Universidad del Azuay

Facultad de Ciencias de la Administración

Carrera de Ingeniería de Sistemas y Telemática

PROCESAMIENTO DE ALTAS PRESTACIONES (PARALELO Y DISTRIBUIDO) EN EL CLÚSTER DE CEDIA UTILIZANDO EL LENGUAJE R CON DATOS DE ÍNDICES CLIMÁTICOS PARA EL ECUADOR CONTINENTAL.

Autores:
Jonnathan Capelo Solano

Directores:
Daniela Ballari

Cuenca – Ecuador
2018

DEDICATORIA

El presente trabajo lo dedicamos principalmente a mis padres que gracias a su trabajo y sacrificio en todos estos años ya que gracias a su apoyo he logrado llegar hasta aquí
A mis hermanas y a mi novia por estar siempre presentes y por el apoyo moral que me brindaron en esta etapa de mi vida.

AGRADECIMIENTO

Agradezco a todas las personas que han hecho que este trabajo se realice con éxito.
Así mismo expresar mi agradecimiento a todos los profesores la escuela de Ingeniería de Sistemas y Telemática por ayudarme a lo largo de mi formación académica, con mención especial a mi tutora Daniela Ballari quien me ha guiado y brindado un valioso aporte para la realización de este trabajo. Y por supuesto, a la Universidad del Azuay, por enriquecerme en conocimiento y haberme brindado tantas oportunidades.

RESUMEN:

El Lenguaje R es especializado para el análisis de datos estadísticos, sin embargo cuando el volumen de datos es muy elevado, R se vuelve ineficiente y requiere de una cantidad considerable de tiempo para el procesamiento Por ello se han desarrollado librerías para el procesamiento en paralelo y distribuido siguiendo las directrices de High Performance Computing (HPC). A pesar de estos esfuerzos, la aplicación de las librerías es limitada, ya sea por el poco acceso a los equipos HPC o porque la documentación es escasa y demasiado técnica. Por ello, este trabajo implementó y documentó el uso de librerías de R para el procesamiento en paralelo y distribuido aplicado al análisis de teleconexiones e índices climáticos para el Ecuador continental. Se utilizaron series temporales de imágenes satelitales de precipitación de diez años e índices climáticos. Se utilizó la librería Parallel y RSLURM en el clúster HPC de CEDIA para aplicar correlación de Spearman y bootstrap. Se midieron los tiempos de procesamiento secuencial, en paralelo y distribuido, observando disminuciones considerables de tiempo a medida que se aumenta el número de procesadores y nodos.

Palabras clave: High Performance Computing, R, parallel, teleconexiones climáticas, clúster, RSLURM.

ABSTRACT

The R Language is specialized for the analysis of statistical data. However, when the volume of data is very high, R becomes inefficient and requires a considerable amount of processing time. Therefore, libraries for parallel and distributed processing have been developed based on the guidelines of High Performance Computing (HPC). Despite these efforts, the application of these libraries is scarce due to the limited access to HPC equipment, infrequent documentation or too technical documentation. This work implemented and documented the use of R libraries for parallel and distributed processing applied to the analysis of teleconnections and climate indices for Ecuador through the use of Parallel and Rslurm.

UNIVERSIDAD DEL AZUAY
Dpto. Idiomas

Translated by
Ing. Paul Arpi

ÍNDICE

1. INTRODUCCIÓN

El Lenguaje "R" se considera uno de los lenguajes de programación para el análisis de datos más populares, ya que fue desarrollado específicamente para el análisis de datos estadísticos (Schmidt, Ostrouchov, Chen, & Patel, 2012). Sin embargo, la capacidad de R para procesar grandes cantidades de datos y computacionalmente exigentes ha sido limitada. Esto se debe a que R lee los datos previamente en la memoria y esto está ligado con la cantidad de memoria que se tiene disponible en el equipo. En este contexto y siguiendo los lineamientos de los sistemas de alto rendimiento o High Performance Computing (HPC) se han desarrollado varias librerías para equipos de escritorio y HPC. Entre ellas constan Parallel (Leach, 2014), Snowfall (Knaus, 2015), Rpmi (Hao, 2018) y multicore (Eugster, 2009). A pesar de estos desarrollos, su aplicación continúa siendo limitada por el acceso a estos equipos y porque la información sobre su aplicación es escasa o demasiado técnica para personas que no disponen de un profundo conocimiento en R y HPC. Este es el caso de este trabajo en que se utilizarán librerías de R para procesamiento en HPC aplicado al análisis de teleconexiones e índices climáticos en el Ecuador.

Tradicionalmente, los sistemas de alto rendimiento fueron especialmente diseñados para el mercado High Performance Computing (HPC), por lo que eran muy superiores en cuanto a rendimiento y precio. Una de las rama HPC se ha centrado en el desarrollo y procesamiento de software para funcionar en estas computadoras (Jing, 2010). Hoy las cosas han cambiado considerablemente, y se puede encontrar en el mercado computadoras multiprocesador y multicore a precios mucho más accesibles. Así, los clúster de alto rendimiento se llegan a utilizar sólo si los requisitos de procesamiento aumenta, por ello es indispensable optimizar procesos y/o cálculos con ayuda de técnicas de procesamiento en paralelo (Hager & Wellein, 2011).

La identificación de patrones espaciales de los índices climáticos para el Ecuador Continental requiere de altos requisitos computacionales, para lo cual es necesario recurrir al procesamiento en paralelo y distribuido. Las teleconexiones climáticas muestran relaciones entre diferentes puntos en la tierra, las cuales permiten predecir variaciones en el clima y anticipar eventos como sequía e inundaciones (Carleton, 2003; Z. Liu & Alexander, 2007).

La teleconexión más importante y conocida en el Ecuador se relacionada con ENSO y la incidencia de la región del niño 1+2 (Océano Pacífico frente a las costas del Ecuador y Perú), ya que a lo largo de la historia ha sido el principal autor de los eventos climáticos suscitados en el Ecuador (Bendix, Gämmerler, Reudenbach, & Bendix, 2003; Morán-Tejeda et al., 2016; Vuille, Bradley, & Keimig, 2000). En el 2017 se llevó a cabo el proceso de corrección y reducción de escala de TRMM de 27km a 5km para el Ecuador con 117 estaciones meteorológicas in-situ e imágenes satelitales como covariables relacionadas con las propiedades de nubles de nubes, índice de vegetación normalizado y humedad del suelo (Ulloa, Ballari, Campozano, & Samaniego, 2017). Este producto corregido y de resolución espacial de 5km es una oportunidad única en nuestro país para representar de una manera continua los patrones espaciales de las teleconexiones climáticas.

El gran volumen de información a utilizar en este trabajo proviene de las imágenes temporales de 10 años (2001 a 2011), y esto se convierte en una limitación para los tiempos la ejecución de los métodos a aplicar. La parte más desafiante es aplicar métodos de correlación de Spearman y correlación con bootstrap pixel a pixel en las imágenes satelitales y visualizar estos resultados a través de un mapa. Esto implica que cada una de las 132 imágenes que conforman la serie temporal contiene 18000 píxeles. Al aplicar el primer método de correlación se requiere 8 segundos por cada índice climático; sin embargo, para el segundo, se necesita realizar múltiples iteraciones (por ejemplo 10000) y por ende requiere 55 minutos por cada índice climático. Por ello, debe recurrirse a la computación de altas prestaciones y a la implementación de procedimientos de procesamiento en paralelo y distribuido para acelerar los tiempos de ejecución. Hasta el momento existen varias herramientas en R que permiten la paralelización pero se han centrado principalmente en estaciones de trabajo. Sin embargo, es necesario escalar a un clúster de altas prestaciones para mayor eficiencia de estos procesos. Dado que la información de estas librerías es escasa, tales herramientas requieren ser exploradas y preparadas para procesar el tipo de datos a utilizar en este proyecto.

Por ello, el objetivo de este proyecto es explorar librerías en R para procesamiento en paralelo y distribuido e implementar los métodos de correlación de índices climáticos antes descritos. Si bien R ha sido adoptado como herramienta para el presente trabajo, el procesamiento de dichos datos requiere de grandes cantidades de cálculos y

procedimientos por lo que también es importante la optimización de estos en los scripts finales a realizar para lo cual se estudiaran las librerías de paralelismo para soluciones R existentes. El procesamiento del scripts para el ordenador personal se realizó en el sistema operativo de distribución Ubuntu 16.04 y el script en el clúster de Cedia con el sistema operativo de distribución Centos 7.4.1708.

2. ANTECEDENTES

2.1. El lenguaje R

El lenguaje R es un entorno de software libre desarrollado en los laboratorios bell por John Chambers para la computación de análisis estadístico. "*Se considera una de las plataformas más populares, ya que fue desarrollado específicamente para el análisis de datos estadísticos"* (Schmidt et al., 2012)**.** La evolución del lenguaje R ha permitido proporcionar una variedad de técnicas estadísticas como modelado lineal y no lineal, análisis de series de tiempo, visualización gráfica, etc.

R se encuentra disponible bajo los licencia de la fundación de software libre GNU-GPL. Por ello puede ser compilado y ejecutado en las distintas plataformas UNIX, Windows y MacOS. Además, esto permite que sea altamente extensible mediante el desarrollo de la comunidad de software libre, que proporcionan diferentes librerías de las cuales su principal contribuyente es el proyecto CRAN. CRAN tiene la función de estandarizar las diferentes librerías desarrolladas de esta forma se caracteriza como un sistema planificado y coherente de librerías para R en lugar de una simple acumulación incremental de herramientas para este lenguaje.

Unas de las mayores desventajas de R es la capacidad para procesar grandes cantidades de datos, la que a su vez ha sido limitada por la memoria disponible en el equipo que se esté usando. Esto es porque R almacena todos sus datos en memoria para poderlos procesar. Otro de los retos a superar en R es que solo utiliza un procesador, ya sea este físico o lógico, nativamente independiente de los que se tengan disponible (Jing, 2010).

2.2. Procesamiento paralelo y distribuido

Tradicionalmente el software construido ha sido escrito para la computación secuencial. De esta forma el software sigue una serie de pasos o instrucciones una atrás de otra para devolver un resultado. Esto quiere decir que solo se ejecuta una instrucción cuando se ha terminado la anterior. En consecuencia, la resolución de un problema complejo y con muchos pasos o instrucciones puede tomar un tiempo considerable.

La computación en paralelo es la forma en que las instrucciones a procesar se ejecutan simultáneamente. Se aplica el principio de "divide y vencerás" en cual permite fraccionar un proceso que demanda mucho tiempo al realizarse secuencialmente, en varios subprocesos que pueden hacerse simultáneamente, y así obtener el mismo resultado. Los esquemas de paralelización pueden ser a nivel de bits, por instrucciones, por tareas o por datos, los cuales se han aplicado sobretodo en la computación de altas prestaciones (Escalera Fariñas, Infante Abreu, André Ampuero, & Rosete Suárez, 2014). Este es el principal paradigma de la arquitectura de las computadoras en la actualidad. La mayoría de procesadores están construidos para ser multinúcleo y multihilo teniendo como mínimo, en cada computador, al menos dos núcleos físicos y dos lógicos, el cual permite trabajar con 4 hilos en un solo procesador. Por esta razón los clúster de alto rendimiento se llegan a utilizar solo si los requisitos de procesamiento aumentan, por ello es indispensable optimizar procesos y/o cálculos con ayuda de técnicas de procesamiento en paralelo (Hager & Wellein, 2011).

