GRANJA DE RENDERIZADO BLENDER CON HTCONDOR SOBRE WINDOWS Y ROCKS SOBRE LINUX

Blender Render Farm with HTCondor over Windows and Linux Rocks

Leonardo Andrés Forero Castillo*
Laforero2@poligran.edu.co
Institución Universitaria Politécnico Grancolombiano, Bogotá (Colombia)
Alexis Rojas Cordero **
arojasco@poligran.edu.co
Institución Universitaria Politécnico Grancolombiano, Bogotá (Colombia)

* Estudiante Ingeniería de Sistemas. Grupo de investigación FICBPG, Institución Universitaria Politécnico Grancolombiano (Bogotá, Colombia).
** Docente / Investigador. Grupo de investigación FICBPG, Institución Universitaria Politécnico Grancolombiano (Bogotá, Colombia).

Resumen

El aumento en la demanda de creación de imágenes con alta calidad y en formato 3D, ha hecho que la solicitud de recursos para su procesamiento, sea cada vez mayor. En el presente documento se demostrará, cómo puede ser usado un grupo de computadoras para hacer computación en clúster o de alto rendimiento, con el fin de realizar el procesamiento de dichas imágenes en un menor tiempo, haciendo uso de los recursos disponibles que pueden tener cierta capacidad de procesamiento y la cual, puede ser usada para este propósito. En este documento se describirá detalladamente como el aprovechamiento de Blender en combinación con HTCondor permite lograr este objetivo.

Palabras clave: Blender, Granja de renderizado, HPC, HTCondor.

Abstract

The growing demand of images creation in 3D format with high quality, has caused the requirement of big amount of resources for its processing to be higher. Within this document, it'll be shown how a group of computers can be used to build up a cluster for heavy computing or a high-performance computing farm to do the processing of those pictures in a lesser time and at the same time use the available resources which could have some processing power that can be used for this purpose. On this document, it'll be described how blender in combination with HTCondor can be used to achieve this objective.

Keywords: Blender, HPC, HTCondor, Render farm.

INTRODUCCIÓN

El renderizado o procesamiento de imágenes en tercera dimensión y con características de alta resolución, demanda el uso de elevados recursos de cómputo para la creación de dichas imágenes, en ocasiones se podría pensar que la única alternativa, es adquirir una cantidad de servidores con altas prestaciones de hardware o pagar por servicios en la nube que permiten el renderizado de imágenes y como opción alternativa, en ocasiones trabajadores independientes e incluso estudiantes de carreras afines a la creación de contenido digital, tienen que armar su propia máquina de cómputo con características muy superiores a las de un computador normal, estas máquinas suelen llevar un alto costo económico y aunque brindan poder de procesamiento suficiente para realizar el trabajo de renderizado en un tiempo relativamente optimo, la capacidad económica puede hacer inaccesible a un estudiante o trabajador independiente a la adquisición de estas.

Para abordar el problema que la necesidad de poder de procesamiento genera, se puede sacar provecho de la computación en malla (o grid) [1], ya que a partir de esta tecnología podemos hacer uso de múltiples máquinas que de forma coordinada usarán recursos heterogéneos dando como resultado una forma de computación distribuida [2], en la cual los equipos conectados o llamados nodos, pueden ser de iguales o diferentes arquitecturas y de esta forma poder tener a disposición una gran cantidad de potencia computacional, para la renderización de una animación que se compone de varios cuadros de imagen o frames. Realizar este tipo de configuración será posible gracias a HTCondor [3], un programa que nos permitirá crear una granja de renderizado gracias a su poder computacional basado en MPI [4] y en la computación en malla.

En la actualidad existen sistemas operativos pre-configurados que pueden ser de ayuda para configurar un sistema de computación de alto desempeño, estos sistemas operativos usan Linux, en específico una de las muchas distribuciones para HPC [5] de la que se hablará es Rocks-HPC [6], el cual facilita la creación y configuración de un clúster de forma sencilla y con tan solo unos cuantos clics. En este documento, se referirá la opción alternativa que se puede usar mediante esta distribución de forma general.

Para el desarrollo del renderizado se usó Blender [7], el cual sirvió para la configuración, puesta en marcha y final renderizado de una animación que dará como resultado un cierto número de imágenes. Se aplicó este programa debido a su popularidad y facilidad de uso, así como el nivel de implementación de línea de comandos [15] que permite realizar el trabajo de renderizado sin la necesidad de tener una interfaz gráfica o emplear un modo interactivo, para la elaboración del renderizado sobre el clúster.

Esta implementación e investigación se enfocó principalmente sobre el sistema operativo Windows, dado que es el sistema operativo más usado y el que comúnmente encontramos instalado en los computadores, además, Windows es ampliamente usado por usuarios del común y por ejemplo, en el campus de una universidad, podríamos hacer uso de los recursos que los computadores tienen, dado que la implementación de HTCondor funciona bajo Windows y a la vez el uso de este sistema operativo, facilitará el trabajo de renderizado con tan solo unos cuantos pasos que no revestirán mayor dificultad para el usuario final.

INSTALACIÓN Y CONFIGURACIÓN DE HTCONDOR SOBRE WINDOWS

Se debe tener en cuenta que para esta instalación de HTCondor se usó Windows de 64 y 32 bits, así como también Blender para Windows de 64 y 32bits.

La instalación y configuración de HTCondor sobre Windows, es en realidad bastante sencilla, gracias al instalador creado por el desarrollador de HTCondor, podemos realizar una instalación y configuración en unos pocos pasos. Para esta instalación se sugiere destinar una máquina como administrador central y que no se ejecute allí ningún trabajo de HTCondor, esto con el fin de tener una estructura clara dentro del pool para HTCondor.

Primero debemos descargar HTCondor de [8].

Una vez allí descargaremos la versión estable para Windows de 64 o 32 bits, al momento de la creación de este artículo la versión estable es la 8.6.2.

Instalación del central Manager

Primero se debe instalar el administrador central (central manager), pues es ahí donde será creado el pool para HTCondor y además, a donde las máquinas se conectarán para unirse al pool. Es recomendable que a este computador se le configure una dirección IP estática con el fin de evitar posibles problemas al usar el nombre de host del pc en la configuración.

