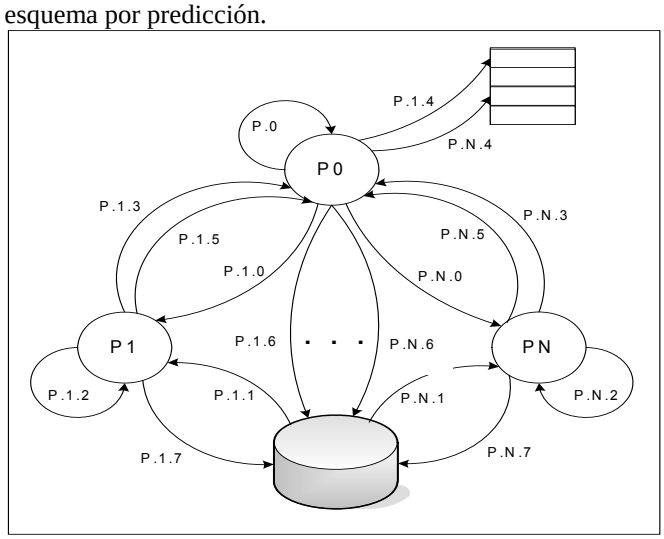

Figura 5 Esquema de funcionalidad del codec

Los *estados* del nuevo esquema de compresión paralela son los siguientes:

- P.0 – P0 determina el GOP a enviar basándose en el esquema por preasignación o predicción, según sea el caso.
- P.1.0 – P0 envía a P1 el GOP a comprimir.
- P.1.1 – P1 lee GOP del disco.
- P.1.2 – P1 inicia la compresión del GOP.
- P.1.3 – P1 envía tiempo estimado de compresión a P0.
- P.1.4 – P0 registra el tiempo estimado de compresión de P1.
- P.1.5 – P1 notifica a P0 la terminación de compresión del GOP para una nueva asignación.

- P.1.6 – P1 almacena el GOP comprimido en el disco.
- P.1.7 – P0 junta el GOP comprimido en el disco dentro de un mismo archivo, en el orden que les corresponde en la secuencia de video y omitiendo su último frame.
- P.0 – P0 determina el GOP a enviar basándose en el esquema por preasignación o predicción, según sea el caso.
- P.N.0 – P0 envía a PN el GOP a comprimir.
- P.N.1 – PN lee GOP del disco.
- P.N.2 – PN inicia la compresión del GOP.
- P.N.3 – PN envía tiempo estimado de compresión a P0.
- P.N.4 – P0 registra el tiempo estimado de compresión de PN.
- P.N.5 – PN notifica a P0 la terminación de compresión del GOP para una nueva asignación.
- P.N.6 – PN almacena el GOP comprimido en el disco.
- P.N.7 – P0 junta el GOP comprimido en el disco dentro de un mismo archivo, en el orden que les corresponde en la secuencia de video y omitiendo su último frame.

2.4. Requerimientos para la implementación y pruebas

El desarrollo del algoritmo paralelo y las pruebas de compresión se realizaron en un cluster llamado Aldebaran de la Universidad Politécnica de Valencia (UPV), el cual cuenta con las siguientes características:

- SGI Altix 3000
- Sistema multiprocesador de 48 procesadores Itanium II (1.46 Ghz).
- Memoria distribuida NUMA
- Conexión directa a la SAN (red de almacenamiento) del CPD (centro de proceso de datos de la UPV)
- Sistema operativo Linux con su distribución RedHat
- Implementación de MPI denominada MPT
- Gestor de trabajos denominado LSF
- Implementación de H264 denomida H26L recompilada con optimización 2 para Intel

3 Resultados Obtenidos

Como se se mencionó, la versión paralela anteriormente descripta fue sometida a una evaluación a fin es determinar su eficiencia en comparación a la versión secuencial del H264 implememtada por el JVT. El banco de pruebas consiste en 6 secuencias de video de dominio público en formato YUV 4.2.0. Algunas de sus características se enuncian en la **Tabla 1.**

