Sistema de Comunicación para Renderización Paralela de Volúmenes en Tiempo Real

C. F. Perez-Monte[1], M. F. Piccoli[2], C. Luciano[3], S. Rizzi[4]

[1] GridTICs - Universidad Tecnológica Nacional (UTN) - Facultad Reg. Mendoza - Mendoza, Argentina

[2]LIDIC - Universidad Nacional de San Luis (UNSL) - San Luis, Argentina

[3]University of Illinois at Chicago (UIC) - Chicago, IL, USA

[4]Argonne Leadership Computing Facility Argonne National Laboratory (ANL) - Lemont, IL, USA

[1]cristian.perez@gridtics.frm.utn.edu.ar, [2]mpiccoli@unsl.edu.ar, [3]clucia1@uic.edu, [4]srizzi@alcf.anl.gov

Resumen: *Para resolver problemas de gran complejidad como la renderización de volúmenes se utilizan sistemas de cómputo paralelos con una elevada potencia de cómputo. En este trabajo se presenta la implementación de un sistema de comunicaciones orientado al mejor esfuerzo, energéticamente eficiente y tolerante a fallas para resolver la renderización fotorrealista en tiempo real. A fin de conseguir estos objetivos se propone un sistema que utiliza en capa de Transporte UDP, en capa de Red IPv6 (unicast - multicast) y en capa de Enlace de Datos Gigabit Ethernet con control de flujo. Al final exponemos algunos resultados experimentales.*

Palabras Claves: *Volumen, Fotorrealista, IPv6, Ethernet, Tiempo Real.*

Abstract: *High performance parallel computing systems are used to solve complex problems such as volume rendering. This paper presents the implementation of communications system with the following characteristics: best effort, energy efficient and fault-tolerant used to solve real-time photorealistic rendering. In order to achieve these objectives, a system is proposed that used UDP in Transport layer, IPv6 (unicast - multicast) in Network layer and Gigabit Ethernet with flow control in Link layer. Finally we present some experimental results.*

Keywords: *Volume, Photorealistic, IPv6, Ethernet, Real Time.*

INTRODUCCIÓN

La renderización fotorrealista de volúmenes en tiempo real requiere una elevada potencia de cómputo para su ejecución debido a la resolución, velocidad de cuadros y efectos de iluminación requeridos. Esta demanda puede ser resuelta mediante la utilización de un sistema paralelo compuesto por numerosos nodos conectados a través de una red. La aplicación de técnicas de paralelización permite distribuir el procesamiento entre los nodos, usando una red con buen desempeño (performance) para las comunicaciones.

En este trabajo planteamos un modelo de sistema con diferentes roles para los nodos conectados y un modelo de distribución de tareas, constituyendo ambos el punto de partida para el diseño del protocolo de comunicación propuesto. Para lograrlo, primero revisamos los conceptos previos, necesarios para el desarrollo del trabajo. Luego se detallan los aspectos de diseño y de implementación considerados en el modelo del sistema y del protocolo desarrollado. Finalmente, se explican los beneficios logrados mostrando algunos resultados experimentales, las conclusiones y futuros trabajos.

RENDERIZACIÓN EN TIEMPO REAL Y FOTORREALISTA

Un modelo virtual es una representación de un objeto 3D, el cual puede estar descrito ya sea a

través de polígonos, de un volumen u otra técnica. En nuestro caso el modelo es un volumen, el cual constituye una matriz tridimensional compuesto de voxels, cada uno con un nivel de intensidad proporcional a la densidad del tejido orgánico. Renderización es un término frecuentemente utilizado para describir el proceso donde a un modelo 3D se le realiza un procesamiento, según las condiciones de entrada especificada, para generar una imagen 2D desde un punto de vista del observador. Tanto la ubicación del observador como su orientación visual forman parte de las condiciones de entrada más importantes, pudiendo existir otras como la ubicación de la fuente de iluminación. En la renderización en tiempo real, estas condiciones pueden variar constantemente debido al cambio de posición del observador o las luces, implicando una generación continua de imágenes 2D.

La renderización fotorrealista tiene como objetivo generar una imagen, cuya percepción visual es lo más cercana a la realidad. Para lograr este objetivo se usan técnicas de iluminación y sombreado, las cuales requieren un procesamiento elevado.

RENDERIZACIÓN PARALELA DISTRIBUIDA

En renderización paralela distribuida existe una clasificación, la taxonomía de Molnar (Cox, 1995) (Molnar et al., 1994) , la cual básicamente establece dos tipos de técnicas basadas en la manera en cómo distribuye el trabajo entre los diferentes nodos de un sistema distribuido. Estas dos técnicas se conocen con el nombre de Sort first o 2D y Sort last o DB, sus principales características son:

- La renderización paralela distribuida Sort first o 2D propone la división de la pantalla en áreas no solapadas, asignando una a cada nodo o unidad de cómputo disponible. De esta forma, cada nodo renderiza una sección determinada de la pantalla completa. Generalmente los algoritmos de renderización utilizados en este caso son del tipo Image Order, existen algunos desarrollos en sistemas con memoria compartida (Palmer et al., 1998) y en sistemas GPU-CPU con memoria distribuida (Bajaj et al., 2000). Si bien en ambos casos se logra un buen desempeño, la resolución de las imágenes no es muy alta. Finalmente con resolución de imágenes muy superiores, así como también volúmenes de gran tamaño, se logran excelentes resultados (Schwarz and Leigh, 2010).
- En la técnica Sort last o DB, la renderización es hecha dividiendo sectores de volúmenes o primitivas, aplicar la renderización y finalmente realizar la composición de los resultados parciales y obtener la imagen final. Este método es especialmente adecuado para algoritmos de renderizado Object Order para resoluciones de la pantalla de salida no demasiado altas y grandes tamaños de volúmenes (Engel et al., 2006) (Marchesin et al., 2008). Ésta es una técnica altamente escalable con una desventaja importante, la necesidad de interacción entre los diferentes nodos para realizar la composición.