La configuración se debe realizar de la siguiente manera para el nodo central:

1. En la segunda pantalla del instalador se dará el nombre al pool:

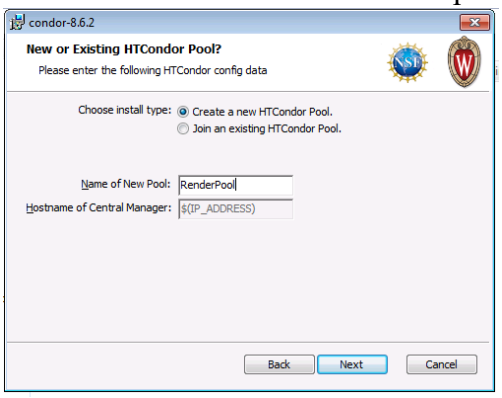

Figura 1. *Instalación de HTCondor*

2. En la configuración de HTCondor para la ejecución de trabajos, se escoge la opción, *submit Jobs to HTCondor Pool.* A la pregunta: *When should HTCondor run Jobs?*, Responder: *Do not run Jobs on this machine*

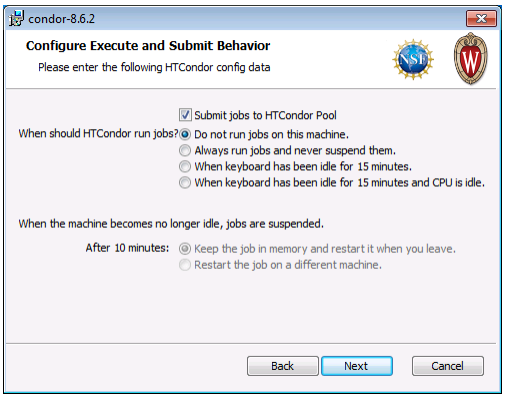

Figura 2. *Instalación de HTCondor*

3. En la ventana de configuración ***Accounting Domain*** se escribe, $(FULL_HOSTNAME), esta opción sirve para configurar, el dominio DNS de la red donde funcionará HTCondor (habilita también funciones de seguridad que pueden ser configuradas en un entorno de dominio) para este caso, no es de interés configurar un dominio, así que se deja por defecto el hostname.

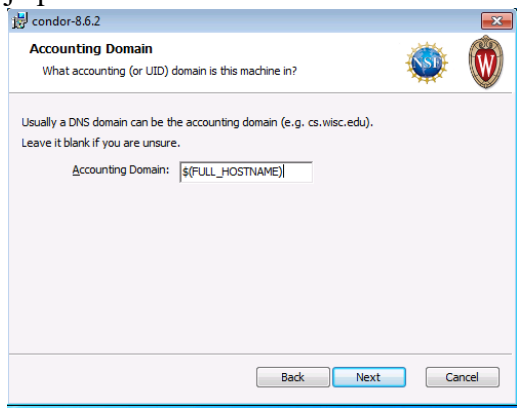

Figura 3*. Instalación de HTCondor*

4. En la venta de configuración ***Email Settings***, se puede configurar un servidor de correo y una dirección de administrador para HTCondor, esto con el fin de que el sistema de HTCondor, pueda enviar alertas o notificaciones que el sistema crea necesarias, pero para este caso de muestra, no se configura dicha opción, entonces las casillas quedan en blanco.
5. En la ventana ***Java Settings*** el instalador pregunta la ruta hacia el ejecutable de java JVM [9], para este caso no es necesario darle dicha ruta, ya que este corresponde al administrador central y sobre esta máquina no se ejecutarán trabajos, por lo tanto, no es necesario realizar dicha configuración.
6. En la ventana ***Host Permission Settings*** se definirá el acceso de otros equipos a este equipo administrador central, aquí es muy importante prestar especial atención a la opción ***Host with write Access,*** pues es esta opción, la que define qué equipos se podrán conectar a esta máquina, en otras palabras, permitirá que los equipos de una subred allí ingresada se conecten al pool, por defecto ya están escritas las siguientes

opciones: $(CONDOR_HOST), $(IP_ADDRESS), *.cs.wisc.edu. De esta manera la configuración permitirá a todos los equipos remotos conectarse al pool, por lo tanto, solo bastará con escribir un * antecedido por una coma y después de *.cs.wisc.edu en la opción ***Host With Administrator Access,*** se adiciona $(CONDOR_HOST) precedido de una coma.

7. En la pantalla de configuración ***VM Universe Settings*** no se hacen modificaciones, es decir, permanecerá seleccionada la opción, No.
8. En la siguiente ventana se selecciona la ruta dónde va a instalarse HTCondor, por lo general se recomienda que se instale sobre el disco C, así que se deja la opción tal como aparece.
9. Finalmente procede la instalación de HTCondor, se reinicia la máquina y ya está el administrador Central listo para enviar trabajos y administrar el pool.

Instalación de HTCondor sobre máquinas que serán miembro del pool.

Para esta instalación, se usa el mismo instalador que se utilizó para el administrador central, pero las opciones a seleccionar varían dependiendo de la configuración que se vaya a dar sobre cada máquina.

La instalación y configuración [10] se realizará de la siguiente manera:

1. En la pantalla ***New or Existing HTCondor pool*** se selecciona la opción *Join an existing HTCondor pool* y en la opción *Hostname of Central Manager* se pueden escribir una de las siguientes dos opciones, a) el nombre del pc que es el administrador central o b) la dirección IP, cabe aclarar que es preferible el uso de la dirección IP, ya que es de suponer, se ha debido asignar una dirección IP fija, a dicho administrador central, para efectos de esta muestra se ha realizado la configuración con el nombre del administrador central.
2. En la pantalla ***Configure Execute and Submit Behavior*** [10] se debe definir cómo se comportará esta máquina, para ello, primero se precisa si, esta permite o no enviar trabajos al pool de HTCondor, si es de preferencia permitirlo, entonces se habilita la opción *Submit Jobs to HTCondor Pool* (esto es opcional y esta consideración de quien lo instala), para esta muestra no se seleccionó dicha opción, pues todos los trabajos serán enviados desde el nodo central o administrador central. En la opción *When should HTCondor run Jobs* tenemos varias opciones disponibles y se debe escoger la que más convenga, según el propósito de la máquina, Para máquinas que estarán dedicadas a la ejecución de trabajos, se toma la opción *Always run Jobs and never suspend them*. Para máquinas que se deban usar cuando estén disponibles y sin uso hay dos opciones, a) escoger *When keyboard has been idle for 15 minutes* o b) escoger la opción *When keyboard has been idle for 15 minutes and CPU is idle,* si se escoge cualquiera de estas dos opciones, es necesario escoger qué hará HTCondor cuando la máquina deje de estar disponible, es decir cuando el trabajo se suspenda por actividad de un usuario, para ello se debe escoger entre si mantener el trabajo en memoria (*Keep the job in memory and restart it when you leave)* o si el trabajo debería reiniciarse en otra máquina (*Restart the jobo n a different machine).* Note que, si se selecciona la

opción para reiniciar los trabajos en otra máquina, esto hará que el trabajo empiece desde cero si el trabajo es del universo vanilla o java.

