



Casa abierta al tiempo

**UNIVERSIDAD AUTÓNOMA METROPOLITANA**  
**Unidad Cuajimalpa**



# K-MEANS CLUSTERING ALGORITHM

SEBASTIÁN MARROQUÍN – [SEBASMARRO10@GMAIL.COM](mailto:SEBASMARRO10@GMAIL.COM)

# AGENDA

1. Introducción
2. Descripción de la implementación
3. Evaluación de rendimiento
4. Conclusiones



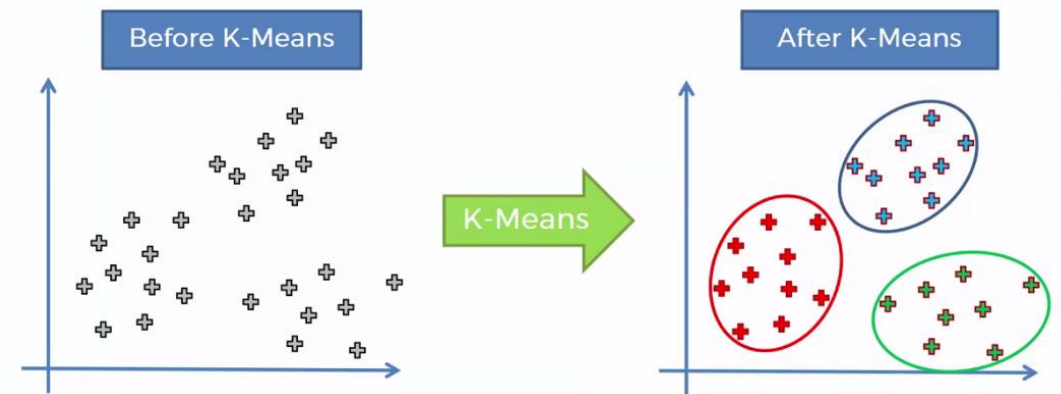
# I. INTRODUCCIÓN

K-MEANS CLUSTERING ALGORITHM



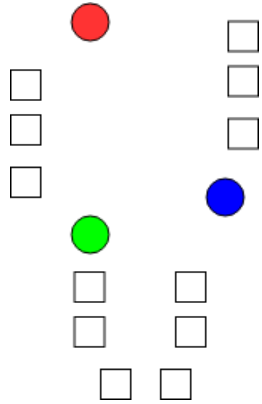
# CONTEXTO

- CLUSTERING: es la tarea de asignar un conjunto de objetos a grupos (llamados clústeres) para que los objetos en el mismo clúster sean más similares (en un sentido u otro) entre sí que a los de otros clústeres.
- El Agrupamiento por K-Means es un método de agrupación que particiona  $n$  puntos de datos en  $k$  clústeres ( $n \gg k$ ) en los que cada observación pertenece al clúster con la media más cercana

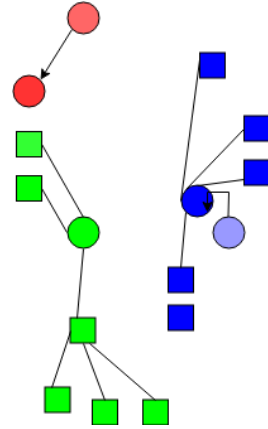


*K Means Clustering : Identifying F.R.I.E.N.D.S in the World of Strangers.* (2018). [Image]. Retrieved from <https://towardsdatascience.com/k-means-clustering-identifying-f-r-i-e-n-d-s-in-the-world-of-strangers-695537505d>