Por último existe, fuera de la taxonomía de Molnar, una tercera técnica de renderización distribuida denominada Alternate Frame Rendering (AFR), muy utilizada en ambientes con varias GPUs en una única PC (Monfort and Grossman, 2009). Esta se basa en la distribución de cuadros de pantalla.

De las tres técnicas antes citadas, 2D y AFR pueden ser aplicadas en nuestro sistema, constituyendo una buena solución para la aplicación en la renderización fotorrealista en tiempo real. Otras técnicas específicas propias de algunos algoritmos de renderización también pueden utilizarse.

Todas estas técnicas pueden combinarse y definir nuevas políticas aplicables a ambientes distribuidos-paralelos para resolver problemas con gran

demanda computacional. El estudio de estas políticas de renderización exceden el alcance de la presente publicación pero el protocolo propuesto está adaptado para funcionar con cualquiera de ellas.

SISTEMA DE RENDERIZACIÓN DISTRIBUIDA EN TIEMPO REAL

El sistema propuesto está compuesto por 3 tipos de nodos, cada uno de los cuales tiene un rol definido. Los roles que puede asumir un nodo en el sistema son:

- Nodo Administrador: Es el nodo que gestiona el procesamiento del sistema al administrar las variables de entrada recibidas desde un dispositivo de entrada para transmitirlas por red a los Nodos Procesadores. Adicionalmente debe anunciar a los Nodos Procesadores qué Nodo o conjunto de Nodos Integradores les corresponde. Opcionalmente puede determinar o no qué procesa cada Nodo Procesador de acuerdo a políticas de implementación.
- Nodo Procesador: Es el nodo que realiza el renderizado y envía el trabajo finalizado por la red al Nodo Integrador. Puede asumir, en el caso de que el Nodo Administrador no lo haga, la función de auto asignarse la tarea a procesar.
- Nodo Integrador: Es el nodo que recibe todas las tareas procesadas y las integra pudiendo ser el encargado también de realizar la visualización en tiempo real de la escena renderizada. Pueden existir múltiples Nodos Integradores pero existe una limitación, cada Nodo Integrador tendrá asignado determinados segmentos de la imagen que no podrán estar asignados simultáneamente a otros Nodos Integradores.

Considerando los distintos tipos de nodos, el sistema de renderización distribuido consta de un Nodo Administrador para coordinar las tareas a realizar por los demás nodos, el cual puede estar separado del resto del sistema en una red alejada permitiendo el control del sistema remotamente; varios Nodos Procesadores, quienes realizan la renderización propiamente dicha, y los Nodos Integradores, responsables de reunir y realizar la visualización en tiempo real del volumen. Estos dos últimos deben estar ubicados físicamente cercanos para su conexión a través de una red de alta performance y baja latencia. En la siguiente sección analizamos el modelo de comunicación propuesto para poder gestionar este sistema.

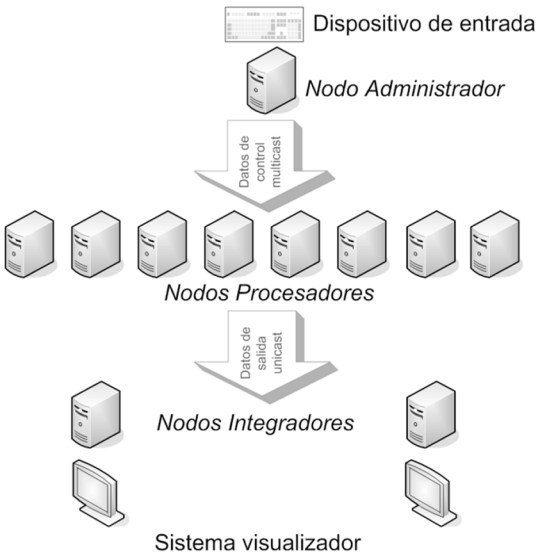

Figura 1. Sistema visualizador.

MODELO DE COMUNICACIÓN

Los nodos en el sistema interaccionan entre sí de una manera específica (ver figura 1), estableciendo de este modo dos tipos de comunicaciones, las cuales son:

- Información de control: Esta comunicación tiene lugar desde el Nodo Administrador a los Nodos Procesadores para enviar la información de las variables de entrada (también denominadas variables de estado del sistema) e

información de pertenencia de los segmentos de tareas que le permite conocer a los Nodos Procesadores a cuáles Nodos Integradores enviar cada segmento. Dado los escasos requisitos necesarios de ancho de banda, esta información puede ser transferida a través de enlaces WAN a redes remotas. La utilización de multicast disminuye los efectos que la latencia de los enlaces WAN pueda ocasionar en el funcionamiento del sistema.