3. En la ventana de configuración ***Accounting Domain*** se debe escribir como configuración $(FULL_HOSTNAME), esta configuración sirve para establecer allí el dominio DNS de la red donde funcionará HTCondor (habilita también funciones de seguridad que pueden ser configuradas en un entorno de dominio, pero en este artículo no se realizará dicha configuración por la complejidad que puede representar), para esta muestra no es de interés tomar un dominio, así que se deja el hostname (aunque no es necesario definir esta configuración y puede dejarse en blanco).
4. En la ventana de configuración ***Email Settings***, se puede configurar un servidor de correo y una dirección de administrador para HTCondor, esto con el fin de que el sistema de HTCondor, pueda enviar alertas o notificaciones que el sistema crea necesarias, pero para esta muestra no será configurada dicha opción, de esta manera quedan en blanco las casillas.
5. En la ventana ***Java Settings*** el instalador pregunta la ruta hacia el ejecutable de java JVM [9], por lo general si Java ya estaba instalado en la máquina, el instalador detectará automáticamente la ruta en donde se encuentra instalada la máquina virtual de Java, pero si no la detecta, se deberá escribir manualmente la ruta hacia el ejecutable de Java.
6. En la ventana ***Host Permission Settings*** se definirá el acceso de otros equipos a este equipo administrador central, aquí es muy importante prestar especial atención a la opción ***Host with write Access,*** pues es esta opción la que define qué equipos se podrán conectar a esta máquina, en otras palabras, permitirá que los equipos de una subred allí ingresada se conecten a nuestro pool, por defecto ya están escritas las siguientes opciones: $(CONDOR_HOST), $(IP_ADDRESS), *.cs.wisc.edu. La configuración realizada permitirá a todos los equipos remotos conectarse al pool, por lo tanto, solo bastará con escribir un * antecedido por una coma y después de *.cs.wisc.edu. Ademas de esto en la opción ***Host With Administrator Access*** adicionaremos $(CONDOR_HOST) precedido de una coma.

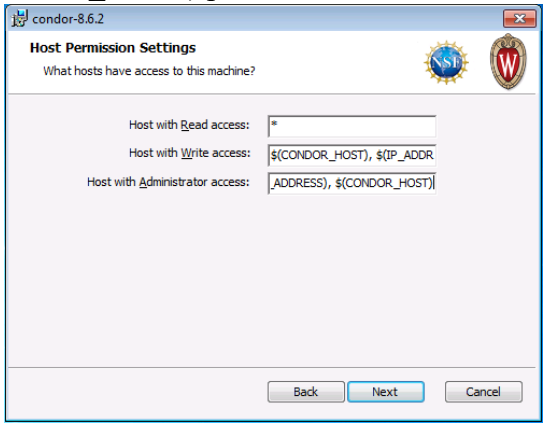

Figura 4. *Instalación nodo HTCondor*

7. En la pantalla de configuración ***VM Universe Settings*** quedara tal cual está, es decir, seleccionada la opción No.
8. En la siguiente ventana se escogerá dónde quiere que se instale HTCondor, por lo general se recomienda que se instale sobre el disco C, así que puede dejar la opción tal como aparece.
9. Finalmente se instala HTCondor, reinicia la máquina y ya tendrá el equipo unido al pool, para comprobar que el equipo se ha unido correctamente al pool, puede ejecutar desde una ventana de comandos la orden ***condor_status*** [13] y verá los equipos que están unidos al pool, así como el equipo que acaba de instalar en unión al pool.

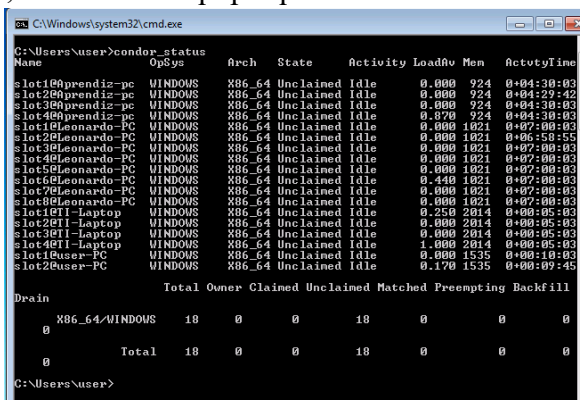

Figura 5. *HTCondor Pool.*

Interfaz gráfica para HTCondor

Dado que el manejo de HTCondor sobre consola puede ser algo complicado, se quiso añadir alguna interfaz que pueda ser de ayuda para administrar el pool de HTCondor de una forma más sencilla y visual, para esto existe una interfaz desarrollada por la universidad de *Otto Von Guericke University Magdeburg*, Alemania. Se puede descargar dicha interfaz (compatible para Windows, Linux y Mac) para ser usada con la instalación de HTCondor en el administrador central [14].

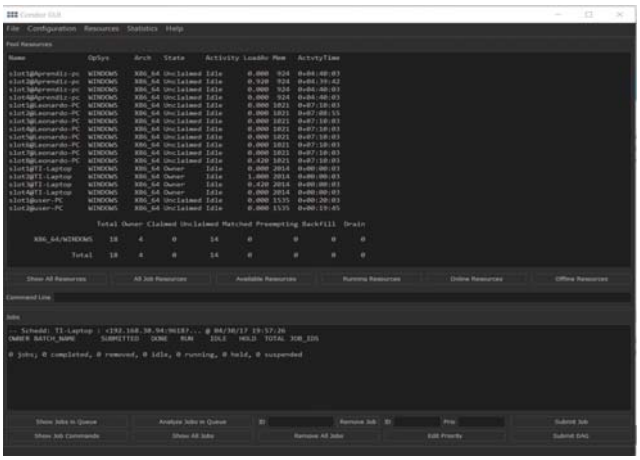

Figura 6. *HTCondor GUI*

Envío de trabajo desde Condor GUI

Enviar un trabajo desde la interfaz Condor GUI es realmente muy sencillo, lo único que se debe hacer es, a) clic sobre el botón ***Submit job***, ubicado en la esquina inferior derecha, b) se abrirá una nueva ventana en la cual navega hasta la ubicación donde se encuentra el archivo job, c) lo selecciona y el trabajo se enviará a la cola de trabajo de HTCondor. La interfaz gráfica es bastante intuitiva y fácil de usar, brindando comodidad y facilidad de uso para administrar trabajos enviados al pool de HTCondor.

