

Modulo “Analisis de Datos Cientificos y Geograficos”

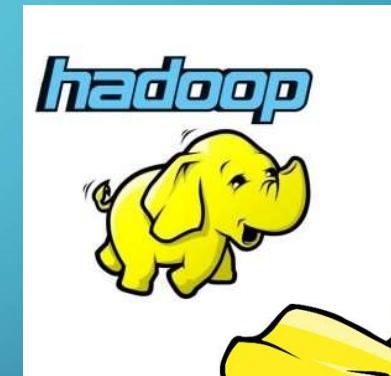
SPATIAL HADOOP

ITBA

BIG DATA EN SISTEMAS ESPACIALES ?



BIG DATA EN SISTEMAS ESPACIALES ?



BIG DATA EN SISTEMAS ESPACIALES ?



Spatial Data
+
Hadoop
=

*Spatial
Hadoop*



HADOOP PARA DATOS ESPACIALES?

El framework Hadoop, con el paradigma MapReduce, es uno de los estandares para analisis en Big data en entornos distribuidos.

Se lo podria adaptar para datos geograficos?

- MAP: mapea la entrada de datos en un conjunto de key-pares (ki, vi)
- REDUCE: toma todos los valores asociados a una misma clave y produce el resultado final



SPATIAL HADOOP

En el 2013, Ahmed Eldawy y Mohamed F. Mokbel presentan un poster en VLDB:

“A Demonstration of Spatial Hadoop: An Efficient MapReduce Framework for Spatial Data”



SPATIAL HADOOP

En ese mismo año, junto con Yuan Li y Ravi Janardan en SIGSPATIAL '2013, presentan

"CG_Hadoop: Computational Geometry in MapReduce"(*), con la idea de manejar estas 5 operaciones geométricas fundamentales:

- **kNN**
- **polygon union**
- **skyline**
- **convex hull**
- **farthest pair**
- **closest pair**

(*)http://spatialhadoop.cs.umn.edu/publications/p0144_Eldawy.pdf



POR QUE SPATIAL HADOOP ?

Los algoritmos geométricos fueron estudiados por años e implementados eficientemente. Muchos productos comerciales y open source (postgis para Postgres, JTS para OpenJump, etc) los han implementado.... Pero...

.... la tasa de producción de imágenes espaciales se ha incrementado notablemente en los últimos años (smart phones, telescopios, etc.) y ya no resultan manejables con los mecanismos tradicionales.



SPATIAL HADOOP

El problema es que los algoritmos geométricos existentes no escalan bien cuando se tratan volúmenes espaciales muy grandes.

Por ejemplo, detectar con el método tradicional los puntos periféricos de una nube de puntos (CONVEX HULL) con 4 billones de puntos (cantidad normal hoy en dia), lleva 3 horas de procesamiento !

Hay que rediseñarlos para permitir manejar esto en un entorno de cluster.



SPATIAL HADOOP

El objetivo de Spatial Hadoop era mostrar que este tipo de operaciones son sujetas a ser paralelizables en el mundo MapReduce.

Pero no solo realizan la implementación distribuida de las operaciones, sino que la propuesta abarca la **indexación espacial distribuida** en un cluster Hadoop.

Tambien realizan una comparación de la performance de usar las operaciones distribuidas en un cluster Hadoop versus un cluster Hadoop con indexación distribuida (Grid File y R-Tree).



SPATIAL HADOOP

Los experimentos mostraron que uso de Operaciones Distribuidas en un cluster de 25 computadoras y datasets de 128 Gb:

- Sin índices resultaron 29 veces más rápido que las tradicionales sin distribuir
- Con indices distribuidos resultaron 260 veces mas rápido que las tradicionales!!

*Algo que tardaba 4 minutos...
... Ahora tarda menos de 1 segundo !*



SPATIAL HADOOP

Los experimentos mostraron que uso de Operaciones Distribuidas en un cluster de 25 computadoras y datasets de 128 Gb:

- Sin índices resultaron 29 veces más rápido que las tradicionales sin distribuir
- Con indices distribuidos resultaron 260 veces mas rápido que las tradicionales!!

**Recuerdan el calculo de convexhull
que tardaba 3 horas?
Con Shadop tarda 1 minuto !!!!**



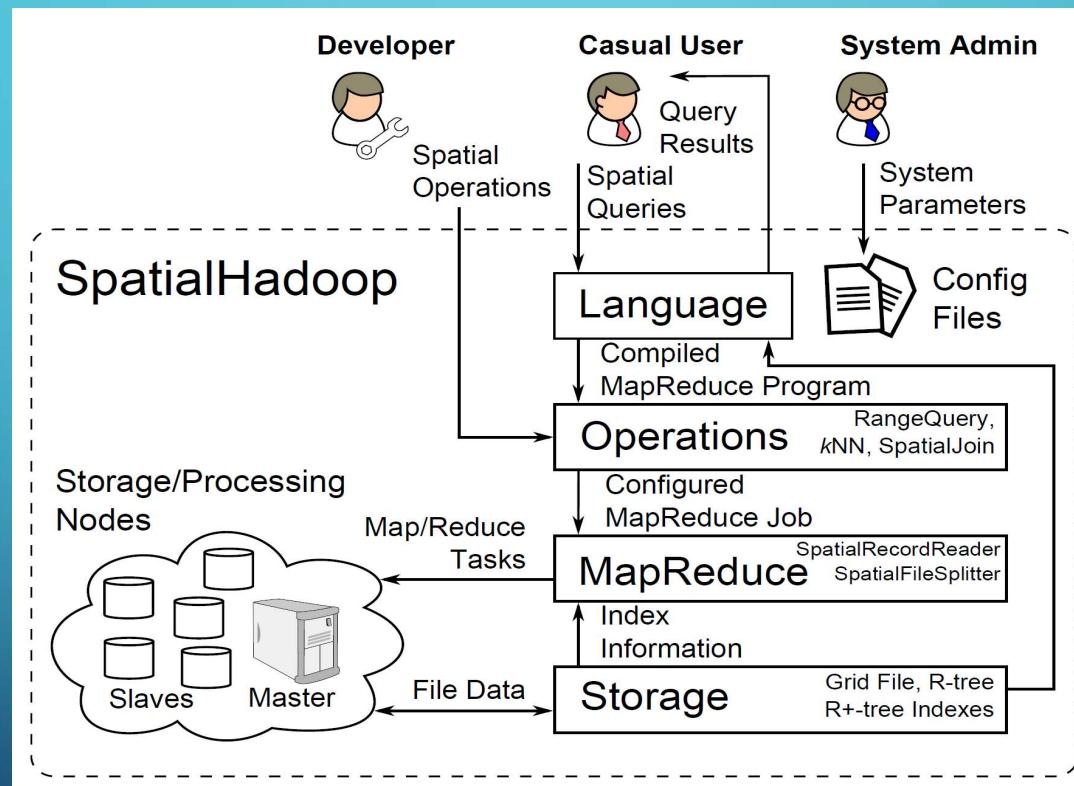
SPATIAL HADOOP

Dado los resultados prometedores, continuaron trabajando en implementar más operaciones e índices.

En el 2015/2016 ofrecen un framework para distribución espacial.

Sobre el mismo se pueden generar nuevos tipos de índices espaciales (es un framework).

SPATIAL HADOOP: ARQUITECTURA





SPATIAL HADOOP

Map Reduce es un paradigma de programación tan aceptado para realizar distribución de procesamiento que pensaron en utilizarlo para adaptar las operaciones geométricas que son tan costosas computacionalmente.

Finalmente generaron una biblioteca MapReduce para las mismas.



SPATIAL HADOOP

En Hadoop, los datos se colocan en el file system distribuido HDFS en chunks (bloques) de 64 Mb.

Así como en un File System regular hay una tabla de bloques (INodes/FAT) y los bloques propiamente dicho, en Hadoop hay 2 tipos de nodos:

- **El Master**
- **Los slaves**



SPATIAL HADOOP

Manejan dos niveles de indices espaciales:

- A nivel local: organiza los datos dentro de un nodo cluster
- A nivel global: particiona datos a lo largo de los nodos cluster

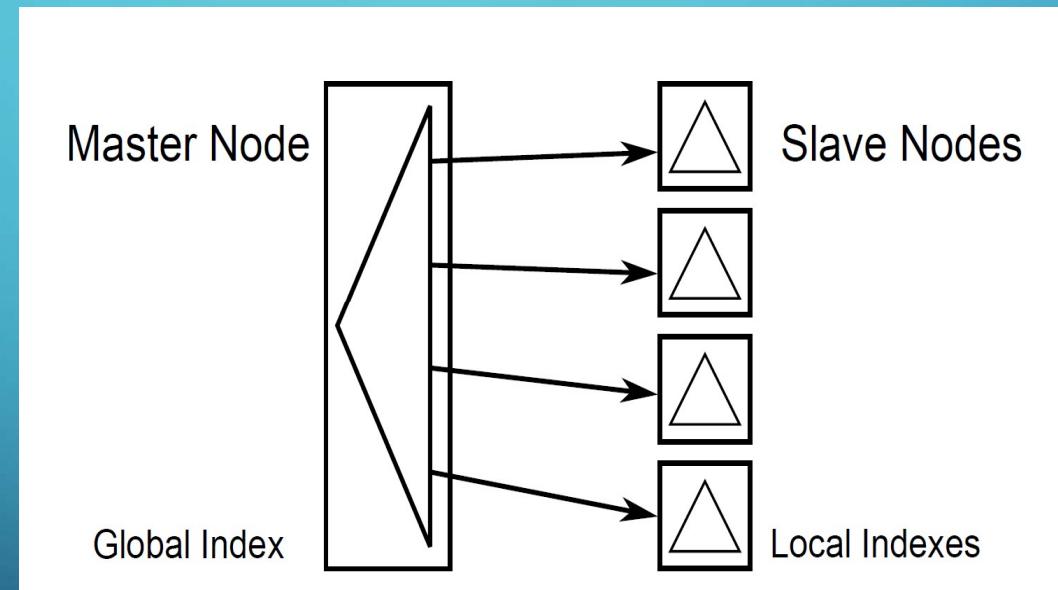
MapReduce utiliza ambos indices para filtrar particiones del archivo de entrada, ganando eficiencia

SPATIAL HADOOP: INDICES ESPACIALES

Mantiene metadata sobre la distribución de los bloques en los nodos slaves

Divide el archivo input en pedazos que asigna a los nodos slaves que realizaran el Map.

Decide qué nodos hacen el Job Map y qué nodos hacen el Job Reduce.