- Información de salida: Esta comunicación se lleva a cabo entre los Nodos Procesadores y los Nodos Integradores. Cada una de ellas contiene la información del flujo de transmisión multimedia distribuido, el cual está compuesto por los sucesivos cuadros en el tiempo generados por cada uno de los Nodos Procesadores. Los requisitos de gran ancho de banda y baja latencia requieren de un enlace LAN exclusivo para la comunicación de salida entre los Nodos Procesadores y los Nodos Integradores.

En las siguientes secciones se detallan las principales características del protocolo y el modo de funcionamiento.

SEGMENTACIÓN DE DATOS

Generalmente en entornos de computación masivamente paralela como el usado en el ambiente de las arquitecturas GPUs (compuesta por grupos de procesadores SIMD) es común usar modelos en los cuales todas las tareas a procesar se dividen en bloques, los cuales son asignados a diferentes procesadores SIMD de la arquitectura (Kirk and Hwu, 2010) . Generalmente es común en sistemas distribuidos contar con nodos con GPU, por ello es importante utilizar un modelo de comunicación el cual mantenga sobre el sistema completo un esquema similar al utilizado por cada nodo. Es por ello que una tarea renderizada (información de salida) que está compuesta por cuadros sucesivos en el tiempo, se puede dividir en segmentos, cada uno de los cuales representa una porción de un cuadro en un momento determinado del tiempo. Denominaremos CS (Compute Segment) a cada una de las estas unidades mínimas de transmisión.

Se puede definir a este CS como una unidad mínima de procesamiento a ser asignada a un Nodo Procesador de la red. Esto es equivalente a los bloques de CUDA o work-group de OpenCL (Unidad mínima que procesamiento que puede asignarse a un Streaming Processor, SM) La diferencia entre ellos está en que un bloque es asignado a un sólo SM de la GPU, en cambio un mismo CS de un mismo cuadro puede ser procesado (de forma diferente o no) en más de un nodo, por lo cual el protocolo debe poder contemplar esta situación. Métodos de renderización iterativos o tolerantes a fallas nodales pueden justificar este modo de funcionamiento. Para los casos de sistemas en los cuales cada Nodo Procesador utiliza su GPU para procesar y conseguir una máxima performance, los CS deberán tener dimensiones múltiplos de los work-group de OpenCL, por ello el protocolo permite que dicho tamaño sea configurable adaptándose a la arquitectura utilizada.

Los Nodos Integradores en cambio sólo pueden ser asignados para recibir CS específicos. Un mismo CS no puede ser recibido por más de un Nodo Integrador. Este tipo de arquitectura permite que los Nodos Integradores, en caso de ser más de uno, realicen también la visualización en forma distribuida.

La segmentación presenta un nuevo paradigma para la computación masiva sin estado interno que denominaremos computación del mejor esfuerzo que guarda similitudes con la entrega del mejor esfuerzo presente en protocolos de comunicaciones de red sin conexión como IP. En el caso de la computación, ésta se realiza sin garantía de sincronismo mientras que en el caso de la comunicación se hace sin garantía de entrega.

ESTADO DEL SISTEMA

El estado completo del sistema es definido por el Nodo Administrador en cada paquete de Información de control, de forma tal que no hay estados internos almacenados en los Nodos Procesadores. Para una correcta gestión y sincronización de la información, cada paquete de Información de control contiene tres diferentes tipos de contadores: el primero varía en función del tiempo, el segundo se incrementa mientras las variables de estado no varíen y se resetea cuando estas cambian y el tercer contador se incrementa cuando las variables de estado se modifican.

Existen básicamente dos estados bien definidos en el sistema, ellos son:

- Estado Estable: Estado en el cual las variables de estado que se envían constantemente como información de control no han variado durante un tiempo lo suficientemente largo permitiendo que todos los Nodos Procesadores posean el mismo estado.
- Estado Inestable: Las variables de estado varían lo suficientemente rápido de forma tal que se asignan diferentes variables de estado a cada uno de los Nodos Procesadores.

MODO DE FUNCIONAMIENTO

El modo de funcionamiento a través del tiempo consta de las varias etapas en donde el estado completo del sistema, compuestos por las variables de entrada y sus correspondiente variables de tiempo, son administradas por el Nodo Administrador. A continuación se describe la función de cada tipo de nodo en el sistema.

Los Nodos Procesadores se anuncian al Nodo Administrador, luego éste envía al grupo multicast de todos los Nodos Procesadores los datos de control (variables de entrada capturadas por un dispositivo de entrada con información del tiempo al momento de la captura). El envío se realiza periódicamente con lapsos de tiempo inferiores al tiempo de refresco del sistema visualizador asegurando así que siempre los Nodos Procesadores tengan tarea para realizar. El Nodo Administrador, también a través de datos de control, le indica a los Nodos Procesadores a cuál Nodo Integrador le corresponde cada CS de pantalla, no pudiendo haber más de un mismo Nodo Integrador asignado a un CS determinado.

Cada Nodo Procesador posee una copia completa del volumen a renderizar. En la actualidad el crecimiento del tamaño de la memoria de video de las placas GPU es suficiente para almacenar una copia completa de casi cualquier volumen médico, además el envío de volúmenes a través de la red reduciría el ancho de banda disponible de la red, por lo cual se considera desaconsejable dada la capacidad de almacenamiento de los nodos en la actualidad. Luego cada Nodo Procesador recibe la información de control, eligiendo la más actual posible y descartando el resto, y a partir de la cual realiza el procesamiento, generando el tráfico con la información de salida, el cual está compuesto por grupos de CS correspondientes a cada cuadro renderizado. Este tipo de comunicación se lleva a cabo mediante envíos unicast a los respectivos Nodos Integrador. El envío de esta información se realiza tan rápido como cada Nodo Procesador es capaz de hacerlo, el límite lo impone el procesamiento o el ancho de banda de la red mediante la cual se conectan los Nodos Procesador con los Nodos Integrador.