Tabla 1. Características de las secuencias de video empleadas para la evaluación

Secuencia	Resolución	#Frames

foreman_cif	352x288	300
news_cif		300
students_cif		1007
flower_720x480	720x480	150
tennis_720x480		150
martin_720x480		240

3.1. Tasa de compresión de frames (FrameRate)

En la **Tabla 2** y en las **Figuras 6 y 7** se observa que al ir aumentando el NPC, también se incrementa aproximadamente con la misma proporción el FrameRate, sin embargo, esto no sucede entre 16 PC y 20 PC, 20 PC y 24 PC, produciendo casi la misma tasa. Esto podría deberse a que la forma en que se distribuye la carga, la cual muy probablemente propicie que se produzcan ventanas de ocio en algunos PC (tiempos de ocio debido a la falta de carga asignada), excepto cuando el NPC no es múltiplo del NG.

Tabla 2 FrameRate con 28 Gops para diferentes secuencias de video

Secuencia	FR 4PC	FR 8PC	FR 12 PC	FR 16 PC	FR 20 PC	FR 24 PC	FR 28PC	NPC
foreman_cif	0.678	1.197	1.581	2.386	2.359	2.393	4.536	**92.15**
news_cif	0.697	1.226	1.621	2.435	2.442	2.439	4.759	**87.84**
students_cif	0.710	1.249	1.648	2.482	2.475	2.496	4.840	**86.37**
flower_720x480	0.190	0.336	0.441	0.658	0.661	0.668	1.293	**323.26**
tennis_720x480	0.190	0.339	0.446	0.663	0.674	0.683	1.261	**331.60**
martin_720x480	0.197	0.346	0.459	0.675	0.689	0.709	1.299	**321.93**

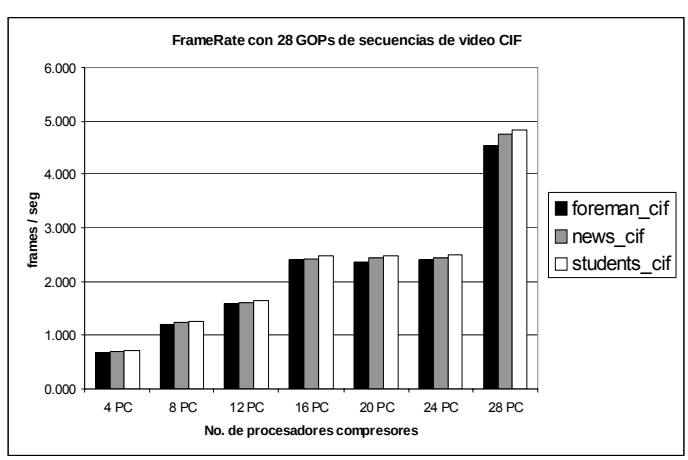

Figura 6 FrameRate con 28 Gops de secuencias de video CIF

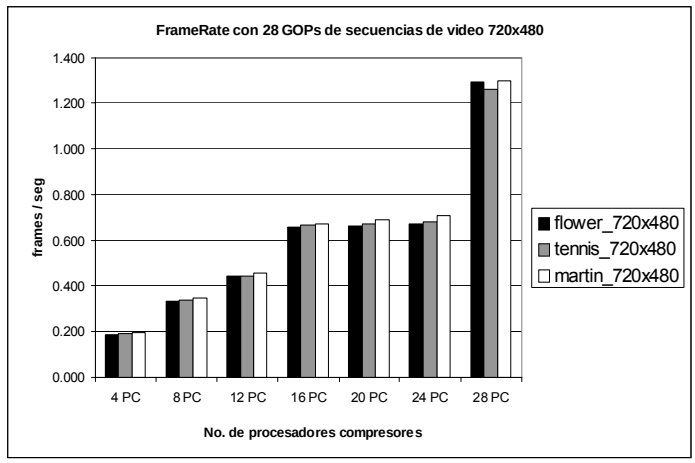

Figura 7 FrameRate con 28 Gops de secuencias de video 720x480

El tiempo real (30 frames/seg) para comprimir 28 GOPs (16 frames/GOP) es de 14.93 segundos; tomando como base lo anterior, se puede deducir los NPC para las diferentes secuencias de video mostrados en la última columna de la **Tabla 2**. Se puede apreciar que con todas las secuencias, hacen falta más PCs; en el caso de las CIF necesita entre 87 PC y 93 PC y con las de 720x480 requieren entre 322 PC y 332 PC.