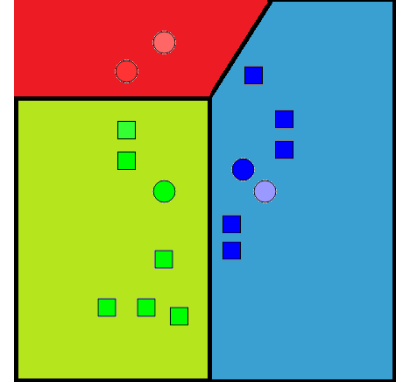
# CONTEXTO



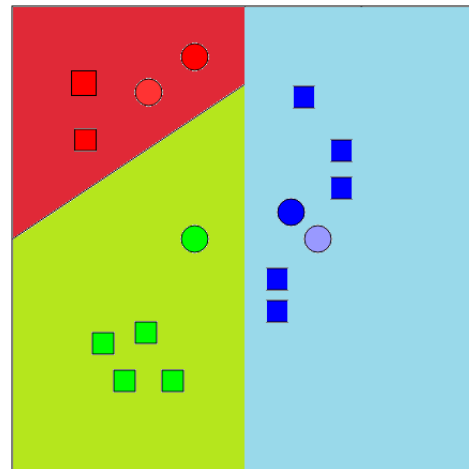
Se escogen K puntos



El centroide de cada K clúster  
se convierte ahora en un  
nuevo mean



Los grupos K se crean al  
asociar cada punto al conjunto  
con la media más cercana




Repetir hasta la  
convergencia

# CONTEXTO

- La proximidad se calcula mediante la función de distancia, que es principalmente la distancia euclidiana.

$$d(P1, P2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$d_{(i,j)}^2 = \sum_{k=1}^k (x_{ik} - x_{jk})^2$$



Una suposición importante que se debe hacer es que los puntos de datos son independientes entre sí. En otras palabras, no existe dependencia entre ningún punto de datos.

# PROBLEMA

- Algoritmo:

Inicializar **los centroides de los clústeres**  $\mu_1, \mu_2, \dots, \mu_k \in \mathbb{R}^n$  de manera aleatoria

Repetir hasta que convergan:

Para cada i, hacer:

$$c^{(i)} := \arg \min_j \|x^i - \mu_j\|^2$$

Para cada j, hacer:

$$\mu_j := \frac{\sum_{i=1}^m 1\{c^i=j\}x^i}{\sum_{i=1}^m 1\{c^i=j\}}$$



## PROBLEMA (2)

- El parámetro  $k$  es el número de grupos que queremos encontrar.
- Los centroides agrupados  $\mu_j$  representan las conjeturas actuales (para las posiciones)
- Para inicializar los centroides del grupo (en el paso I del algoritmo anterior), podríamos elegir  $k$  ejemplos de entrenamiento al azar y con ellos establecer los centroides del grupo para que sean iguales a los valores de  $k$ .

# JUSTIFICACIÓN

- Lo que se trata de hacer es resolver el algoritmo del agrupamiento por K-Means de manera paralela, con paso de mensajes, mismo proporcionado por la herramienta MPI4PY.
- La manera en como podemos encontrar este tipo de problemas es: **BIG DATA, DATA MINING, etc.**
- **TENDRA ÉXITO POR QUE EL OBJETIVO DEL COMPUTO CONCURRENTES ES ACELERAR UN PROGRAMA SECUENCIAL, ADEMÁS DE IMPLEMENTARLO DE MANERA CONCURRENTES.**

## JUSTIFICACIÓN (2)

- Uno de los riesgos es que **FALLE LA IMPLEMENTACIÓN.**
- La implementación tomo en tiempo: **3 semanas (aunque no es cierto)**
- La manera en como se verificara el éxito será **con las evaluaciones de rendimiento.**



# OBJETIVO PRINCIPAL



- *Aplicar los conceptos básicos del cómputo concurrente en la programación de algoritmos para la solución de problemas planteamos los siguientes objetivos particulares que deben de cumplirse:*

BLOGECONOMISTA.COM. (2011). *Gestión por Objetivos* [Image]. Retrieved from <http://blogeconomista.com/gestion-por-objetivos/>

# OBJETIVOS SECUNDARIOS

- Implementación de la versión secuencial del algoritmo K-Means.
- Diseño e Implementación de un algoritmo concurrente que solucione el problema de los K-Means
- Evaluar el desempeño de un algoritmo concurrente tomando como base un algoritmo secuencial.