CREACIÓN DE UN TRABAJO EN HTCONDOR

Para el procesamiento de trabajos sobre HTCondor, se debe crear un archivo job en el cual se configurará el trabajo, así como las variables necesarias para la ejecución del mismo.

Un archivo de job de HTCondor tiene una estructura como la siguiente:

```
executable     = mathematica
universe       = vanilla
input          = test.data
output         = loop.out
error          = loop.error
log            = loop.log
request_memory = 1 GB
initialdir     = run_1
queue
```

Cuadro 1. *Ejemplo archivo de trabajo HTCondor*

Para entender esta estructura, es necesario tener en cuenta la configuración que se puede realizar sobre dicho archivo, por esto es indispensable conocer cada opción:

Universe: en HTCondor la opción universo es aquella que define el entorno de ejecución, HTCondor soporta diferentes tipos de universo para procesar los trabajos, entre ellos encontramos:

- Standard
- Vanilla
- Grid
- Java
- Parallel
- Local
- Scheduler
- Vm
- Docker

Este universo es el que se específica en el archivo de configuración del archivo job, si no se especifica el universo bajo el cual debe ser ejecutado el trabajo, HTCondor tomará el universo por defecto el cual es vanilla (a menos que esta configuración sea cambiada en la instalación de HTCondor).

El universo Estandar provee migración y confiabilidad, pero tiene algunas restricciones sobre los programas que puede ejecutar. El universo grid permite que los usuarios puedan enviar

trabajos usando la interfaz de HTCondor, estos trabajos son enviados para su ejecución en recursos de tipo grid. El universo Java permite que los usuarios puedan ejecutar trabajos creados para la máquina virtual de java (JVM). El universo scheduler permite a los usuarios enviar trabajos livianos para ser encolados por un programa conocido como, el demonio sobre la máquina misma, que envía el trabajo (submit host). El universo parallel es para programas que requieren el uso de múltiples máquinas para un trabajo. El universo vm permite que los usuarios puedan ejecutar trabajos donde el trabajo no es un simple ejecutable, sino una imagen de disco, facilitando la ejecución de una máquina virtual. El universo Docker ejecutar un contenedor Docker como un trabajo de HTCondor.

Universo Vanilla

El propósito del universo vanilla en HTCondor es ser usado para programas que no pueden ser re-enlazados satisfactoriamente. Los scripts de consola son otro caso donde el universo vanilla es usable. Desafortunadamente, los trabajos que se ejecutan bajo el universo vanilla no pueden realizar puntos de control o usar llamadas de sistema remoto. Esto tiene consecuencias desafortunadas para un trabajo que es parcialmente completado cuando la máquina que está ejecutando el trabajo debe ser devuelta al usuario. HTCondor tiene solamente dos opciones, puede suspender el trabajo esperando a completarlo más tarde o puede reiniciar el trabajo *desde el inicio* en otra máquina del pool.

El hecho de que el sistema de llamada remota de HTcondor no puede ser usado con el universo vanilla hace que el acceso a los archivos de entradas y salidas sea una preocupación. Una opción es dejar que HTCondor dependa de un sistema compartido de archivos tal como NFS o AFS. Alternativamente, HTCondor tiene un mecanismo para trasferir archivos en nombre del usuario. En este caso, HTCondor realizará la transferencia de los archivos que sean necesario para el trabajo hacia el sitio de ejecución, ejecuta el trabajo y devuelve la salida hacia el equipo que envió el trabajo a la máquina.

Bajo Unix, HTCondor asume que hay un sistema compartido de archivos para los trabajos del universo vanilla, sin embargo, si no está disponible un sistema de archivos compartido, el usuario puede habilitar el mecanismo de trasferencia de archivos de HTCondor. En plataformas Windows, el método por defecto es el mecanismo de transferencia.

Variables en el envío de un trabajo

Hay variables [11] automáticas para el uso en el archivo descriptivo de envío de un trabajo:

- ***$(Cluster)*** o ***$(ClusterId):*** cada grupo de trabajos encolados de un usuario específico, enviados desde un host único de envío de trabajos, que comparte un ejecutable tiene el mismo valor de $(Cluster) o $(ClusterId). El primer cluster de trabajos son asignados al cluster 0 y el valor es incrementado uno por uno a cada nuevo cluster de trabajos.
- ***$(Process)*** o ***$(ProcId)*** dentro de un cluster de trabajos, cada uno toma un valor $(Process) o $(ProcId) único. El primer trabajo dentro del grupo de trabajos toma valor 0.

- ***$(Item)*** El nombre por defecto de la variable cuando no se provee ningún <varname> al comando queue.
- ***$(ItemIndex)*** Representa un índice dentro de una lista de ítems.
- ***$(Step)*** para la <int expr> (variable de expresión inicial) especificada, $(Step) cuenta empezando en 0.
- ***$(Row)*** Cuando una lista de ítems es especificada poniendo cada ítem en su propia línea en el archivo descriptor de envío, $(Row) identifica en cual línea está el ítem. El primer ítem (primera línea de la lista) es $(Row) 0, el segundo ítem (segunda línea de la lista) es $(row) 1.

