Distributed visualisation of oceanographic data using Apache Spark and LustreFS

L. Astudillo*, C. Núñez*
*Universidad Técnica Federico Santa María, Santiago, Chile

***Abstract*—Oceanographic data domain has gained an increasingly interest in the big data community and distributed applications, becoming a latent field of study. In this work, we proposed an approach for visualization of the Global operational Real-Time Ocean Forecast System (Global RTOFS) data, using *Apache Spark*, *LustreFS* and *Jupyter Notebook*, to optimize the pipeline of imaging and reduction processes.**

***Index Terms*—Apache Spark, LustreFS, Jupyter Notebook, PySpark, Oceanographic Data, Global RTOFS.**

I. Introducción

El procesamiento y visualización de datos a través del tiempo y a gran escala es un desafio permanente de la era contemporánea, debido al volumen y rapidez con la cual se genera estos. Este desafio se ha puesto a prueba a lo largo de múltiples disciplinas, como el procesamiento de datos astronómicos[1] o el procesamiento de lenguaje natural para la detección de *fakenews* en redes sociales[2].

A medida que la tecnología aumenta y mejora, los datos generados por los instrumentos actuales, demandan un mayor uso de las tecnologías existentes. Unas de las principales fuentes de datos, son los sistemas sensoriales oceanográficos, capaces de captar mediciones de su entorno y transmitirlas en tiempo real, demandando una plataforma capaz de recibir estas grandes cantidades de datos y procesarlos en tiempo real para su visualización y posterior análisis.

Frente a esto, el presente trabajo busca contribuir en el pipeline de almacenamiento y procesamientos de estos datos. Particularmente, la contribución de este trabajo es crear una plataforma distribuida para el procesamiento imágenes en datos oceanográficos, usando (1) *LustreFS* como filesystem distribuido para su almacenamiento, (2) el framework *Apache Spark* para el procesamiento distribuido, y (3) *Jupyter Notebook* para la edición de código.

II. Background y Trabajos Relacionados

El trabajo que conlleva procesar datos a gran escala, implica a su vez un desarrollo de nuevas tecnologías, las cuales ya se encuentra disponible en gran parte de los datacenter de HPC a lo largo de Chile y el mundo.

Una de estas tecnologías corresponde a los filesystem distribuidos, los cuales son una abstracción de los clásicos sistemas de POSIX que se han utilizado a lo largo de la historia en los sistemas Unix. *LustreFS*[3] es uno de los mejores ejemplos de tecnología en los filesystem distribuidos, además de ser el utilizado durante esta investigación.

LustreFS es el filesystem distribuido por excelencia en los cluster. Se encuentra en muchos de los clusters de HPC en el mundo, con decenas de petabytes(PB) de almacenamiento y cientos de gbps en sus redes de I/O.

Por otro lado, otra de las grandes tecnologías que han sido implementadas para la resolución de problemas de alto nivel es *Apache Spark*[4], un framework para clusters el cual funciona por medio de colecciones de elementos o RDD los cuales realizan cálculos en la memoria volátil de los clusters.

Ambas herramientas, tanto *LustreFS* como *Apache Spark*, son utilizadas en diversos proyectos a gran escala. *LustreFS* ha sido utilizado para el almacenamiento distribuido de datos astronómicos, los cuales son procesados en plataformas HA orientadas a trabajar con *Jupyter Notebook*[5][6], como lo consignado en el trabajo de [7] en la plataforma *Jovial*. Por otro lado *Apache Spark* ha servido en el creciente de desarrollo del computo en el análisis de datos oceanográficos, como puede ver visto en el trabajo de [8]. Ambas investigaciones sirvieron de base para la elaboración de este proyecto.

Si bien el problema propuesto en este trabajo se ha extrapolado a otras áreas, como astroinformática [9], nuestro aporte se focaliza en la implementación de las herramientas y en su rápido deployment en cualquier tipo de infraestructura. Por otro lado, los trabajos ya explorados, son complementarios al desarrollo que se logro durante nuestra investigación.

III. Diseño del sistema

El desarrollo de nuestro proyecto, busca confeccionar un ambiente de *data science* para el procesamiento distribuido de datos oceanográficos. Para ello, nuestra plataforma se divide en tres componentes: Un módulo de almacenamiento, el cual corresponde al archive dispuesto en *LustreFS*, un módulo de computo con *Apache Spark*, en cual será el encargado de procesar los RDD de forma distribuida, y un componente de frontend para edición, el cual en nuestro caso será la aplicación web *Jupyter*.

El módulo de almacenamiento o `archive`, estará encargado de almacenar los datos oceanográficos que se descarguen desde el sistema Global RTOFS (Global Real-Time Ocean Forecast System). El `archive` cuenta

con *LustreFS* 2.10, y posee un total de 146[TB] para uso académico, desplegado a lo largo de dos OSS, dos MDS y 24 OST; todo esto instalado en el datacenter Chi2AD.