## 2. DESCRIPCIÓN DE LA IMPLEMENTACIÓN

K-MEANS CLUSTERING ALGORITHM



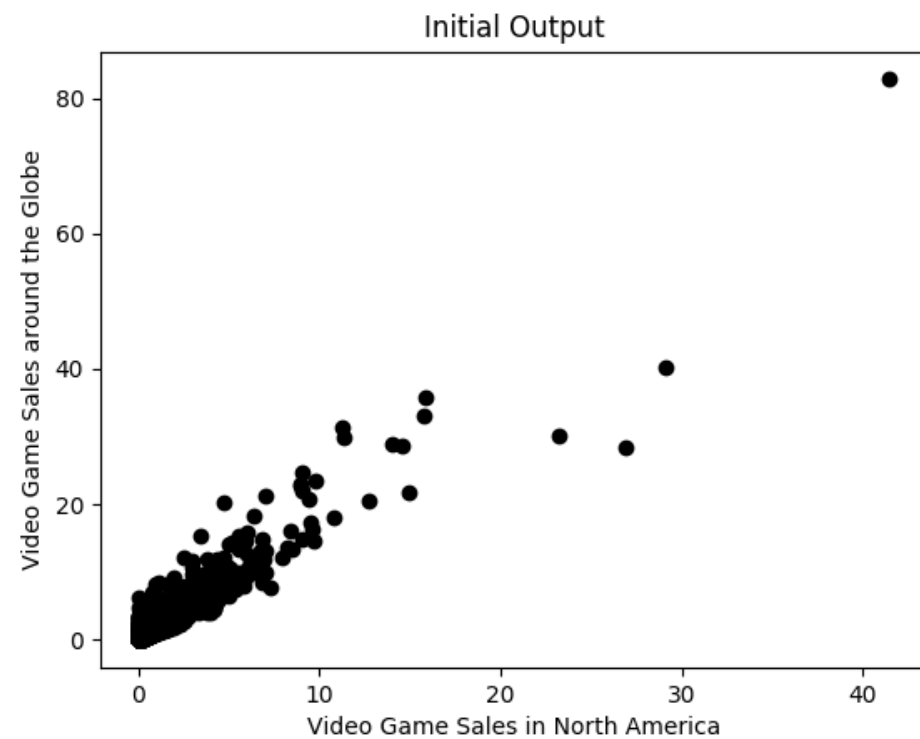
- Con motivos de la implementación, decidí utilizar un conjunto de datos ubicado en la plataforma de Kaggle.
- Este conjunto de datos contiene una lista de videojuegos con ventas superiores a 100,000 copias.



OPEN DATA SCIENCE. (2014). *Kaggle* [Image]. Retrieved from <https://medium.com/@ODSC/10-tips-to-get-started-with-kaggle-fc7cb9316d27>

# DATA & PLOTTING

	Platform	Year	Genre	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	2006	1	41.49	29.02	3.77	8.46	82.74
1	2	1985	2	29.08	3.58	6.81	0.77	40.24
2	1	2008	3	15.85	12.88	3.79	3.31	35.82
3	1	2009	1	15.75	11.01	3.28	2.96	33
4	3	1996	4	11.27	8.89	10.22	1	31.37
5	3	1989	5	23.2	2.26	4.22	0.58	30.26
6	4	2006	2	11.38	9.23	6.5	2.9	30.01
7	1	2006	6	14.03	9.2	2.93	2.85	29.02
8	1	2009	2	14.59	7.06	4.7	2.26	28.62
9	2	1984	7	26.93	0.63	0.28	0.47	28.31
10	4	2005	8	9.07	11	1.93	2.75	24.76
11	4	2005	3	9.81	7.57	4.13	1.92	23.42
12	3	1999	4	9	6.18	7.2	0.71	23.1
13	1	2007	1	8.94	8.03	3.6	2.15	22.72
14	1	2009	1	9.09	8.59	2.53	1.79	22
15	5	2010	6	14.97	4.94	0.24	1.67	21.82
16	6	2013	9	7.01	9.27	0.97	4.14	21.4
17	7	2004	9	9.43	0.4	0.41	10.57	20.81
18	8	1990	2	12.78	3.75	3.54	0.55	20.61
19	4	2005	6	4.75	9.26	4.16	2.05	20.22
20	4	2006	4	6.42	4.52	6.04	1.37	18.36





# K-Means Clustering Algorithm Design

Sebastián Marroquín

## 1 Sequential K-Means

---

**Algorithm 1** k-Means Clustering

---

**Input**

1.  $X = x_1, \dots, x_n$

2.  $k$

1: Inicializar: Centroides de Clusters  $\mu_1, \mu_2, \dots, \mu_k$

2: **for** every  $x_i \in 1, \dots, n$  **do**

3:    $c^{(i)} := \arg \min_j \|x^i - \mu_j\|^2$

4: **for** every  $c^i \in 1, \dots, k$  **do**

5:   
$$\frac{\sum_{i=1}^n 1\{c^i=j\} \cdot x^i}{\sum_{i=1}^n 1\{c^i=j\}}$$

6: Repetir los pasos 3 y 4 hasta que los centroides se agrupen (i.e la optima solución sea hayada)