Otras variables a usar en el descriptor de trabajo

Hay variables adicionales [12] que se usarán para el envío del trabajo de renderizado a HTCondor, estas variables son:

- ***Environment:*** Son aquellas variables de entorno que puedan ser necesarias para la ejecución de un trabajo de HTCondor, para esta muestra se definió un path, para evitar que Windows no reconozca la ruta, en donde se encuentra el ejecutable del programa, en este caso, el ejecutable de blender.
- ***Transfer_input_files:*** aquí es donde se indicarán los archivos a transferir, es de mencionar, que este comando hace uso del mecanismo de transferencia de archivos de HTCondor sin tener un sistema de archivos compartido, por esto es necesario usar este comando en el archivo descriptor de trabajo. Por tanto, aquí se coloca el nombre del archivo de blender que se enviará al pool para ser renderizado.
- ***Should_transfer_files:*** este comando es el que habilita o deshabilita el mecanismo de transferencia de HTCondor, para este comando hay tres posibles valores: IF_NEEDED, YES, y NO, el primero habilita el mecanismo en caso de que el trabajo necesite transferir archivos (opción usada cuando se trabaja sobre un sistema compartido de archivos) y con las otras dos opciones se habilita o deshabilita la transferencia de archivos. Para esta muestra se usa este comando con valor en yes.
- ***Transfer_executable:*** Con este comando le indica al método de transferencia de HTCondor si debe o no transferir el ejecutable que se defina en el archivo, en el caso de los archivos tipo batch y similares será necesario que se transfiera para su posterior ejecución, pero en el caso en que se quiere iniciar un ejecutable en las máquinas donde se procesará el trabajo, la opción deberá deshabilitarse dándole como valor FALSE.
- ***Transfer_output_files:*** Mediante este comando en el archivo descriptor del trabajo, se le indica a HTCondor qué archivos debe transferir de las máquinas donde se ejecute el trabajo, por defecto HTCondor detecta las carpetas y ficheros creados en las máquinas remotas en la ejecución de un trabajo, aplicado en esta muestra en particular; será necesario realizar la transferencia de dichos archivos creados en cada máquina, pues se crearán archivos de imagen (frames) que corresponden al trabajo de renderizado que se envía a HTCondor en cada máquina y la efectividad del renderizado está en poder obtener de vuelta en la máquina central (o desde donde se envió el trabajo) todas las imágenes que se han procesado en el pool, por lo que se

hace imperativo el uso y definición de este comando en el archivo descriptor de trabajo de HTCondor.

- ***when_to transfer_output:*** Este comando le dice a HTCondor cuando los archivos de salida creados por el trabajo ejecutado deben ser enviados de vuelta a la máquina que envió el trabajo. Para la ejecución de este ejemplo se emplea la siguiente opción:
  ON_EXIT: HTCondor transfiere devuelta a la máquina que envió el trabajo los archivos que se definan en el comando output, así como otros archivos creados por el trabajo en las máquinas remotas.
- ***Executable:*** A partir de este comando se define el ejecutable del programa que se desea iniciar para la consecución de la tarea, en esta muestra es mediante este comando, que se le dirá a HTCondor donde puede encontrar el ejecutable de blender. Este comando funciona en conjunto con el comando *transfer_executable*, pues cuando no se define si transferir o no, por defecto HTCondor buscará en la máquina local dicho ejecutable y lo transferirá a la máquina remota, lo cual para este propósito es inviable ya que blender es un programa que depende, de mucho más que su ejecutable para funcionar.
- ***Output:*** Mediante este comando se precisa el nombre del archivo de salida de HTCondor, es allí donde se registrarán los eventos que tengan que ver con la salida de HTCondor, es decir, el resultado del procesamiento del trabajo en las máquinas remotas.
- ***Error:*** Este comando indica el nombre del archivo de error de HTCondor, es allí donde se registrarán los errores del archivo cuando se ejecute en las máquinas remotas.
- ***Log:*** A partir de este comando se define el nombre del archivo de seguimiento o log de HTCondor, es donde se registrarán los sucesos normales de comportamiento del trabajo, como por ejemplo las máquinas que aceptan el trabajo y demás información general relativa al trabajo enviado al pool de HTCondor.
- ***Arguments:*** Con este comando se define los argumentos con los que se ejecutará el trabajo, en esta muestra es allí donde se hará uso de los argumentos que se pueden pasar al ejecutable de blender, para posterior a este, realizar el procesamiento del render el cual se está trabajando, nótese que es necesario conocer un poco los argumentos que se necesitan para el render en función del resultado que se quiere lograr.
- ***Priority:*** Este comando define la prioridad que tendrá el trabajo en la cola de HTCondor, su número varía desde -20 hasta 20, siendo -20 la prioridad más baja y 20 la prioridad máxima. Si no se define una prioridad, HTCondor tomará como prioridad un valor por defecto de 0 el cual le da una prioridad normal al trabajo.
- ***Queue:*** Este comando finaliza el archivo descriptor de HTCondor y especifica el grupo de trabajos o el trabajo a enviar a HTCondor, es decir que mediante el comando *queue* podemos ejecutar n número de veces el mismo trabajo siendo tomado cada uno como un trabajo por realizar, por ejemplo, para esta muestra es un valor de 60 ya que son 60 los cuadros de imágenes que queremos renderizar.

Creación del archivo de trabajo clockRender.job

Luego de ya haber visto un poco de los conceptos necesarios para poder entender el funcionamiento de un archivo descriptor de trabajo de HTCondor, se debe crear un archivo [11] de nombre clockRender.job para poder enviar el trabajo a nuestro pool. En la creación de este archivo se puede usar cualquier editor de texto, como el bloc de notas de Windows u otra alternativa, que nos permita guardar texto sin formato y al finalizar la escritura del archivo es indispensable asegurarse que quede con el nombre terminado en extensión job, de esta manera puede ser reconocido y leído por HTCondor.

```
universe = vanilla
environment = path=C:\Program Files\Blender Foundation\Blender;c:\windows;c:\windows\system32
transfer_input_files = clockFinal.blend
transfer_output_files = /tmp/
transfer_executable = False
should_transfer_files = yes
when_to_transfer_output = on_exit
executable = C:\Program Files\Blender Foundation\Blender\blender.exe
output = ClockRender.out
error = ClockRender.err
log = ClockRender.log
arguments = -b clockFinal.blend -o /tmp/file -f $(Process)
request_disk = 0
priority = 0
queue 60
```

Cuadro 2. *Archivo de trabajo de HTCondor*

En el archivo existe la siguiente configuración:

universe = vanilla

Se hace uso del universo vanilla, ya que es el que mejor se ajusta al esquema de ejecución para el trabajo de render.

transfer_input_files = clockFinal.blend

Mediante este comando se transfiere el archivo a renderizar hacia cada máquina remota.

transfer_output_files = /tmp/clock/

Mediante este comando se hace la transferencia de la carpeta de salida que creará Blender en cada máquina remota.