Los Nodos Integrador tienen como función reunir el flujo de transmisión multimedia distribuido para finalmente visualizarlo en un sistema compuesto de tantas pantallas como Nodos Integrador existan.

CARACTERÍSTICAS E INTER-OPERATIVIDAD DEL PROTOCOLO

El sistema de comunicación diseñado tiene las siguientes características:

- Mejor esfuerzo: El protocolo, al igual que el sistema en su conjunto, aplica la filosofía del "mejor esfuerzo computacional". Esto significa que ante limitaciones reales (ancho de banda de enlace, Nodos Procesador con diferente potencia de cómputo, nodos ocupados, etc.), el sistema lleva a cabo la tarea aprovechando los recursos y obteniendo los mejores resultados posibles. Razón por la cual el protocolo IPv6-UDP es orientado a datagramas, siguiendo un modelo de entrega con la misma filosofía. Así como el protocolo de comunicación es orientado a no-conexión y sin garantía de entrega, la computación es sin estado interno y sin garantía de sincronismo.
- Tolerante a saturación de enlace: Relacionado con el concepto anterior, el protocolo es capaz de aprovechar al máximo el ancho de banda de red y ante su saturación, el protocolo es capaz de continuar funcionando normalmente. La pérdida de información no repercute en el sistema ya que cada datagrama incluye toda la información necesaria sobre el estado del sistema.
- Energéticamente eficiente: El procesamiento de información tiene un considerable consumo de energía, principalmente en los Nodos Procesador. Como el sistema de comunicación realiza el máximo esfuerzo por transportar los datos de salida, no siempre es posible lograrlo, resultando conveniente procesar sólo la cantidad de datos que el enlace entre los Nodos Procesador y los Nodos Integrador sea capaz de transportar.
- Económicamente eficiente: En cuanto a la posibilidad de aprovechar hardware de diferentes generaciones, logrando su integración en sistemas heterogéneos.
- Tolerante a fallas de Nodos Procesador: Cuando la cantidad de Nodos Procesador crece, las posibilidades de fallo de alguno de ellos se incrementa, razón por la cual ante ciertas políticas de renderización, el protocolo está preparado para tolerar la falla de un nodo y continuar trabajando.
- Jerárquico: Siguiendo metodología de encabezados opcionales luego de capa de transporte y acompañando los lenguajes de computación de procesamiento universal para plataformas heterogéneas, se utiliza una jerarquía compatible con los niveles de abstracción de software de las mismas.

En las próximas secciones se demuestra la capacidad del sistema para cumplir con estas características.

ARQUITECTURA DE LA PILA DE PROTOCOLOS

La arquitectura del stack de protocolos que se utiliza para cumplir con los objetivos antes mencionados es la siguiente:

- Capa de enlace de datos: En el caso de los enlaces entre los Nodos Procesador y Nodos Integrador se utiliza el protocolo Ethernet, con control de flujo (opcional) (IEEE Standards for Local and Metropolitan Area Networks, 1997). Si se tiene control de flujo se permite, ante saturación del enlace, a los Nodos Procesador sólo procesar aquello que son capaces de enviar por la red, reduciendo el consumo energético.
- Capa de red: Aplica el protocolo IPv6 (Deering and Hinden, 1998) (el estándar de red de alcance mundial en su versión más reciente), siendo adecuado para computación distribuida. Además usa multicast para reducir el tráfico de información de control.
- Capa de transporte; Protocolo UDP (Postel,

1980). Implementa un protocolo con modelo de entrega de mejor esfuerzo y tolerante a la saturación del enlace.

La arquitectura propuesta está diseñada utilizando protocolos de gran difusión permitiendo su implementación en sistemas heterogéneos de bajo costo habituales en entornos de investigación universitaria. La utilización de protocolos de red de capa de enlace de datos como Infiniband o 10 Gbit Ethernet pueden ser utilizados en un futuro cuando sean de menor costo y mayor difusión.

DESCRIPCIÓN DETALLADA DEL PROTOCOLO

En el protocolo se distinguen dos tipos de comunicación, las cuales:

- Información de Control Multicast:

Está constituida por datagramas que tienen como función principal el envío de los datos de entrada. Incluyen 3 cabeceras, de las cuales dos son opcionales para el envío de los datos. Ellas son:

- Cabecera de control (CCm): Es obligatoria, contiene la información de cada CS de control.
- Cabecera de control (CCGo): Es opcional de índole general. Se usa para anunciar a los Nodos Procesador. Esta información puede referirse a los propios Nodos Procesador (qué CS debe procesar cada uno) o sobre los Nodos Integrador (qué CS tienen asignado cada Nodo Integrador). En CCGo se indica cuántas cabeceras CCEo hay.
- Cabecera de control específica por nodo (CCEo): Es opcional con características específicas, contiene información específica para cada Nodo Procesador o Nodo Integrador.

El datagrama de la información de control se muestra en la tabla 1.