3.2 Aprovechamiento de los procesadores (SpeedUp)

En la **Tabla 3** y en la **Figuras 8 y 9** se muestra el SpeedUp de las secuencias de video para los diferentes NPC. De la misma forma que su FrameRate mostrado en la **Tabla 2**, al escalar de 16 PC a 24 PC, el SpeedUp no aumenta proporcionalmente. Esto pudiera presentarse debido a las misma ventanas de ocio que se pudieran estar formando dada la distribución de la carga.

También se puede apreciar que cuando el NPC es de 4 PC y 28 PC se presenta un SpeedUp que sobre pasa lo deseado (ya que éste sobrepasa el ideal). Esto quiere decir que tal situación se presenta, cada vez que el NPC es múltiplo del NGC, debido a la ausencia de ventenas de ocio.

Tabla 3 SpeedUp ($SU = TCS / TCP$) con 28 Gops

Secuencia	4 PC	8 PC	12 PC	16 PC	20 PC	24 PC	28 PC
foreman_cif	4.332	7.649	10.105	15.251	15.078	15.295	28.994
news_cif	4.340	7.634	10.097	15.166	15.213	15.194	29.646
students_cif	4.323	7.601	10.028	15.100	15.062	15.188	29.451
flower_720x480	4.221	7.480	9.815	14.651	14.708	14.874	28.786
tennis_720x480	4.212	7.528	9.882	14.712	14.945	15.142	27.956
martin_720x480	4.326	7.592	10.073	14.818	15.136	15.561	28.511
Ideal	**04.00**	**08.00**	**12.00**	**16.00**	**20.00**	**24.00**	**28.00**