transfer_executable = False

Se configura que no se transfiera el ejecutable de blender, pues este debe estar instalado en cada máquina remota.

executable = C:\Program Files\Blender Foundation\Blender\blender.exe

Este comando le dice a HTCondor cuál es la ruta del ejecutable de Blender.

output = ClockRender.out, error = ClockRender.err, y log = ClockRender.log

Estos comandos definen el nombre para cada tipo de archivo log de HTCondor.

arguments = -b clockFinal.blend -o /tmp/clock/file -f $(Process)

Este comando define los argumentos que se le pasarán al ejecutable de Blender [15], el -b indica el nombre del archivo a procesar, -o indica la salida del archivo (al no dársele una ruta específica como C:/ o D:/, etc., el programa tomará la ruta raíz donde el programa está instalado) y -f le indica a Blender que sólo debe renderizar un frame específico, en este caso el número de frame estará definido por $(Process) qué será el número del proceso creado por HTCondor; esta opción depende del comando queue.

queue 60

Con este comando se le ordena a HTCondor que se quiere ejecutar este trabajo 60 veces, ya que son 60 las imágenes que se van a renderizar. Esto quiere decir que para el trabajo habrá una especie de sub trabajos que serán para este caso 60 en total, siendo así, cada trabajo tendrá un número de proceso, el cual hará que con cada ejecución se renderice un frame específico que será el correspondiente al número de trabajo dentro del sistema de ejecución de HTCondor.

EJECUCIÓN DE UN TRABAJO EN UN ENTORNO HETEROGENEO

HTCondor brinda la posibilidad de realizar la ejecución de una tarea bajo un ambiente que tenga máquinas remotas tanto de 64 bits como de 32 bits, para ello se deberá hacer uso de una macro que se configura dentro del archivo descriptor de trabajo de HTCondor y para eso se hará uso del comando *requirements*, en el cual podremos definir la arquitectura de procesador a usar para el trabajo [11].

El siguiente es el ejemplo del archivo descriptor que permitirá que el trabajo se ejecute en máquinas con diferente arquitectura:

```
universe = vanilla
environment = path=C:\Program Files\Blender
Foundation\Blender;c:\windows;c:\windows\system32
transfer_input_files = clockFinal.blend
transfer_output_files = /tmp/
transfer_executable = False
should_transfer_files = yes
when_to_transfer_output = on_exit
executable = C:\Program Files\Blender Foundation\Blender\blender.exe
output = ClockRender.out
error = ClockRender.err
log = ClockRender.log
arguments = -b clockFinal.blend -o /tmp/file -F PNG -f $(Process)
request_disk = 0
priority = 20
Requirements = (Arch == "INTEL") || \ (Arch == "X86_64")
queue 60
```

***Cuadro 3**. Archivo de trabajo para ejecución heterogénea*

EJECUCIÓN DEL TRABAJO EN HTCONDOR

Para realizar la ejecución del trabajo se sugiere crear una carpeta que contenga tanto el archivo de trabajo como el archivo a renderizar, para ejemplificar mejor, en nuestro caso muestra, se ha creado una carpeta de nombre RENDER y se han puesto allí los archivos clockRender.job y clockFinal.blend

***Figura 7.** Carpeta de trabajo*

Adicional a esto se debe tener en cuenta que, ya que el directorio de salida para HTCondor está configurado como /tmp, será necesario crear esta carpeta en la raíz del disco donde se encuentre HTCondor instalado (En este caso disco C).

Ahora que el archivo de trabajo y el archivo blender estan juntos en una misma carpeta, se procede al envió del trabajo al pool de HTCondor, para ello se abre una consola de comandos (cmd), en paralelo se dirige al directorio donde está guardado el archivo job y el archivo blend, luego ejecutando la orden condor_submit [13] ClockRender.job .

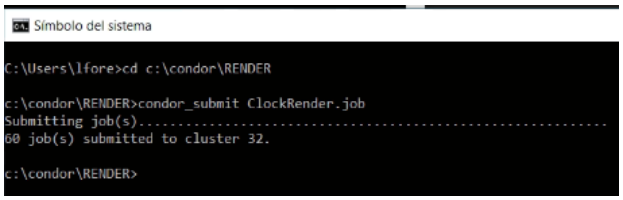

***Figura 8**. Envío de trabajo en HTCondor*

También como solución alternativa se puede usar la interfaz gráfica para HTCondor.

Después de unos minutos se recibirá en el administrador central los archivos que han sido generados por el trabajo:

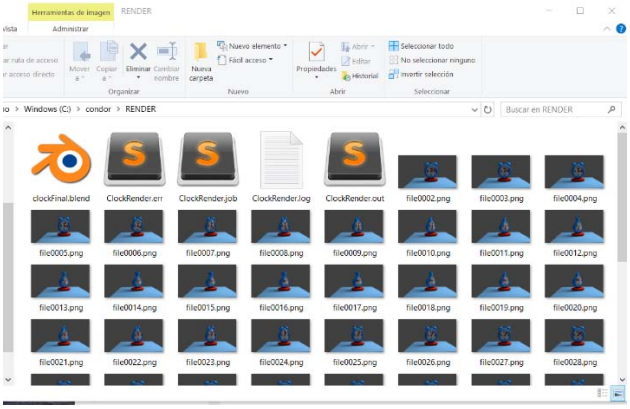

***Figura 9.** Tarea de render finalizada*

Es posible que antes de la ejecución del trabajo se muestre un error de HTCondor que dirá ERROR: No credential stored for user youruser@yourpc, esto se corrige mediante la ejecución del comando ***condor_store_cred add***, al ejecutar este comando HTCondor preguntará la contraseña del usuario que ha iniciado sesión en el equipo, como restricción no pueden haber contraseñas vacías pues este es un método de seguridad usado implícitamente por HTCondor, esto quiere decir que se debe iniciar sesión en la máquina con un usuario con contraseña para poder realizar la ejecución del trabajo y la adición de las credenciales para el uso de HTCondor.

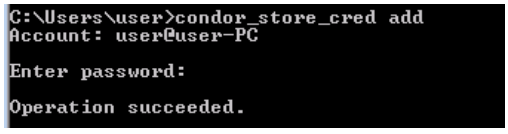

***Figura 10**. Credenciales para HTCondor*

Luego de ejecutar este comando [13] y de obtener la confirmación de operación exitosa, puede proceder nuevamente a enviar el trabajo al pool de HTCondor.

