Restrepo, Alex; Montoya, Alejandro; Trefftz, Helmuth

Asignación dinámica de servidores en ambientes virtuales, usando "octrees" para la partición dinámica del espacio

Avances en Sistemas e Informática, vol. 6, núm. 2, septiembre, 2009, pp. 1-4
Universidad Nacional de Colombia
Colombia

Avances en Sistemas e Informática
ISSN (Versión impresa): 1657-7663
mprada@unalmed.edu.co;avances@unalmed.edu.co
Universidad Nacional de Colombia
Colombia

Asignación dinámica de servidores en ambientes virtuales, usando "octrees" para la partición dinámica del espacio

Dynamic server allocation in virtual environments, using octrees for dynamic space partition

Alex Restrepo[1] Ing. Alejandro Montoya[2] Ing. & Helmuth Trefftz[3] PhD.
1. Senior Software Engineer Brainchild
2. Asistente Investigación Universidad Eafit
3. Profesor Universidad Eafit
Medellín, Colombia
alexrestrepo@mac.com; alejomontoya@gmail.com; htrefftz@eafit.edu.co

Recibido para revisión 29 de Septiembre de 2008, aceptado 25 de Agosto de 2009, versión final 14 de Septiembre de 2009

Resumen— **Uno de los problemas a los cuales recientemente se ha dado atención en las aplicaciones de ambientes virtuales, es la forma de asignar o distribuir recursos computacionales a los participantes del sistema. En la mayoría de los casos se usan asignaciones estáticas de los recursos, lo cual, a pesar de ser sencillo, presenta una serie de limitantes, entre estas esta la subutilización de los recursos.**
Con la presente investigación pretendemos hacer una prueba de concepto, la cual consiste en demostrar que usando un algoritmo de partición de espacio como los "octrees", y un algoritmo de balanceo de la asignación de los clientes que maneja un servidor, pueden asignarse recursos de una forma dinámica y con esto mejorar la distribución de los participantes en un ambiente virtual distribuido.

Palabras Clave— **Asignación Dinámica de Servidores, "Octrees", Ambientes Virtuales Distribuidos, Balanceo de Carga.**

Abstract— **One of the challenges to which has recently been paid attention in the application of virtual environments is how to assign or distribute computational resources among the participants of the system. In most cases it is used static resources allocations, which, despite being simple, has a number of constraints, including the resources underutilization. In this research, we attempt to make a proof of concept, which consists of demonstrating that using a space portioning algorithm e.g. "octrees", and an algorithm for balancing the allocation of the clients handled by the server, may be allocated the resources in a dynamic way, and with this, improve the distribution of the participants in a distributed virtual environment.**

Keywords— **Dynamic allocation of Servers, "Octrees", Distributed Virtual Environments, Load Balancing.**

I.INTRODUCCIÓN

En un ambiente virtual es necesario tener y compartir un registro de los participantes. Generalmente un servidor tiene asignado para su control una parte estática de espacio virtual, la cual puede estar vacía o con usuarios. En esta investigación tratamos de reducir el número de "trozos" virtuales vacíos, pues cuando una partición esta vacía, no es necesario dedicarle un servidor o gastar tiempo de procesamiento para controlarla. Los servidores libres pueden ser asignados a espacios que no estén vacíos. Las ventajas de usar un algoritmo de partición dinámica es que se utiliza de una mejor manera los recursos. Adicionalmente se demostró que de esta manera el ambiente virtual, bajo ciertas condiciones, puede llegar a ser más escalable.

II.TRABAJO PREVIO

En un trabajo previo, los autores [6] propusimos una implementación del problema en dos dimensiones usando "quadtrees", esta forma de asignación demostró ser más poderosa en el uso de los recursos que un método estático. Con esta propuesta se hace una mejor utilización del sistema representando en promedio una mejora del 24% con respecto a un modelo estático, la investigación presentada en este documento es una extensión a esta solución.

