



# Learning on the Edge: Unlocking the Storage Bottleneck with a Divide and Conquer Approach

Jalil Boukhobza,  
ENSTA-Bretagne, Lab-STICC UMR 6285

# Who am I ?

<https://www.ensta-bretagne.fr/boukhobza/>  
jalil.boukhobza@ensta-bretagne.fr

## / Education

- / 1999 – Engineer in Electronics, INELEC, Algeria
- / 2000 – Master in computer science, Univ. Versailles
- / 2004 – PhD Univ. Versailles, PRISM Lab., Storage Systems

## / Prof. Exp.

- / 2004-2006 – Research and teaching assistant, Univ. Versailles
- / 2006-2020 – Associate Professor, Univ. Bretagne Occidentale
- / 2013-\* – Part time researcher at IRT b<>com, Rennes
- / 2016 Invited researcher, Hong Kong Polytechnic University
- / 2024 Invited scholar at Academia Sinica Taipei
- / 2020-\* Professor, ENSTA-Bretagne Lab-STICC
- / Team leader of SHAKER (Software/HARdware and unKnown Environment inteRactions)

## / Research topics:

- / Storage and memory systems
  - / Modeling / benchmarking / data placement / I/O optimization / I/O tracing
- / **Domains:** Cloud and Fog resource management, Embedded systems, HPC

## / Current projects:

- / CEA: DataMeSS, data placement in multi-tiered storage systems
- / Atos: Energy I/O optimization for HPC with frugal and federated learning
- / NIST: cache optimization for NDN networks
- / DGA-AID project: DISPEED Intrusion Detection and Security/Performance/Energy tradeoff: a Study for Drone Swarms
- / IRT b<>com: service scheduling in heterogeneous systems (FPGA, GPU, GPP)



# Outline

1. Edge intelligence
2. Problem statement and general pattern
3. Case studies
  - a) K-means
  - b) Gaussian mixture model
  - c) Random forest
4. Time limited learning and energy optimization
5. Conclusions and perspectives

# 1. Edge intelligence - More data on the edge

## Global Data Creation is About to Explode

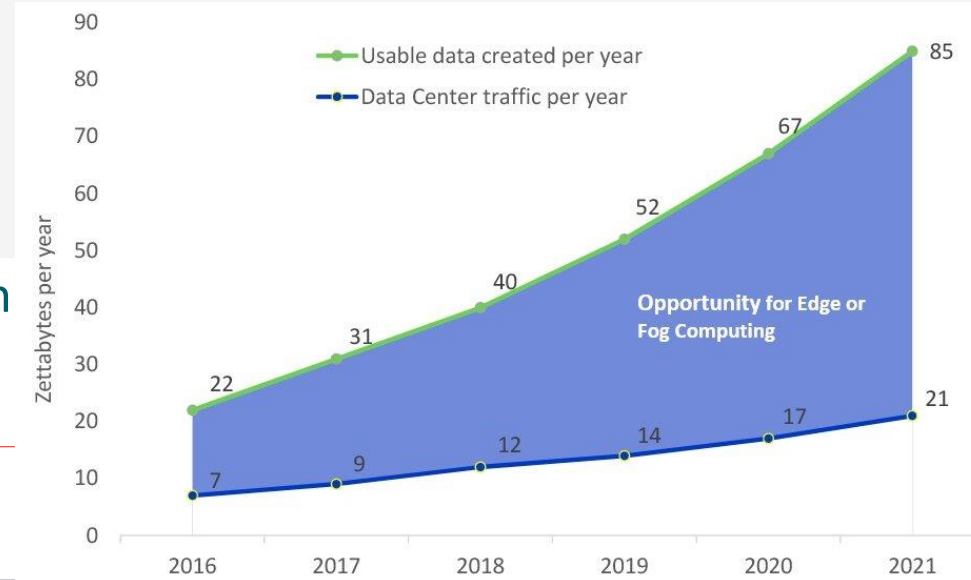
Actual and forecast amount of data created worldwide 2010-2035 (in zettabytes)



1 zettabyte is equal to 1 billion terabytes.