Nota: En Linux la ejecución de un trabajo no se puede realizar desde el usuario root debido a restricciones de seguridad en la ejecución de la tarea.

SOLUCIÓN ALTERNATIVA BAJO LINUX

Cómo solución alternativa para crear una granja de renderizado, podemos implementar un esquema bajo Linux, este esquema se recomienda crearlo con la distribución HPC de Linux llamada ROCKS HPC [6].

Configurar este entorno requerirá conocimientos avanzados en el uso de sistemas operativos Linux, así como algunos conocimientos redes e instalación de sistemas operativos. Para realizar la instalación de Rocks se puede ir a [16]. En la instalación se debe seleccionar la instalación de HTCondor, al seleccionar esta opción no será necesario configurar nada ya que la configuración es automática.

Una vez se tenga Rocks plenamente instalado y configurado lo único que se debe hacer es lo siguiente:

1. Crear un usuario normal para ingresar al sistema, esto debido a la restricción que tiene HTCondor para el envío de trabajos que no pueden ser enviados como root.
2. Se puede implementar un sistema compartido de archivos como Lustre [17] o simplemente compartir una carpeta en el maestro de Rocks y montarla en cada una de las máquinas nodo, esto puede hacerse tal como lo indica el manual en [18].
3. Una vez definida la carpeta que se compartirá con las máquinas nodo, procederemos a descargar blender, para ello desde la consola de comandos debe ubicarse en el directorio que ha compartido con los nodos y ejecutar la orden:
   wget http://mirror.cs.umn.edu/blender.org/release/Blender2.78/blender-2.78-linux-glibc211-x86_64.tar.bz2
   Se debe descargar la versión aquí mencionada ya que Rocks trae por defecto la librería glibc212 y carece de versiones superiores por lo que al descargar una versión con glibc mayor, esta no funcionará sobre esta distribución de Linux.

3.1 Ahora se descomprime el archivo descargado con la siguiente orden:
tar xvjf blender-2.78-linux-glibc211-x86_64.tar.bz2
y seguido a esto se debe renombrar la carpeta para mayor facilidad de uso.

4. Para verificar que blender funciona, hay que entrar en la carpeta recién renombrada con el comando cd /blender y una vez allí ejecutaremos el comando blender -v, si todo ha ido bien se observa la información de versión de blender.
5. Como ya se ha instalado Blender y se ha configurado la carpeta compartida, hay que proceder a hacer una pequeña edición del archivo de configuración de trabajo para HTCondor, en él solo se cambia las líneas necesarias para ajustarlo a Linux (suponiendo que el directorio compartido es /home/apps para ajustar la ruta del ejecutable):

```
universe = vanilla
transfer_input_files = clockFinal.blend
transfer_output_files = /tmp/
transfer_executable = False
should_transfer_files = yes
when_to_transfer_output = on_exit
executable = /home/apps/blender/blender
output = ClockRender.out
error = ClockRender.err
log = ClockRender.log
arguments = -b clockFinal.blend -o /tmp/file -F PNG -f $(Process)
request_disk = 0
priority = 20
queue 60
```

Cuadro 4. *Archivo de configuración de trabajo HTCondor bajo Linux*

6. Finalmente se guarda el archivo de trabajo con extensión .job en un lugar como el home del usuario o en el escritorio, se abre una terminal de consola se ubica en la carpeta donde se guardó el archivo y se ejecuta el trabajo con el comando *condor_submit* nombre.job.

Resultados y discusión.

Para un trabajo de prueba se usó la siguiente configuración de hardware:

Componente	PC1	PC2	PC3	Totales
Procesador	Intel Core I7	Intel Core I5	Intel Core I5	-
Modelo	2600K	2500s	5200u	-
Núcleos	4 (8 lógicos)	4 (4 lógicos)	2 (4 lógicos)	16
Memoria	8 Gb	4 Gb	4 Gb	16 Gb
Disco	120 Gb	320 Gb	200 Gb	-

Tabla 1. *Especificaciones de Hardware por equipo*

El tiempo total que tardó el renderizado en este clúster fue de 27 minutos, cada cuadro de imagen procesado tiene una resolución de 1920*1080 (Full HD).

Esta renderización se ejecutó sobre un pc con altas especificaciones de Hardware:

Características del PC	
Tipo	Pc de sobremesa
Procesador	Intel Core i7 2600K 3,4 GHz
Memoria	8 Gb DDR3
Tarjeta Vídeo	MSI Nvidia Gforce GTX760 2Gb RAM
Disco Duro	320 Gb SSD
Sistema Operativo	Windows 10 Pro 64 bits

Tabla 2*. Especificaciones de hardware Pc de Escritorio*

Esta renderización se ejecutó sobre un Clúster con 1 nodo bajo Rocks-HPC:

Características del Nodo Cluster Linux	
Tipo	Servidor
Procesador	Intel Xeon 5130 2,0 GHz
Memoria Ram	4 Gb
Disco Duro	140 Gb SAS
Sistema Operativo	Rocks Client (Centos 6,5)

***Tabla 3**. Especificaciones de hardware Servidor nodo*

El tiempo total que tardó este renderizado en ser finalizado fue de 39 minutos. Este renderizado se ejecutó sobre la tarjeta de vídeo del pc.