Sin embargo el aumento de procesadores en la computación en paralelo no reduce el tiempo de procesamiento infinitamente. Lo ideal fuera que al duplicar el número de procesadores el tiempo de ejecución del proceso se reduzca a la mitad, pero esta aceleración tiene un límite y a partir de este punto la aceleración es casi lineal, tal como lo expresa la ley de Amdahl. Ningún proceso puede ser paralelizado por completo dado que principalmente la parte no paralelizada será la que va a limitar su aceleración óptima. Otra limitante para la aceleración es la comunicación entre los diferentes nodos o procesadores ya que también necesitan un tiempo para pasar los datos de un nodo a otro y de la misma forma obtener sus resultados.

2.3. High Performance Computing (HPC)

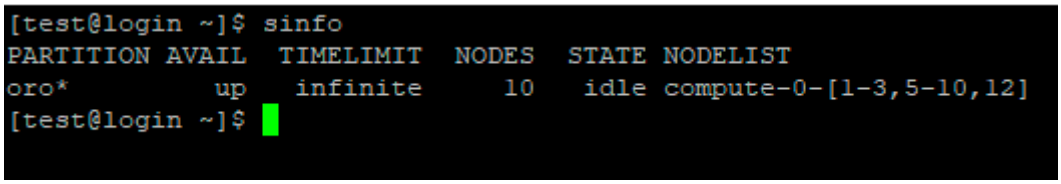
Es un conjunto de ordenadores interconectados entre sí, normalmente por una red de alta velocidad. Por lo que comparten la memoria, almacenamiento, y procesamiento, en consecuencia el acceso a ellos es como si fuesen una única computadora.

En la actualidad estas supercomputadoras están en las áreas donde se necesita procesar grandes cantidades de datos donde a un ordenador personal le costase mucho tiempo realizarlo. Además del desarrollo del hardware, una rama importante HPC se centra en el desarrollo y procesamiento de software para funcionar en estas computadoras (Jing, 2010).

Hasta hace unos años tener acceso a equipos de HPC era prácticamente imposible, por lo que son muy superiores en cuanto a rendimiento y precio. Sin embargo, hoy las cosas han cambiado considerablemente y se puede encontrar en el mercado supercomputadoras a precios mucho más accesibles, y a más del doble de capacidad que hace cinco años. Por tal razón un sin número de instituciones pueden poseer una de estas supercomputadoras como es el caso de CEDIA, el cual se usó en este trabajo. Este consta del nodo un para iniciar sesión en este se conectan los usuarios y por tal razón no es recomendable trabajar sobre este nodo. Además, el clúster consta con un total de doce nodos de cómputo y una conexión GigaEthernet. Los nodos poseen procesadores Intel de 16, 32 y 64 núcleos en total y 96 GB de memoria RAM. En lo referente a software, el clúster está basado en el sistema operativo Linux Centos y dispone de compiladores GNU e Intel para C, C++, Fortran y MPI.

Un comando importante a utilizar en un clúster es *sinfo* el cual muestra los nodos disponibles para poder trabajar. En la Figura 1 se pueden observar los nodos que están disponibles, que en este caso son del 1 al 3, 5 al 10 y el nodo 12.

Figura 1
Comando sinfo



Fuente: Elaboración propia

El principal reto a superar en este trabajo es la paralelización y distribución de los scripts en el lenguaje R. Cuando se trabaja con R es común encontrar que procesos o cálculos que son repetitivos, y por ello el proceso se vuelve lento dada la restricción del uso de un solo procesador en R. Por lo que se han desarrollado varias librerías para realizar procesamiento en paralelo y distribuido, aprovechando todo el potencial de los procesadores disponibles en nuestra computadora y en la computación de altas prestaciones. Una de las librerías más destacadas y que ya se encuentra disponible por defecto en la instalación de R Base es el paquete *parallel* (April, 2018) esta librería está desarrollada a partir de las librerías *Snow* (Tierney, 2016) y *multicore* (Eugster, 2009). Además, para el procesamiento distribuido se dispone de la librería *RSLURM* (Smorul, Carroll, & Carroll, 2017), que es útil cuando se realiza procesamiento en un entorno HPC. A continuación, se detallan las librerías *parallel y RSLURM* que son aplicadas en este trabajo.

2.4. Librería Parallel

Esta librería establece tareas para cada nodo, incluso en computadoras unidas por Ethernet. Esta espera que los nodos terminen sus tareas y pide los resultados a cada uno. Las computadoras no necesitan tener el mismo SO, se comunica por socialización para enviar los objetos. A continuación, se detallan algunas de las funciones que implementa *parallel*.

- mclapply(X, FUN, ..., mc.cores)

Esta función está disponible solo para sistemas GNU/Linux y MacOS. Se encarga de crear automáticamente la secuencia de trabajo para cada uno de los procesadores a utilizar, por lo que no es necesario exportar o reproducir los datos a cada uno de los procesadores. Esta función devuelve el resultado en forma de lista. Existen funciones similares a *mcapply* para el sistema operativo Windows pero no serán usadas en este trabajo, porque se ha utilizado únicamente en Sistema GNU/Linux, estas funciones son:

parSapply: Envía los procesos a cada procesador y se comunican solo cuando todos los procesadores hayan terminado. Devuelve su resultado en forma de vector que es la estructura más simple de agrupación de datos. Es decir, une todos los resultados de cada procesador y muestra en un solo vector.

parLapply: Su funcionamiento es similar a *parSapply*, pero los resultados de cada procesador son organizados en una estructura de lista de R.

La función *mclapply* utiliza el modelo Fork que se detalla a continuación.

FORK.- este modelo crea una copia exacta de todas las funciones y variables que se encuentran en el entorno de trabajo para ser usados en todos los núcleos del procesador. Este modelo solo funciona para sistemas GNU/Linux y MacOS.

PSOCK.- este modelo viene por defecto en la librería para las funciones que se detallan a continuación. Están pensadas para funcionar en sistemas Windows por lo que adicionalmente se necesita usar una función para declarar las variables y funciones personalizadas en cada uno de los procesadores.

Tabla 1
Descripción de la función mclapply

Argumentos	**Descripción**
X	Un vector o lista de expresiones que será aplicada a una función.
FUN	Función que será aplicada a cada elemento de X en paralelo.
mc.cores	Indica la cantidad de núcleos a usar en la paralelización, esta requiere de al menos 2 núcleos para la paralelización

2.5. Librería RSLURM

Simple Linux Utility for Resource Management (*SLURM*) fue desarrollado para la comunicación entre los nodos de un clúster. Su código fue escrito en el lenguaje C bajo la licencia GPL y se lo puede encontrar en el repositorio de *Github* https://github.com/SESYNC-ci/rslurm. También está provisto de *Autoconf* que realiza una adaptación automática al entorno en el que será instalado Sin embargo, también puede ser configurado según requerimientos personalizados. Además, no necesita modificaciones del *kernel* para su funcionamiento. Funciona tanto en clústeres Linux grandes o pequeños, por lo que es altamente escalable. *SLURM* es el encargado de administrar la carga de trabajo en el clúster, cual asigna un uso exclusivo de los recursos durante el tiempo que dure el trabajo. Es común que, aunque un usuario no use el 100% de los recursos, otro usuario deba esperar a que el anterior termine para poder realizar su trabajo. Provee comandos a nivel de consola para el monitoreo de las tareas ejecutadas, además de coordinar diferentes tareas en cola de tareas pendientes.

Para este trabajo se usaron las siguientes funciones

- **`slurm_apply(f, params, jobname = NA, nodes = 2, cpus_per_node = 2, add_objects = NULL, pkgs = rev(.packages()),slurm_options = list(), submit = TRUE)`**

Esta función permite la creación del trabajo y los archivos necesarios para ejecutar en el clúster, el script Bash crea un *job array* con el cual *SLURM* configurara cada nodo del clúster. Estos generan resultados en cada uno por lo que es necesario unir sus resultados con la función *get_slurm_out*. Es importante saber que los parámetros de la función deben tener el mismo nombre o corresponder a las columnas de *params.*

Tabla 2
Descripción de la función slurm-apply

Argumentos	Descripción
f	Función que será aplicada a cada iteración en paralelo.
params	Data frame de valores a aplicar en f cada columna corresponde a un parámetro de f.
jobname	Indica el nombre del proceso, si este no se asigna este tendrá un valor aleatorio de forma "slr***".
nodes	Indica el número de nodos que usara el proceso.
cpus_oer_node	Indica el número de núcleos por nodos que usara el proceso.
add_objects	Carga objetos R adicionales para f.
pkgs	Carga paquetes adicionales a usar en f.
slurm_options	Se usó para establecer la lista de nodos del clúster a usar.
submit	Envía un mensaje que el script debe ser ejecutado o solo creado para después ejecutarlo.

- **`print_job_status(slr_job)`**

Esta función imprime el estado de un proceso de *SLURM* e indica si este se ha completado.

Tabla 3
Descripción de la función print_job_status

Argumentos	Descripción
slr_job	Es el nombre del proceso en ejecución.

- **`get_slurm_out(slr_job, outtype = "raw", wait = TRUE)`**

Esta función lee todos los archivos de salida de cada uno de los nodos de un proceso de *SLURM* especifico este puede devolver el resultado en forma de data frame o lista.

Tabla 4
Descripción de la función get_slurm_out

Argumentos	Descripción
slr_job	Establece el nombre del proceso Slurm.
outtype	Establece el tipo de los datos que retornara, data frame= “table”, lista= “raw”
wait	Indica si bloquea R hasta terminar de unir los resultados.

- `cleanup_files(slr_job, wait = TRUE)`

Borra todos los archivos que ha generado *slurm_apply*

Tabla 5
Descripción de la función cleanup_files

Argumentos	Descripción
slr_job	Establece el nombre del proceso Slurm.
wait	Indica si bloquea R hasta terminar de unir los resultados.

Existen dos comandos básicos de monitoreo desde consola cuando se ejecuta un proceso con *SLURM*:

Squeue: muestra los procesos que se están ejecutándose y los que se encuentran en la cola de tareas.

Scancel: ayuda a forzar la detención de la tarea creada.

3. MATERIALES

3.1. Teleconexiones Climaticas

Un índice climático es un valor que puede ser usado para describir el estado y los cambios en el sistema climático (Quiroz, 2011). Estas muestran relaciones remotas y a gran escala entre puntos distantes en la tierra. Son conocidas por producir anomalías y variabilidad en el clima superficial de una región, entre ellas la precipitación (Carleton, 2003; Z. Liu & Alexander, 2007). Una de las teleconexiones climáticas más conocidas y estudiadas son las relacionadas con ENSO - El Niño South Oscilation (Diaz, Hoerling, & Eischeid, 2001), que surge de la interacción periódica entre la atmósfera y el océano en el Pacífico tropical. Otras teleconexiones que también influencian la variabilidad climática superficial (Fierro, 2014) son la Oscilación Ártica (AO) que es un modo de variabilidad climática a gran escala, también conocido como el modo anular del Hemisferio Norte.