- Información de Salida Unicast:

Como esta información se divide en CS, cada uno de ellos tienen, además, 3 cabeceras compuestas de varios campos, estos son:

- Cabecera de información general del sistema (CIGS): Esta cabecera contiene una descripción general de las variables del sistema completo. Dicha información se repite en cada CS permitiendo la tolerancia a fallas de nodos o de red.
- Cabecera de información general del cuadro (CIGC): La información enviada en cada CS, generalmente no varía de CS a CS de un mismo cuadro.
- Cabecera específica de CS (CES): Es información específica de cada CS.

Los requisitos del sistema incluyen entre otros que todos los cuadros del flujo de transmisión de salida tengan la misma resolución y que todos los segmentos de un mismo cuadro tengan el mismo tamaño.

El datagrama de la información de salida se muestra en la tabla 2.

REQUISITOS DE ANCHO DE BANDA DE RED TEÓRICOS Y LATENCIA MÁXIMA DEL SISTEMA

En la presente sección se determinan los requisitos mínimos de ancho de banda del sistema para que la red no sea limitante del rendimiento y la latencia que existe desde el momento que una variable de entrada es ingresada por el dispositivo de entrada hasta que la misma se hace efectiva en el sistema visualizador. Por último el rendimiento que poseerá el sistema en caso de que la red sea limitante del rendimiento.

Siendo:

m : Número de Nodos Procesador.

l : Número de Nodos Integrador.

R_x : Resolución horizontal.

R_y : Resolución vertical.

B_s : Cantidad de bytes por cada pixel.

	CAMPOS	DESCRIPCIÓN
CABECERA IPV6 + CABECERAS OPCIONES	DEFINIDOS POR EL PROTOCOLO	DEFINIDOS POR EL PROTOCOLO SEGÚN RFC 2460
	DIRECCIÓN IPV6 DE DESTINO	DIRECCIÓN IPV6 DE DESTINO MULTICAST DEL GRUPO DE LOS NODOS PROCESADOR SEGÚN RFC 3306
CABECERA UDP	DEFINIDOS POR EL PROTOCOLO	DEFINIDOS POR EL PROTOCOLO SEGÚN RFC 768
CCM	INPUT VAR TYPE	DESCRIPTOR DE LAS VARIABLES DE ENTRADA
	TIME STAMP	CAMPO DE MARCA DE TIEMPO DE VARIABLE DE ENTRADA
	CONTROL COUNTER	CAMPO DE NÚMERO DE DATO DE CONTROL ENVIADO
	NO VAR COUNTER	CAMPO DE NÚMERO DE DATO DE CONTROL ENVIADO CON IDÉNTICA VARIABLE DE ENTRADA
	NEXT HEADER	SIGUIENTE CABECERA
CCGo	NODE TYPE	TIPO DE NODO (PROCESADOR O INTEGRADOR)
	NODE COUNTER	CANTIDAD DE NODOS
	NODE INFO	TIPO Y CANTIDAD DE INFORMACIÓN DEL NODO
	NEXT HEADER	SIGUIENTE CABECERA
CCEo	ID NODE	IDENTIFICADOR ÚNICO DE NODO: 64 BITS DE MENOR PESO DE DIRECCIÓN IPV6 DE NODO PROCESADOR O INTEGRADOR
	SEGMENT INFO	CAMPO DE INFORMACIÓN ESPECÍFICA PARA NODO: MAPA DE BITS DE INFORMACIÓN DE CS ASIGNADOS PARA RECIBIR POR PARTE DE NODO INTEGRADOR O PARA PROCESAR POR PARTE DE NODO PROCESADOR
	NEXT HEADER	SIGUIENTE CABECERA
CUERPO	-	DATOS DE VARIABLES DE ENTRADA

Tabla 1. Información de control multicast.

R : Rendimiento en cantidad de cuadros por segundo.

BW : Ancho de banda de red.

T_s : Latencia de renderizado.

T_{BW}: Tiempo de transferencia del cuadro.

T_n : Latencia de red.

T_{nap} : Latencia de red entre Nodo Administrador y Nodo Procesador.

T_{npi} : Latencia de red entre Nodo Procesador y Nodo Integrador.

T_1 : Latencia de interactividad total.

El caso extremo de ancho de banda utilizado por el canal de comunicaciones en la información de salida estará dado en su funcionamiento con la técnica AFR o políticas que la utilicen y será:

$$BW = \frac{R_x \times R_y \times B_s \times R \times m}{l}$$

La latencia total del sistema estará comprendida por la suma de la latencia en la renderización de cada cuadro de la pantalla sumada a la latencia de la red y al tiempo de transferencia del cuadro de pantalla en la red. El tiempo de latencia de red está determinado por la suma de la latencia Nodo Administrador – Nodo Procesador y Nodo Procesador – Nodo Integrador, en la cual esta última, por ser pequeña, puede despreciarse. El tiempo de transferencia del cuadro en la red puede determinarse según el ancho de banda de la red usada mientras que el tiempo de renderizado depende del hardware de los Nodos Procesador.

$$T_t = T_s + T_{BW} + T_n = T_s + T_{BW} + T_{npi} + T_{nap}$$

$$T_t = T_s + T_{BW} + T_{nap} = T_s + \frac{R_x \times R_y \times B_s}{l \times BW} + T_{nap}$$

Si bien la información de salida puede transferirse a través de redes de baja velocidad, el ancho de banda en este caso será un limitante en el rendimiento máximo del sistema expresado en cuadros por segundo (R), siendo este:

$$R = \frac{BW \times l}{m \times R_x \times R_y \times B_s}$$

Los requisitos antes expuestos no son limitantes del funcionamiento del sistema, sino umbrales a partir del cual el sistema estará limitado por un recurso (potencia computacional) u otro recurso (ancho de banda de red).