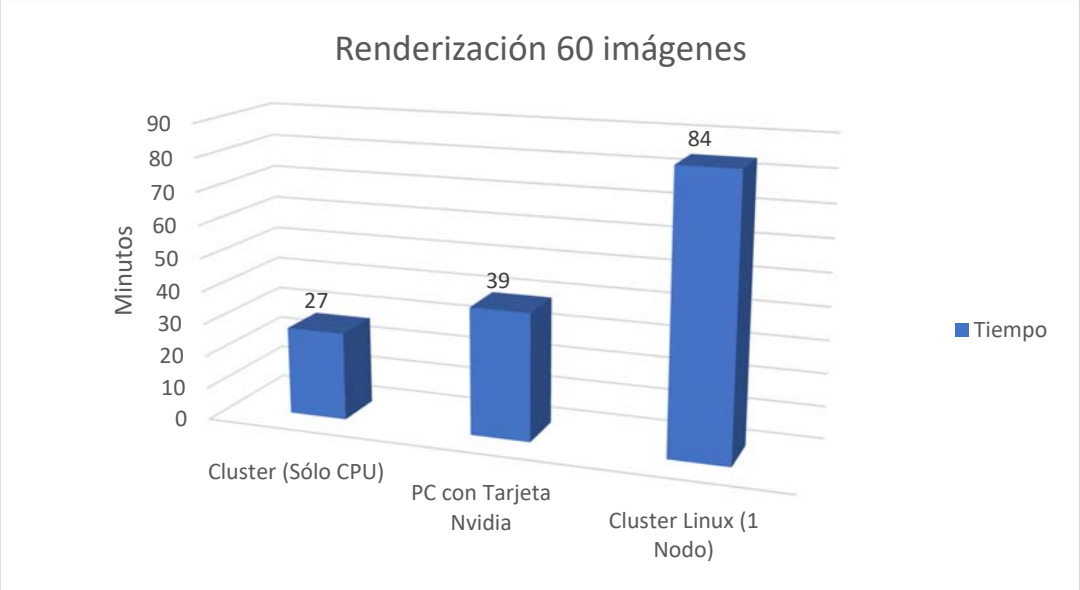

***Figura 11.** Comparación de tiempo en procesamiento de imágenes*

Se pudo observar que, sobre el clúster casero de tan solo 3 equipos, el tiempo que tardó en terminar la tarea de renderizado, fue menor al tiempo que tardó el pc con altas especificaciones y el Clúster Linux de 1 nodo, por ello se podría decir que un clúster de mayor tamaño podría realizar esta tarea en un tiempo aún menor que el que se obtuvo al realizar esta prueba.

A pesar de que la prueba realizada sobre el clúster Linux, muestra el peor de los tiempos, ya que este clúster solo tenía un nodo funcional, de esta manera se infiere que con dos nodos más de similares características de hardware el tiempo podría rondar los 28 minutos, dándonos un resultado de rendimiento muy cercano al clúster casero que estaba compuesto por 3 equipos.

CONCLUSIONES

De acuerdo a los resultados obtenidos podemos observar que una granja de renderizado puede ser más efectiva que un computador con altas prestaciones, además se le da un mejor uso a los recursos que puedan estar siendo desperdiciados en máquinas que alcanzan un estado ocioso e incluso darle vida a equipos que ya habían sido descartados, debido a que son demasiado viejos o lentos para el uso en tareas de alto procesamiento.

Las tareas complejas se pueden convertir en tareas sencillas que a su vez logran ser procesadas por múltiples equipos lo cual redunda en una gran mejora en eficiencia y uso de recursos, por lo tanto realizar una implementación de HTCondor en el campus de una universidad brinda herramientas para procesamiento de tareas complejas, puesto que HTCondor brinda el poder necesario para ejecutar todo tipo de aplicaciones que se puedan paralelizar, brindado potencia, eficacia y mejorando costos, ya que las tareas que antes se debían contratar ahora se podrían ejecutar sobre una red propia cuya administración es más sencilla y hace que la herramienta esté disponible prácticamente para todos.

REFERENCIAS

[1] Universidad Técnica Federico Santa María, "*Computación en grilla", UTFSM,* 2012. [On-Line]. Disponible en: http://wiki.inf.utfsm.cl/index.php?title=Computaci%C3%B3n_en_grilla_(grid_computing)

[2] George Couloris *et al*, "Characterization of distributed systems," in *Distributed systems: Concepts and design,* 5th ed. Boston: Pearson, 2011, pp. 2-8.

[3] University of Wisconsin-Madison, "*What is HTCondor*", UW-Madison, 2012. [On-Line]. Disponible en: https://research.cs.wisc.edu/htcondor/description.html

[4] George Couloris *et al*, "Interprocess Communication," in *Distributed systems: Concepts and design,* 5th ed. Boston: Pearson, 2011, pp. 178-180.

[5] InsideHPC, "*What is HPC",* InsideHPC, 2017. [On-Line]. Disponible en: http://insidehpc.com/hpc-basic-training/what-is-hpc/

[6] Rocks Clusters, "About Rocks", University of California, 2014. [On-Line]. Disponible en: http://www.rocksclusters.org/wordpress/?page_id=57

[7] Blender, "*About Blender*", Blender, 2014. [On-Line]. Disponible en: https://www.blender.org/about/

[8] UW-Madison, "*HTCondor Dowloads"*, UW-Madison, 2017. [On-Line]. Disponible en: https://research.cs.wisc.edu/htcondor/downloads/

[9] Java, "*¿Qué es Java?",* Java, 2017. [On-Line]. Disponible en: https://www.java.com/es/about/whatis_java.jsp

[10] UW-Madison, "*Installation, Start Up, Shut Down, and Reconfiguration*", UW-Madison, 2017. [On-Line]. Disponible en: http://research.cs.wisc.edu/htcondor/manual/v8.6/index.html

[11] UW-Madison, "*Submitting a Job"*, UW-Madison, 2017. [On-Line]. Disponible en: http://research.cs.wisc.edu/htcondor/manual/v8.6/2_5Submitting_Job.html

[12] UW-Madison, "*Running a Job: the Steps To Take*", UW-Madison, 2017. [On-Line]. Disponible en: http://research.cs.wisc.edu/htcondor/manual/v8.6/2_4Running_Job.html

[13] UW- Madison, "*Command Reference Manual"*, UW-Madison, 2017. [On-Line]. Disponible en: http://research.cs.wisc.edu/htcondor/manual/v8.6/11_Command_Reference.html

[14] Condor GUI, Otto von Guericke University Magdeburg, Germany, 2016. [Online]. Disponible en: https://sourceforge.net/projects/condor-gui/

[15] Blender, "*Command Line",* Blender, 2014. [On-Line]. Disponible en: https://docs.blender.org/manual/de/dev/render/workflows/command_line.html

[16] Rocks Clusters, "*Installing a Rocks Cluster"*, University of California, 2017. [On-Line]. Disponible en: http://central6.rocksclusters.org/roll-documentation/base/6.1.1/installing.html

[17] Jorge Enrique Ruiz Navarro, "*Instalación de Lustre en Centos 6.5",* Systerminal, 2014. [On-line]. Disponible en: http://www.systerminal.com/2014/07/21/instalacion-de-lustre-en-centos-6-5/

[18] Rocks Clusters, "*Frequently Asked Questions"*, University of California, 2017. [On-Line]. Disponible en: http://central6.rocksclusters.org/roll-documentation/base/6.1.1/faq-configuration.html#EXPORT