Tienen realmente los datos a procesar



OPERACION KNN EN SPATIAL HADOOP

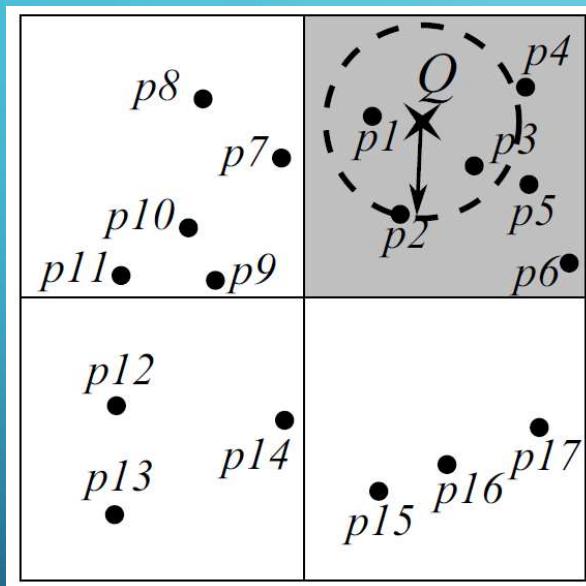
Operador kNN : encuentra dentro de un conjunto de puntos P los k puntos mas cercanos a un punto dado Q

En Hadoop: se escanean todos los puntos, se calcula su distancia a Q y se hace un ranking, eligiendo los k del tope.

En SpatialHadoop:

- 1) Respuesta inicial: se toman los k mas cercanos a Q en el bloque de Q
- 2) Chequeo de correctitud: se imagina un circulo C de testeo alrededor de Q con un radio igual a la distancia de Q a su k vecino mas lejano encontrado inicialmente. Si C no hace overlaps con otras particiones, la respuesta es la final. Caso contrario, se pasa al paso 3.
- 3) Refinamiento: se ejecuta una busqueda de puntos mas cercanos dentro del MBR del circulo C, y se toman los k mas cercanos

EJEMPLO KNN EN SPACIAL HADOOP

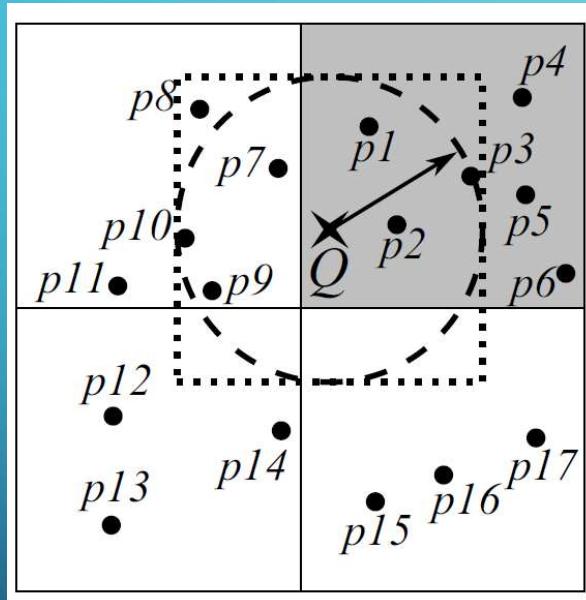


kNN (Q, 3)

Donde toma ventaja Spatial Hadoop?

Que hubiera hecho Hadoop?

EJEMPLO KNN EN SPACIAL HADOOP



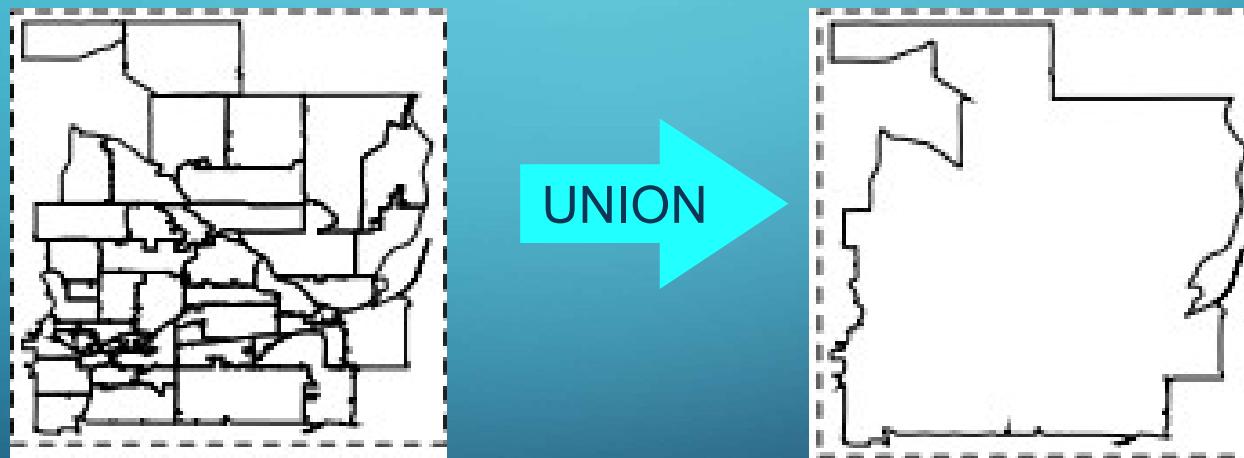
kNN (Q, 3)

Donde toma ventaja Spatial Hadoop?

No hubiera sido lo mismo con Hadoop?

OPERACION UNION

La unión de un conjunto Pol de polígonos es el conjunto de todos los puntos que pertenecen por lo menos a un polígono de Pol y donde sólo el perímetro de todos los puntos se mantiene y los segmentos internos se eliminan.





OPERACION UNION

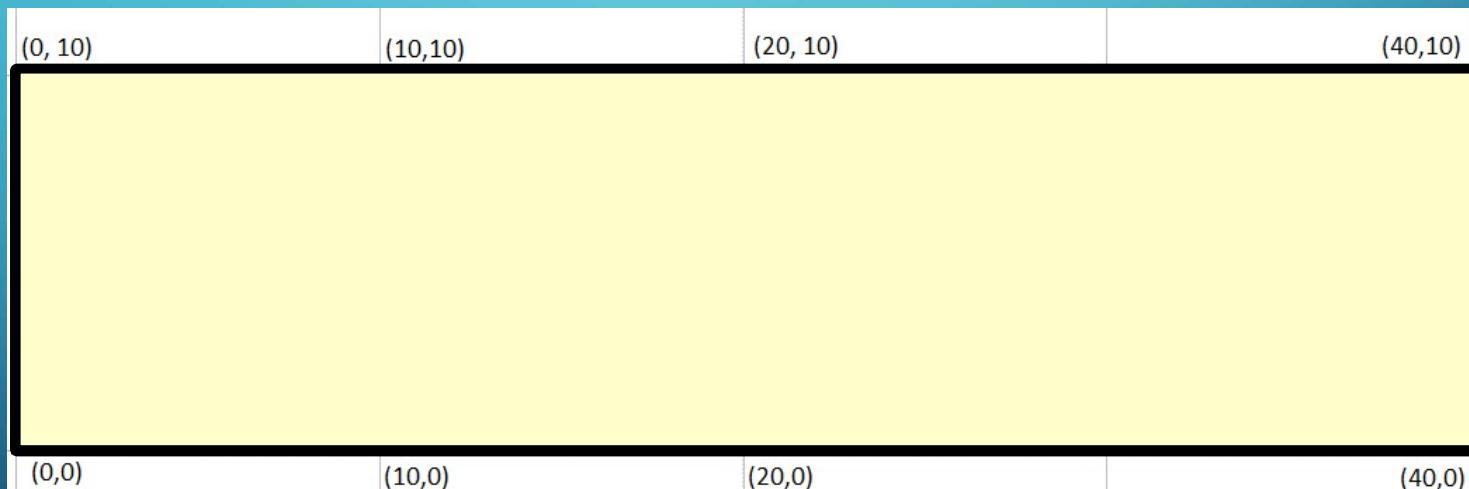
Algoritmo Tradicional para Unión de Polígonos

- a) Se calcula la unión de todos ejes
- b) Se eliminan los segmentos internos
- c) Se devuelven los segmentos que quedan, que son los del perímetro

EJEMPLO 1

POL 1 = POLYGON ((0,0), (0,10), (20,10), (20,0), (0,0))

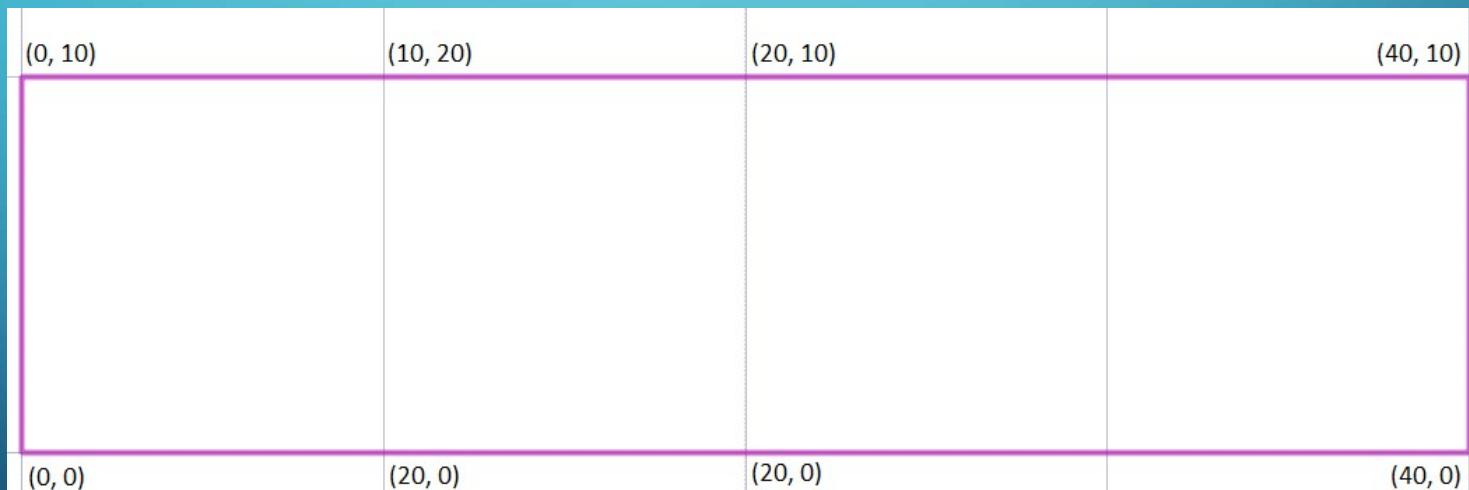
POL 2 = POLYGON ((10,0), (10,10), (40,10), (40,0), (10,0))