Para obtener y almacenar los datos, se programó un crawler en Python para descargar datos del sistema Global RTOFS (Global Real-Time Ocean Forecast System) parte del NOAA National Weather Service [10].

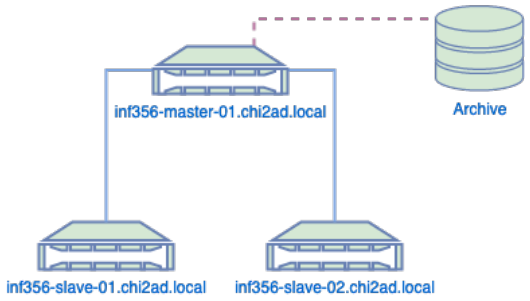


Fig. 1: Diagrama de alto nivel de la arquitectura. En linea puntiaguda roja se demarca la conexión entre el nodo maestro y el archive. En linea azul se demarca la conexión entre los workers y el nodo maestro.

El modulo de computo fue implementado con *Apache Spark* 3 y *PySpark*. La implementación de *Apache Spark* fue por medio de un nodo maestro y dos workers; a su vez, el nodo maestro fue conectado al `archive`, tal como se explica en el diagrama de alto nivel en la Figura 1.

PySpark es un wrapper para Python de *Apache Spark*, y su objetivo en el desarrollo de este proyecto fue crear las `SessionContext` con los cuales se logro paralelizar el procesamiento de las imágenes. La librería fue importada en los módulos de *Jupyter*. Para ver la carga del cluster al momento de iniciar un `SessionContext`, se utilizó el dashboard que ofrece *Apache Spark*, disponible en la Figura 2.

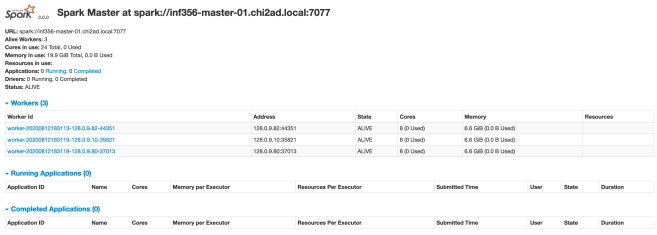


Fig. 2: Dashboard con de *Apache Spark* con la configuración dispuesta para el desarrollo del proyecto.

Finalmente, el componente de frontend fue el editor *Jupyter*. En el, se crearon los `notebooks` necesarios para el procesamiento de las imagenes. El componente de frontend tiene conexión directa al `archive`, debido a que los `notebooks` fueron ejecutados en un directorio dedicado dentro del filesystem.

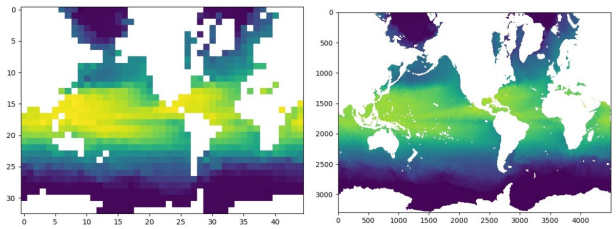
El código fuente del sistema es posible encontrarlo en el repositorio de GitHub[1] del proyecto.

IV. Experimentos

Para esta parte, primero se habla de la tarea de obtención de datos, estos fueron obtenidos por el *crawler* mencionado en la sección anterior. Este script esta programado para obtener los datos a las 15:00 hrs todos los días. Los datos descargados son archivos `.nc` [11] que contienen registros oceanográficos, asociados con metadata del tipo latitud, longitud, dimensiones del rango de medición, timestamp de la medición, entre otros. Cada uno de estos archivos pesa aproximadamente *0,5[GB]* por cada instante de tiempo. La cantidad de archivos recibida es más de 100 archivos por día.

Luego se procede con la tarea de procesamiento y visualización de datos, para esto se trabaja con la herramienta *Jupyter Notebooks* con kernel de *Python* para un mejor manejo de los inputs/outputs y para facilitar las pruebas, se procede a extraer el conjunto de datos, entregados en una lista de rutas en donde están contenidos los archivos oceanográficos, luego se hace un procesamiento con la librería `netCDF4`, la cual identifica los tipos de mediciones y se extrae los datos de la temperatura de la superficie del mar (`Sea_Surface_Temperature`).

Se utilizó la librería `xarray` para estructurar los datos y etiquetarlos para ser manejados por la librería de `pandas` para finalmente aplicar una función *plot* con la librería `matplotlib` dando como resultado dos tipos imágenes que es una visualización en baja y alta resolución de la temperatura superficial marina como lo muestra la Figura 3.



(a) low resolution (b) high resolution

Fig. 3: Visualizaciones de la temperatura marina superficial.

Esta tarea es realizada por cada archivo `nc` recolectado, por lo que se utiliza *Apache Spark* para poder procesar este conjunto de archivos.