Las teleconexiones climáticas han sido ampliamente estudiadas desde comienzos del siglo XX (e.g. Ångström, 1935). A pesar de la larga trayectoria de estudios al respecto, es reconocido que la influencia de las teleconexiones en el clima local es altamente variable a través de grandes áreas geográficas (Kiem & Franks, 2001). Por ello, ciertas teleconexiones pueden evidenciarse sólo en ciertas regiones y, además, en ciertos períodos de tiempo o estacionalidades. Esto requiere que en cada región en particular se

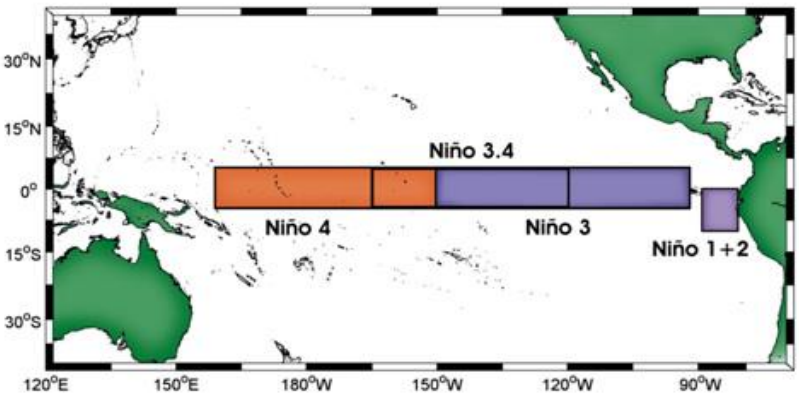
desarrollen estudios para detectar las teleconexiones presentes. La siguiente tabla detalla los índices climáticos considerados, que fueron seleccionados a partir de los trabajos de (Córdoba-Machado, Palomino-Lemus, Gámiz-Fortis, Castro-Diez, & Esteban-Parra, 2016; Fierro, 2014; Y.-C. Liu, Di, Chen, & DaMassa, 2018). La Tabla 6 muestra los índices climáticos utilizados:

Tabla 6
Descripción de los índices climáticos usados

Índice	Nombre	Acceso	Descripción
Niño 1+2	El Niño región 1+2	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este índice es la temperatura del mar en el extremo este del pacifico tropical
Niño 3	El Niño región 3	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este índice es la temperatura del mar en el este del pacifico tropical
Niño 4	El Niño región 4	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este índice es la temperatura del mar en el centro del pacifico tropical
Niño 3.4	El Niño región 3,4	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este índice es la temperatura del mar en el centro este del pacifico tropical
SOI	Southern Oscillation Index	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este valor calcula la diferencia entre la presión atmosférica estandarizada, medida al nivel del mar, entre Tahití y Darwin
TNI	Trans-Niño	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este valor calcula la diferencia entre las anomalías de región niño 1+2 y niño 3.4
MEI	ENSO multivariado	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Este índice es una combinación lineal de seis variables atmosféricas y marinas medidas en la región Niño 3. Los datos se presentan en periodos bimestrales.
ONI	Niño oceánico	https://www.esrl.noaa.gov/psd/data/climateindices/list/	Índice de El Niño oceánico mensual. Es una serie de tiempo elaborada a partir de las anomalías de la temperatura superficial marina reconstruidas por la NOAA a partir de datos medidos in situ, en la región

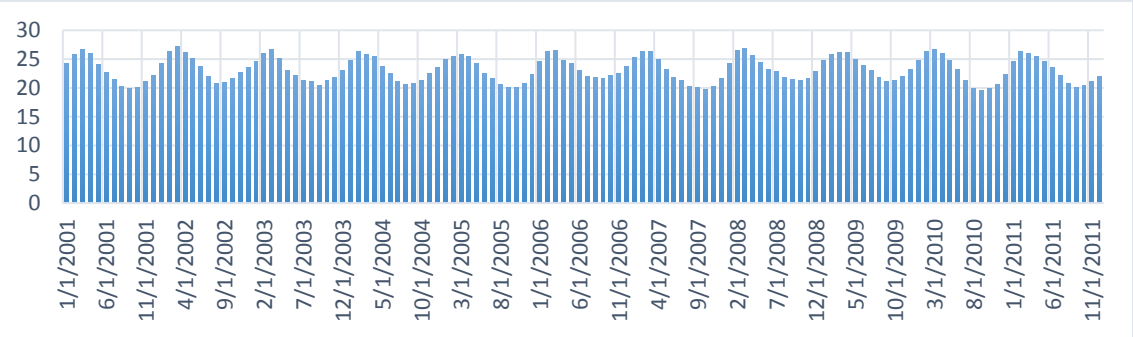
La figura 2 muestra las regiones del pacifico de las cuales se obtienen los datos. Ver más en http://www.inocar.mil.ec/modelamiento/elnino/nino_generalidades.php.

Figura 2.
Regiones Niño 1+2, Niño 3, Niño 3.4 y Niño 4.



En la Figura 3 la variación de la temperatura en la región del niño 1+2 en el periodo 2001-2011.

Figura 3.
Índice Niño 1+2, período 2001-2011.



3.2. Series de Tiempo de datos satelitales con reducción de escala

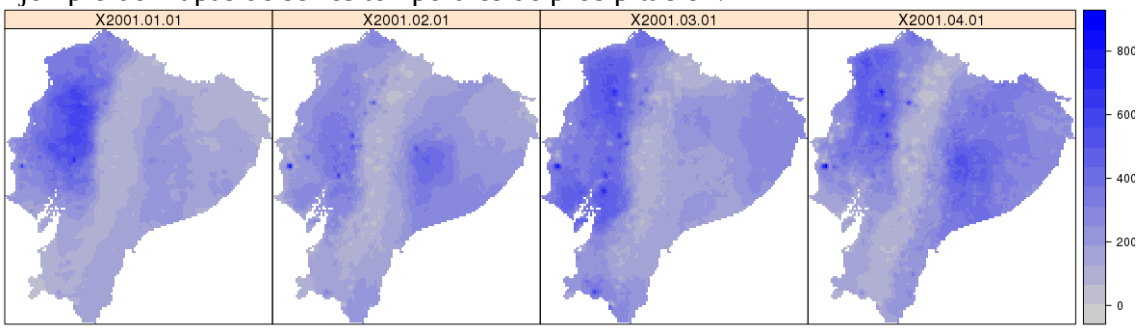
Esta serie temporal de imágenes satelitales contiene la precipitación en el ecuador continental desde los años 2001-2011. Son imágenes satelitales TRMM con reducción de escala a 5km. Este es el resultado del análisis de reducción de escala con covariables. Para más información consultar: Ulloa, Jacinto; Ballari, Daniela; Campozano, Lenin; Samaniego, Esteban. 2017. "Two-Step Downscaling of TRMM 3b43 V7 Precipitation in Contrasting Climatic Regions with Sparse Monitoring: The Case of Ecuador in Tropical South America." Remote Sens. 9, no. 7: 758.

En la tabla 7 se visualiza una muestra los datos de la serie temporal. A su vez, la figura 4 muestra en forma de mapas los datos de la tabla 7

Tabla 7
Serie temporal de precipitación de imágenes satelitales. Filas corresponden con pixeles y columnas con fechas.

X2001.01.01	X2001.02.01	X2001.03.01	X2001.04.01	X2001.05.01	X2001.06.01	X2001.07.01	X2001.08.01
291.6214	258.6525	371.9906	286.4558	612.0491	184.0009	66.54870	21.00197
306.7239	269.8632	392.0780	346.6487	666.5654	259.7679	113.29682	22.69778
268.1269	221.6607	354.7491	282.1624	418.2309	169.1354	118.79079	22.55301
291.3280	254.2240	370.2113	294.3328	611.4422	199.6670	84.72182	21.47448
307.3873	272.8849	398.9675	356.8422	642.2419	230.9799	122.20178	23.45309
326.4174	288.4452	404.3640	369.0166	608.4638	234.0149	113.33208	21.34983

Figura 4
Ejemplo de mapas de series temporales de precipitación.



4. MÉTODO.

Se utilizó métodos estadísticos de correlación para relacionar los índices climáticos con las series de imágenes satelitales de precipitación. En particular se aplicó correlación simple no paramétrica de Spearman y correlación con bootstrap para determinar intervalos de confianza de esta correlación. Este análisis se realizó pixel a pixel, donde cada uno representa una resolución espacial de 5km sobre el terreno. Para esto se utilizó el lenguaje R que es especializado para este tipo de análisis a realizar. Dado el volumen de datos (9919 pixel por mes y por 10 años) y el número de repeticiones (1000) para el análisis con boostrap, se realizó la paralelización en el Cluster de CEDIA.

A continuación, primero se describe la función de boot y luego los tres scripts desarrollados: secuencial, paralelizado y distribuido. Además, se detalla la evaluación del rendimiento de los scripts.

4.1. Función boot

- **`boot(data=data, statistic=F, R=1000, parallel="multicore",ncpus=3)`**

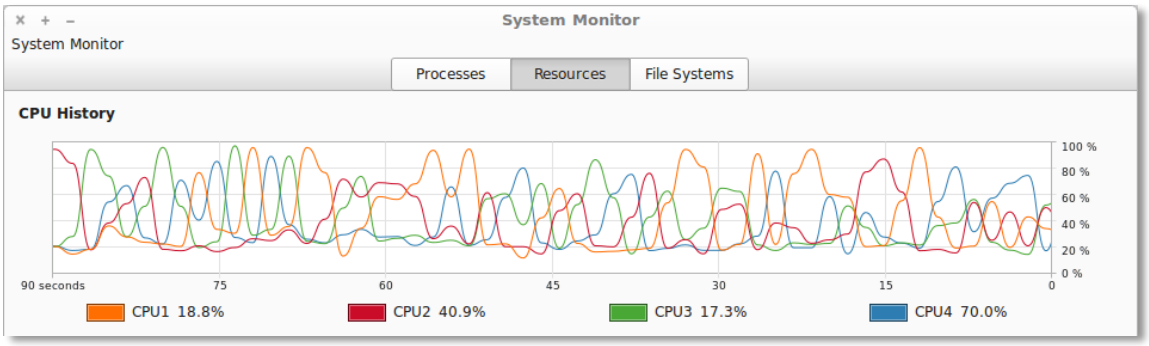
Tabla 8
Descripción de la función boot

Argumentos	Descripción
data	Los datos con los que trabajara statistic
statistic	Se establece la función que se usara en la función boot
R	Indica el número de repeticiónes que realizará la función boot.
parallel	Indica que librería usar multicore o snow o ninguna, para paralelización implícita.
ncpus	Indica el número de cpus a usar

Si bien la función boot puede utilizar paralelización implícita, no es recomendable ya que esta no usa el 100 % de los recursos que se le establece. Por ello el script de paralelización de este trabajo no utiliza esta funcionalidad.

La figura 5 es un gráfico de *System Monitor* del paquete GNU del sistema Ubuntu. Esta muestra el uso de los recursos del ordenador con función boot, en la que se puede observar que no necesariamente paraleliza los procesos, sino que realiza una alternancia de los recursos utilizados por esta función.

Figura 5.
Paralelización implícita de la función boot.



4.2. Scripts

Se desarrollaron tres scripts:

- Script base para procesamiento secuencial
- Script para procesamiento en paralelo
- Script para procesamiento en distribuido

1. Script base para procesamiento secuencial.

A continuación, se describe el script, que para facilitar la comprensión fue dividido en 3 partes: Carga de datos, correlación de Spearman, y bootstrap.