EJEMPLO 1

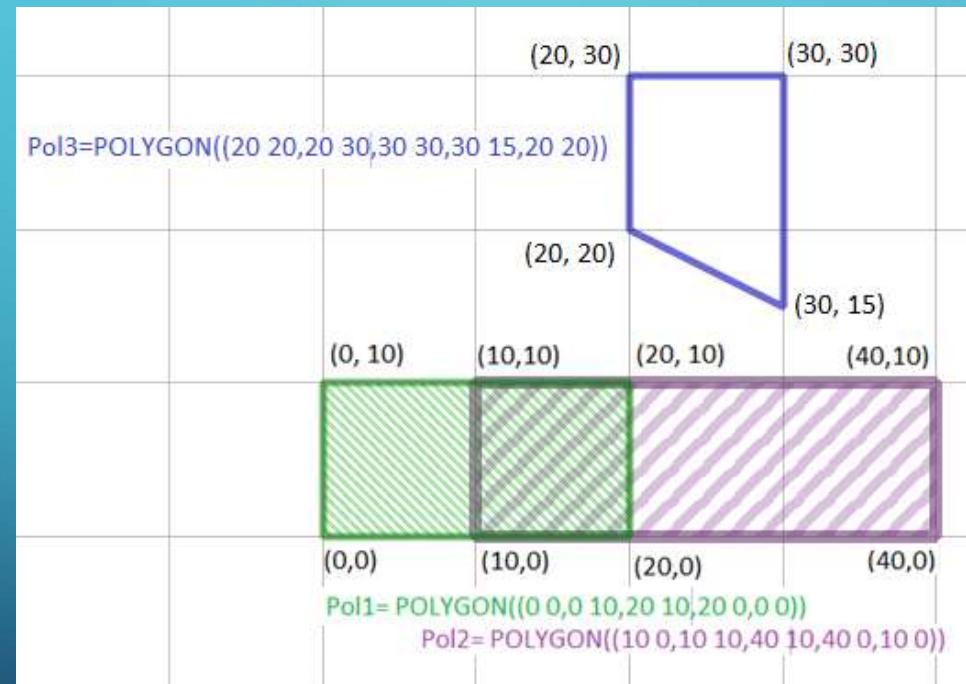
Resultado de la unión: **POLYGON ((0,0), (0,10), (40,10), (40,0), (0,0))**

Menos información: antes tenía $5 * 2$ puntos. Ahora tengo 9 puntos...



EJEMPLO 2

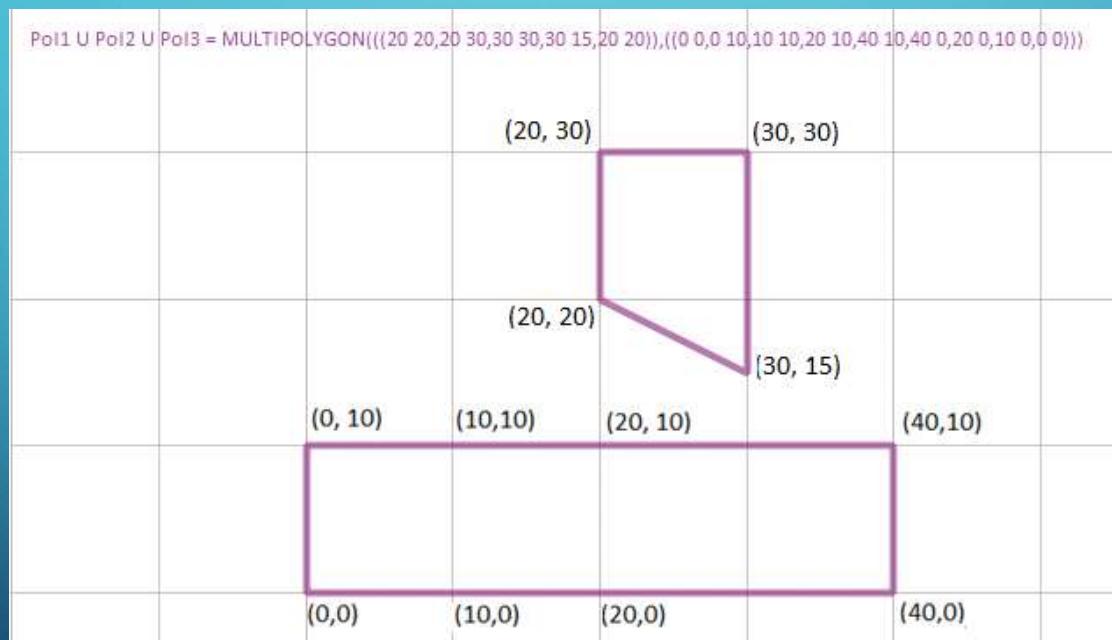
Que hubiera pasado en este caso?



EJEMPLO 2

Resultado de la unión: un multipolygon

Menos información: antes tenía $5 * 3$ puntos. Ahora tengo 14 puntos...

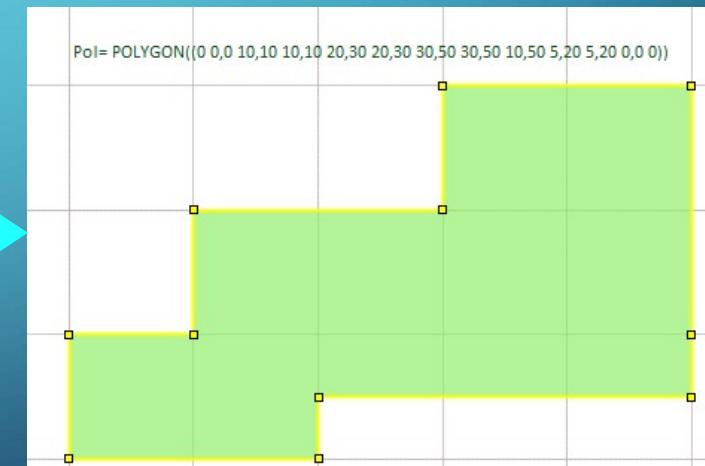
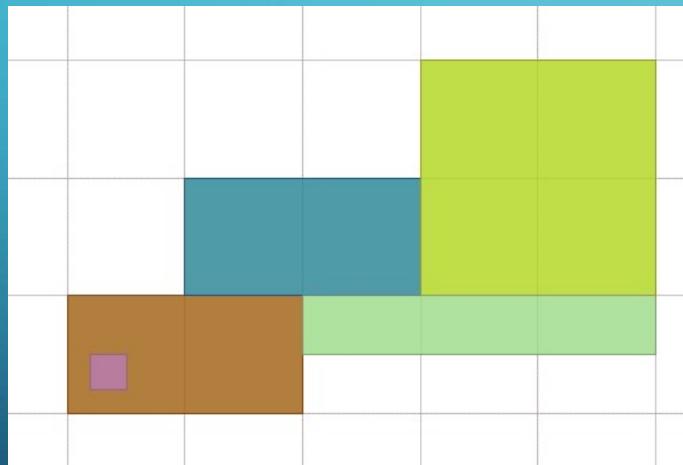


OPERACION UNION

Siempre ahorro 1 punto ? En qué casos tengo máximo beneficio?

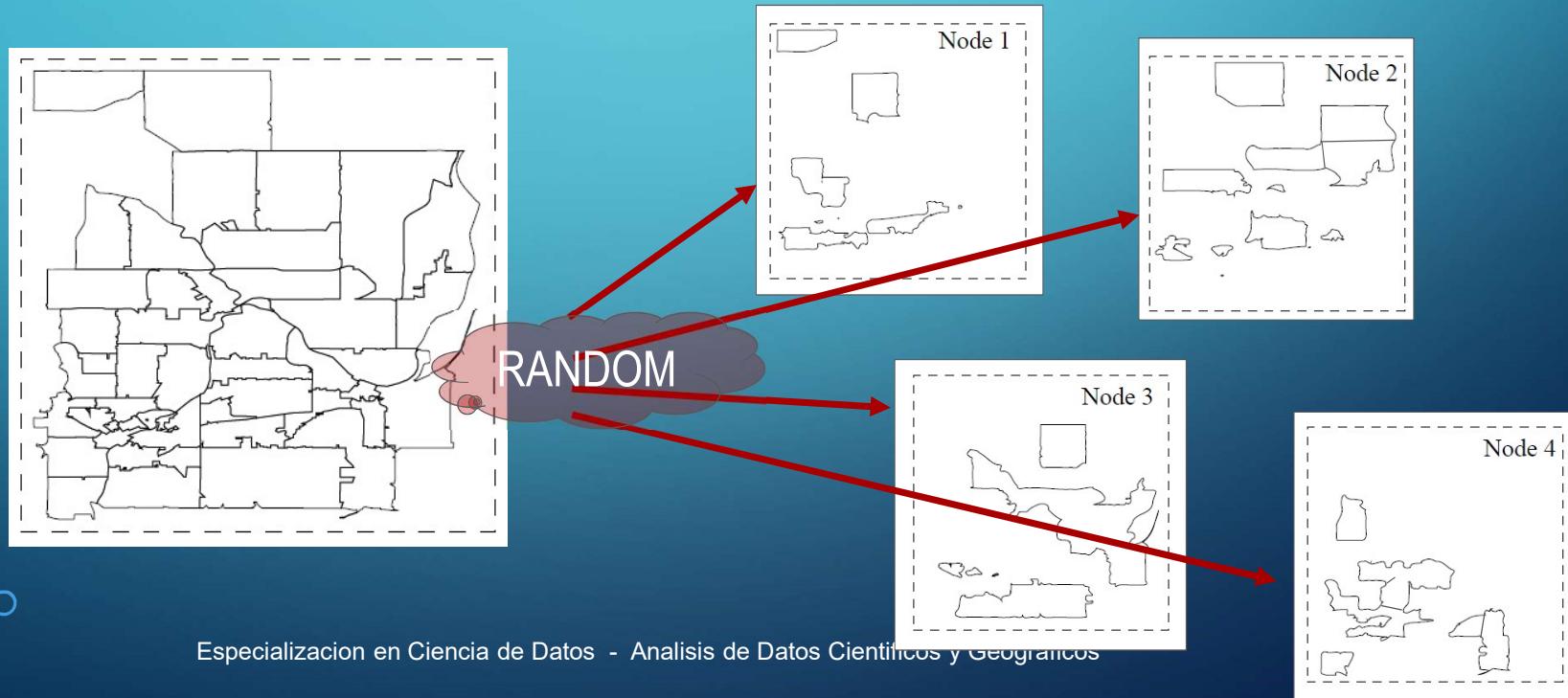
En el siguiente caso tengo un beneficio del 50%...
Pasamos de $5 * 5$ a sólo 12 puntos !