Funkhouser presenta varias tipologías para el uso en ambientes virtuales [3]. Sin embargo, estas tipologías solo varían el esquema a seguir al momento de montar un ambiente virtual; no presenta una forma de modificar dinámicamente la asignación

de los servidores en función del espacio virtual y a las entidades participantes. Así mismo, Macedonia [5] presenta un conjunto de factores que afectan el desempeño de un ambiente virtual, entre los cuales podemos contar la latencia, ancho de banda, confiabilidad, y cómo un buen desempeño es vital para el correcto funcionamiento de un ambiente virtual.

Otro factor interesante en el análisis de un ambiente virtual es el de balanceo de cargas ("Load Balancín") en donde la cantidad de trabajo que el computador tiene que hacer se reparte en dos o más computadores, de forma general todos los usuarios obtienen una mejor respuesta [1].

Srisawat y Alexandridis proponen un sistema para la asignación dinámica de capacidad de procesamiento en máquinas en paralelo conectadas en forma de malla, por medio de "quadtrees" [8], en su modelo, procesos son asignados y desasignados con búsquedas usando "quadtrees", obteniendo una mejora significativa en el porcentaje de asignación, en comparación con métodos estáticos.

En el campo de la realidad virtual, se ha encontrado que la asignación del espacio de acción de cada servidor se hace de forma estática, existiendo diferentes formas de partición, donde las mas conocidas son aquellas realizadas por medio de mallas rectangulares y hexagonales [2], en un trabajo previo nuestro se propone una implementación dinámica para 2 dimensiones [6], sin embargo, la mayoría de ambientes virtuales actuales son mundos tridimensionales, por lo que esta investigación es más aplicable y tiene una serie de ventajas que se explican en el transcurso de los experimentos.

III. PARTICIÓN DINÁMICA BASADA EN "OCTREES"

La partición del espacio usando "ocurres", es un algoritmo recursivo que funciona encapsulando todo el espacio virtual posible del problema en un nodo raíz; a medida que a esté nodo entran entidades, el algoritmo evalúa si es necesario partirlo en 8 subnodos de igual tamaño, dependiendo de la información contenida en él. El proceso se repite recursivamente y al final se obtiene una estructura tipo árbol, la cual permite acceder a los contenidos del espacio encapsulado de una manera rápida y eficiente [4], así mismo nos ayuda a determinar cuáles porciones de espacio están siendo usadas y cuales no, es por esto que el algoritmo "octree" sobresale como una solución natural al problema de asignación dinámica de espacio.

Es de notar que, en un ambiente virtual compartido, es más probable que se establezca comunicación entre entidades que están más cercanas, por lo tanto la partición espacial es útil. En el caso de un "octree", podemos asignar un nodo a aquellos usuarios que estén situados relativamente cerca en el espacio virtual.

En un ambiente con asignación estática, todos los servidores (existe un servidor por cada nodo) están asignados pero hay algunos sin entidades adentro, esto significa que están sin trabajo. Por el contrario en un ambiente con la partición dinámica no todos los servidores están asignados, por lo que habrá servidores disponibles para otras tareas. Aquellos usuarios contenidos en un mismo nodo, serán asignados al mismo servidor, por lo que reduce la necesidad de comunicarse con entidades fuera de su servidor asignado, disminuyendo el número de mensajes que pasa entre servidores.

En nuestro caso, se crea un "octree" que encapsula todo el espacio virtual disponible, y que inicialmente contiene un solo nodo. A medida que los participantes ingresan al espacio y luego de sobrepasar el máximo de participantes definido por nodo, este será dividido en 8 subnodos iguales, de los cuales algunos pueden quedar vacíos, estos nodos son liberados para uso posterior, el proceso se repite de forma recursiva.

IV. EL EXPERIMENTO

Para poder probar la validez del modelo se simuló un ambiente virtual cerrado, una pecera, donde había tres tipos de entidades con comportamientos diferentes que interactuaron. El modelo fue implementado de forma estática y dinámica, para realizar las comparaciones pertinentes.