- Parte 1 Carga de datos

```
#########################
#Script base para realizar paralelización de procesos
#Daniela Ballari
#26 de abril de 2018
#Universidad del Azuay
#################
```

```
#Se realiza la carga de las librerías
library(raster)
library(sp)

#2 Se establece el directorio donde se encuentra nuestros datos
setwd("/media/climateindexes")

#Se realiza la lectura de los mapas
load("predEcuadorCorr.Rdata")
prec5km <- predEcuadorCorr
#Se visualiza para confirmar su lectura
spplot(prec5km[[1:4]],col.regions=colorRampPalette(c('gray80','blue')))

#Se convierte de raster a SpatialPointsDataFrame para obtener las coordenadas para la posterior visualización de la correlación con el índice
prec5km.df<- as(prec5km, "SpatialPointsDataFrame")
head(prec5km.df@data)

#Se crea el data frame “S” de los datos el cual usaremos para nuestros procesos
s=as.data.frame(t(prec5km.df@data))

#Se verifica si los datos presentan una distribución normalidad
#Se realiza la carga de la librería
library(nortest)

#se procede a verificar
normalidad<- NULL
for(i in 1:length(s)){
  normalidad[i]<- ad.test(as.numeric(s[,i]))$p.value>0.05
}
table(normalidad)*100/nrow(prec5km.df)
#   FALSE     TRUE  %
#  82.59      17.4
# El 82% de los pixeles no presentan una distribución de tipo normal, por ello se utilizarán métodos de correlación no paramétricos, en particular correlación de spearman

#Se realiza la cargar de los índices climáticos y se selecciona las fechas entre 2001 y 2011
setwd("/media/paul/Ubuntu-MATE 16.04.2 LTS amd64/google Drive/uda/Tesis/New folder/climateindexes/climateindexes")

index12 <- read.table("nino12.data", sep="", dec=".", header=FALSE)
index12 <- as.vector(t(subset(index12, V1>=2001 & V1<=2011,)[,2:13]))

indextni <- read.table("tni.data", sep="", dec=".", header=FALSE)
indextni <- as.vector(t(subset(indextni, V1>=2001 & V1<=2011,)[,2:13]))

index34 <- read.table("nina34.data", sep="", dec=".", header=FALSE)
index34 <- as.vector(t(subset(index34, V1>=2001 & V1<=2011,)[,2:13]))

indexmei <- read.table("mei.data", sep="", dec=".", header=FALSE)
indexmei <- as.vector(t(subset(indexmei, V1>=2001 & V1<=2011,)[,2:13]))

indexoni <- read.table("oni.data", sep="", dec=".", header=FALSE)
indexoni <- as.vector(t(subset(indexoni, V1>=2001 & V1<=2011,)[,2:13]))

indexpdo <- read.table("pdo.data", sep="", dec=".", header=FALSE)
indexpdo <- as.vector(t(subset(indexpdo, V1>=2001 & V1<=2011,)[,2:13]))
```

```
indexsoi <- read.table("soi.data", sep="", dec=".", header=FALSE)
indexsoi <- as.vector(t(subset(indexsoi, V1>=2001 & V1<=2011,)[,2:13]))

indextna <- read.table("tna.data", sep="", dec=".", header=FALSE)
indextna <- as.vector(t(subset(indextna, V1>=2001 & V1<=2011,)[,2:13]))
```

- Parte 2 Se procede a realizar la correlación de Spearman

```
## Se obtienen los datos de las coordenadas del raster
cormap<-prec5km.df[,1]
gridded(cormap)<-  TRUE

#Se asigna a la variable 'start,time' la fecha y hora actual ("2018-09-13 04:35:57 -05")
start.time <- Sys.time()

# Se realiza la correlación con el índice "index12" por medio de un bucle for para iterar sobre los datos, este resultado será guardado en la variable cor12.
#NINO 12
cor12<- NULL
for(i in 1:length(s)){
  cor12[i]<- cor(as.numeric(s[,i]), index12, method = "spearman")
}

end.time <- Sys.time()
time.taken <- end.time - start.time
time.taken
# Se imprime la diferencia de del tiempo
#Time difference of 9.469945 secs

# Se grafica el Boxplot de dispersión de la correlación para todo el Ecuador
boxplot(cor12)

#Se añade este resultado de correlación a la variable "cormap" que se fue cargada previamente con las coordenadas del mapa.
cormap$nino12<-cor12
spplot(cormap, "nino12",col.regions=colorRampPalette(c('red','gray80','blue')),
      at=c(-1,-.9,-.8,-.7,-.6,-.5,-.4,-.3,-.2,-.1,0,.1,.2,.3,.4,.5,.6,.7,.8,.9,1))
```

- Parte 3 Correlación aplicando Bootstrap

```
# Se carga la librería boot la cual nos sirve para aplicar la correlación con bootstrap para este trabajo se estableció para cada iteración con 1000 muestras
library(boot)

#función de correlación para que sea usado por bootstrap
bcorr = function(datos, i)
        {cor(datos[,2][i], datos[,1][i], method="spearman")}

#Se establece en la variable la hora actual para la toma de tiempo
start.time2 <- Sys.time()

#Se establece el tamaño que ocuparan las variables de este proceso
bcor12<- data.frame(matrix(,nrow = nrow(prec5km.df), ncol = 1000))
bci12<- data.frame(matrix(,nrow = nrow(prec5km.df), ncol = 2))

#Este proceso se realiza mediante un bucle for el cual iterara cada pixel
for(i in 1:nrow(prec5km.df)){
```

```
  #Se crea la variable datos cual es un data frame de la serie temporal de un pixel con el índice.
  datos<- cbind(as.numeric(prec5km.df@data[i,]), index12)
  #Se realiza la correlación aplicando bootstrap para el cual se ayudó de la paralelización que
tiene la propia función pero la cual no es la más óptima.
  cor.b = boot(data=datos, statistic=bcorr, R=1000,parallel="multicore", ncpus = 3)
#Se guarda la correlación generada por cada iteración bootstrap
  bcor12[i,]<- as.numeric(t(cor.b$t))
# Se obtiene y se guarda el intervalo de confianza de bootstrap mayor al 95%
  bci12[i,]<-boot.ci(cor.b, conf=0.95, type = "bca")$bca[4:5]
}

# Se obtiene la diferencia del tiempo de ejecución de este.
end.time2 <- Sys.time()
time.taken2 <- end.time2 - start.time2
time.taken2
```

2. Script con procesamiento en paralelo.

Este script cuenta con procesamiento en paralelo para un ordenador personal, usando todos los procesadores o cores disponibles. Para este script se utilizó de igual manera las dos primeras partes del script anterior, por lo que a continuación sólo se describe la optimización de la tercera parte en la que se aplica la correlación con bootstrap paralelizado.

- Parte 3 Paralelización de la correlación aplicando Bootstrap

```
#Se crea la función funboot, esta función será la encargada de realizar todo el proceso de
correlación con bootstrap
 funboot<-function(i){

#Se crea la función bcorr que realizara la correlación.
  bcorr=function(datos, i){cor(datos[,2][i], datos[,1][i], method="spearman")}

#Se guarda la correlación en cor12 que es el retorno de la function boot.
        #data: dataframe de la serie temporal de un pixel con el índice index12.
        #statistic: se pasa nuestra función bcorr.
        #R: la cantidad de muestras con las que se realizara la correlación.
  cor12<-(boot(data=cbind(s[,i], index$ind.12), statistic=bcorr, R=1000))

#Se guarda en ci.12 el intervalo de confianza de la correlación.
  ci.12<-t(boot.ci(cor12, conf=0.95, type = "bca")$bca[4:5])

#Se obtiene t0 de la correlación.
  cor.12<-cor12$t0

#se retorna en forma de data.frame t0 de la correlación y el intervalo de confianza
  return(data.frame(ci.12=ci.12,cor.12=cor.12))
}

# Se inicializa la variable que guardara nuestra respuesta
result=NULL
```

```
#Se establece en a la hora actual
a=Sys.time()

# Para la paralelización se usa mclapply y se guardará en result que será de tipo lista.
        #El primer argumento es el tamaño del dataframe “s”
        #FUN: establecemos la función funboot antes creada
        #mc.cores: establecemos en 3 ya que nuestro ordenador cuenta con 4 cores
result=mclapply(1: length(s),FUN =funboot, mc.cores = 3)

#por ultimo sacamos la diferencia del tiempo actual con a.
Sys.time()-a
```

3. Script para procesamiento distribuido

Este script se ejecuta en el clúster de CEDIA para el cual se usó la librería *RSLURM*. Esta librería permite usar los diferentes nodos y procesadores con los que cuenta este clúster. Al igual que el script anterior se reutiliza la carga de datos del primer script y la función *funboot* del segundo script.

```
#Se carga la librería RSlurm
library(rslurm)

#Creamos params, será un dataframe con una secuencia del tamaño de nuestros datos.
params <- data.frame("i" = seq(length(s)))

#Creamos el job con los siguientes parámetros
#funboot: la función que creamos anteriormente,
#params: nuestra función del mismo nombre
#jobname: asignamos un nombre con cual identificarlo
#nodes: asignamos 4 nodos para el procesamiento,
#cpus_per_node: asignamos 10 cpus por nodo lo que nos daría en total 40 cpus,
#add_objects: agregamos un vector con los nombres de nuestros datos.
#slurm_options: pasamos una lista de los nodos que no estaban disponibles ese momento.
#submit : establecemos en true para que se ejecute en seguida

sjob21 <- slurm_apply(funboot, params =params , jobname = 'test_correlacion',
                      nodes = 4, cpus_per_node = 10,
                      add_objects = c("s","index"),
                      slurm_options = list(exclude = c("compute-0-6","compute-0-1","compute-0-2")),
                      submit = TRUE)

#Imprimimos el status del job de esta manera podemos saber si se está ejecutando
print_job_status(sjob21)

#Determinamos el tiempo de ejecución del Job.
for (i in 1:1000) {
  print_job_status(sjob21)
  Sys.sleep(0.5)
}

#Podemos listar los archivos que contienen nuestros resultados se crea uno por cada nodo.
list.files('_rslurm_test_correlacion', 'results')

#Recuperamos los resultados de nuestros archivos en forma de dataframe.
res <- get_slurm_out(sjob21, outtype = 'table')
```

```
#Eliminamos todos los archivos que se han creado.
cleanup_files(sjob21)

#Obtenemos los valores de la correlación y el intervalo de confianza por separado e inicializamos
las variables sinino12 y finalnino12.
bcor12=res$cor.12
bci121=res$ci.12.1
bci122=res$ci.12.2

signino12=NULL
finalnino12=NULL

#creamos un bucle for para obtener solo los valores que están dentro el intervalo de confianza.
for(i in 1:length(bcor12)){
  #Guarda true o false si entre el intervalo de confianza existe el 0
  signino12[i]<- bci121[i]*bci122[i]>0
  #Crea la lista con los datos de bcor si signino12 es true o establece NA
  finalnino12[i] <- ifelse(signino12[i],bcor12[i],NA)
}

#Agregamos a nuestro cormap las 3 variables
cormap$finalnino12<-finalnino12
cormap$nino12<-bcor12
cormap$signino12<-signino12

#Graficamos nuestro resultado de la correlacion con bootstrap
spplot(cormap, c("nino12","signino12", "finalnino12"),
col.regions=colorRampPalette(c('red','gray80','blue')), at=c(-1,-.9,-.8,-.7,-.6,-.5,-.4,-.3,-.2,-
.1,0,.1,.2,.3,.4,.5,.6,.7,.8,.9,1))
```

4.3. Evaluación del rendimiento de los scripts

La evaluación del rendimiento de los scripts se realiza a través del tiempo que necesita R para completar un proceso. Para esto se usó la función *Sys.time()* que retorna la hora actual y permite calcular la diferencia de tiempos desde comienza hasta que finaliza un proceso.