*Cuanto mas contigüidad
hay, mas compacta
resulta la union*



OPERACION UNION EN HADOOP SIN INDICE

1) Partición: Nodo Master distribuye el conjunto de polígonos input en cada nodo para el Mapper:

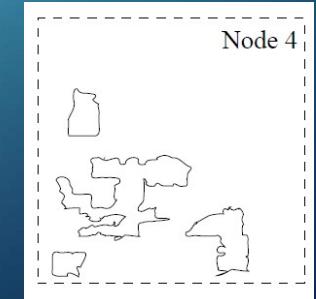
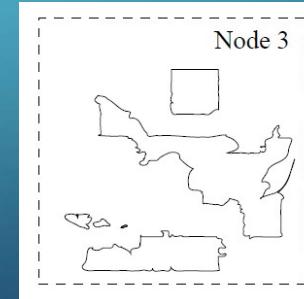
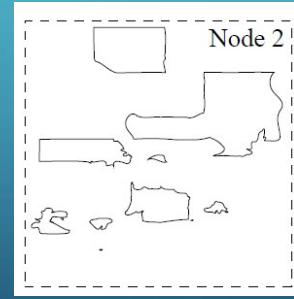
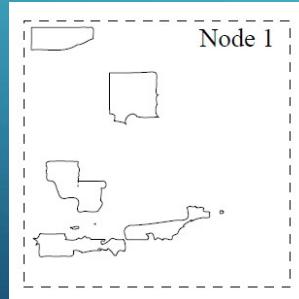


OPERACION UNION EN HADOOP SIN INDICE

2) **Unión Local:** Cada nodo debe calcular la Union Local de todos los polígonos que le llegan y arrojar una solo polígono de respuesta:

2.1) **Mapper (“Ki”, Polygon) => (“Ki”, Polygon)**

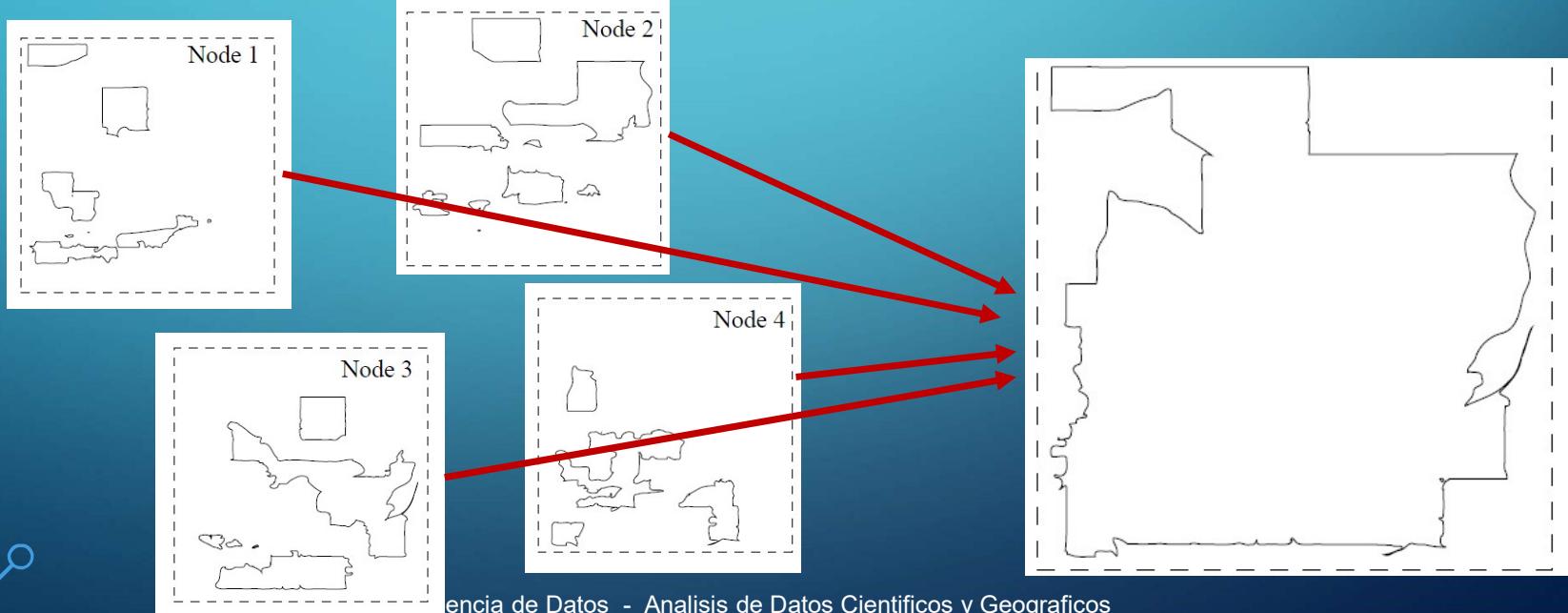
2.2) **Combiner (“Ki”, Lista(Polygon) => (“Ki”, NewPolygon) // union local**



OPERACION UNION EN HADOOP SIN INDICE

3) **Unión Global:** El nodo Reducer (único) calcular la unión de los polígonos que le llegan (resultado del local unión). Son muchos menos que los originales:

Reducer(“Ki”, Lista(NewPolygon)) => (“Ki”, PoligonoRespuesta)





OPERACION UNION EN HADOOP CON INDICE

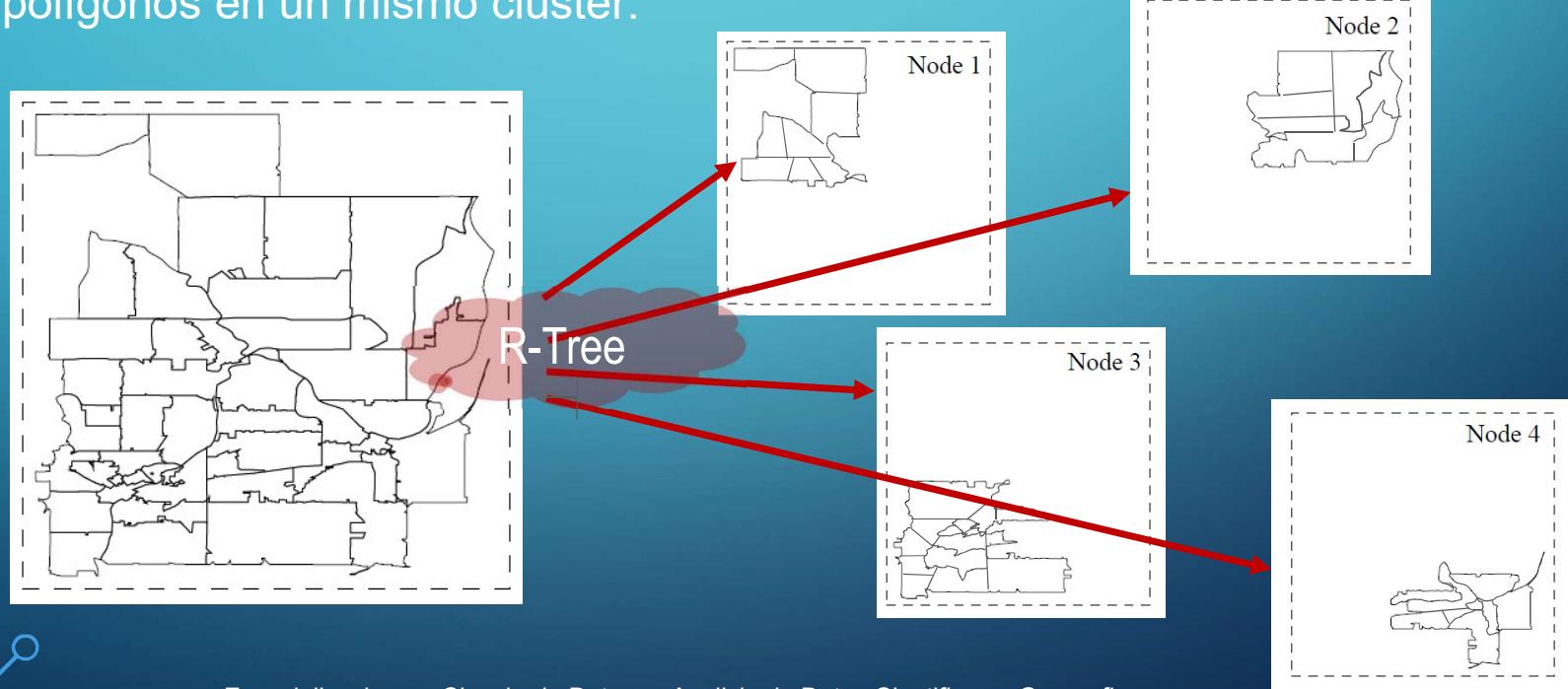
Los pasos son los mismos pero la idea es tomar ventaja de PARTICIONAR teniendo en cuenta la localidad espacial de los polígonos: polígonos adyacentes deben asignarse al mismo nodo.

Para ello se usa un índice R-Tree. Cada nodo R-Tree tiene el tamaño 64 M (bloque hadoop), así se asigna a un nodo.

R-Tree con nodo igual
al nodo cluster

OPERACION UNION EN HADOOP CON INDICE

1) Partición: Nodo Master distribuye el conjunto de polígonos input en cada nodo para el Mapper, usando un índice R-Tree que asegura contigüidad de polígonos en un mismo cluster:

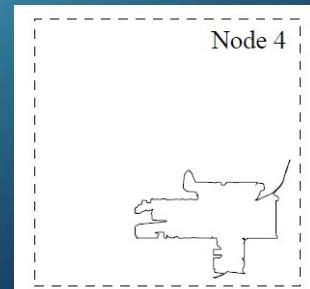
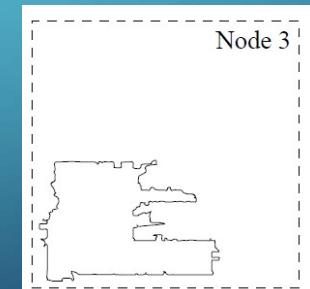
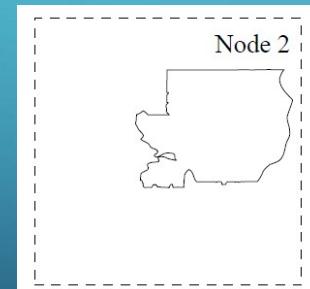
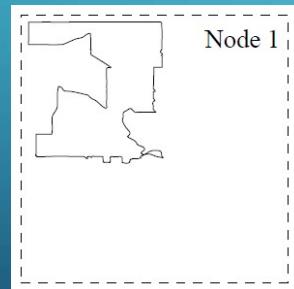


OPERACION UNION EN HADOOP CON INDICE

2) **Unión Local:** Cada nodo debe calcular la Union Local de todos los polígonos que le llegan y arrojar una solo polígono de respuesta:

2.1) **Mapper (“Ki”, Polygon) => (“Ki”, Polygon)**

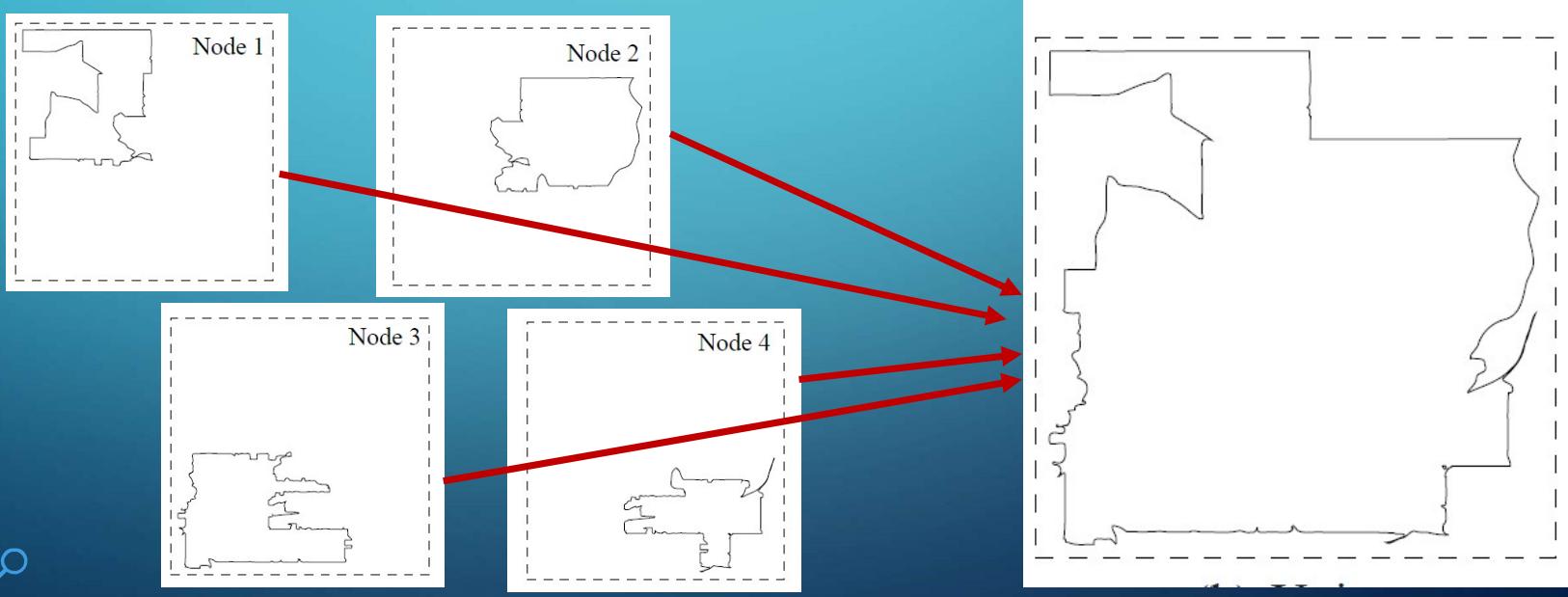
2.2) **Combiner (“Ki”, Lista(Polygon)) => (“Ki”, NewPolygon) // union local**



OPERACION UNION EN HADOOP CON INDICE

3) **Unión Global:** El nodo Reducer (único) calcular la unión de los polígonos que le llegan (resultado del local unión). Son muchos menos que los originales:

Reducer(“Ki”, Lista(NewPolygon)) => (“Ki”, PoligonoRespuesta)





OPERACION UNION EN HADOOP CON INDICE

Dónde estuvo la ventaja ?

En que los nodos Combiner (union local) generan un **polígono compacto** en vez de un multipolígono...

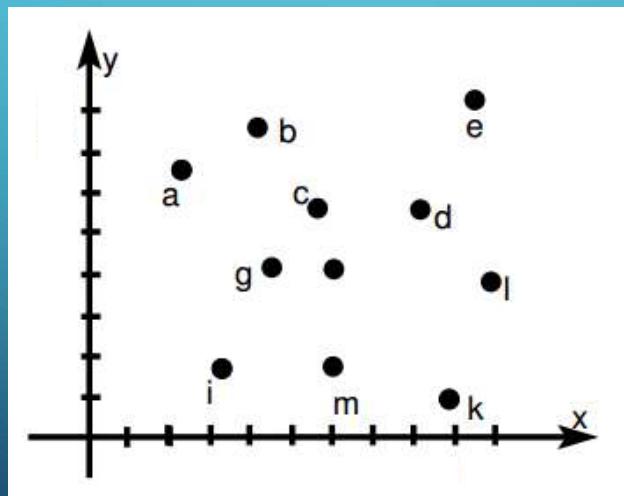
Menos procesamiento para la salida del Combiner y menos procesamiento para el Reducer

Bien por el uso de R Tree !!!

OPERACION SKYLINE

Punto Dominante:

Dado un conjunto de puntos Pts, se dice que un punto $P_i \in Pts$ domina a otro $P_j \in Pts \Leftrightarrow$ cada coordenada de P_i es \leq (o \geq , según se busque minimizar o maximizar) que la correspondiente coordenada de P_j



Con relacion \leq :

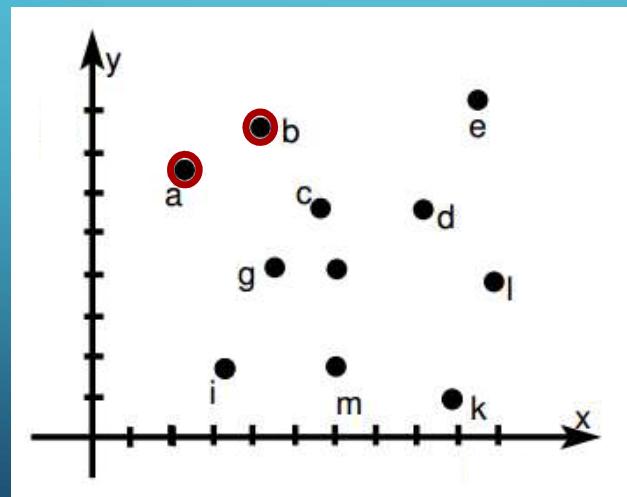
A a B

L a D

OPERACION SKYLINE

Punto Dominante:

Dado un conjunto de puntos Pts, se dice que un punto $P_i \in Pts$ domina a otro $P_j \in Pts \Leftrightarrow$ cada coordenada de P_i es \leq (o \geq , según se busque minimizar o maximizar) que la correspondiente coordenada de P_j



Con relación \leq :

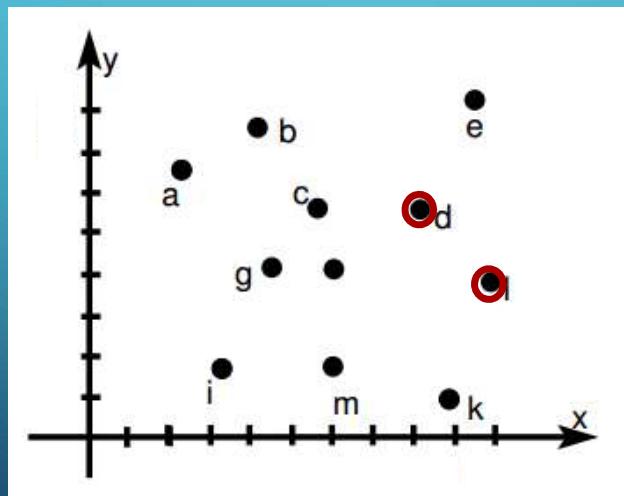
A domina a B

L a D

OPERACION SKYLINE

Punto Dominante:

Dado un conjunto de puntos Pts, se dice que un punto $P_i \in Pts$ domina a otro $P_j \in Pts \Leftrightarrow$ cada coordenada de P_i es \leq (o \geq , según se busque minimizar o maximizar) que la correspondiente coordenada de P_j



Con relacion \leq :

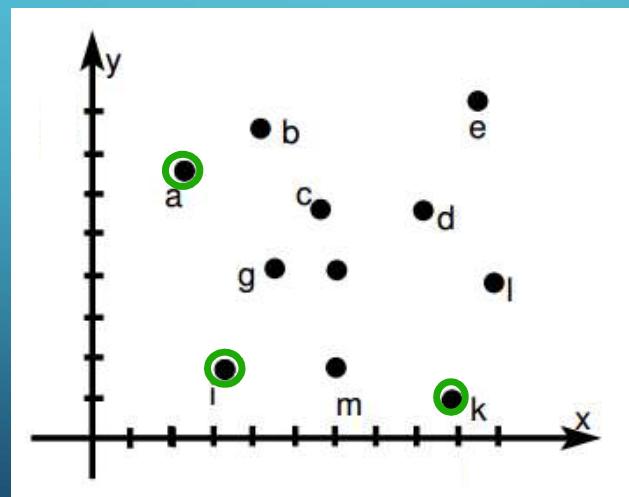
A domina a B

L no domina a D

OPERACION SKYLINE

Skyline de un conjunto Pts:

Es el conjunto de puntos de Pts que **NO son dominados por ningún otro punto** de Pts



Skyline $\leq ?$

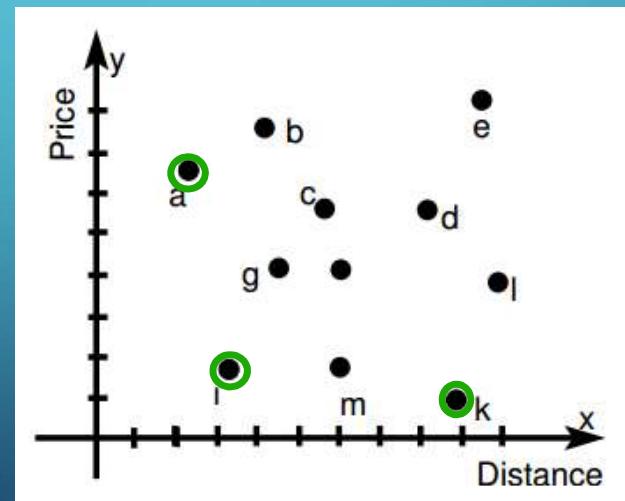
Skyline = { a, i, k }

EJEMPLO SKYLINE

Supongamos hoteles con 2 propiedades: precio y distancia a la playa.

Los puntos **a**, **i**, **k** son buenas alternativas...