✓ IDC forecasts that 80 billion IoT devices will be connected by 2025



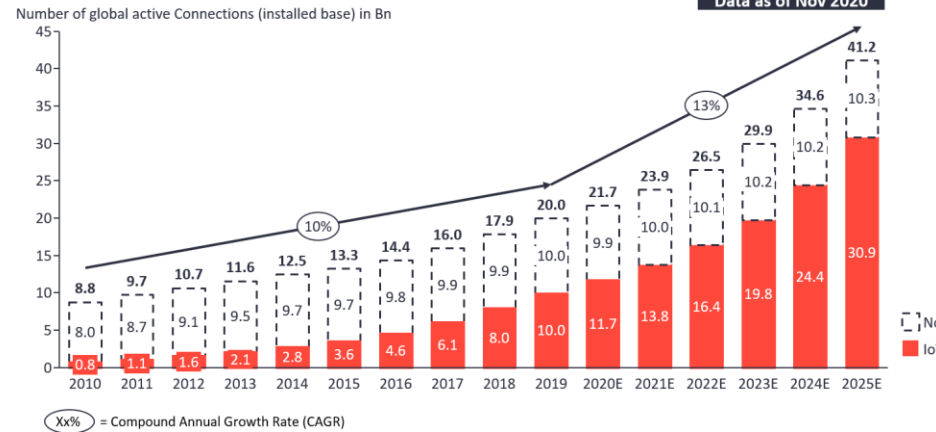
Source: Statista Digital Economy Compass 2019

✓ Data volume grows exponentially >500 Zettabytes by 2030!

IOT ANALYTICS

## Total number of device connections (incl. Non-IoT)

20.0Bn in 2019— expected to grow 13% to 41.2Bn in 2025

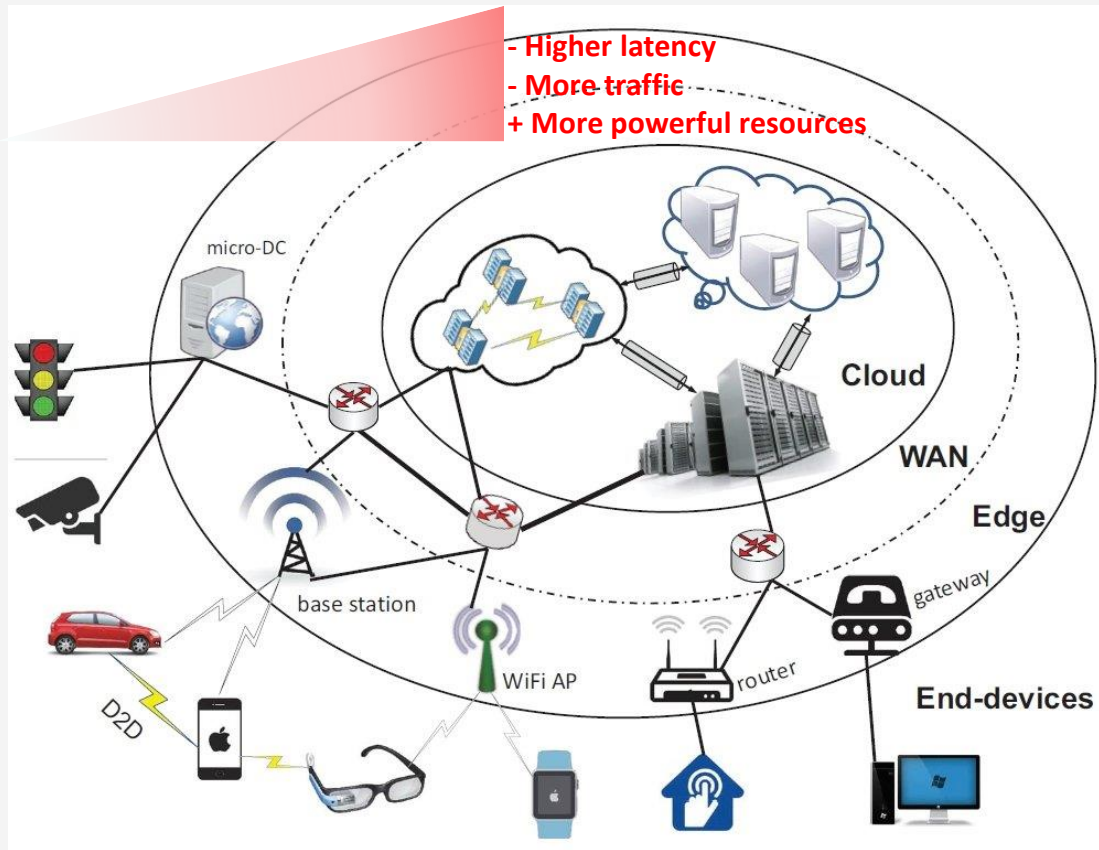


Note: Non-IoT includes all mobile phones, tablets, PCs, laptops, and fixed line phones. IoT includes all consumer and B2B devices connected – see IoT break-down for further details