Se asigna a la variable ‘start.time’ la fecha y hora actual, la que es devuelta en este formato "2018-09-13 04:35:57 -05"

```
start.time <- Sys.time()
   #Se realiza el proceso a medir
```

Se asigna a la variable ‘end.time’ la fecha y hora que se ha terminado el proceso

```
end.time <- Sys.time()
time.taken <- end.time - start.time
```

Se obtiene la diferencia entre start.time y end.time, la cual retorna el siguiente texto:

Time difference of 9.469945 secs

Para la evaluación del tiempo de ejecución en el Cluster no se encontró una librería que realice la toma de tiempo o métricas, ya que este funciona a nivel de comandos bash. Por lo que fue necesario utilizar un script que consta de un bucle para obtener el estado del proceso en ejecución cada medio segundo y, de esta manera, obtener un valor aproximado del tiempo en el que se termina de realizar dicha operación.

Se crea un bucle con una secuencia para el tiempo, en el que se estima que termine el proceso. Se imprime el estado del job o tarea, y se detiene al bucle por 0.5 segundos. El estado del job muestra el tiempo que se está ejecutando y cuando termina.

```
for (i in 1:1000) {
  print_job_status(sjob21)
  Sys.sleep(0.5)}
```

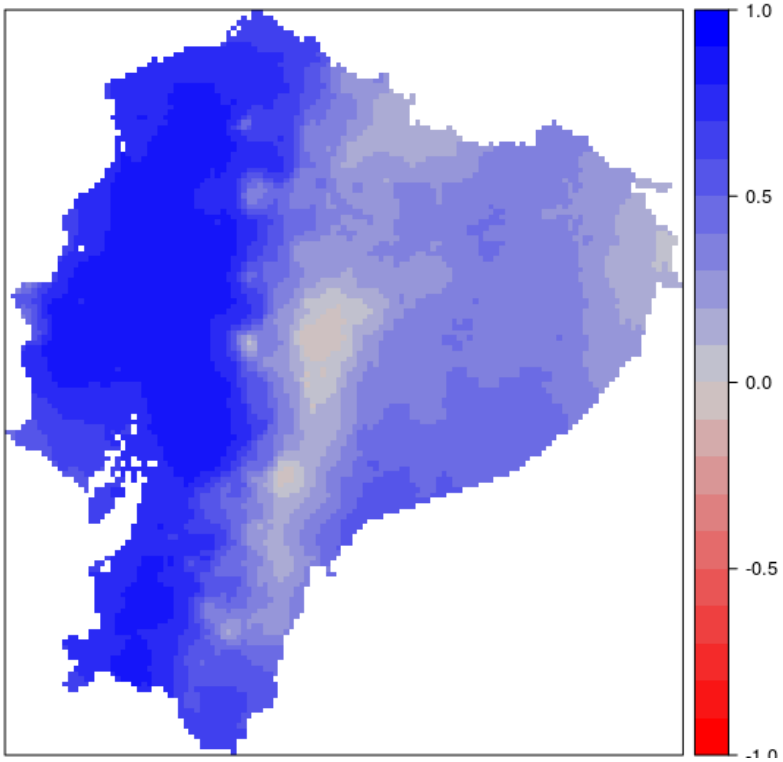
Finalmente, según como lo expresa la ley de Amdahl, la optimización del script tiene un límite en el procesamiento, ya que después de determinado número de cpus su aceleración será lineal debido principalmente a la porción de código no paralelizado y a la comunicación de los diferentes nodos, por lo que no tiene sentido seguir aumentando cpus, si lo que se ganará son unos pocos segundos. Por esta razón se realizaran pruebas para evaluar un el número cpus que se pueden usar con nuestros datos obteniendo una mejora significativa en el tiempo de ejecución de los scripts.

5. RESULTADOS Y DISCUSIÓN

5.1. Resultados de mapas

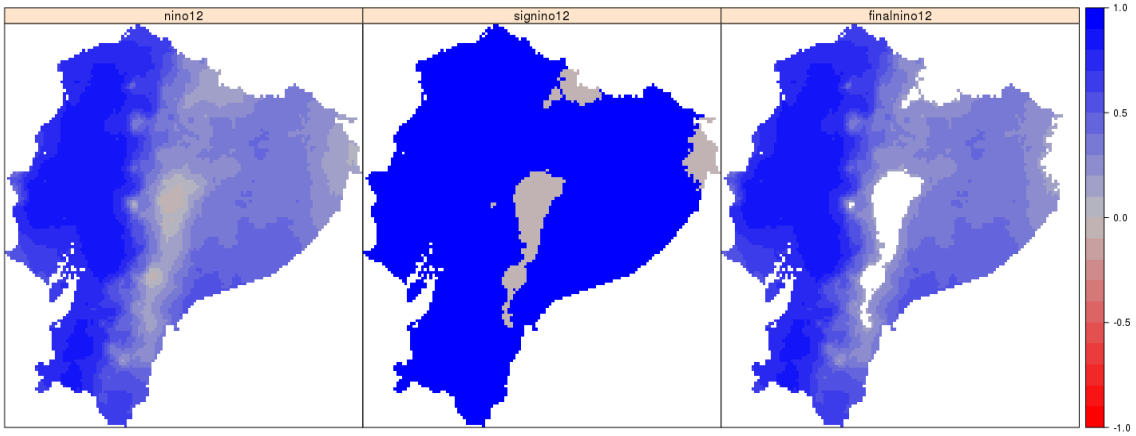
Los mapas obtenidos de los 3 scripts son similares, ya que independientemente que el procesamiento se haya realizado en un único procesador, paralelizado o distribuido, los tres procesos devuelven resultados idénticos. La figura 6 muestra la correlación de Spearman con la distribución espacial de la serie temporal de precipitación con el índice Niño 1+2. En ella se observa que la precipitación tiene correlación positiva en caso todo el Ecuador con el índice Niño 1+2, con valores superiores a 0.5 en la región costa. En varias zonas de la sierra se observa falta de correlación con valores cercanos a 0.

Figura 6.
Mapa de correlación de Spearman



La figura 7 muestra la correlación con Bootstrap con los datos distribución espacial de la serie temporal de precipitación con el índice Niño 1+2. Se puede observar, en primer lugar, la correlación de la precipitación con el índice Niño 1+2, en segundo lugar los sitios que en los que la correlación es significante al 95%; y en tercer lugar, se grafica la correlación sólo en los sitios con significancia estadística.

Figura 7.
Mapas de correlación de bootstrap

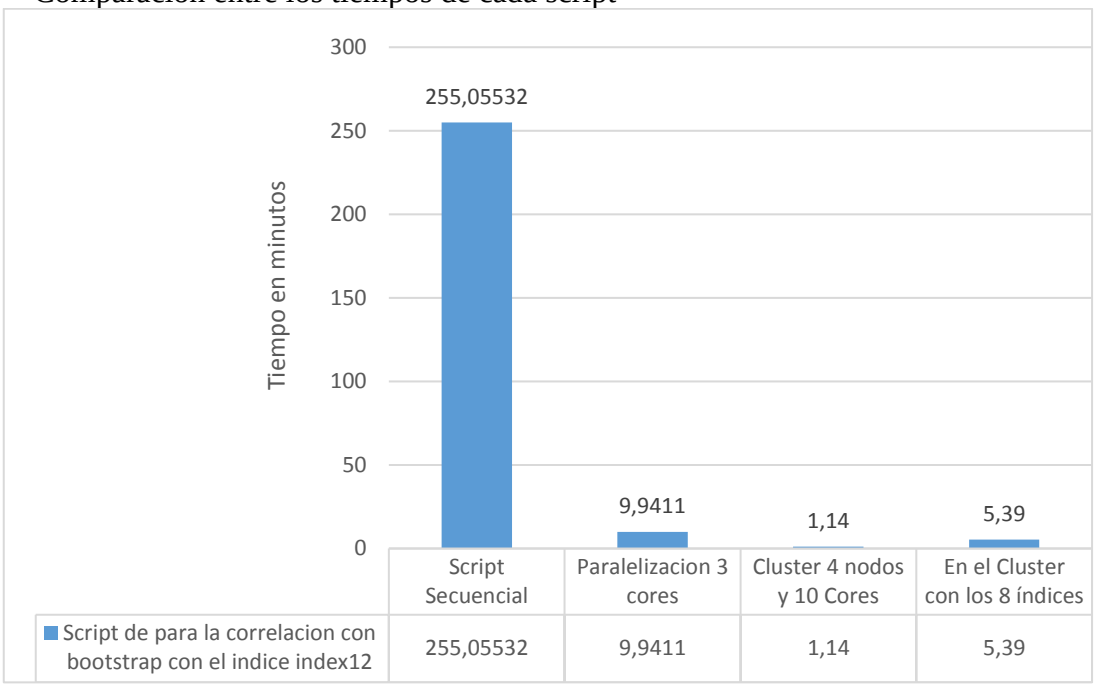


5.2. Tiempos de ejecución de los Scripts

La figura 8 muestra los tiempos de ejecución de los tres scripts. Las tres primeras columnas del gráfico muestran los tiempos únicamente para el índice Niño 1+2, y la última columna para todos los índices utilizados en este trabajo. Para un índice, el script secuencial requiere de 255 minutos para ejecutarse; el script paralelizado con 3 cores, de casi 10 minutos; y el script distribuido en 4 nodos con 10 cores cada uno, de 1 minuto. Al aplicar el script distribuido para los 8 índices se requirió de 5.4 minutos.

Es decir que la paralelización y optimización del código resultó en una disminución del tiempo muy importante, ya que el proceso secuencial es de aproximadamente cuatro horas y con la paralelización se obtiene el resultado en aproximadamente diez minutos. De igual forma se puede concluir que el procesamiento en el clúster disminuyó el tiempo de procesamiento en casi diez veces, ya que el resultado se puede obtener incluso en menos de un minuto. Así el proceso de paralelización y distribución de proceso permite ahorrar varias horas de espera en el procesamiento.