Se utilizó la librería `PySpark` para crear una `SessionContext` que se utiliza para la paralelización de los datos; de esta manera se paralelizo la tarea antes

[1] https://github.com/camilo-nunez/INF356-proyecto-final

mencionada para generar una serie de visualizaciones a través del tiempo. Este proceso se realizó en un tiempo de computo considerablemente menor que en un caso serializado.

Una vez realizada esta experiencia, podemos decir que este sistema es mucho más viable, respeto al sistema clásico serial de procesamiento, para las visualizaciones en tiempo real debido a que puede procesar una gran cantidad de datos en un tiempo aceptable. Varias propiedades que tiene este sistema permiten generar un dashboard completo mostrando las principales informaciones obtenidos por estos datos, lo que permitiría fabricar un sistema de monitoreo marítimo.

V. Conclusiones

Se creó una plataforma distribuida con `Apache Spark` y `LustreFS` de manera exitosa, para procesar datos oceanográficos en tiempo real, descargando datos actualizados en periodos de 24 horas, y datasets de aproximadamente 50 [GB], sin tener mayores problemas con memoria o complejidad operacional. Se logro implementar de manera correcta el uso de un archive por medio de `LustreFS`, por otro, fue posible procesar de manera distribuidas funciones de gráficos utilizando `Apache Spark` a lo largo de un master y dos workers. Finalmente se cumplió con el objetivo de presentar un interfaz de edición como *Jupyter Notebooks* para que el usuario pueda ejecutar el código.

Cabe recordar que el desarrollo de este proyecto corresponde a un prototipo de plataforma, y no a una solución para un ambiente de producción. Dado que muchos de los elementos utilizados, fueron configurados para ambientes de testing.

Finalmente como trabajo futuro, se platea la opción de utilizar *JupyterHub* para habilitar múltiples usuarios concurrentes capaces de utilizar el cluster de forma masiva, de manera similar a lo implementado en [9].

VI. Agradecimientos

El desarrollo de esta investigación fue gracias a la infraestructura proporcionada por el Chilean Virtual Observatory, y el asesoramiento de su grupo de system administrators.

References

1 Solar, M., Araya, M., Farias, H., Mardones, D., and Wang, Z., "Cloud services on an astronomy data center," in *Software and Cyberinfrastructure for Astronomy IV*, Chiozzi, G. and Guzman, J. C., Eds., vol. 9913, International Society for Optics and Photonics. SPIE, 2016, pp. 552 – 558. [Online]. Available: https://doi.org/10.1117/12.2232705

2 Castillo, C., Mendoza, M., and Poblete, B., "Information credibility on twitter," in *Proceedings of the 20th International Conference on World Wide Web*, ser. WWW '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 675–684. [Online]. Available: https://doi.org/10.1145/1963405.1963500

3 Schwan, P., "Lustre: Building a file system for 1000-node clusters," 2003.

4 Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., Stoica, I. *et al.*, "Spark: Cluster computing with working sets." *HotCloud*, vol. 10, no. 10-10, p. 95, 2010.

5 Pérez, F. and Granger, B. E., "IPython: a system for interactive scientific computing," *Computing in Science and Engineering*, vol. 9, no. 3, pp. 21–29, May 2007. [Online]. Available: https://ipython.org

6 Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., Willing, C., and development team, J., "Jupyter notebooks ? a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, Loizides, F. and Scmidt, B., Eds. IOS Press, 2016, pp. 87–90. [Online]. Available: https://eprints.soton.ac.uk/403913/

7 Araya, M., Osorio, M., Díaz, M., Ponce, C., Villanueva, M., Valenzuela, C., and Solar, M., "Jovial: Notebook-based astronomical data analysis in the cloud," *Astronomy and Computing*, vol. 25, pp. 110 – 117, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2213133718300180

8 Dahal, J., Ioup, E., Arifuzzaman, S., and Abdelguerfi, M., "Distributed streaming analytics on large-scale oceanographic data using apache spark," 2019.

9 Sr., H. A. F., Sr., D. O., Núñez, C., Sr., M. S., and Bugueno, M., "ChiVOLabs: cloud service that offer interactive environment for reprocessing astronomical data," in *Software and Cyberinfrastructure for Astronomy V*, Guzman, J. C. and Ibsen, J., Eds., vol. 10707, International Society for Optics and Photonics. SPIE, 2018, pp. 787 – 796. [Online]. Available: https://doi.org/10.1117/12.2313304

10 Ryan, A., Regnier, C., Divakaran, P., Spindler, T., Mehra, A., Smith, G., Davidson, F., Hernandez, F., Maksymczuk, J., and Liu, Y., "Godae oceanview class 4 forecast verification framework: global ocean intercomparison," *Journal of Operational Oceanography*, vol. 8, no. sup1, pp. s98–s111, 2015.

11 Rew, R. and Davis, G., "Netcdf: an interface for scientific data access," *IEEE computer graphics and applications*, vol. 10, no. 4, pp. 76–82, 1990.