	CAMPOS	DESCRIPCIÓN
CABECERA IPv6 + CABECERAS OPCIONES	DEFINIDOS POR EL PROTOCOLO	DEFINIDOS POR EL PROTOCOLO SEGÚN RFC 2460
	DIRECCIÓN IPV6 DE DESTINO	DIRECCIÓN IPV6 DE DESTINO UNICAST DE NODO INTEGRADOR CORRESPONDIENTE AL CS TRANSMITIDO.
CABECERA UDP	DEFINIDOS POR EL PROTOCOLO	DEFINIDOS POR EL PROTOCOLO SEGÚN RFC 768
CIGS	FRAME DIM	CAMPO DE CANTIDAD DE DIMENSIONES DEL FLUJO DE TRANSMISIÓN DE SALIDA (GENERALMENTE ES 2)
	FRAME RESOLUTION	CAMPOS DE RESOLUCIÓN TOTAL DEL CUADRO EN CADA DIMENSIÓN
	PIXEL TYPE	CAMPO DE CANTIDAD DE BIT POR PIXEL Y TIPO DE REPRESENTACIÓN
	NEXT HEADER	SIGUIENTE CABECERA
CIGC	SEGMENT DIM	CANTIDAD DE DIMENSIONES DEL CS
	SEGMENT RESOLUTION	TAMAÑO DEL CS EN CADA DIMENSIÓN
	TOTAL SEGMENT	CANTIDAD TOTAL DE CS DEL CUADRO
	NEXT HEADER	SIGUIENTE CABECERA
CES	ID SEGMENT	IDENTIFICADOR DE CS
	REPRESENTATION SEGMENT	REPRESENTACIÓN DEL CS. CAMPO DE CS COMPRIMIDO O NO. COMPRESIÓN CON PÉRDIDA O NO DE CALIDAD. TIPO DE COMPRESIÓN
	SEGMENT SIZE	TAMAÑO COMPRIMIDO DEL CS
	TIME STAMP	CAMPO DE MARCA DE TIEMPO DE VARIABLE DE ENTRADA: CADA CS TENDRÁ MARCAS DE TIEMPO QUE LO IDENTIFIQUEN COMO PERTENECIENTE A DETERMINADA VARIABLE DE ENTRADA Y POR LO TANTO PERTENECIENTE A DETERMINADO CUADRO EN EL TIEMPO
	SEGMENT COUNTER	CAMPO DE NÚMERO DE CS GENERADO CON IGUAL IDENTIFICADOR DE CS
	RENDER TYPE	TIPO DE MÉTODO USADO PARA LA RENDERIZACIÓN
	ITERATION COUNTER	SEGÚN CORRESPONDA NIVEL DE PROCESAMIENTO EJERCIDO COMO CANTIDAD DE ITERACIONES DEL CS
	NO VAR COUNTER	CAMPO DE NÚMERO DE CS PROCESADO CON IDÉNTICA MARCA DE TIEMPO E IDENTIFICADOR D E CS ÚTIL PARA POLÍTICAS ESPECÍFICAS DE RENDERIZADO
	NEXT HEADER	SIGUIENTE CABECERA
CUERPO	CS	DATOS DEL SEGMENTO

Tabla 2. Información de salida unicast.

TRÁFICO DE SALIDA DESPERDICIADO

Se le denomina tráfico desperdiciado al tráfico generado por los Nodos Procesador, el cual no puede ser procesado por los Nodos Integrador.

Existen básicamente dos tipos de tráfico desperdiciado:

- Tráfico desperdiciado por Nodo Integrador: Este tráfico generalmente es por falta de sincronización. Como los Nodos Procesador trabajan en un modelo asincrónico de mejor esfuerzo, puede ocurrir envíen CS, los cuales son recibidos tardíamente. En tal caso, los Nodos Integrador los descartan. Ocurre lo mismo si los buffers de recepción del Nodo Integrador no tienen el tamaño suficiente o trabaja a una velocidad reducida.
- Tráfico desperdiciado por pérdidas de enlace: Es todo aquel tráfico generado por los Nodos Procesador y no recibido por los Nodos Integrador. Un caso donde ocurre esto es cuando se produce la saturación del enlace en un medio sin control del flujo, por lo cual la pérdida de una trama que contiene un CS o parte del mismo hará que dicho CS no pueda ser reconstruido en el Nodo Integrador.

Si bien el sistema es capaz de funcionar con tráfico desperdiciando, el mismo genera un desperdicio de potencia computacional de los Nodos Procesador por lo cual es de suma importancia su reducción.

RESULTADOS

Los resultados experimentales se realizaron en un cluster de computación gráfica de renderizado distribuido compuesto por 2 Nodos Procesadores, un Nodo Administrador y un Nodo Integrador conectados mediante una red Ethernet Gigabit. Se utilizó un MTU estándar de 1528 bytes. El tamaño del CS incluido en la carga del datagrama UDP fue definido en 48 Kbytes.

Se realizaron modificaciones en parámetros de kernel en el Nodo Integrador para incrementar los buffers de recepción evitando Tráfico desperdiciado por Nodo Integrador.