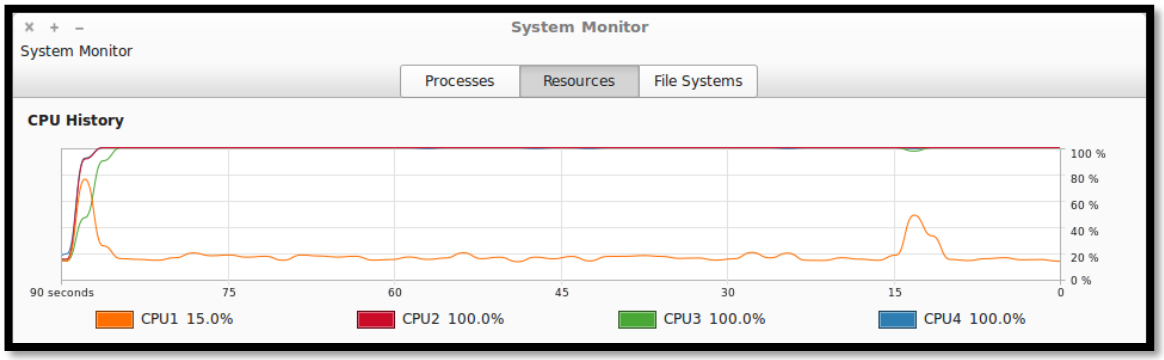
Figura 8
Comparación entre los tiempos de cada script



5.3. Utilización de los recurso en paralelo

La figura 9 muestra el estado de los recursos, estos fueron tomados con la herramienta del sistema Ubuntu: System-Monitor. Aquí se puede analizar el porqué del tiempo de procesamiento en el segundo script mejora significativamente en comparación con el script inicial. En este se pueden observar los cuatro hilos de ejecución que tiene el procesador, de los cuales se establecieron tres cpus (cpu 2, 3 y 4) para que realicen el procesamiento del script y dejando uno libre para el uso general del computador (cpu 1). Los tres cpus restantes en los que se están ejecutando el script, se observa en el lapso de 90 segundos, que el uso de estos sube rápidamente hasta llegar al cien por ciento y prácticamente se mantienen en ese nivel durante todo el procesamiento.

Figura 9
Uso de los cpus en paralelo



5.4. Tiempo de ejecución en el clúster de CEDIA con diferente número de procesadores.

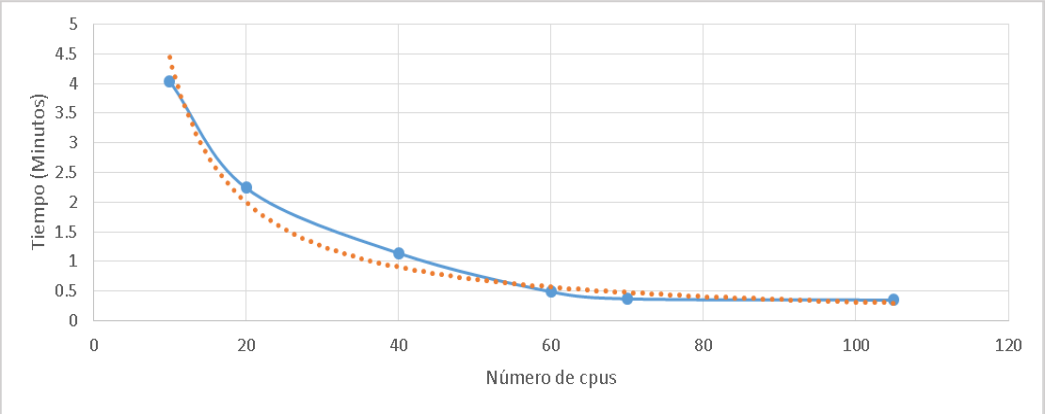
La siguiente tabla 9 y figura 10 muestran el tiempo de ejecución del tercer script con diferente número de cpus (10, 15, 20 y 24) y diferentes número de nodos del clúster (1, 2, 4, 6 y 7). Al efectuar esta comparación se puede determinar que, a medida que aumenta el número de nodos y de cpus, el tiempo de ejecución del script se ha reducido Sin embargo, cuando se alcanza los 6 nodos y 10 cpus el rendimiento no sigue aumentando de la misma forma. A partir de este punto, se observa una leve mejora, por lo tanto, la mejora podría ser imperceptible al aumentar más nodos o cpus.

Tabla 9
Tabla de cpus y nodos

No de Nodos	No de Cpus	Total	Tiempo(min)
1	10	10	4.04
2	10	20	2.24
4	10	40	1.14
6	10	60	0.49
7	10	70	0.37
7	15	105	0.35
7	20	140	0.32
7	24	168	0.31

En la figura 10 muestra la representación gráfica de las columnas Total y Tiempo de la tabla 9 y la línea punteada nos muestra la tendencia que siguen nuestros datos, que es una desaceleración. Esta tendencia es potencial que está determinada por la ecuación $y=38.088X^{-0.994}$ y con un $R^2 = 0.9427$ el cual es un ajuste bueno de la línea hacia nuestros datos.

Figura 10
Relación entre cpus y tiempo



6. CONCLUSIONES

El lenguaje R está especializado para el procesamiento de datos estadísticos, pero al estar basado en un lenguaje de programación, la curva de aprendizaje, para personas que no están familiarizadas con los paradigmas de programación de software, es muy alta pero aún con esta desventaja, en general, es fácil su aprendizaje.

La paralización y distribución de procesos en el clúster es muy recomendable dado que el tiempo de procesamiento disminuye significativamente. Para que esto suceda, se requiere que los scripts sean eficientes. Si el código es ineficiente, la mejora se verá disminuida lo que implica aumento en el tiempo de procesamiento. Una de las principales funciones que hacen que el código en R se torne ineficiente son los bucles. Cómo alternativa a esta problemática R proporciona la familia de los *apply, esta es una función propia del lenguaje que está optimizada para realizar trabajos repetitivos.

Vale destacar que en el procesamiento en un clúster, aunque se aumente la cantidad de nodos y/o procesadores con los que se ejecutara el script, no necesariamente crece el rendimiento de forma infinita, ya que este rendimiento se ve afectado por el tiempo de sobrecarga hacia los distintos nodos, por lo que resulta muy necesario probar los scripts con distintos números de cpus. Una buena manera de medir el rendimiento óptimo sería que el rendimiento en tiempo de procesamiento debe reducirse a la mitad al duplicar los procesadores, si esta meta no se consigue no es recomendable seguir aumentando los cpus, ya que el rendimiento no disminuirá significativamente. Tal como se vio en la figura de relación entre cpus y tiempo podemos determinar que para nuestros datos con 60 cpus el rendimiento aumenta y, a partir de este, el rendimiento se incrementa más lentamente.

Este trabajo comprobó que scripts en R previamente desarrollados de manera secuencial pueden ser fácilmente paralelizados y distribuidos, de manera que se ahorre tiempo de procesamiento. El material y procedimientos documentados en este trabajo pueden ser de utilidad a otros programadores e investigadores que deseen incurrir en la programación HPC dentro del entorno del lenguaje R.

7. BIBLIOGRAFÍA

Ångström, A. (1935). Teleconnections of climatic changes in present time. *Geografiska Annaler*, *17*(3–4), 242–258.

April, R. (2018). Package ‘ parallel ,’ 1–14.

Bendix, J., Gämmerler, S., Reudenbach, C., & Bendix, A. (2003). A Case Study on Rainfall Dynamics during El Niño/la Niña 1997/99 in Ecuador and Surrounding Areas as Inferred from GOES-8 and TRMM-PR Observations Niederschlagsdynamik während El Niño/La Niña 1997/99 in Ecuador und benachbarten Gebieten—Eine Fallstudie au. *Erdkunde*, 81–93.

Carleton, A. M. (2003). Atmospheric teleconnections involving the Southern Ocean. *Journal of Geophysical Research: Oceans*, *108*(C4).

Córdoba-Machado, S., Palomino-Lemus, R., Gámiz-Fortis, S. R., Castro-Diez, Y., & Esteban-Parra, M. J. (2016). Seasonal streamflow prediction in Colombia using atmospheric and oceanic patterns. *Journal of Hydrology*, *538*, 1–12.

Diaz, H. F., Hoerling, M. P., & Eischeid, J. K. (2001). ENSO variability, teleconnections and climate change. *International Journal of Climatology*, *21*(15), 1845–1862.

Escalera Fariñas, K., Infante Abreu, A. L., André Ampuero, M., & Rosete Suárez, A. (2014). Uso de estrategias de paralelización en algoritmos metaheur{\'\i}sticos para la conformación de equipos de software. *Revista Cubana de Ciencias Informáticas*, *8*(3), 56–72.

Eugster, M. J. A. (2009). The multicore package Script. *Imagine*, 1–6.

Fierro, A. O. (2014). Relationships between California rainfall variability and large-scale climate drivers. *International Journal of Climatology*, *34*(13), 3626–3640.

Hager, G., & Wellein, G. (2011). *Introduction to High Performance Computing for Scientists and Engineers*. *Book*. https://doi.org/10.1201/EBK1439811924

Hao, Y. (2018). Package ‘ Rmpi .’ *CRAN*, *2*, 56. Retrieved from http://www.stats.uwo.ca/faculty/yu/Rmpi

Jing, L. (2010). Parallel Computing with R and How to Use it on High Performance Computing Cluster, 1–11.

Knaus, J. (2015). Package ‘ snowfall .’

Leach, C. (2014). Introduction to parallel computing in R Parallel backends, 1–6. Retrieved from http://cran.r-project.org/web/views/HighPerformanceComputing.html%0A6

Liu, Y.-C., Di, P., Chen, S.-H., & DaMassa, J. (2018). Relationships of Rainy Season Precipitation and Temperature to Climate Indices in California: Long-Term Variability and Extreme Events. *Journal of Climate*, *31*(5), 1921–1942.

Liu, Z., & Alexander, M. (2007). Atmospheric bridge, oceanic tunnel, and global climatic teleconnections. *Reviews of Geophysics*, *45*(2).

Morán-Tejeda, E., Bazo, J., López-Moreno, J. I., Aguilar, E., Azor\’\in-Molina, C., Sanchez-Lorenzo, A., … others. (2016). Climate trends and variability in Ecuador (1966--2011). *International Journal of Climatology*, *36*(11), 3839–3855.

Quiroz, M. C. (2011). Anexo del Informe Técnico: Elaboración de un bolet{\’\i}n con información hidroclimática de los mares de México.". *Recuperado a Partir de Http://Www. Inapesca. Gob. Mx/Portal/Documentos/Publicaciones/BOLETINES/Hidroclimatico/INDIC ES-CLIMATICOS. Pdf.*

Schmidt, D., Ostrouchov, G., Chen, W.-C., & Patel, P. (2012). Tight coupling of r and distributed linear algebra for high-level programming with big data. In *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:* (pp. 811–815).

Smorul, M., Carroll, I., & Carroll, M. I. (2017). Package ‘ rslurm .’

Tierney, M. L. (2016). Package ‘ snow ,’ *0.4-2*.

Ulloa, J., Ballari, D., Campozano, L., & Samaniego, E. (2017). Two-step downscaling of Trmm 3b43 V7 precipitation in contrasting climatic regions with sparse monitoring: The case of Ecuador in Tropical South America. *Remote Sensing, 9*(7), 758.

Vuille, M., Bradley, R. S., & Keimig, F. (2000). Climate variability in the Andes of Ecuador and its relation to tropical Pacific and Atlantic sea surface temperature anomalies. *Journal of Climate, 13*(14), 2520–2535.

UNIVERSIDAD DEL AZUAY

Doctora María Elena Ramírez Aguilar, Secretaria de la Facultad de Ciencias de la Administración de la Universidad del Azuay

C E R T I F I C A:

Que, el Consejo de Facultad en sesión del 11 de mayo de 2018, conoció y aprobó la solicitud para realización del trabajo de titulación, presentada por :

Estudiante: Jonnathan Paul Capelo Solano (cód. 60799)
Tema: **"Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental"**
Previo a la obtención del título de Ingeniero de Sistemas y Telemática
Director: Agrim. Daniela Ballari
Tribunal: Ing. Marcos Orellana
Ing. Andrés Patiño

Plazo de presentación del trabajo de titulación, con la calificación del Director: seis meses a partir de la fecha de aprobación, esto es hasta el 11 de noviembre de 2018.