El algoritmo usado por el modelo estático consiste en dividir el espacio en nodos de igual tamaño y de ubicación fija, cada uno de estos nodos representa un servidor asignado. En el modelo propuesto, el cual es dinámico, se usa el algoritmo "octree" para dividir el espacio, los nodos resultantes cambian tanto en tamaño como en ubicación, de acuerdo con la evaluación de la cantidad de trabajo asignado a cada uno, la cual está relacionada con el número de entidades dentro de cada nodo. En esta simulación la partición dinámica se reevaluó cada 1/30 segundos, y se uso un máximo de 64 servidores.

Para el experimento, los agentes tuvieron las siguientes características:

Peces: Existen dos especies, ambas con las mismas características, son agentes de baja velocidad que nadan libremente en el sistema, al encontrarse dos o más peces en el mismo nodo y de la misma especie, seguirán su recorrido en la misma dirección usando el algoritmo conocido como Boids "Simulated flocking" [7]. Si son de diferente especie simplemente se ignoran. En el momento de encontrar en su nodo un agente de otro tipo (hostiles) emprenderá huida en dirección contraria.

Tiburones: Agentes capaces de lograr una alta velocidad, nadan libremente en el sistema. De encontrar en su nodo un agente de otro tipo sale en busca de este hasta atraparlo y comerlo, siempre y cuando este dentro de su rango de visión, el cual esta demarcado por una constante definida.

La figura 1 muestra el experimento realizado, donde los entes azules y los entes rojos son los peces y los entes grises los tiburones. Adicional estamos representando el rango de visión de los tiburones con una esfera alrededor de cada uno

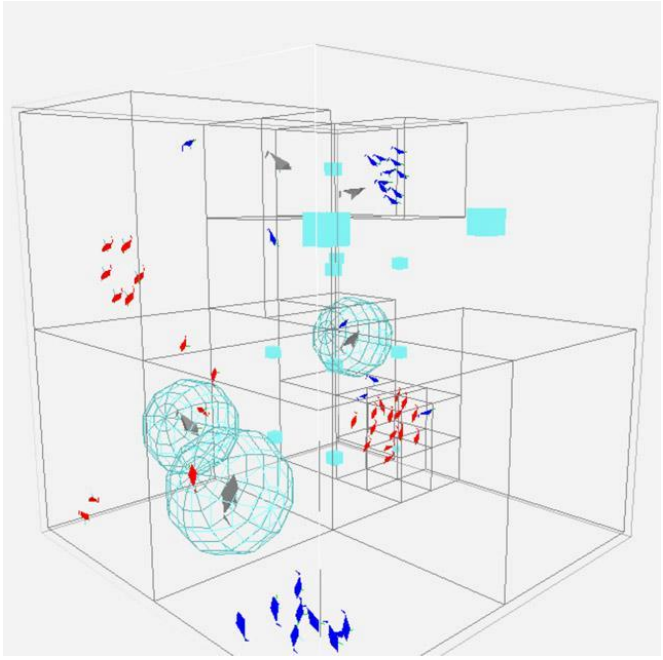

Figura 1. Simulación

Con este comportamiento se buscó probar la suposición que el modelo propuesto es mejor y definir bajo qué condiciones lo es, ya que se asegura que las entidades estarán cambiando de nodos a medida que deambulan por el sistema. Además, con la característica de los peces de nadar en cardumen, se asegura que estarán cerca varias entidades y así la partición del espacio virtual se ejecutará.

De la simulación, se recopiló la siguiente información:

a. Número de particiones por unidad de tiempo

Con el número de particiones por unidad de tiempo, se demostró, que en el modelo propuesto no siempre son asignados todos los servidores. Por el contrario en la partición estática del espacio virtual, el total de servidores esta asignado todo el tiempo.

b. Número de mensajes entre servidores por unidad de tiempo

Cada vez que una entidad cambia de nodo, el servidor que tenía asignado el nodo al que pertenecía, debe enviar un mensaje al servidor que tiene asignado el nodo al que se dirige. Este proceso es igual para ambos modelos, estático y dinámico.

Esta medida sirvió para probar si el modelo que se plantea en este trabajo reduce el tráfico de mensajes.

c. Factor de optimización

El problema principal que se desea solucionar es la optimización en el uso de los recursos, es decir que los servidores asignados si estén ocupados.