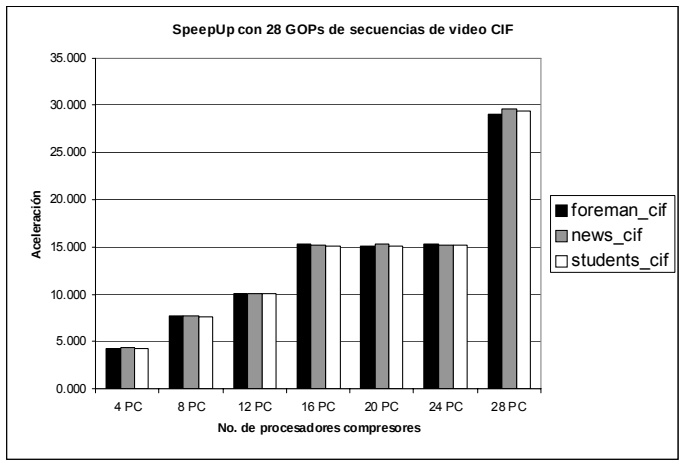

Figura 8 SpeedUp con 28 Gops de secuencias de video CIF

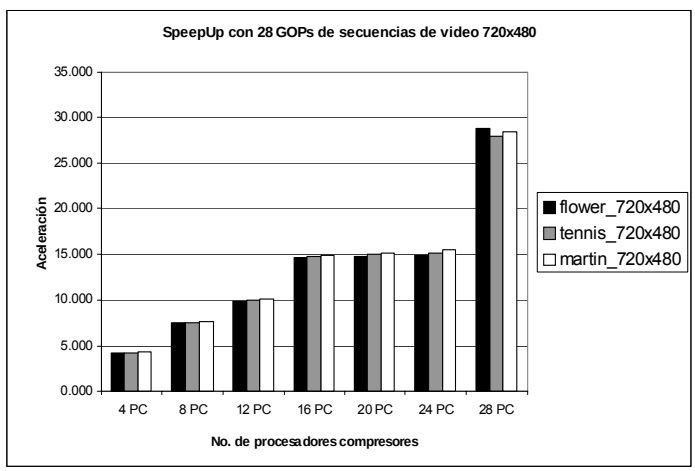

Figura 9 SpeedUp con 28 Gops de secuencias de video 720x480

4 Trabajos Futuros

Ahora bien, en un futuro, se propone realizar pruebas más exhaustivas con un mayor NG y NPC, a fin de analizar el balanceo de carga del algoritmo predictivo y así determinar si la formación de ventanas de ocio son la causa del comportamiento lineal que se presenta del FrameRate y SpeepUp entre 16 PC y 24 PC, así como del sobreaprovechamiento que se presenta cuando el NG es múltimplo del NPC. También para explicar la causa del comportamiento variable que se tienen de ambas variables cuando el NPC es el mismo para diferentes videos de una misma resolución. La idea es finalmente determinar para que tipo de video es es más adecuado, considerando otras características de los mismos, tales como: No. de fondos dinámicos y/o estáticos, No.

de figuras centrales, No. de objetos en movimiento. Así como determinar la Escalabilidad del algoritmo de tal forma que se pueda concluir si su rendimiento mejora conforme se van incrementando el número de nodos al cluster.

Referencias

1. Dongarra, J., Foster, I., Fox, G., Gropp-William, K. K., Torczon, L. y White A. “Source-Book of Parallel Computing”, Ed. Morgan Kaufmann (2003).[ES99] Effelsberg, W. y Steinmetz, R. “Video Compression Techniques”. Ed. Morgan Kauffman (1999).
2. J. C. Fernández y M. P. Malumbres, "A Parallel Implementation of H.26L Video Encoder", Lectures Notes on Computer Sciences No. 2400, Springer-Verlag pp. 830-833, 2002.
3. Genis-Triana C., Rodríguez-León A. “Evaluación de una versión paralela para el Codec H.264/AVC”. Artículo publicado en las memorias del CORE del CIC-IPN (Congreso Nacional del Centro de Investigación en Computación del Instituto Politécnico Nacional) cuyo título es: Recientes avances en la ciencia de la computación en México con el ISBN: 970-36-0149-9. Mayo del 2004.
4. Genis-Triana C., Rodríguez-León A. “Characterization of the load balance of a parallel implementation of the H264 by the GAP of DISCA-UPV”. Artículo publicado en las memorias del CIICC’04 del ITTLA (11vo. Congreso Internacional de Investigación en Ciencias Computacionales del Instituto Tecnológico de Tlalnepantla) con el ISBN: 968-5823-10-3. Septiembre de 2004.
5. Genis-Triana C. “Caracterización de un Algoritmo Paralelo con Balanceo Predictivo de Carga para la Compresión de Secuencias de Video con H.264”. Tesis para obtener al Grado de Maestro en Ciencias de la Computación en el ITV. Mayo del 2006.
6. Rodríguez-León Abelardo. “Diseño e implementación de algoritmos paralelos para la compresión de secuencias de vídeo MPEG4”. Reporte Interno de Investigación DISCA - Universidad Politécnica de Valencia. Noviembre de 2002.
7. Sánchez-Zavaleta M. “Monografía de MPI”. Tecnológico de Monterrey campus Morelos. Enero de 2001
8. Schäfer R., Wiegand T. and Schwarz H. “H.264/AVC The emerging standard”. Heinrich Hertz Institute, Berlin, Germany. January 2003.
9. Wiegand, T., Sullivan, G.J., Bjontegaard, G. y Luthra, A., “Overview of the H.264/AVC Video Coding Standard”, en IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, pp. 560-576, July 2003.