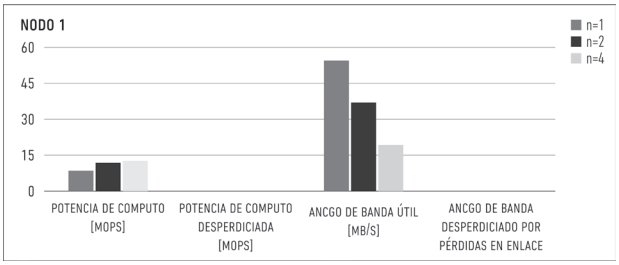

Fig. 2. Potencia de cómputo y Ancho de banda de Nodo 1.

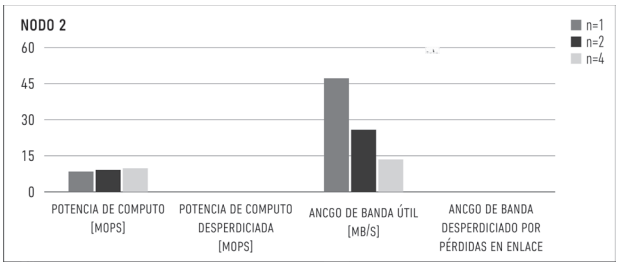

Fig. 3. Potencia de cómputo y Ancho de banda de Nodo 2.

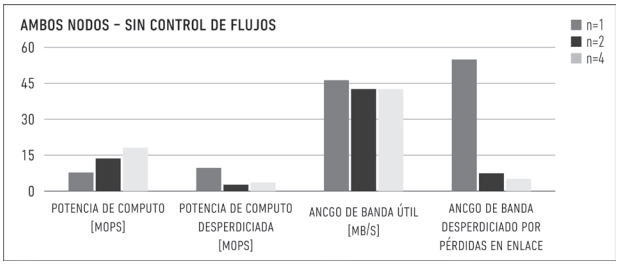

Fig. 4. Potencia de cómputo y Ancho de banda de Ambos nodos sin control de flujo.

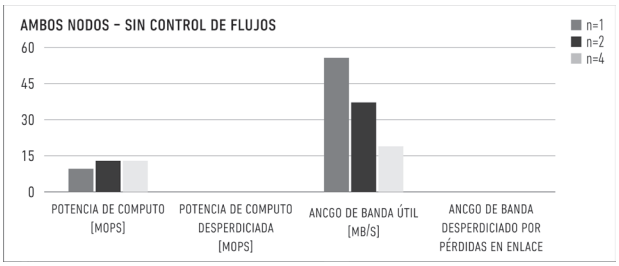

Fig. 5. Potencia de cómputo y Ancho de banda de Ambos nodos con control de flujo.

Las características de la arquitectura de cada uno de los Nodos Procesadores del sistema son:

- Nodo Procesador 1: Intel Core 2 Duo 4 GB de RAM GPU GTX560 1GB de RAM.
- Nodo Procesador 2: AMD Fx6200 4 GB de RAM GPU GTX650 1 GB de RAM.

La elección de estos nodos fue con la intención de demostrar el funcionamiento con hardware CPU de diferentes fabricantes y hardware GPU de diferente generación.

Las figuras 2 y 3 muestran respectivamente las potencias de cómputo y Ancho de banda de los nodos 1 y 2.

Las figuras 4 y 5 muestra la potencia de cómputo combinada y anchos de banda medidos sobre el Nodo Integrador con diferentes composiciones del sistema, valor de n y control de flujo en Ethernet activado o no.

n es la cantidad de iteraciones del método de renderizado, la cual permite incrementar el nivel del realismo en forma previa al envío de cada cuadro y la potencia de cómputo de combinada de todos los Nodos Procesadores intervinientes se presenta en MOPS (millones de operaciones /segundo).

La potencia de cómputo es definida como la cantidad de pixels procesados por la cantidad n de operaciones realizadas por cada uno de ellos por unidad de tiempo. Dicho parámetro es más objetivo que cuadros por segundo (FPS) a la hora de medir el rendimiento de los Nodos Procesadores.

En primer lugar se de la observación de los resultados de las figuras 2 y 3 es posible determinar que los nodos 1 y 2 en funcionamiento independiente no poseen pérdidas de potencia de cómputo ni tráfico.

En segundo lugar, a partir de la figura 4 es posible determinar que sin control de flujo existe un ancho de banda desperdiciado por pérdidas de enlace lo que repercute en Potencia de cómputo desperdiciada.

Por último, es posible determinar que en estado estable, según se observa en la figura 5, con la configuración con control de flujo activada no hay pérdidas de CS y, por lo tanto, no existe desperdicio en potencia de cómputo, siendo así, toda la energía utilizada en el procesamiento de Nodos Procesadores aprovechada. También se observa que la potencia de cómputo global está limitada en el caso de n=1 por el ancho de banda de red mientras que para n superiores está limitada por la potencia

computacional de los mismos. Así se demuestra su funcionamiento de computación de mejor esfuerzo.

En los casos de saturación de enlace, para n=1 y con funcionamiento de los dos nodos en simultáneo, se pudo comprobar una reducción:

En la velocidad del flujo de transmisión con respecto a la suma del tráfico de los dos nodos separados.

En la potencia combinada con respecto a la potencia de cómputo de los dos nodos separados.

Si bien en los resultados no ha podido mostrarse debido a que no es un parámetro cuantitativo, se puede señalar que en estado inestable con fuertes variaciones del estado del sistema se pudieron observar pérdidas de CS por Tráfico desperdiciado por pérdidas en el enlace, las cuales en todos los casos fueron inferiores al 2% del tráfico total

Si bien la limitación de la potencia de cómputo combinada la impone el ancho de banda de la red utilizada, no hay desperdicio de energía. El ancho de banda utilizado es muy cercano al máximo de la red por lo cual se puede observar un mejor esfuerzo por conseguir el mayor rendimiento posible.

En las pruebas en las cuales se ha desactivado el control de flujo de ethernet, se puede observar que en los casos de saturación de enlace existe una pérdida considerable de tráfico. Esto es debido a que existe una gran fragmentación del tráfico IPv6 en tramas ethernet, ya que el MTU es mucho menor al tamaño del paquete IPv6, por lo cual, la pérdida de una única trama ethernet generará la pérdida de todo un CS. Incrementar el MTU o decrementar el tamaño del CS son posibles soluciones en estos casos.

En estas últimas pruebas, si bien el sistema funcionó con menor performance debido a la gran pérdida de tramas, mostró tolerancia a falla de CS, simulando además un ambiente similar al caso en el cual hay fallas de Nodos Procesador, pudiendo demostrarse su funcionamiento con tolerancia a fallas de nodos procesador y saturación de enlace.

CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo planteamos un protocolo, el cual fue diseñado para un sistema de renderizado distribuido/paralelo en tiempo real. El protocolo cumple con los requisitos necesarios para el control y transmisión de flujo de transmisión multimedia distribuido. Del análisis de los resultados derivados de la segmentación de datos, se pudo observar que la técnica usada es apropiada tanto para la transmisión de datos en tiempo real como para distribuir las tareas de procesamiento y de visualización. Además el protocolo propuesto no sólo logra conseguir el mejor esfuerzo, sino también ser tolerante tanto a saturación de red como fallas de los Nodos Procesador.

Entre las posibles extensiones de este trabajo se encuentra el análisis del comportamiento cuanto la conexión de los nodos es a través de Internet, por ejemplo el Nodo Administrador está ubicado en redes geográficamente alejadas de la red en donde se ubican los Nodos Procesador y los Nodos Integrador. Además se analizará la posibilidad de generalizar el protocolo de manera de poder realizar la gestión remota de cualquier tipo de procesamiento masivamente paralelo en aplicaciones de tiempo real.

AGRADECIMIENTOS

Se agradece el asesoramiento y soporte con infraestructura de los integrantes de los Grupos de Investigación GridTICs y LICPaD de UTN-FRM y LIDIC de UNSL.

El trabajo es sostenido económicamente gracias al financiamiento de los proyectos 25J084 “SARA Operation”, PICT2010/29 “Procesamiento para visualización utilizando algoritmos paralelos en GPU y distribuidos en red” ambos de UTN-FRM, PROICO-30310 de UNSL y otras fuentes de financiamiento de grupo GridTICs.

Este proyecto ha sido realizado gracias al aporte

económico de UTN a través de una beca de doctorado y al soporte académico de la carrera de posgrado de UNSL.

REFERENCIAS

Cox M."Algorithms for Parallel Rendering". PhD thesis, Department of Computer Science, Princeton University (1995).

Molnar S., Cox M., Ellsworth D., Fuchs H."A Sorting Classification of Parallel Rendering." IEEE Computer Graphics and Algorithms. Pp 23-32 (1994).

Palmer M., Totty B., Taylor S. "Ray casting on shared-memory architectures: Memoryhierarchy considerations in volume rendering". IEEE Concurrency, Vol 6, nro 1. Pp 20–35 (1998).

Bajaj C., Ihm I., Park S., Song D.. "Compression-based ray casting of very large volume data in distributed environments." In HPC '00: Proceedings of the The Fourth International Conference on High-Performance Computing in the Asia-Pacific Region-Volume 2. Pp 720–725 (2000).

Schwarz N., Leigh J.,"Distributed Volume Rendering for Scalable High-resolution Display Arrays", Grapp – International Conference on Computer Graphics Theory and Applications (2010).

Engel K., Hadwiger M., Kniss J. M., Rezk-Salama C., Weiskopf D. "Real-Time Volume Graphics." A K Peters, Ltd (2006).

Marchesin S., Mongenet C., Dischler J. "Multi-gpu sort-last volume visualization". In Eurographics Symposium on Parallel Graphics and Visualization (EGPGV08) (2008).

Monfort J., Grossman M."Scaling of 3D Game Engine Workloads on Modern Multi-GPU Systems" - The 1st ACM Conference on High Performance Graphics (HPG-09) (2009).

Kirk D., Hwu W. "Programming Massively Parallel Processors, A Hands on Approach" , Elsevier, Morgan Kaufmann (2010).

IEEE Standards for Local and Metropolitan Area Networks: "Supplements to Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Specification for 802.3 Full Duplex Operation and Physical Layer Specification for 100 Mb/s Operation on Two Pairs of Category 3 Or Better Balanced Twisted Pair Cable (100BASE-T2)," IEEE Std 802.3x-1997 and IEEE Std 802.3y-1997 (Supplement to ISO/IEC 8802-3: 1996; ANSI/IEEE Std 802.3, 1996 Edition) (1997).

Deering S., Hinden R."Internet Protocol, Version 6 (Ipv6) Specification" IETF Standards Track (1998).

Postel J., "User Datagram Protocol" IETF Standard protocol (1980).

Fecha de Presentación: 15/05/15
Fecha de Aprobación: 08/06/15