En particular, el punto **i** es el mejor en cuanto a trade-off entre precio y distancia a la playa

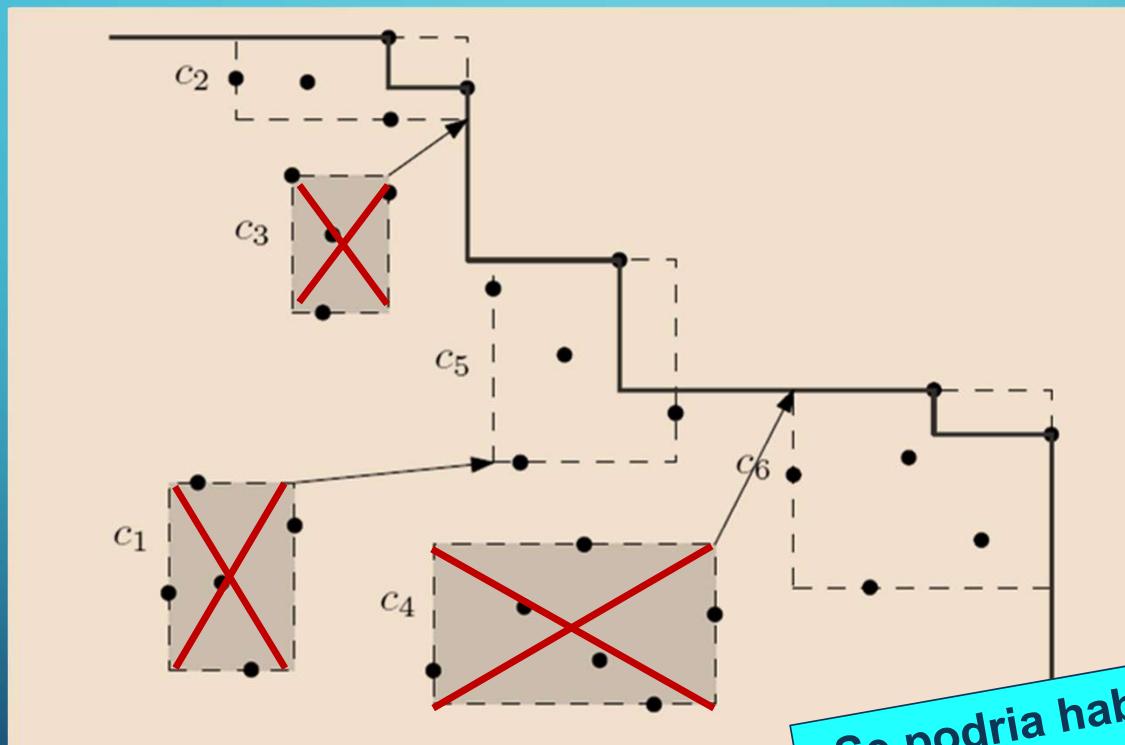




EJEMPLO SKYLINE CON HADOOP SPATIAL

- **Particion** Se partitionan los puntos usando un R-Tree.
- **Filtrado** Antes del mapeo se filtran las celdas: si una celda C_i domina a otra celda C_j (al menos un punto de C_i domina a todos los puntos de C_j), entonces se elimina C_j .
- **Local** Dado que al menos hay un punto en cada lado de la celda (por la construccion del Rtree), se comparan los vertices de las celdas.
- **Global** Este paso poda las celdas dominadas por otras, que no se computaran en el Skyline Local, y en consecuencia, tampoco en el Global.

EJEMPLO SKYLINE CON HADOOP SPATIAL



C2 domina a C3

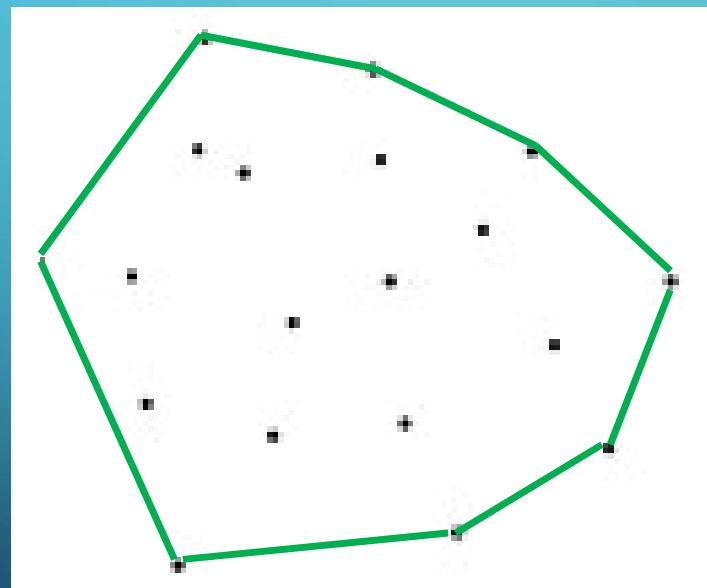
C5 domina a C1

C6 domina a C4

Se podria haber utilizado este filtrado con
Hadoop tradicional?

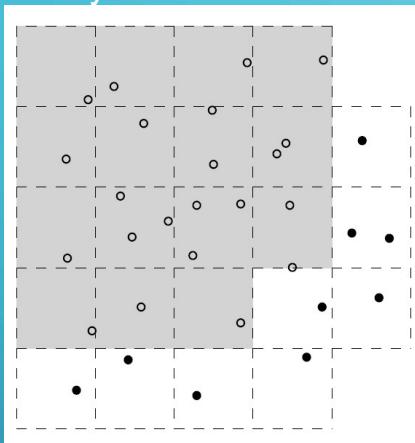
OPERACION CONVEX HULL

Dado un conjunto de puntos Pts, es el polígono convexo más pequeño que contiene todos los puntos de Pts.

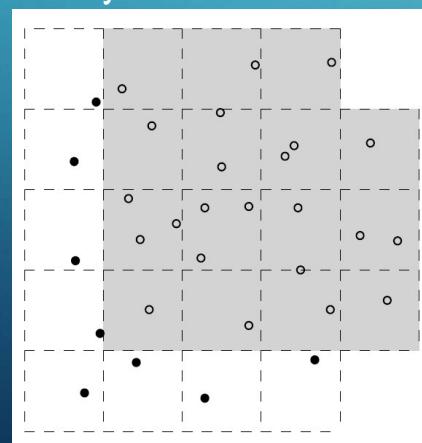


OPERACION CONVEX HULL EN SPATIAL HADOOP

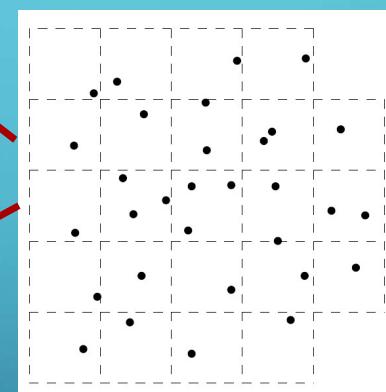
Skyline Xmax-Ymin



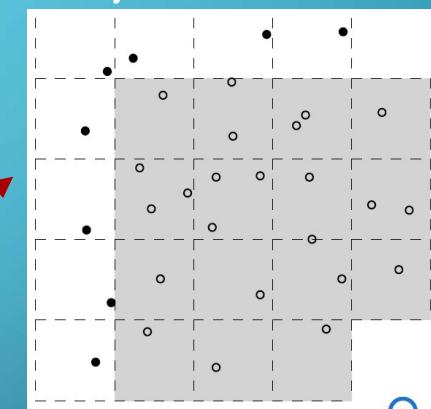
Skyline Xmin-Ymin



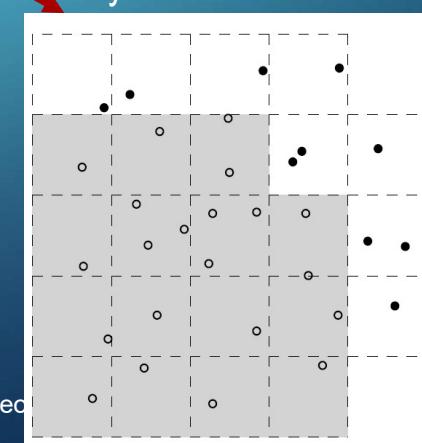
Input



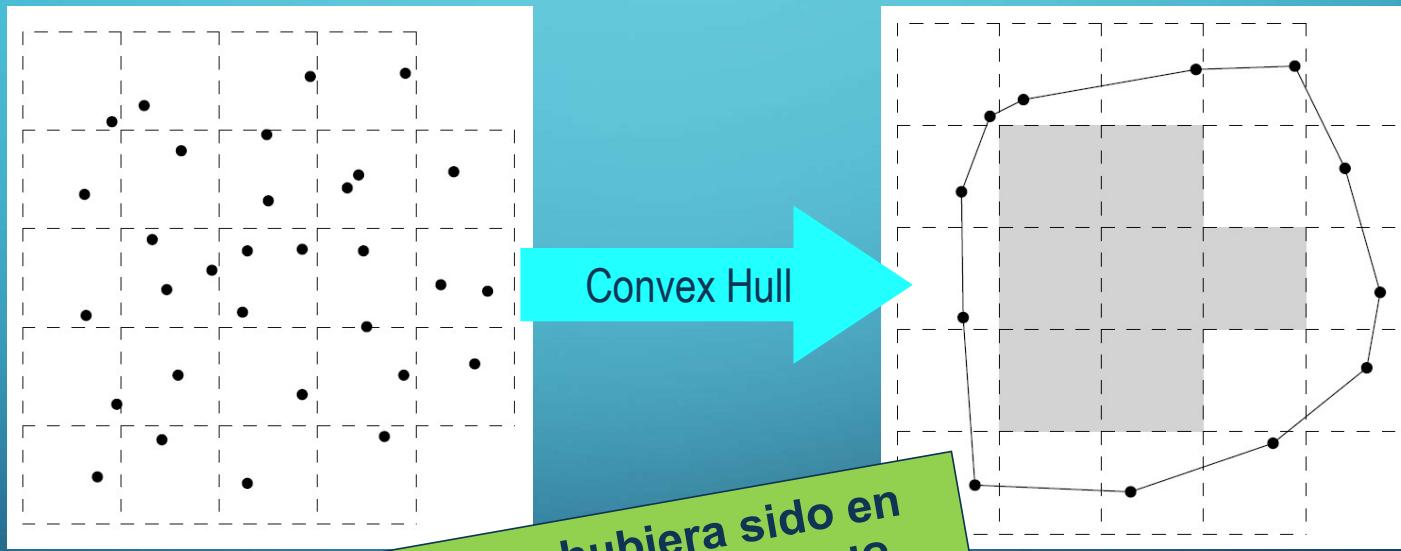
Skyline Xmin-Ymax



Skyline Xmax-Ymax



OPERACION CONVEX HULL EN SPATIAL HADOOP

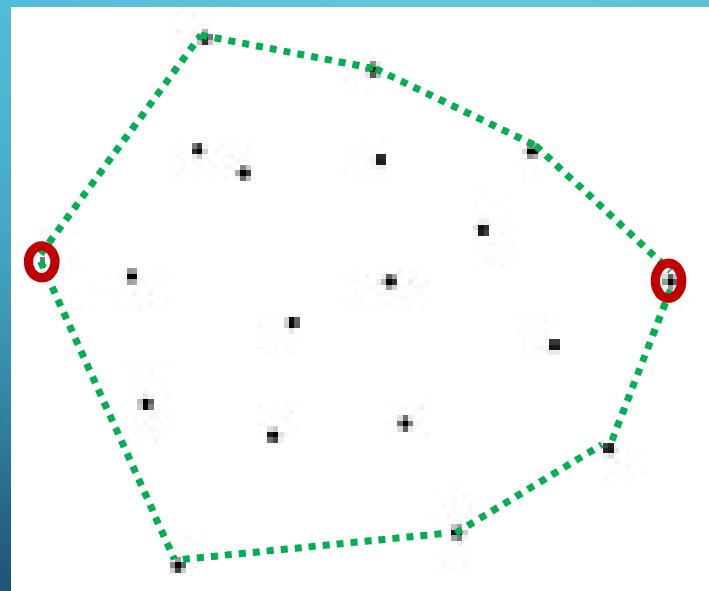


Convex Hull

Como hubiera sido en
Hadoop tradicional?