7: **Output**

8:   1.  $C = c_1, \dots, c_k \in \mathbb{R}^n$  (centroides encontrados)

      2.  $y = y_1, \dots, y_n$

---

IMPLEMENTACIÓN  
SECUENCIAL

# IMPLEMENTACIÓN SECUENCIAL (2)

sebastian@DESKTOP-M1LRMKV: /mnt/c/Users/sebas/Desktop/UAM/19-I/ComputoConcurrente/PRESENTACIÓN/Code/Seq

```
sebastian@DESKTOP-M1LRMKV:/mnt/c/Users/sebas/Desktop/UAM/19-I/ComputoConcurrente/PRESENTACIÓN/Code/Seq$ python3 sequential_kmeans.py 5
```

Final Centers are:

```
[[1658.0, 8.522158577027435, 2005.2274012508774, 5.694603557431414, 0.9852607778112747, 0.5824962315345179, 0.25423876997286493, 0.18909255351220763, 2.011091347603256], [4  
975.5, 9.091320072332731, 2005.6060159209226, 6.005424954792043, 0.20142555756479755, 0.09420132610006057, 0.06792646172393024, 0.033604581072935714, 0.39712477396020934],  
[8294.0, 9.53449834287436, 2006.3040576284836, 6.034046399517927, 0.09169629406447766, 0.03638144019282932, 0.035203374510394805, 0.01238324796625494, 0.17574269358240419],  
[11614.0, 9.920204757603132, 2006.6898931202536, 6.404095152062632, 0.0379223125564591, 0.014619090635350827, 0.021348991267690413, 0.004766636555254357, 0.078912978018671  
77], [14936.0, 10.978332831778513, 2008.2015627572061, 7.16190189587722, 0.00789346975624446, 0.006057779115257367, 0.010409268733072356, 0.0006319590731266925, 0.026069816  
430936213]]
```

Execution time 28.247859239578247 seconds

```
sebastian@DESKTOP-M1LRMKV:/mnt/c/Users/sebas/Desktop/UAM/19-I/ComputoConcurrente/PRESENTACIÓN/Code/Seq$
```

# SOLUCIÓN PARALELA - AGENDA

1. Descripción del Algoritmo implementado
2. Implementación del diseño del algoritmo paralelo

# ALGORITMO PARALELO

Mientras sea **VERDADERO**, hacer:

1. **ROOT** hace *bcast* a los vectores  $k$  a todos los procesadores.
2. Cada proceso calcula la distancia de cada vector a los vectores de los centroides  $k$ .
3. Cada proceso vuelve a calcular los vectores de centroides  $k$  según los centroides reasignados.
4. Cada proceso realiza una reducción de **ALL to ALL** de los centroides  $k$ . Después de cada iteración, una reducción de **ALL to ALL** sincroniza los vectores centroides.

# IMPLEMENTACIÓN PARALELA

```
sebastian@DESKTOP-M1LRMKV:/mnt/c/Users/sebas/Desktop/UAM/19-I/ComputoConcurrente/PRESENTACIÓN/Code/Parallel$ mpiexec -n 8 python parallel_kmeans.py
```

```
-----  
WARNING: Linux kernel CMA support was requested via the  
btl_vader_single_copy_mechanism MCA variable, but CMA support is  
not available due to restrictive ptrace settings.
```

The vader shared memory BTL will fall back on another single-copy  
mechanism if one is available. This may result in lower performance.

Local host: DESKTOP-M1LRMKV

```
-----  
[DESKTOP-M1LRMKV:00221] 7 more processes have sent help message help-btl-vader.txt / cma-permission-denied  
[DESKTOP-M1LRMKV:00221] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error messages  
Final centers are:
```

```
[[1034.5, 8.369565217391305, 2004.6951409434414, 5.66328502415459, 1.344009661835748, 0.8020628019323675, 0.35033816425120534, 0.25800000000000034, 2.7543719806763374], [310  
5.5, 8.77364864864865, 2005.9085906094383, 5.776061776061776, 0.34403957528957485, 0.18620173745173677, 0.08854729729729742, 0.06364864864864896, 0.6825772200772103], [5177.  
5, 9.195945945945946, 2005.6026371505325, 6.074324324324325, 0.18650579150579147, 0.08501447876447939, 0.06473938223938219, 0.031100386100386443, 0.36712837837837375], [7250  
.5, 9.374156219864995, 2006.1492921693975, 6.012536162005786, 0.11304725168755979, 0.04783027965284473, 0.04378977820636458, 0.016364513018322258, 0.2208823529411756], [9325  
.0, 9.670843373493977, 2006.3440993667812, 6.1248192771084335, 0.07185060240963866, 0.02504578313253023, 0.029836144578313215, 0.008930120481927735, 0.13614457831325447], [1  
3479.0, 10.54978354978355, 2007.2457042196727, 6.622895622895623, 0.017258297258297417, 0.009292929292929294, 0.015372775372775429, 0.0017412217412217253, 0.0446801346801354  
3], [11401.0, 9.76600866634569, 2006.7146907218996, 6.453538757823784, 0.03965816080885902, 0.015373134328358264, 0.022291766971593612, 0.005291285507944062, 0.0823784304285  
017], [15558.0, 11.168831168831169, 2008.579158547002, 7.34968734968735, 0.004694564694564634, 0.004559884559884498, 0.008282828282828253, 0.00013949013949013953, 0.01890331  
890331816]]
```

Execution time 66.5158331394 seconds



# 3. EVALUACIÓN DE RENDIMIENTO

K-MEANS CLUSTERING ALGORITHM



# EVALUACIÓN DE RENDIMIENTO



Los resultados que se presentarán a continuación serán sobre la implementación del algoritmo de agrupamiento de los K-Means.



Se mostrarán graficas de rendimiento

# SPEEDUP

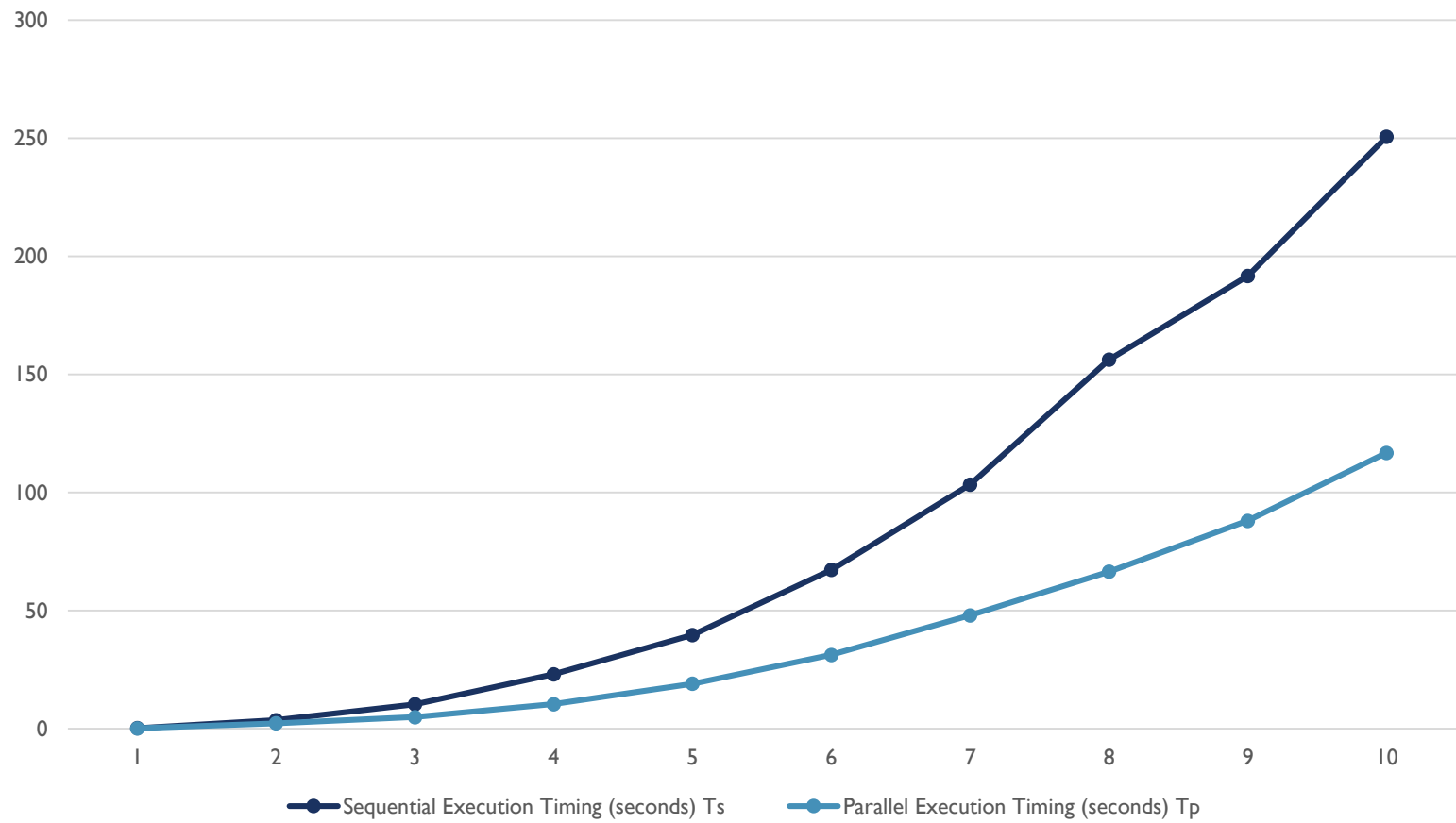
## ANALYSIS

Number of clusters	Sequential Execution Timing (seconds) $T_s$	Parallel Execution Timing (seconds) $T_p$	Speedup ( $T_s/T_p$ )
1	0.31026	0.19863	1
2	3.68241	2.34552	1
3	10.39307	4.97802	2
4	23.13027	10.4285	2
5	39.70104	19.0684	2
6	67.29214	31.2834	2
7	103.42623	48.05842	2
8	156.26017	66.51583	2
9	191.71026	88.09356	2
10	250.65177	116.84431	2



# GRAFICA DE RENDIMIENTO

Time Analysis



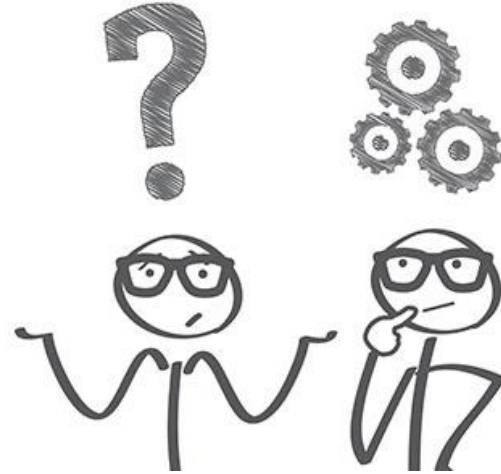


## 4. CONCLUSIONES

K-MEANS CLUSTERING ALGORITHM



FICA. (2017). *Si quieres resolver un problema con alguien, sigue estos cuatro pasos* [Image]. Retrieved from <http://www.ficaconsulting.com.do/cw/publicaciones/17-otros/801-si-quieres-resolver-un-problema-con-alguien-sigue-estos-cuatro-pasos>



Diseñar e Implementar de manera paralela el algoritmo de agrupamiento de los K-Means, utilizando el paso de mensajes.

PROBLEMA A  
RESOLVER

FICA. (2017). *Si quieres resolver un problema con alguien, sigue estos cuatro pasos* [Image]. Retrieved from <http://www.ficaconsulting.com.do/cw/publicaciones/17-otros/801-si-quieres-resolver-un-problema-con-alguien-sigue-estos-cuatro-pasos>



## RESULTADOS OBTENIDOS

Como se pudo ver en el apartado de **Evaluación de Rendimiento**, pudimos observar que existe una aceleración mayor a la implementación del algoritmo secuencial.



# GRACIAS POR SU ATENCIÓN

Presentó: Sebastián Marroquín Martínez, [sebasmarro10@gmail.com](mailto:sebasmarro10@gmail.com)

Nombre del proyecto: K-Means Clustering Algorithm