E INFORMA:

Que en aplicación de la Disposición General Cuarta del Reglamento de Régimen Académico vigente, en caso de que el estudiante no culmine y apruebe el trabajo de titulación luego de dos períodos académicos contados a partir de la fecha de culminación de estudios, deberá realizar la actualización de conocimientos previa a su titulación.

Cuenca, 14 de mayo de 2018

UNIVERSIDAD DEL AZUAY | Ciencias de la Administración Facultad
Dra. María Elena Ramírez Aguilar
Secretaria - Abogada

CONVOCATORIA

Por disposición de la Junta Académica de la escuela de Ingeniería de Sistemas y Telemática se convoca a los Miembros del Tribunal Examinador, a la sustentación del Protocolo del Trabajo de Titulación: **"Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental",** presentado por el estudiante Capelo Solano Jonnathan Paul con códigos 60799 , previa a la obtención del título de Ingeniero de Sistemas y Telemática, para el día **Miércoles, 09 de mayo de 2018 a las 12:00.**

Tomar en cuenta que posterior a la sustentación del Diseño del Trabajo de Titulación, por ningún concepto se puede realizar modificaciones ni cambios en los documentos; únicamente, en caso de diseño aprobado con modificación, el Director adjuntará al esquema un oficio indicando que se procede con los cambios sugeridos.

Cuenca, 09 de mayo de 2018

Dra. María Elena Ramírez Aguilar
Secretaria de la Facultad

Agrim. Daniela Ballari .....

Ing. Marcos Orellana .....

Ing. Andrés Patiño .....

ACTA

SUSTENTACIÓN DE PROTOCOLO/DENUNCIA DEL TRABAJO DE TITULACIÓN

Fecha de sustentación: **Miércoles, 09 de mayo de 2018 a las 12:00**

1.1. Nombre del estudiante: Capelo Solano Jonnathan Paul

1.2. Código: 60799

1.3. Director sugerido: Agrim. Daniela Ballari

1.4. Codirector (opcional): _____

1.4.1. Tribunal: Ing. Marcos Orellana e Ing. Andrés Patiño

1.4.2. Título propuesto: **"Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental"**

1.4.3. Aceptado sin modificaciones :

Aceptado.

1.4.4. Aceptado con las siguientes modificaciones:

_____

_____

_____

1.4.5. No aceptado

1.4.6. Justificación:

_____

_____

_____

Tribunal

Agrim. Daniela Ballari

Ing. Marcos Orellana

Ing. Andrés Patiño

Sr. Jonnathan Paul Capelo Solano

Dra. María Elena Ramírez Aguilar
Secretaria de la Facultad

RÚBRICA PARA LA EVALUACIÓN DEL PROTOCOLO DE TRABAJO DE TITULACIÓN
(Tribunal)

1.57. Nombre del estudiante: Capelo Solano Jonnathan Paul
1.58. Código : 60799
1.59. Director sugerido:
1.59.1. Codirector (opcional): Agrim. Daniela Ballari
1.60. Título propuesto: Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental
1.61. Revisores tribunal: Ing. Marcos Orellana e Ing. Andrés Patiño
1.62. Recomendaciones generales de la revisión:

	Cumple	No cumple
Problemática y/o pregunta de investigación		
155. ¿Presenta una descripción precisa y clara?	✓	
156. ¿Tiene relevancia profesional y social?	✓	
Objetivo general		
157. ¿Concuerda con el problema formulado?	✓	
158. ¿Se encuentra redactado en tiempo verbal infinitivo?	✓	
Objetivos específicos		
159. ¿Permiten cumplir con el objetivo general?	✓	
160. ¿Son comprobables cualitativa o cuantitativamente?	✓	
Metodología		
161. ¿Se encuentran disponibles los datos y materiales mencionados?	✓	
162. ¿Las actividades se presentan siguiendo una secuencia lógica?	✓	
163. ¿Las actividades permitirán la consecución de los objetivos específicos planteados?	✓	
164. ¿Las técnicas planteadas están de acuerdo con el tipo de investigación?	✓	
Resultados esperados		
165. ¿Son relevantes para resolver o contribuir con el problema formulado?	✓	
166. ¿Concuerdan con los objetivos específicos?	✓	
167. ¿Se detalla la forma de presentación de los resultados?	✓	
168. ¿Los resultados esperados son consecuencia, en todos los casos, de las actividades mencionadas?	✓	

Agrim. Daniela Ballari

Ing. Marcos Orellana

Ing. Andrés Patiño

Oficio Nro. 020-2018-DIST-UDA

Cuenca, 2 de mayo de 2018

Señor Ingeniero
Oswaldo Merchán Manzano
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN
Presente.-

De nuestras consideraciones:

La Junta Académica de la Escuela de Ingeniería de Sistemas y Telemática, reunida el día 2 de mayo del 2018, recibió el proyecto de tesis titulado "Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental", presentado por Jonnathan Capelo, estudiante de la Escuela de Ingeniería de Sistemas y Telemática, y revisado por la Ing. Daniela Ballari Ph.D. previo a la obtención del título de Ingeniero de Sistemas y Telemática.

Por lo expuesto, y de conformidad con el Reglamento de Graduación de la Facultad, recomendamos como director y responsable de aplicar cualquier modificación al diseño del trabajo de graduación posterior a la Ing. Daniela Ballari Ph.D., y como miembros del tribunal al Ing.Marcos Orellana e Ing. Andrés Patiño.

Atentamente,

Ing. Catalina Astudillo
Junta Académica de la Escuela de
Ingeniería de Sistemas y Telemática
Universidad del Azuay

Ing. Esteban Crespo
Junta Académica de la Escuela de
Ingeniería de Sistemas y Telemática
Universidad del Azuay

UNIVERSIDAD DEL AZUAY

Ingeniero,

Oswaldo Merchán Manzano

DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACION

UNIVERSIDAD DEL AZUAY

De mi consideración,

Estimado Señor Decano, yo Jonnathan Paul Capelo Solano con C.I 0106051865, código estudiantil 69259; estudiante de la Carrera de Sistemas y Telemática, solicito muy comedidamente a usted y por su intermedio al Consejo de la Facultad, la aprobación del protocolo de trabajo de titulación con el tema **"Explorar mecanismos de procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental"** previo a la obtención del título de Ingeniero en Sistemas y Telemática para lo cual adjunto la documentación respectiva.

Por la favorable acogida que brinde a la presente, anticipo mi agradecimiento.

Atentamente:

Jonnathan Paul Capelo Solano

N° 0849531

UNIVERSIDAD DEL AZUAY

DOCTORA MARÍA ELENA RAMÍREZ AGUILAR, SECRETARIA DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN DE LA UNIVERSIDAD DEL AZUAY

C E R T I F I C A:

Que, el señor **CAPELO SOLANO JONNATHAN PAUL** con código **60799,** alumno de la carrera de **INGENIERIA DE SISTEMAS Y TELEMATICA,** tiene aprobado el **91,11%** de créditos de su malla curricular.

Cuenca, 08 de mayo de 2018

Dra. María Elena Ramírez Aguilar
SECRETARIA DE LA FACULTAD
DE CIENCIAS DE LA ADMINISTRACIÓN

UNIVERSIDAD DEL AZUAY
FACULTAD DE ADMINISTRACIÓN
SECRETARIA

Derecho No. **001-010-000135821**
mjmr.-

[redacted] 0848419

Cuenca, 4 de Mayo de 2018

Señor Ingeniero
Oswaldo Merchán Manzano
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN
Presente. -

De mis consideraciones:

Luego de revisar el diseño del trabajo de titulación denominado *"Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental"*, presentado por Jonnathan Capelo, estudiante de la Escuela de Ingeniería de Sistemas y Telemática, considero que el documento cumple con las normas legales y reglamentarias de la Universidad y de la Facultad de Ciencias de la Administración, por lo que recomiendo su aprobación por parte del Consejo de Facultad.

Atentamente,

Agrim. Daniela Ballari, PhD
Directora

UNIVERSIDAD DEL AZUAY

UNIVERSIDAD DEL AZUAY
INGENIERIA EN SISTEMAS Y TELEMATICA
DISEÑO DE TESIS

1. DATOS GENERALES

1.1. Nombre del estudiante: Jonnathan Capelo

1.1.1.1. Código: 60799

1.1.1.2. **Contacto:** jcapelo@es.uazuay.edu.ec

1.2. Director sugerido: Daniela Ballari

1.2.1. Contacto: dballari@uazuay.edu.ec

1.3. Co-director sugerido:

1.4. Tribunal designado:

1.5. Aprobación:

1.6. Línea de Investigación de la carrera:

1.6.1. Código UNESCO: 1203 Informática de Computadoras

1.6.2. Tipo de trabajo: 1203.17 Informática

1.7. Área de estudio:

Procesamiento de datos, teleconexiones climáticas, R, clúster de altas prestaciones.

1.8. Título propuesto:

Procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R con datos de índices climáticos para el Ecuador Continental.

1.9. Estado del proyecto:

Nuevo

2. CONTENIDO

2.1. Motivación de la investigación:

La identificación de patrones espaciales de los índices climáticos para el Ecuador Continental requiere de altos requisitos computacionales, para lo cual es necesario recurrir al procesamiento en paralelo y distribuido. Actualmente existen herramientas en R que permiten realizar tales tipos de procesamiento en estaciones de trabajo, sin embargo es necesario escalar a un clúster de altas prestaciones para mayor eficiencia de estos procesos. Por lo tanto, tales herramientas requieren ser exploradas y preparadas para procesar el tipo de datos a utilizar en este proyecto.

2.2. Problemática:

Si bien existen librerías en R para el procesamiento en clúster de alta prestaciones, estas librerías deben ser integradas y probadas con el tipo de datos a procesar en el presente trabajo, es decir con series temporales de imágenes satelitales.

2.3. Resumen:

Este trabajo explorará diferentes librerías en R para el procesamiento distribuido en el clúster de altas prestaciones de CEDIA. Los datos a utilizar consistirán en imágenes satelitales de precipitación e índices climáticos relacionados con ENSO (El Niño South Oscilation). Se buscará encontrar los coeficientes de correlación relevantes entre ambas fuentes de información. Para los procesamientos se utilizará el software estadístico R y librerías para paralelizar y distribuir procesos como Paralell y rSlum. Los resultados del proyecto reportará métricas de rendimientos de los diferentes análisis ejecutados en un ordenador personal, con paralelización de procesos, en el clúster de altas prestaciones en un nodo único paralizando y en el clúster con distribución de procesos en varios nodos.

2.4. Estado del Arte

Las teleconexiones climáticas muestran relaciones entre diferentes puntos en la tierra, las cuales permiten predecir variaciones en el clima y anticipar eventos como sequía e inundaciones(Carleton, 2003; Liu & Alexander, 2007). La teleconexión más importante y conocida en el Ecuador se relacionada con ENSO y la incidencia de la región del niño 1+2 (Océano Pacífico frente a las costas del Ecuador y Perú), ya que a lo largo de la historia ha sido el principal autor de los eventos climáticos suscitados en el Ecuador(Bendix, Gämmerler, Reudenbach, & Bendix, 2003; Morán-Tejeda et al., 2016; Vuille, Bradley, & Keimig, 2000).