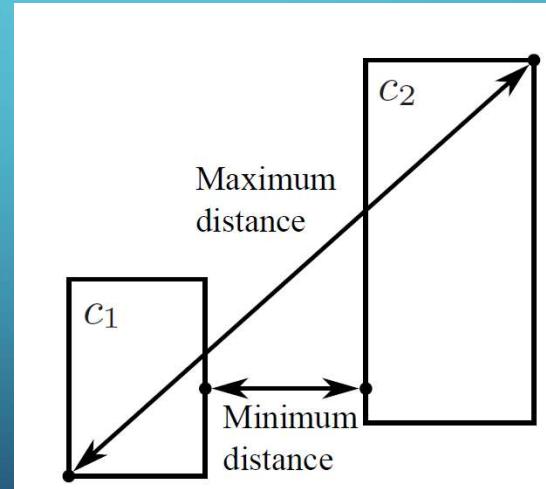
OPERACION FARTHEST PAIR

Dado un conjunto de puntos Pts, el **Farthest Pair** es el par de puntos que tiene la distancia Euclídea más grande entre ellos.



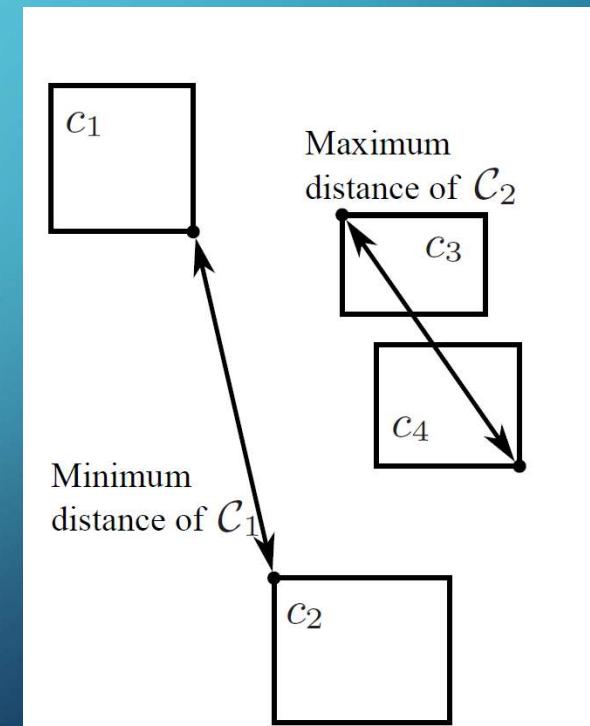
OPERACION FARTHEST PAIR EN SPATIAL HADOOP

- **Particion** Se partitionan los puntos con RTree
- **Filtrado** Para cada par de celdas, se calcula la máxima y minima distancia entre un punto de una y un punto de la otra.
- **Local**
- **Global**



OPERACION FARTHEST PAIR EN SPATIAL HADOOP

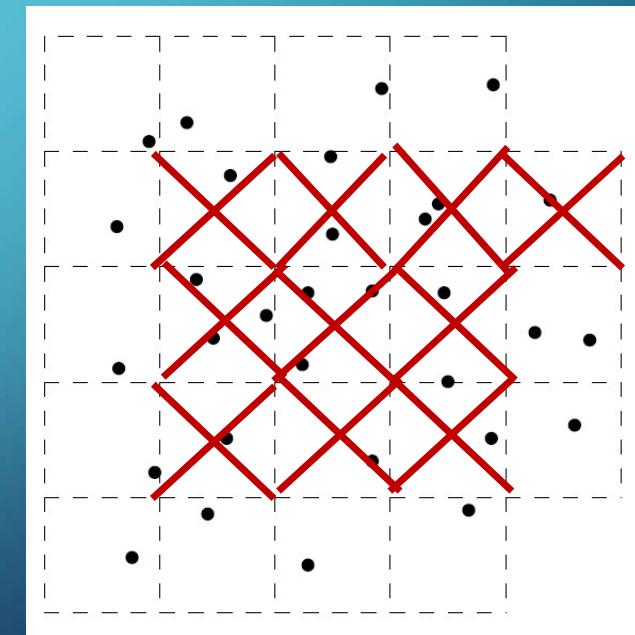
- **Particion** Se partitionan los puntos con RTree
- **Filtrado** Para cada par de celdas, se calcula la máxima y minima distancia entre un punto de una y un punto de la otra.
- **Local**
- **Global** Si la minima distancia de un par de celdas (c_1, c_2) es mayor que la maxima distancia de un par (c_3, c_4), se dice que el primer par domina, y se poda el segundo.



OPERACION FARTHEST PAIR EN SPATIAL HADOOP

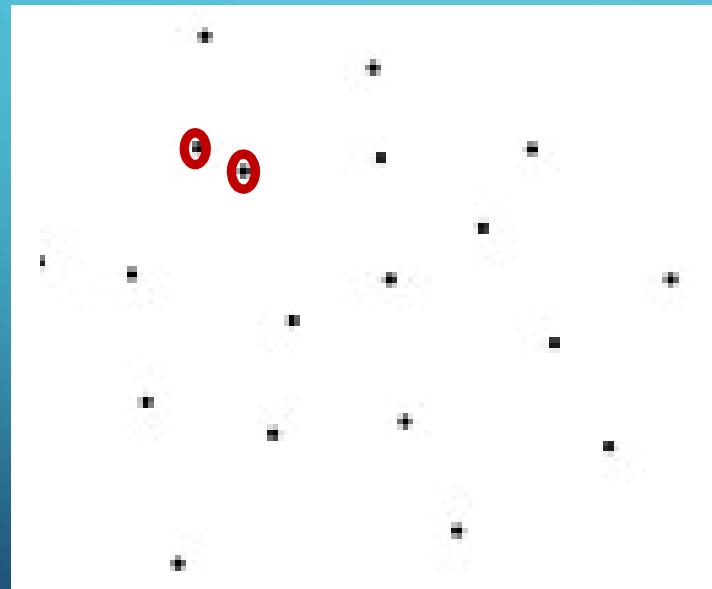
- **Particion** Se partitionan los puntos con Rtree
- **Filtrado** Para cada par de celdas, se calcula la máxima y minima distancia entre un punto de una y un punto de la otra.
- **Local**
- **Global** Si el par de celdas (c_1, c_2) domina al par (c_3, c_4), se poda el segundo.

Sobre los pares que permanecen se calcula el Local, y sobre los resultados parciales el Global.



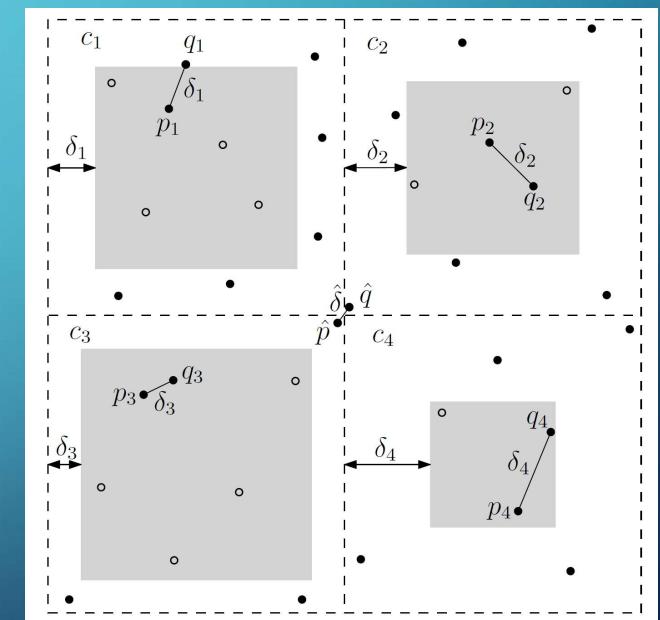
OPERACION SHORTEST PAIR

Dado un conjunto de puntos Pts, el **Shortest Pair** es el par de puntos que tiene la distancia Euclídea más pequeña entre ellos.



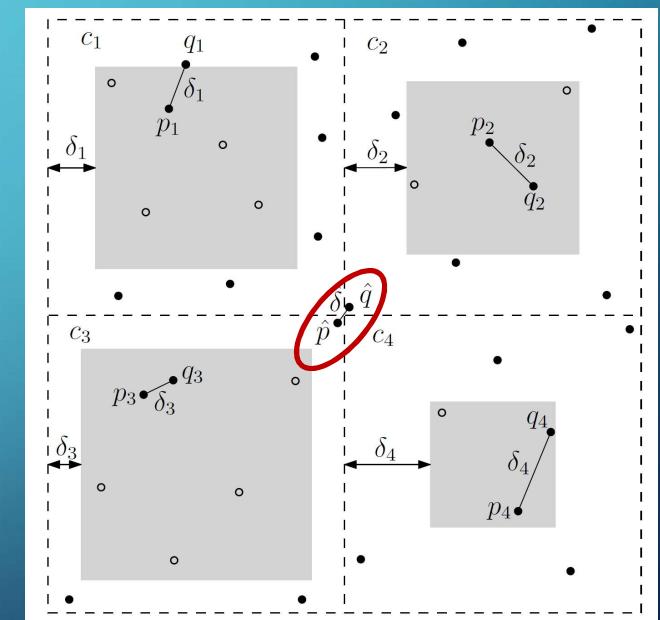
OPERACION CLOSEST PAIR EN SPATIAL HADOOP

- **Particion** Se partitionan los puntos con RTr
- **Local** Cada celda C_i retorna localmente sus puntos mas cercanos ($dist\ di$) y los retorna junto con los puntos de celdas vecinas que caen dentro de un buffer de radio di .
- **Global**



OPERACION CLOSEST PAIR EN SPATIAL HADOOP

- **Particion** Se partitionan los puntos con RTr
- **Local** Cada celda C_i retorna localmente sus puntos mas cercanos ($dist\ di$) y los retorna junto con los puntos de celdas vecinas que caen dentro de un buffer de radio di .
- **Global** En el paso Global se toman todos los puntos recibidos y se busca el par mas cercano.



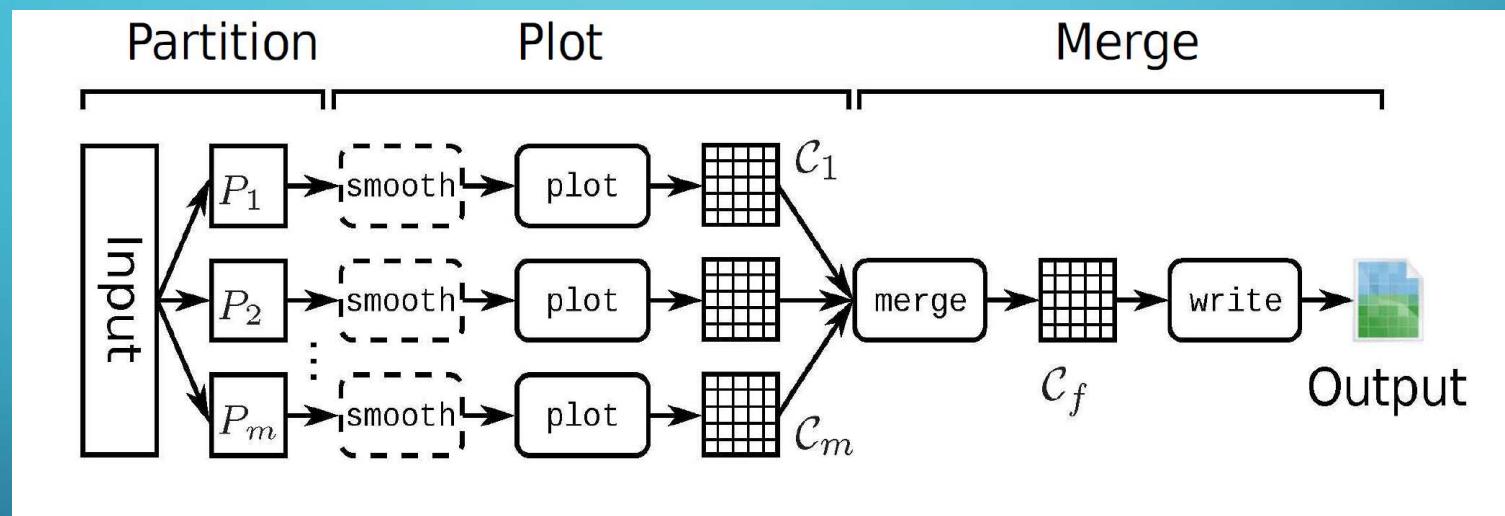


VISUALIZACION CON HADOOPVIZ

En el 2016, Ahmed Eldawy, Mohamed F. Mokbel y Christopher Jonathan presentaron HadoopViz, un framework extensible que no solo visualiza eficazmente datos espaciales, sino que permite que el usuario implemente su logica a traves de 5 funciones abstractas:

- Smooth
- Create-canvas
- Plot
- Merge
- Write

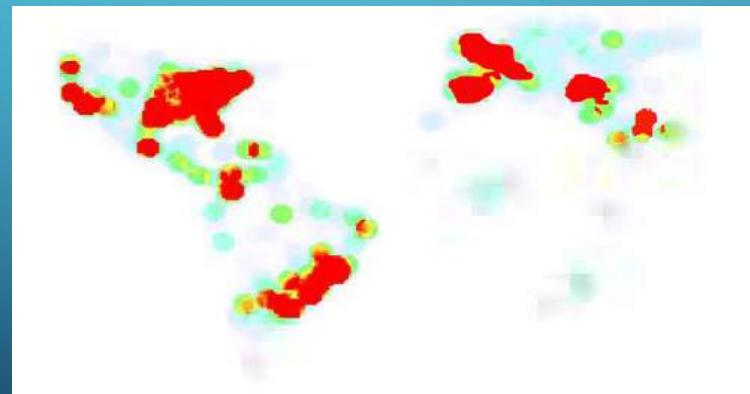
VISUALIZACION CON HADOOPVIZ



EJEMPLO EN HADOOPVIZ: FREQUENCY HEAT MAP

Input: conjunto de puntos representando algun evento
(por ejemplo tweets geo-tageados)

Output: mapa coloreado por densidad
(por ejemplo, zonas densas ne rojo, zonas esparcidas en azul)



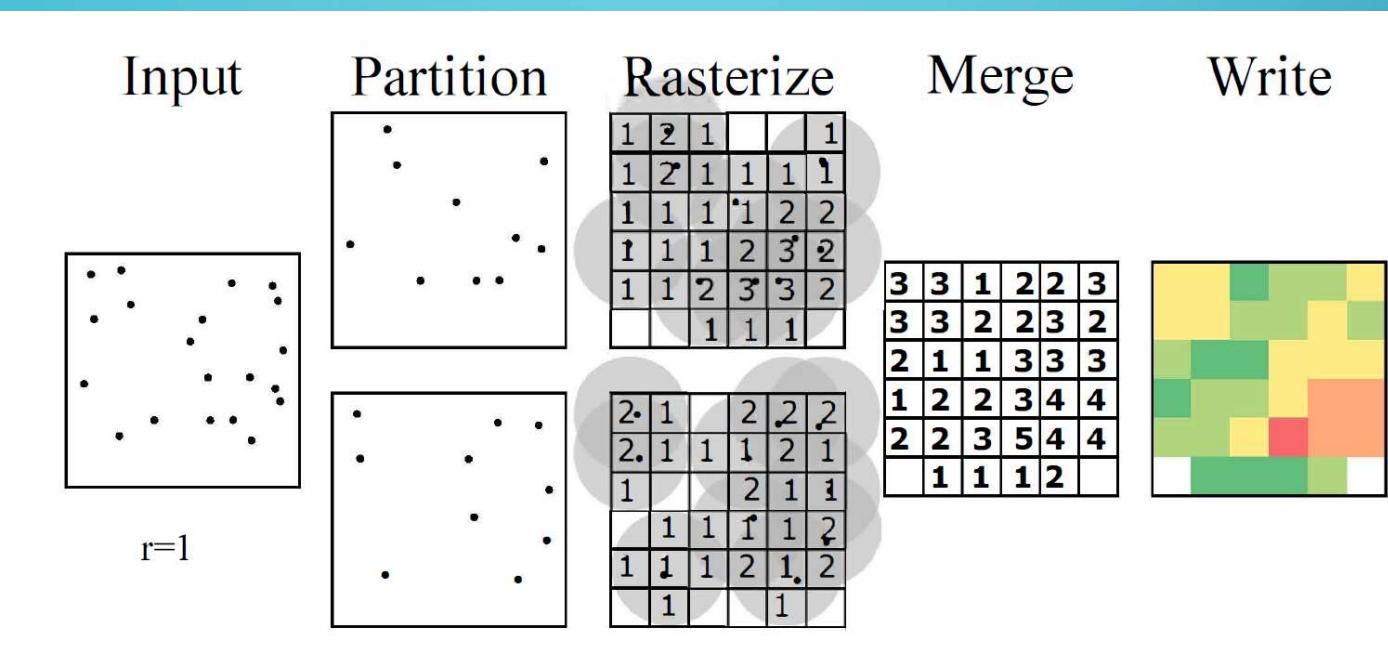


VISUALIZACION CON HADOOPVIZ

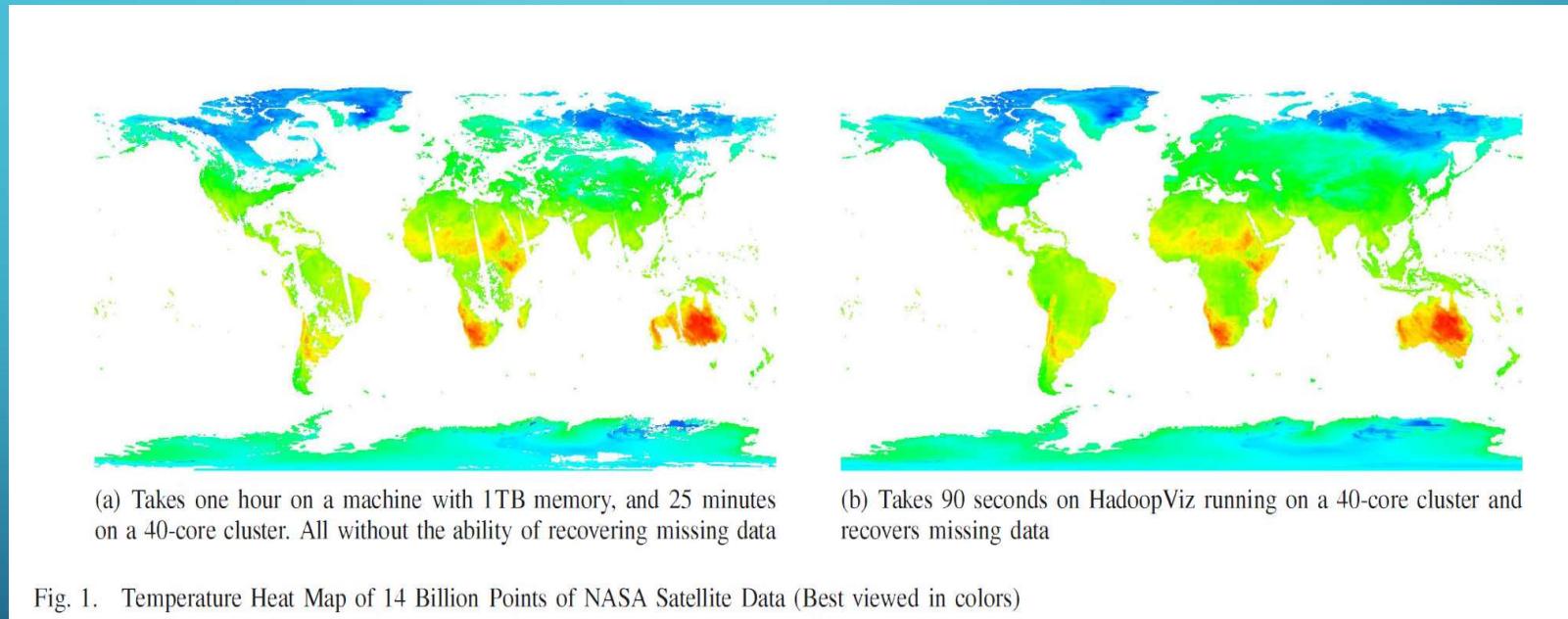
Debemos definir las 5 funciones:

- 1) Smooth: no necesaria
- 2) Create-canvas: arreglo 2D donde cada valor representa el numero de registros en dicha coordenada. Se inicializa en cero.
- 3) Plot: para cada punto, lo proyecta en el arreglo 2D e incrementa todas las entradas en un radio determinado P.
- 4) Merge: toma dos arreglos 2D y los suma celda a celda
- 5) Write: toma el arreglo 2D y lo convierte en una imagen PNG, normalizan do las densidades en el rango [0, 1] , y luego calculando un color para cada pixel (interpolando entre rojo y azul).

EJEMPLO EN HADOOPVIZ: FREQUENCY HEAT MAP



HADOOPVIZ: VENTAJA DEL SMOOTH





MANOS A LA OBRA !!!