El factor de optimización consiste en el número de servidores utilizados dividido por el número de servidores asignados, está dado en porcentaje y es mejor en la medida que se acerca a 100%.

Es de notar que las entidades tienen un tamaño relativamente alto en proporción a su espacio virtual, haciendo que pasen de un nodo a otro más frecuentemente, lo que genera un número mayor de mensajes. Los resultados demuestran que nuestro modelo hace una mejor utilización de los recursos y disminuye en forma significativa el número de mensajes en la red, los resultados se mostraran en la siguiente sección.

V. RESULTADOS

5.1 Mensajes por Segundo

En la figura 2 se observa solo una diferencia considerable en el inicio del experimento, esto ocurre dado que inicialmente para el modelo dinámico las entidades están más dispersas por el ambiente, por lo que las particiones son de mayor tamaño causando que una entidad demore más tiempo en cambiar de un nodo a otro disminuyendo así el número de mensajes, a medida que el comportamiento de moverse en cardumen de las entidades va evolucionando, se nivela la cantidad de mensajes por segundo entre ambos modelos, dinámico y estático, sin embargo la generación de mensajes es proporcional, en promedio se generaron 123 mensajes por segundo en el modelo dinámico y 149 en el modelo estático.

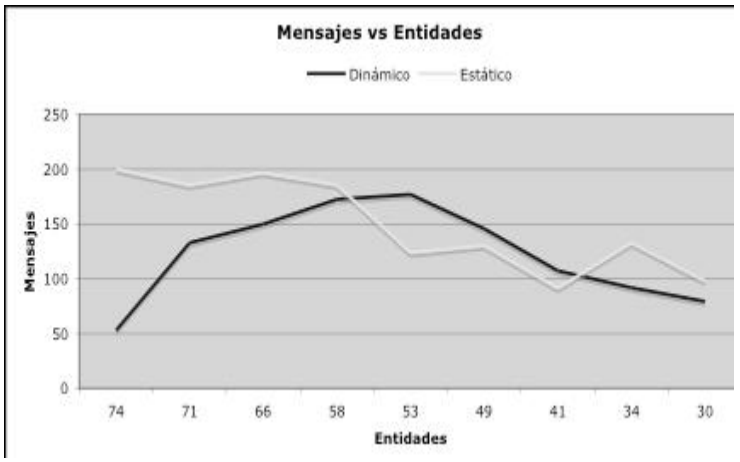

Figura 2. Mensajes por segundo vs. Número de entidades

Servidores Asignados

En la figura 3 se observa que mientras en el modelo estático todos los servidores permanecen asignados, en el dinámico no todos lo están y pueden ser utilizados para realizar otra tarea. También es importante notar que a medida que el número de entidades disminuye, la utilización de servidores disminuye en el modelo dinámico.

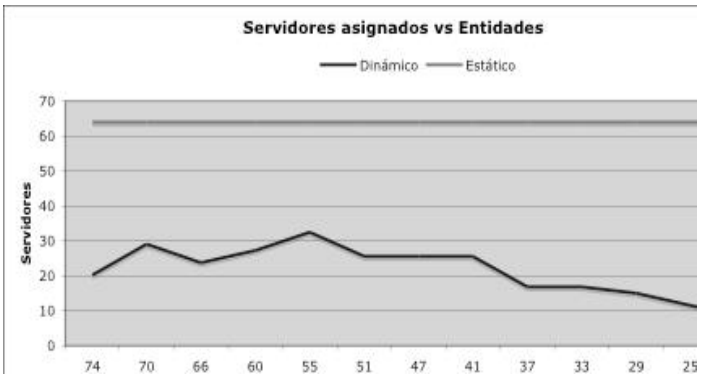

Figura 3. Servidores asignados vs. Número de entidades

Factor de Optimización

Los resultados mostrados en las figura 4 indican una mejor utilización de los recursos por el modelo propuesto, además mejora a medida que el número de entidades disminuye. En el modelo estático se alcanza una optimización máxima de cerca del 50% y un promedio de 22.62%, mientras que en el dinámico se alcanza en un máximo de rendimiento cerca del 80% y en promedio se registra una utilización de los recursos del 56.93%. En general el modelo dinámico muestra en promedio una mejora del 34.23% con respecto al estático, en el factor de optimización de recursos, lo cual es un 10.32% mejor que el modelo equivalente en 2D propuesto por Restrepo [6].

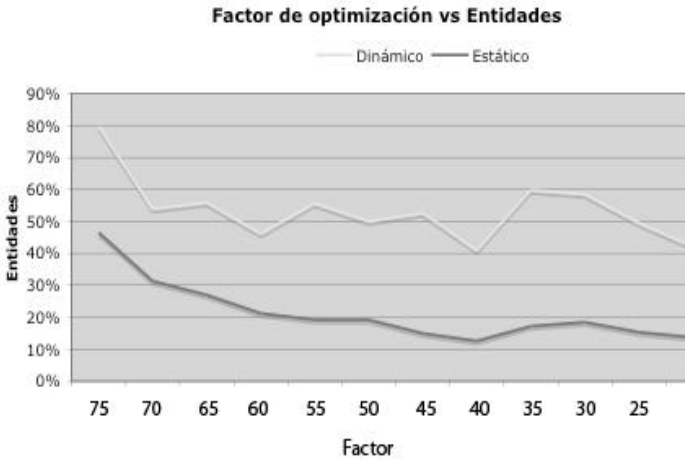

Figura 4. Factor de optimización vs. Número de entidades

VI. CONCLUSIONES

En este artículo introducimos un nuevo método para la asignación dinámica de servidores basado en "octrees". Este método prueba ser más poderoso en el uso de los recursos que un método estático. La simulación realizada arrojó como resultado que el modelo propuesto usando "octrees" retornaba la mejor utilización del sistema representando en promedio una mejora del 34.23% con respecto a un modelo estático.

VI. TRABAJO FUTURO

Podría evaluarse el método usando sistemas como juegos, donde las entidades estén a cargo de personas de modo que su movimiento no sea errático ya que los diferentes jugadores estarán atentos a su ambiente inmediato de acción.

Puede también ser usado en otras áreas como las comunicaciones, donde es esencial contar con recursos para atender la demanda, el modelo prueba ser útil para el balanceo de cargas, como caso práctico se propone el escenario de asignación de señales para telefonía celular. De manera similar, se podría explorar un algoritmo semejante para el establecimiento de redes "Ad-hoc".

REFERENCIAS

[1] Bourke, T., 2001. Server Load Balancing. O´Reilly.

[2] Broll, W., 1997. Distributed Virtual Reality for Everyone - a framework for Networked VR on the internet. IEEE Computer Society Press.

[3] Funkhouser, T., 1996. Network Topologies for scalable multi-user virtual environments. Bell Laboratories.

[4] Humphrey, B., 2005. Octree tutorial (DigiBen) http://www.gametutorials.com/Tutorials/OpenGL/Octree.htm

[5] Macedonia, M. y Zyda, M., 1997. Taxonomy for networked virtual environments. Fraunhofer Center for Research in Computer Graphics. Naval Postgraduate School.

[6] Restrepo, A., Montoya, A. y Trefftz, H., 2003. Dynamic Server Allocation in Virtual Environments, Using Quadtrees for Dynamic Space Partition. Proceedings of IASTED Computer Science and Technology, Cancún, México.

[7] Reynolds, C., 1987. http://www.red3d.com/dwr/boids/

[8] Srisawat, J. y Nikitas, A., 1998. A Quadtree Based Dynamic Processor Allocation Scheme for Mesh-Connected Parallel Machines. Conference in Computational Applications, Las Vegas, Nevada, USA.

[9] Srisawat, J. y Nikitas, A., 1999. A New Quadtree Based subsystem Allocation Technique for Mesh-Connected Parallel Machines. ACM-SIGARCH conference in super computing Rhodes, Greece.