En el 2017 se llevó a cabo el proceso de corrección y reducción de escala de TRMM de 27km a 5km para el Ecuador con 117 estaciones meteorológicas in-situ e imágenes satelitales como covariables relacionadas con las propiedades de nubles de nubes, índice de vegetación normalizado y humedad del suelo(Ulloa, Ballari, Campozano, & Samaniego, 2017). Este producto corregido y de resolución espacial de 5km es una oportunidad única en nuestro país para representar de una manera continua los patrones espaciales de las teleconexiones climáticas; Es por ello que el presente trabajo utilizará esta fuente de información. El gran volumen de información proviene de las imágenes temporales de 13 años, sin embargo, es una limitación para la ejecución de los métodos a aplicar. Por ello, debe recurrirse a la computación de altas prestaciones.

El lenguaje que se utilizará en el siguiente trabajo es "R" y dado que este es específico para el análisis de datos. Se considera una de las plataformas más populares, ya que fue desarrollado específicamente para el análisis de datos estadísticos(Schmidt, Ostrouchov, Chen, & Patel, 2012). La evolución del lenguaje en los últimos años ha combinado

paradigmas funcionales y combinaciones de alto nivel con lo cual se puede ejecutar instrucciones C en código nativo, convirtiéndose en un lenguaje de componentes interpretados compilados. La capacidad de R para procesar grandes cantidades de datos ha sido limitada, sobretodo en equipos especializados en computación de altas prestaciones. Sin embargo, en la actualidad se han desarrollado varias librerías para equipos de escritorio multinúcleo. Estas funcionalidades tienen dos importantes limitaciones. Una es que lee los datos en la memoria, y esto va de la mano de la cantidad de memoria que se tiene disponible. La segunda es que R solo utiliza un procesador nativamente independiente de los que se tengan disponibles (Jing, 2010).

Los sistemas de alto rendimiento fueron especialmente diseñados para el mercado High Performance Computing (HPC), por lo que eran muy superiores en cuanto a rendimiento y precio. Una rama HPC se centra en el desarrollo y procesamiento de software para funcionar en estas computadoras(Jing, 2010). Hoy las cosas han cambiado considerablemente, y se puede encontrar en el mercado computadoras multiprocesador y multicore a precios mucho más accesibles, por lo que los clúster de alto rendimiento se llegan a utilizar solo si los requisitos de procesamiento aumenta, por ello es indispensable optimizar procesos y/0 cálculos con ayuda de técnicas de procesamiento en paralelo (Hager & Wellein, 2011).

Algunas librerías existentes a utilizar son "Rmpi" es un sistema estandarizado de paso de mensajes diseñado para funcionar en una amplia variedad de computadoras paralelas. Rmpi proporciona una interfaz R, además proporciona funciones para que los usuarios R no tengan que preocuparse por los detalles de las implementaciones MPI. "Parallel" establece tareas para cada nodo incluso en computadoras unidas por Ethernet en el cual se espera que

terminen sus tareas y se pide los resultados a cada uno; las computadoras no necesitan tener el mismo SO, se comunica por socialización para enviar los objetos. "Snowfall" está basado en SNOW que trabaja principalmente en Pc, pero este está especializado para Clusters, añade métodos para establecer maestros y esclavos para el procesamiento en paralelo. "Multicore" Realiza el paralelismo y la configuración del sistema de forma implícita y separando los datos por rasgos comunes como ventaja que ningún dato debe ser inicializado, pero a la vez esto limita su uso en sistemas Windows.

2.5. Objetivo general:

- Explorar mecanismos de procesamiento de altas prestaciones (paralelo y distribuido) en el clúster de CEDIA utilizando el software R y datos de índices climáticos para el Ecuador Continental.

2.6. Objetivos específicos:

- Explorar librerías en R para procesamiento en paralelo y distribuido.
- Implementar métodos de correlación de índices climáticos para procesamiento en paralelo y distribuido.
- Reportar métricas de rendimientos para los distintos métodos empleados.

2.7. Metodología:

Script inicial: El trabajo comenzará con un script inicial de R preparado para trabajar en un computador personal y sin ningún tipo de optimización en su ejecución. Dicho script implementa dos métodos de correlación entre una serie temporal de imágenes satelitales de precipitación obtenido de (Ulloa et al., 2017) y series temporales de diversos índices climáticos de libre acceso. Uno de los métodos de correlación es correlación no paramétrica de Spearman que se aplica pixel a pixel en las imágenes satelitales, y cuyo

resultado se visualiza en forma de un mapa. El otro método de correlación es correlación de Spearman con bootstrapping, el cual es un método no paramétrico que permite calcular intervalos de confianza de los coeficientes de correlación. Dado que cada una de las 132 imágenes que conforman la serie temporal contiene 18000 píxeles, el primer método de correlación requiere 8 segundos por cada índice climático; sin embargo el segundo, con 10000 iteraciones, requiere 55 minutos por cada índice climático.

Los índices climáticos a utilizar son:

Índices Climáticos

Indice	Nombre	Acceso
Niño 1+2	El Niño región 1+2	https://www.esrl.noaa.gov/psd/data/climateindices/list/
Niño 3	El Niño región 3	https://www.esrl.noaa.gov/psd/data/climateindices/list/
Niño 4	El Niño región 4	https://www.esrl.noaa.gov/psd/data/climateindices/list/
Niño 3.4	El Niño región 3,4	https://www.esrl.noaa.gov/psd/data/climateindices/list/
SOI	Southern Oscillation Index	https://www.esrl.noaa.gov/psd/data/climateindices/list/
TNI	Trans-Niño	https://www.esrl.noaa.gov/psd/data/climateindices/list/
MEI	ENSO multivariado	https://www.esrl.noaa.gov/psd/data/climateindices/list/
ONI	Niño oceánico	https://www.esrl.noaa.gov/psd/data/climateindices/list/

Librerías en R procesamiento en paralelo y distribuido: Se recopilará información de las librerías existentes para el procesamiento en paralelo y distribuido en R. Se estudiará el funcionamiento de cada una mediante su implementación para adquirir una adecuada comprensión y determinar las ventajas y desventajas de cada una. Esta implementación se realizará en 3 etapas: Primero se implementará en paralelo en un computador personal, luego en paralelo en un nodo del clúster HPC de CEDIA utilizando para ello la herramienta de RStudio Server. Para estas 2 etapas se utilizarán las librerías Parallel, SnowFall y Multicore. Finalmente se implementará el procesamiento distribuido en los diferentes nodos del clúster, utilizando para ello la libraría rSlurm.

Métricas de rendimiento: La eficiencia de las diferentes librerías exploradas, tanto en computadores personales como en el clúster, serán reportadas a través de métricas que permitan una comparación objetiva. Se reportará el tiempo de ejecución, la carga de memoria, la aceleración del proceso y la aceleración máxima (Jing, 2010).

2.8. Alcances y resultados esperados:

Scripts implementados para el procesamiento de los datos de índices climáticos del Ecuador continental, en paralelo y distribuido en el clúster de Cedia.

Evaluación del rendimiento de los diferentes scripts implementados, a través de las métricas.

2.9. Cronograma:

OBJETIVO	ACTIVIDAD	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	FECHAS
		7-11	14-18	21-25	28-1	4-8	11-15	18-22	25-29	2-6	9-13	16-20	23-27	30-3	6-10	13-17	20-24	27-31	3-7	10-14	17-21	24-28	1-5	8-12	15-19	22-26	
		MAYO				JUNIO				JULIO					AGOSTO				SEPTIEMBRE				OCTUBRE				
	Sustentación Diseño Trabajo de Titulación ante Tribunal (director, 2 tribunales) y entrega de documentación en secetaría para aprobación del Consejo de																										del 7 al 18 de mayo
	Desarrollo Trabajo de Titulación Tutorio		1	2	3	4	5	6	7	8	9	10	11	12			13	14	15	16							del 14 de mayo al 14 de septiembre
Explorar librerías en R para procesamiento en paralelo y distribuido.	Revision de librerias para el desarrollo del trabajo																										del 14 de mayo al 25 de mayo
Implementar métodos de correlación de índices climáticos para procesamiento en paralelo y distribuido.	Estudiar metodos de correlacion para el analisis de los datos del siguente trabajo																										del 21 de mayo al 1 de junio
	Implementacion en pc de las librerias y metodos estudiados																										del 4 de junio al 22 de junio
	Experimentacion en cluster de cedia lo realizado en la pc																										del 18 de junio al 31 de agosto
Reportar métricas de rendimientos para los distintos métodos empleados.	Analizar las metricas obtenidas en pc y en el cluster de cedia																										del 20 de agosto al 7 de septiembre
	Entrega de resumen para su traducción en la Unidad de																										del 10 al 21 de septiembre
	Revisión de documento final por parte del tribunal																										del 17 de septiembre al 5 de octubre
	Sustentación final ante Tribunal																										del 8 al 26 de octubre

2.10. Referencias:

Bendix, J., Gämmerler, S., Reudenbach, C., & Bendix, A. (2003). A Case Study on Rainfall Dynamics during El Niño/la Niña 1997/99 in Ecuador and Surrounding Areas as Inferred from GOES-8 and TRMM-PR Observations Niederschlagsdynamik während El Niño/La Niña 1997/99 in Ecuador und benachbarten Gebieten—Eine Fallstudie auf der Basis von GOES-8-und TRMM-PR-Daten. *Erdkunde*, 81–93.

Carleton, A. M. (2003). Atmospheric teleconnections involving the Southern Ocean. *Journal of Geophysical Research: Oceans*, *108*(C4).

Hager, G., & Wellein, G. (2011). *Introduction to High Performance Computing for Scientists and Engineers*. *Book*. https://doi.org/10.1201/EBK1439811924

Jing, L. (2010). Parallel Computing with R and How to Use it on High Performance Computing Cluster, 1–11.

Liu, Z., & Alexander, M. (2007). Atmospheric bridge, oceanic tunnel, and global climatic teleconnections. *Reviews of Geophysics*, *45*(2).

Morán-Tejeda, E., Bazo, J., López-Moreno, J. I., Aguilar, E., Azor'\in-Molina, C., Sanchez-Lorenzo, A., ... others. (2016). Climate trends and variability in Ecuador (1966–2011). *International Journal of Climatology*, *36*(11), 3839–3855.

Schmidt, D., Ostrouchov, G., Chen, W.-C., & Patel, P. (2012). Tight coupling of r and distributed linear algebra for high-level programming with big data. In *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:* (pp. 811–815).

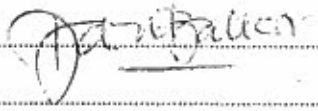
Ulloa, J., Ballari, D., Campozano, L., & Samaniego, E. (2017). Two-step downscaling of Trmm 3b43 V7 precipitation in contrasting climatic regions with sparse monitoring: The case of Ecuador in Tropical South America. *Remote Sensing*, *9*(7), 758.

Vuille, M., Bradley, R. S., & Keimig, F. (2000). Climate variability in the Andes of Ecuador and its relation to tropical Pacific and Atlantic sea surface temperature anomalies. *Journal of Climate*, *13*(14), 2520–2535.

2.11. Anexos: para casos en los que se requiera respaldar el proyecto.

2.12. Firma de responsabilidad (estudiante)

2.13. Firma de responsabilidad (director sugerido)



2.14. Firma de responsabilidad (Asesor Metodológico)

2.15. Fecha de entrega: