



# Programa de formación **MACHINE LEARNING AND DATA SCIENCE MLDS**

Facultad de  
**INGENIERÍA**



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA





# Módulo 3

## Big Data

### Unidad 4

### Procesamiento Big Data I

### Clase sincrónica

Facultad de  
**INGENIERÍA**



## > Bienvenida

# Jorge Eliécer Camargo Mendoza, PhD.

<https://dis.unal.edu.co/~jecamargom/>  
[jecamargom@unal.edu.co](mailto:jecamargom@unal.edu.co)

Departamento de Ingeniería de Sistemas e Industrial

Facultad de Ingeniería

Universidad Nacional de Colombia

Sede Bogotá



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA

## > Tabla de contenidos

- 1 ¿Qué es el procesamiento distribuido?
- 2 Diferencias entre procesamiento en línea y persistencia.
- 3 Definición de DAGs para procesamiento de datos
- 4 ¿Qué es *Dask*?
- 5 Arquitectura de Procesamiento Distribuido en *Dask*
- 6 ¿Cómo funciona *Dask*?
- 7 ¿Quién usa *Dask*?
- 8 Arreglos de *Dask*
- 9 Dataframes de *Dask*
- 10 Machine Learning con *Dask*

## Objetivos de aprendizaje

## Unidad 4 – Procesamiento Distribuido I

Al finalizar la unidad usted deberá ser capaz de:

1



Entender conceptos de computación distribuida con operaciones optimizadas automáticamente por medio de grafos dirigidos acíclicos (DAGs).

2



Utilizar arreglos multidimensionales de forma distribuida y optimizada desde la librería *Dask* para el manejo de grandes cantidades de datos numéricos.

3

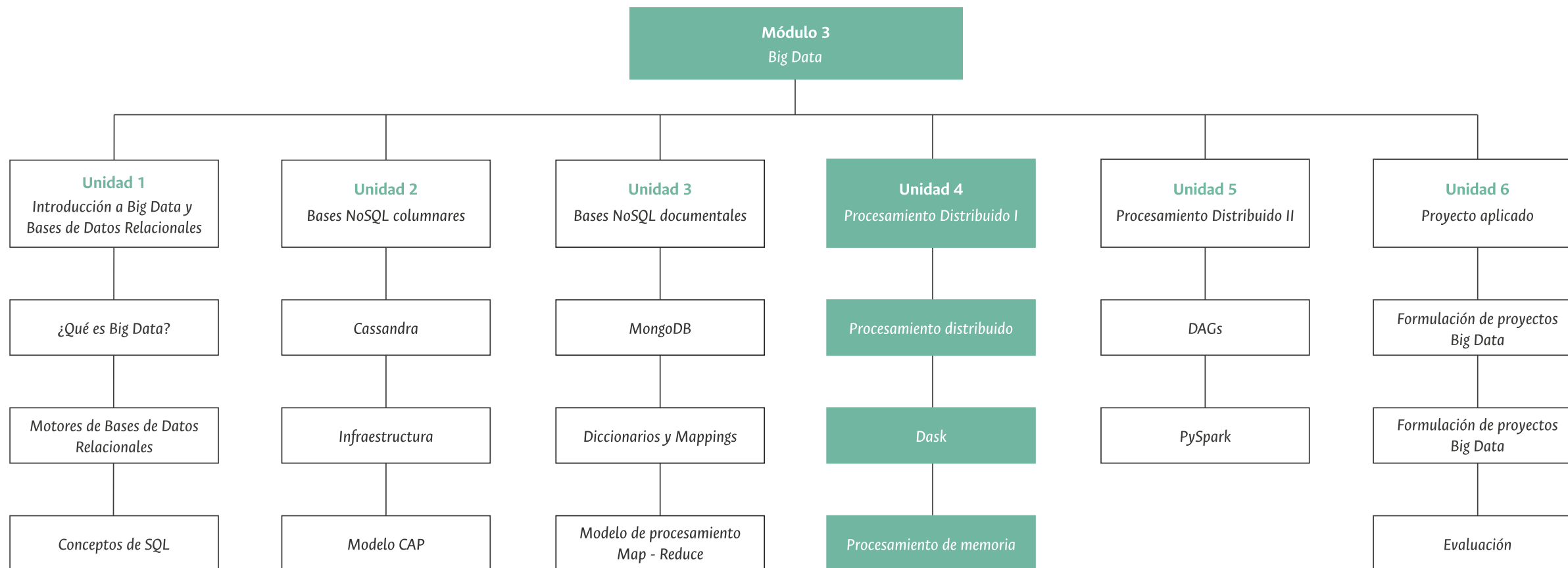


Utilizar *dataframes* distribuidos y optimizados desde la librería *Dask* para manejo de grandes cantidades de datos tabulares.



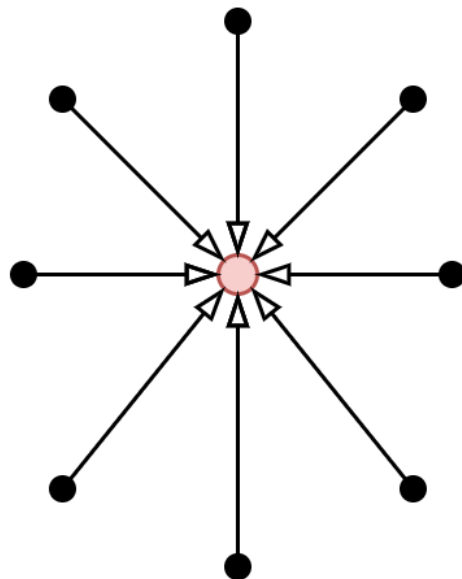


## Mapa de contenidos de la unidad

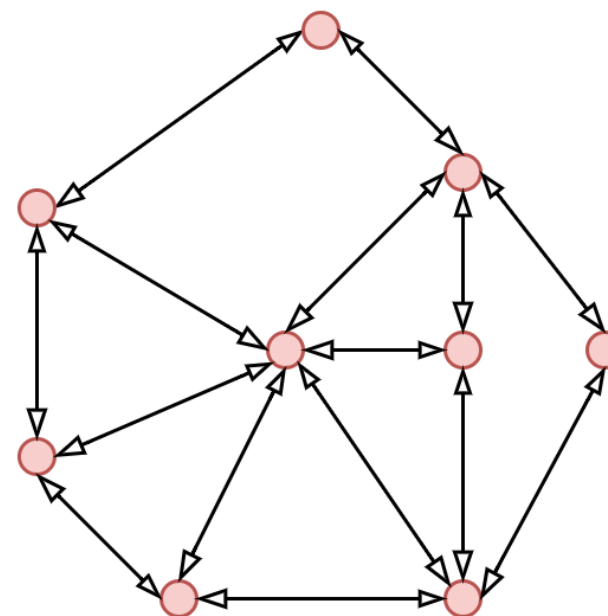


1

## ¿Qué es el procesamiento distribuido?



Sistema centralizado



Sistema distribuido

Procesamiento de información que se da en un grupo de *nodos* conectados en red, los cuales comparten *hardware* y *software* para cumplir un objetivo común. El propósito principal de este tipo de procesamiento es mejorar el rendimiento y evitar los errores centralizados que se dan en sistemas monolíticos.

## 2

## Diferencias entre procesamiento en línea y persistencia

### Procesamiento en línea

Consiste en que los programas se ejecuten de tal forma que los datos se actualicen de inmediato en los archivos del sistema. A este tipo de procesamiento se le conoce también como tiempo real. Por lo general, este tipo de procesamiento responde al instante al recibir un comando o una entrada.

### Persistencia de datos

Cuando una operación cambia datos en un sistema distribuido esto debe reflejarse en todos los nodos del sistema. La persistencia es el mecanismo que se usa para mantener información almacenada pero también debe poder recuperarse dicha información para que pueda ser utilizada nuevamente.



## 3

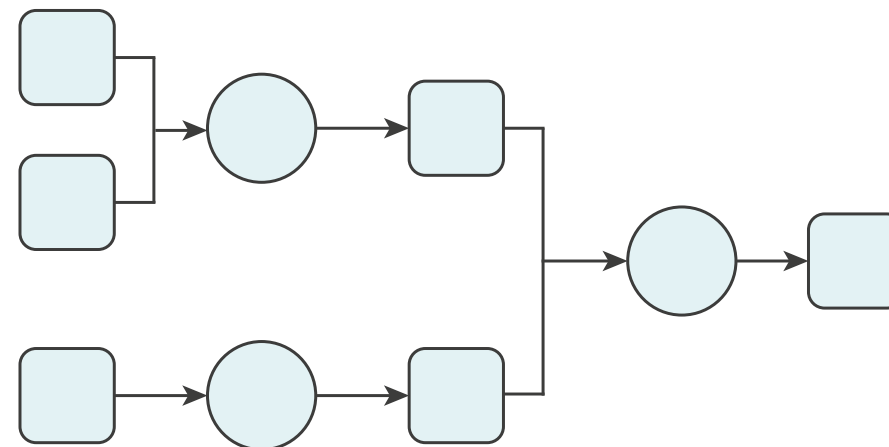
## DAGs y procesamiento de datos

Directed Acyclic Graph o Grafo Acíclico Dirigido, es un *grafo de nodos y aristas en el que no se retrocede*. Es una representación conceptual de una serie de procesos o actividades (*data pipeline*) que se aplican a los datos de forma única.

Cada nodo representa un conjunto de datos y cada arista un flujo de datos único.

El DAG puede representar, por ejemplo:

- Camino recorrido por los datos en un sistema distribuido.
- Camino óptimo usado por una *query*.
- Orden de las funciones de agregación aplicadas a un conjunto de datos.



**Grafo Acíclico Dirigido**

## 4

## ¿Qué es Dask?

Dask es una librería flexible de código abierto escrita en *Python* que se utiliza para la computación paralela y analítica.

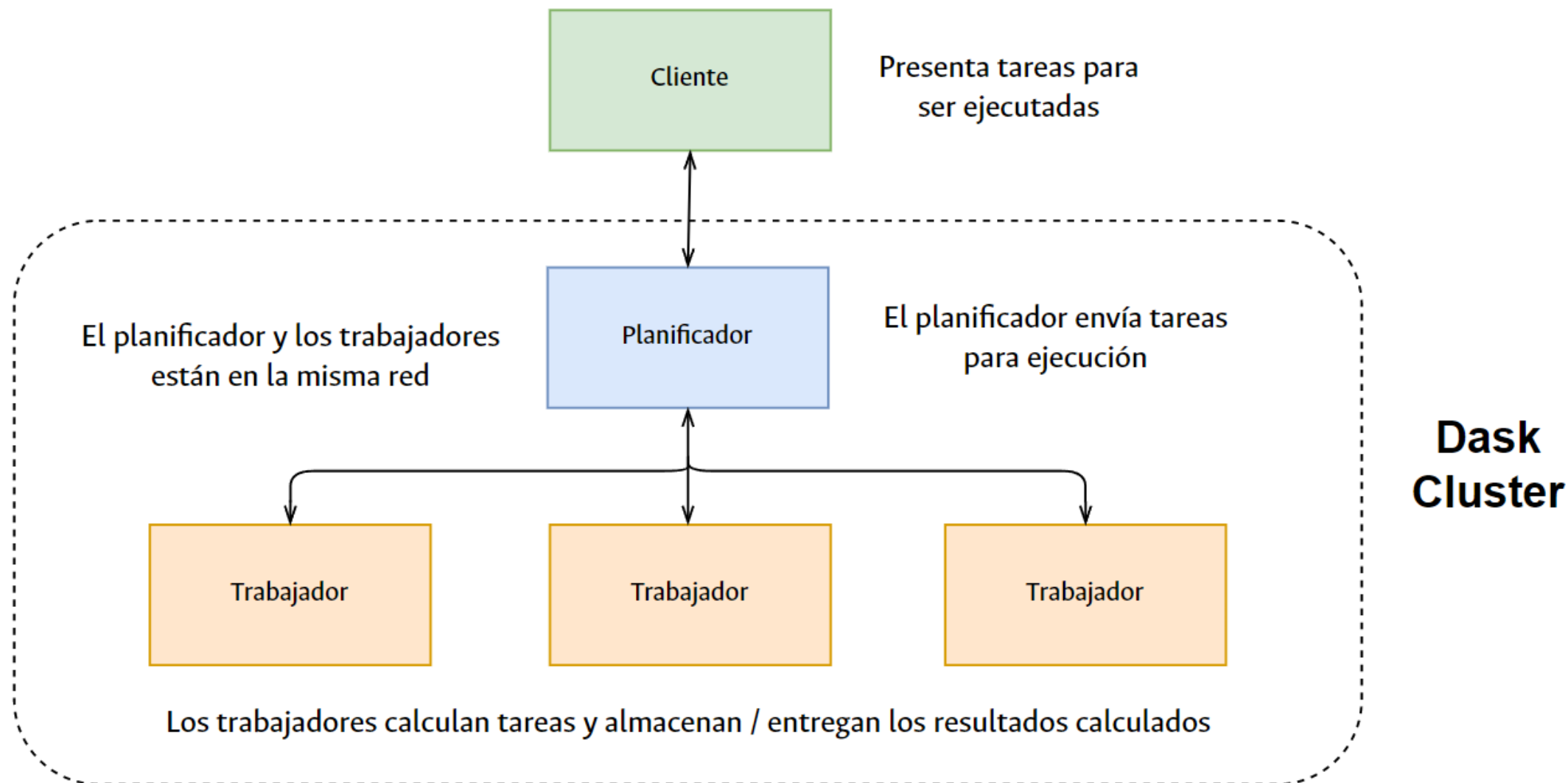
Esta librería ayuda a escalar los flujos de trabajo de *data science* y *machine learning*. Dask facilita el trabajo con otras librerías como *Numpy*, *Pandas* y *Scikit-Learn*.

Cuenta con programación dinámica de tareas, lo cual optimiza las cargas computacionales. También tiene colecciones de *Big Data* como *arrays* y *dataframes* que se pueden procesar en paralelo utilizando múltiples núcleos, así como con nodos en un clúster. Esto le da al usuario el poder de escalar el procesamiento de datos a cientos de máquinas en un clúster.



5

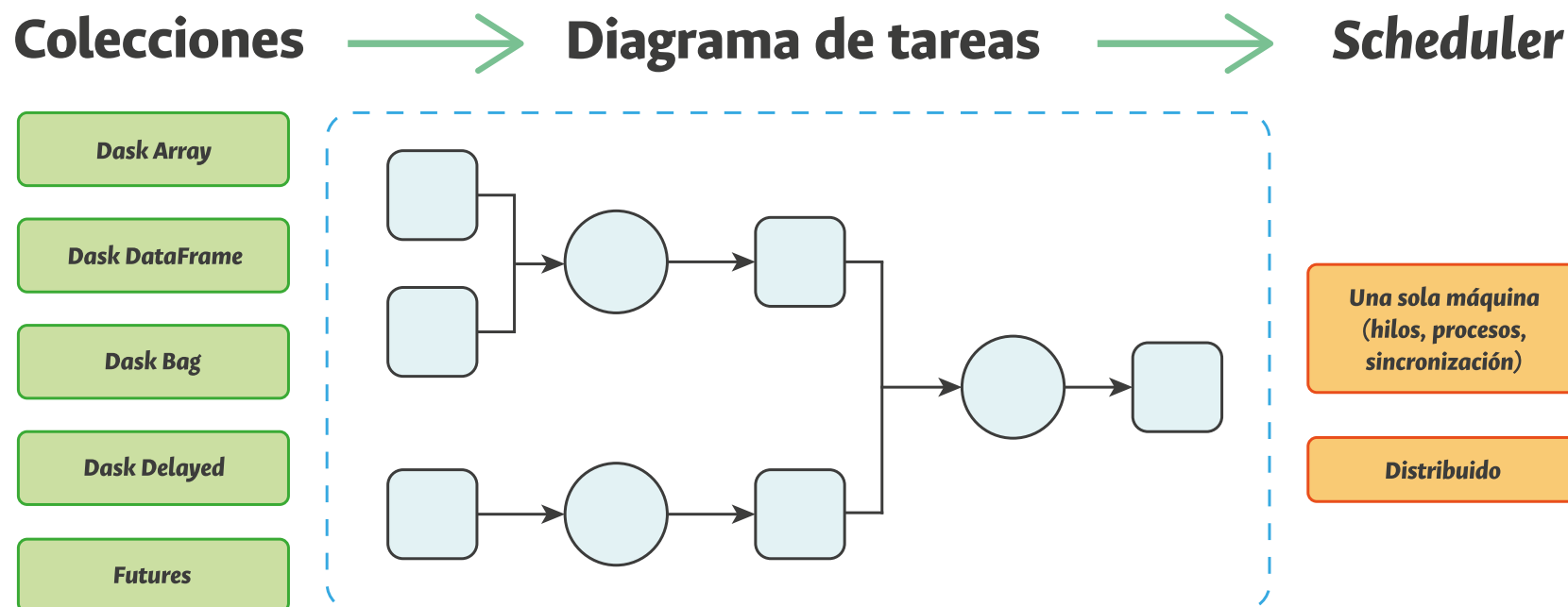
## Arquitectura de procesamiento distribuido en *Dask*



## 6

## ¿Cómo funciona Dask?

- Las colecciones de alto nivel se utilizan para generar gráficos de tareas que los programadores pueden ejecutar en una sola máquina o en un clúster.
- El *scheduler* distribuido de Dask es un programador de tareas dinámico, asíncrono y está basado en eventos. Coordina las acciones de varios procesos de trabajo de Dask distribuidos en varias máquinas




Internamente, el planificador realiza un seguimiento de todo el trabajo como un diagrama de tareas acíclico dirigido que cambia constantemente. Una tarea es una función de Python que opera en sus objetos que a su vez pueden ser el resultado de otras tareas. Este gráfico de tareas crece a medida que los usuarios envían más cálculos y se completa a medida que los trabajadores realizan las tareas; se reduce a medida que los usuarios abandonan o pierden el interés en los resultados anteriores.



7

## ¿Quién usa Dask?

	<p>NVIDIA usa Dask con el objetivo de escalar horizontalmente las cargas de trabajo de análisis de datos acelerados a múltiples GPU y GPU-based systems. Debido a que Dask puede trabajar óptimamente con los cientos de núcleos que contienen las GPUs y así manejar miles de subprocesos simultáneamente.</p>
	<p>Con Dask, los oceanógrafos pueden producir conjuntos de datos masivos a partir de la simulación de los océanos de la Tierra y los investigadores pueden observar grandes conjuntos de datos de sismología que se obtienen a través de sensores de todo el mundo. También se recopila una gran cantidad de observaciones de satélites y de estaciones meteorológicas para así ejecutar simulaciones complejas.</p>
	<p>Dask pronostica la demanda de más de las 500 millones de combinaciones entre un artículo y la tienda a la que pertenece. Para proporcionar la cantidad de artículos en demanda necesaria en todos sus puntos de venta, necesitan realizar cálculos enormes.</p>
	<p>Estos centros médicos registran la evolución y los movimientos de las células con el máximo detalle y tridimensionalmente a lo largo del tiempo. Esto genera grandes cantidades de datos que son difíciles de analizar con métodos tradicionales. Dask les ayuda a escalar sus flujos de trabajo de análisis de datos y a escalar prototipos de machine learning con estos datos.</p>

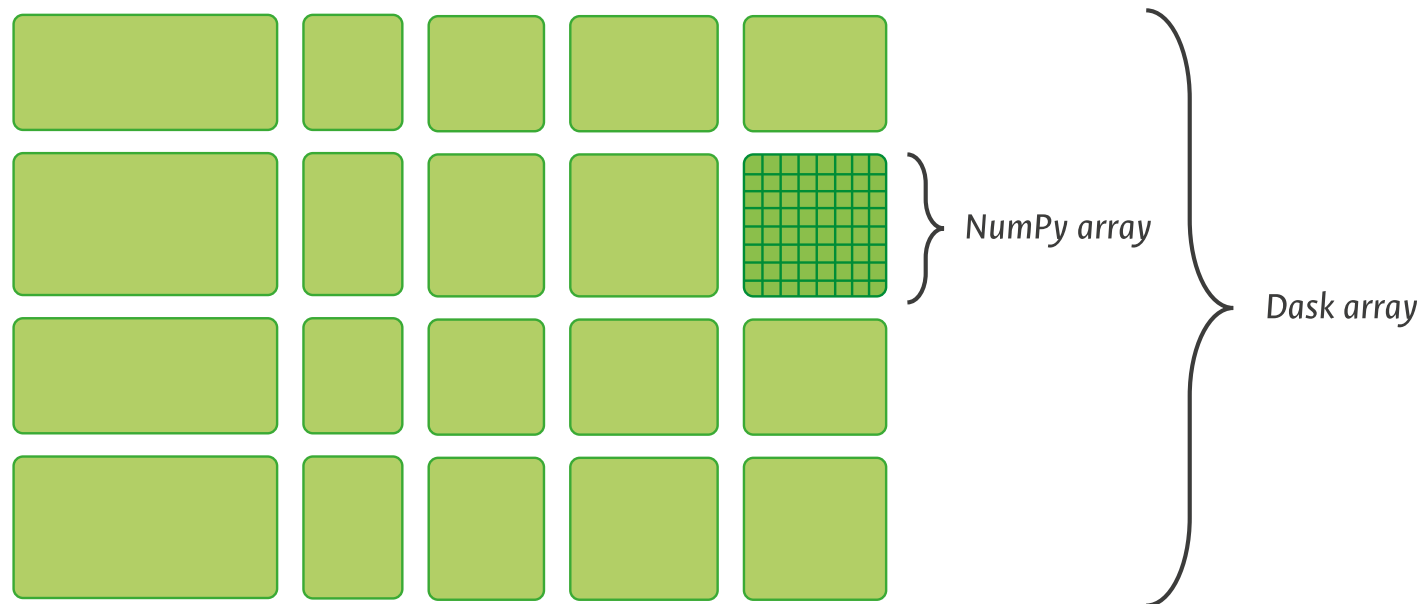
## 8

## Arreglos de Dask

- Los *arrays* en *Dask* imitan a los arreglos de *NumPy*. Estos son una colección de alto nivel que paralelizan las cargas de trabajo los cuales pueden residir en el disco o en otras máquinas.
- Dask* utiliza los *blocked algorithms* para que se puedan computar arreglos de mayor tamaño que la memoria utilizando todos los núcleos del sistema.



Los arreglos de *Dask* admiten la mayoría de las funciones de *NumPy* como el slicing, operaciones de aritmética y matemática escalar, algunas operaciones de álgebra lineal, etc.



Durante la ejecución del *blocked algorithm*, *Dask* transforma el arreglo en un diagrama de tareas, divide los *arrays* de gran tamaño de *NumPy* en varios fragmentos más pequeños y ejecuta el trabajo en cada fragmento paralelamente. Los resultados de cada fragmento se combinan para producir el resultado final.

## Arreglos en Dask

## Arreglo de Numpy vs. Arreglo de Dask

Los arreglos de *Dask* brindan mucha más flexibilidad que los de *Numpy*. Estos permiten trabajar con objetos más grandes que la memoria y el tiempo de cálculo es significativamente más rápido debido a la paralelización.

### Versión de Dask:

```
%%time
import dask.array as da
da_arr = da.random.normal(10, 0.1, size=(30_000, 30_000), chunks=(3000, 3000))
da_mn = da_arr.mean(axis=0)
da_mn

>>> CPU times: user 438 ms, sys: 70.6 ms, total: 509 ms
>>> Wall time: 11.6 s
>>> array([10.00058981, 10.00057394,  9.99906814, ...,  9.99991088,
          9.99966384, 10.00015255])
```

### Versión de Numpy:

```
%%time
import numpy as np
np_arr = np.random.normal(10, 0.1, size=(30_000, 30_000))
np_mn = np_arr.mean(axis=0)
np_mn

>>> CPU times: user 39.8 s, sys: 3.3 s, total: 43.1 s
>>> Wall time: 43 s
>>> array([ 9.99873418,  9.99910296,  9.99858814, ...,  9.99983802,
          10.00028087, 10.00124687])
```

## 9

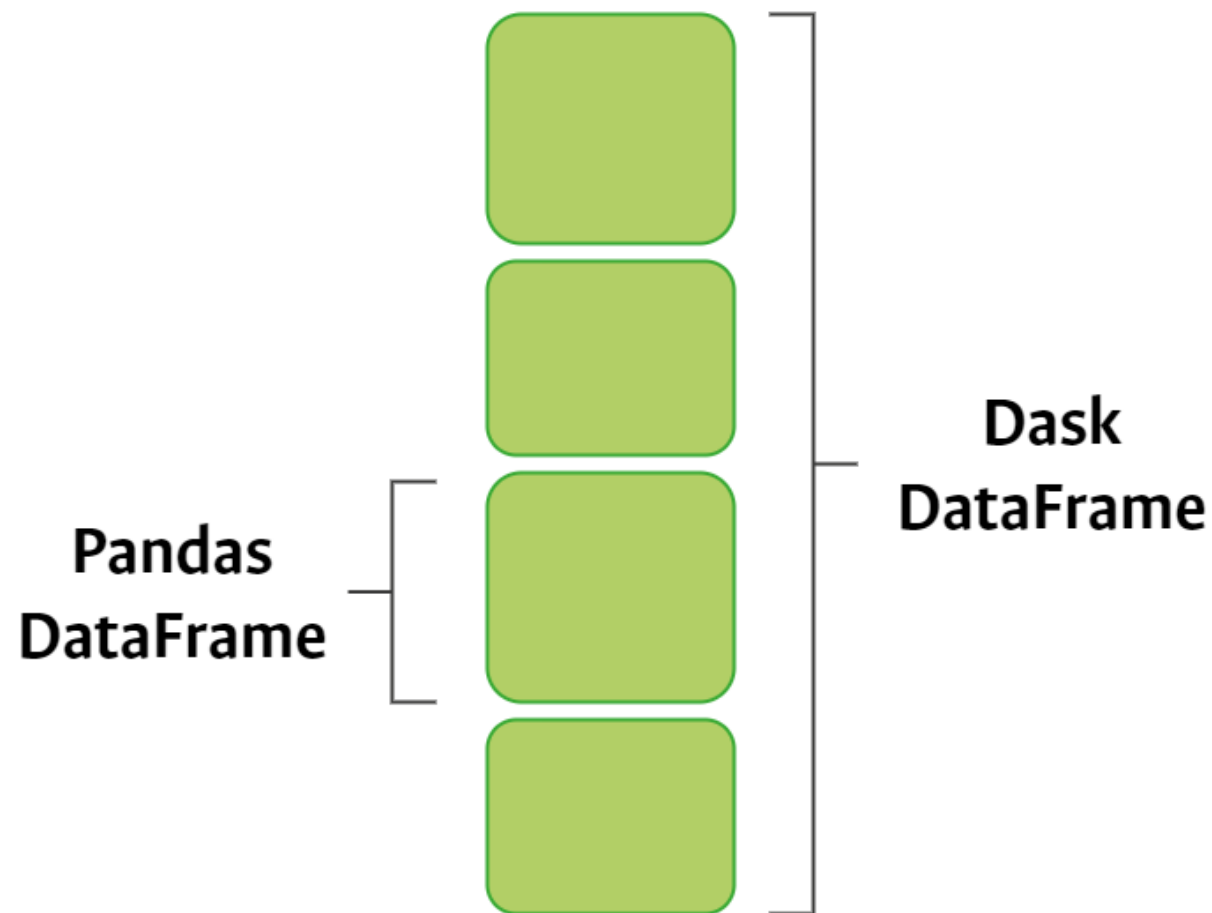
## Dataframes de *Dask*

Los DataFrames de *dask* son una estructura de datos tabular que está compuesta de múltiples DataFrames de *pandas*.

Este tipo de estructura de datos permite coordinar, paralelizar y distribuir series y DataFrames de *pandas* dando una forma de uso similar a los *pd.DataFrame*.

Generalmente se usan los *DataFrame* de *dask* cuando:

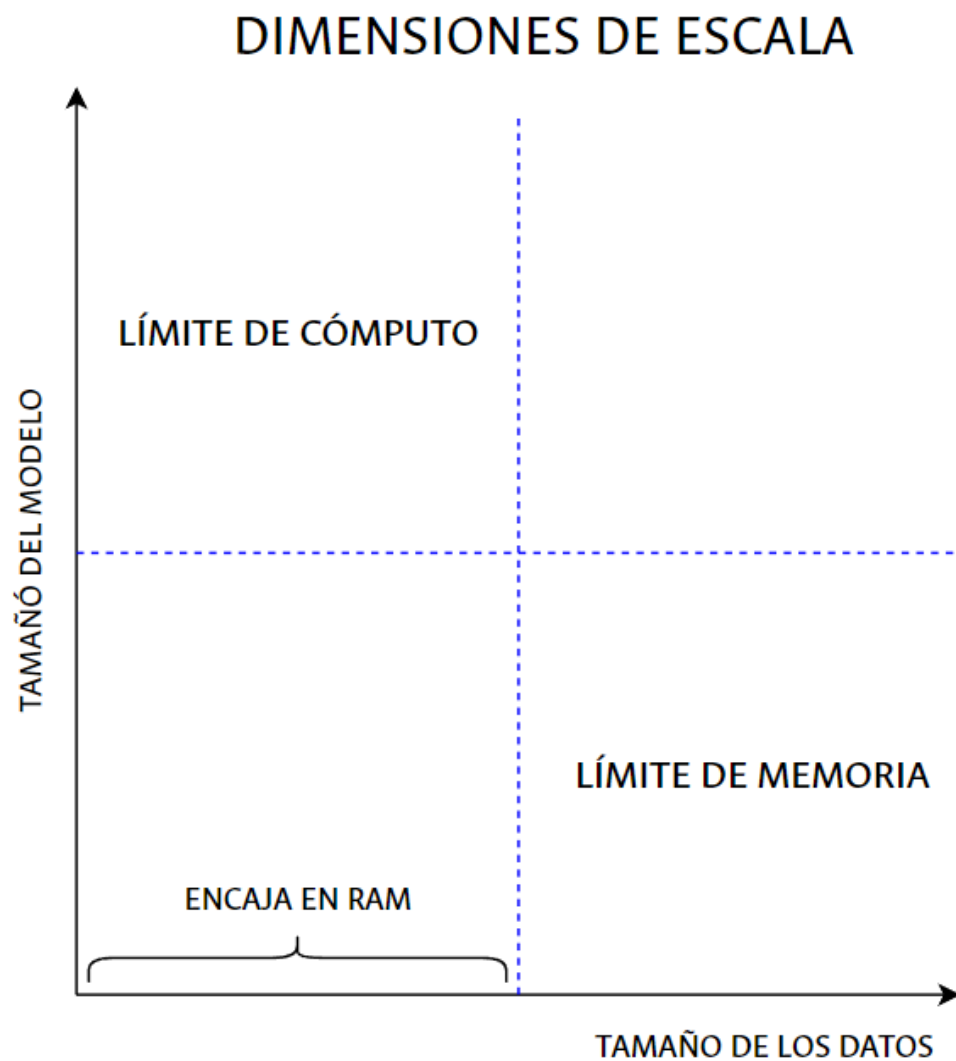
- Se tienen conjuntos de datos grandes que no caben en la memoria RAM.
- Se quiere acelerar operaciones sobre *datasets* usando varios núcleos de un computador o varios nodos.





## 10

## Machine Learning con Dask



Dask es usado en el aprendizaje automático, ya que también permite el entrenamiento y la previsión de modelos paralelos.

## > Despedida

**¡Gracias por su atención!**

**Jorge Eliécer Camargo  
Mendoza, PhD.**

<https://dis.unal.edu.co/~jecamargom/>  
[jecamargom@unal.edu.co](mailto:jecamargom@unal.edu.co)

Departamento de Ingeniería de Sistemas e Industrial  
Facultad de Ingeniería  
Universidad Nacional de Colombia  
Sede Bogotá



UNIVERSIDAD  
**NACIONAL**  
DE COLOMBIA



## Referencias

Dean, J. & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters [MapReduce: procesamiento de datos simplificado en grandes grupos].

<https://static.googleusercontent.com/media/research.google.com/es//archive/mapreduce-osdi04.pdf>

Katuzka, J. (2016). Data Locality in Hadoop. (tesis de maestría). Universidad Politécnica de Cataluña. Barcelona.

The Apache Software Foundation. (s. f.). Apache Hadoop Documentation.

<https://hadoop.apache.org/docs/stable/>

MapReduce. (2020, 16 de abril). En Wikipedia.

<https://es.wikipedia.org/wiki/MapReduce>

<https://www.nvidia.com/en-us/glossary/data-science/dask/>

[https://en.wikipedia.org/wiki/Dask\\_\(software\)](https://en.wikipedia.org/wiki/Dask_(software))



## Recursos adicionales

Dean, J. & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters [MapReduce: procesamiento de datos simplificado en grandes grupos].

<https://static.googleusercontent.com/media/research.google.com/es//archive/mapreduce-osdi04.pdf>

White, T. (2015). Hadoop: The definitive Guide. (Fourth Edition). <https://piiazza-resources.s3.amazonaws.com/ist3pwd6k8p5t/iu5gqbsh8re6mj/OReilly.Hadoop.The.Definitive.Guide.4th.Edition.2015.pdf>





## Derechos de imágenes

Freepik. (s.f.). Datos del usuario. [Icono].

[https://www.flaticon.es/icono-gratis/datos-del-usuario\\_3056329?term=data&page=1&position=74](https://www.flaticon.es/icono-gratis/datos-del-usuario_3056329?term=data&page=1&position=74)

Freepik. (s.f.). Intercambio. [Icono].

[https://www.flaticon.es/icono-gratis/intercambio\\_1372840?term=cambios&page=1&position=3](https://www.flaticon.es/icono-gratis/intercambio_1372840?term=cambios&page=1&position=3)

Smashicons. (s.f.). Tareas. [Icono].

[https://www.flaticon.es/icono-gratis/tareas\\_1032320?term=tareas&page=2&position=23](https://www.flaticon.es/icono-gratis/tareas_1032320?term=tareas&page=2&position=23)

Becris. (s.f.). Date free icon. [Icono].

[https://www.flaticon.com/free-icon/date\\_876805?term=calendar&page=1&position=22](https://www.flaticon.com/free-icon/date_876805?term=calendar&page=1&position=22)

Nvidia. Nidia Logo 2019 .SVG [Logo] <https://es.logodownload.org/nvidia-logo/>

National Aeronautics and Space Administration. (30 de julio de 2013). NASA logo.svg [Logo].

[https://en.wikipedia.org/wiki/File:NASA\\_logo.svg](https://en.wikipedia.org/wiki/File:NASA_logo.svg)

HHMI Communications team. (9 de enero de 2014). Howard Hughes Medical Institute logo.svg [Logo].

[https://en.wikipedia.org/wiki/File:Howard\\_Hughes\\_Medical\\_Institute\\_logo.svg](https://en.wikipedia.org/wiki/File:Howard_Hughes_Medical_Institute_logo.svg)

Harvard Medical School. (s.f.). hms\_logo\_final\_rgb.png [Logo]. <https://identityguide.hms.harvard.edu/logo>

PANGEO: Earth Science. (s.f.). Pangeo\_logo.png [Logo]. <https://avatars.githubusercontent.com/u/23299451?s=200&v=4>



## Derechos de imágenes

Freepik. (s.f.). Lista. [Icono].

[https://www.flaticon.es/icono-gratis/lista\\_2285586?term=tareas&page=2&position=12](https://www.flaticon.es/icono-gratis/lista_2285586?term=tareas&page=2&position=12)

Icongeek26. (s.f.). Carpeta. [Icono].

[https://www.flaticon.es/icono-gratis/carpeta\\_1333742?term=folders&page=4&position=78](https://www.flaticon.es/icono-gratis/carpeta_1333742?term=folders&page=4&position=78)

Clém IAGL (s.f.) Mapreduce.png [imagen]

<https://commons.wikimedia.org/wiki/File:Mapreduce.png>

## > Créditos

Facultad de

**INGENIERÍA**

### **Autores**

Jorge Eliécer Camargo Mendoza, PhD

### **Asistente docente**

Leonardo Avendaño Rocha

Alberto Nicolai Romero Martínez

### **Diseño instruccional**

Claudia Patricia Rodríguez Sánchez

### **Diseño gráfico**

Clara Valeria Suárez Caballero

Milton R. Pachón Pinzón

Rosa Alejandra Superlano Esquibel

Brian Chaparro Cetina

### **Diagramadora PPT**

Daniela Duque

**Fecha**  
2022-II