Source(s): IoT Analytics - Cellular IoT & LPWA Connectivity Market Tracker 2010-25

✓ Data generated on devices > 4x data generated in data centers (2021 according to Cisco Global Cloud Index)

# 1. Edge intelligence - Edge computing



## / Edge computing definition

/ « ...paradigm that pushes computing tasks and services from the network core to the network edge » [1]

/ « ...capturing, storing, processing, analyzing data near its source » [2]

/ Several neighboring concepts: Cloudlets, Fog computing, Micro datacenters ...

/ **Move computation capabilities to the data (Vs data to the computing resources)**

/ Why ?

/ Reducing latency

/ Preserving privacy + security

/ Reducing network traffic

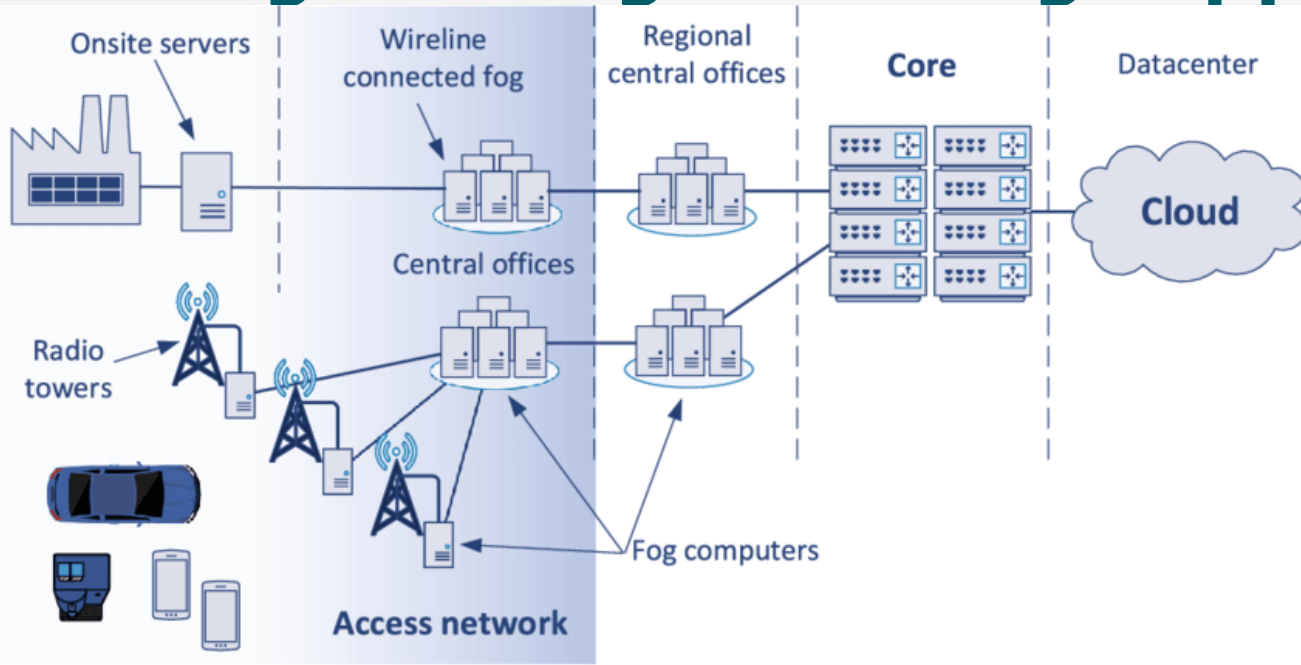
/ “cloud computing may lack the capacity to meet data processing needs” [3]

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1738-1762, Aug. 2019,.

[2] G. Boesch, **Edge Intelligence: Edge Computing and ML (2024 Guide)**, <https://viso.ai/edge-ai/edge-intelligence-deep-learning-with-edge-computing/>, 2023, accessed last on 21 sept 2024

[3] Thomas Bittman, Gartner analyst, [The Edge Will Eat the Cloud](#), 2017

# 1. Edge intelligence - Edge applications

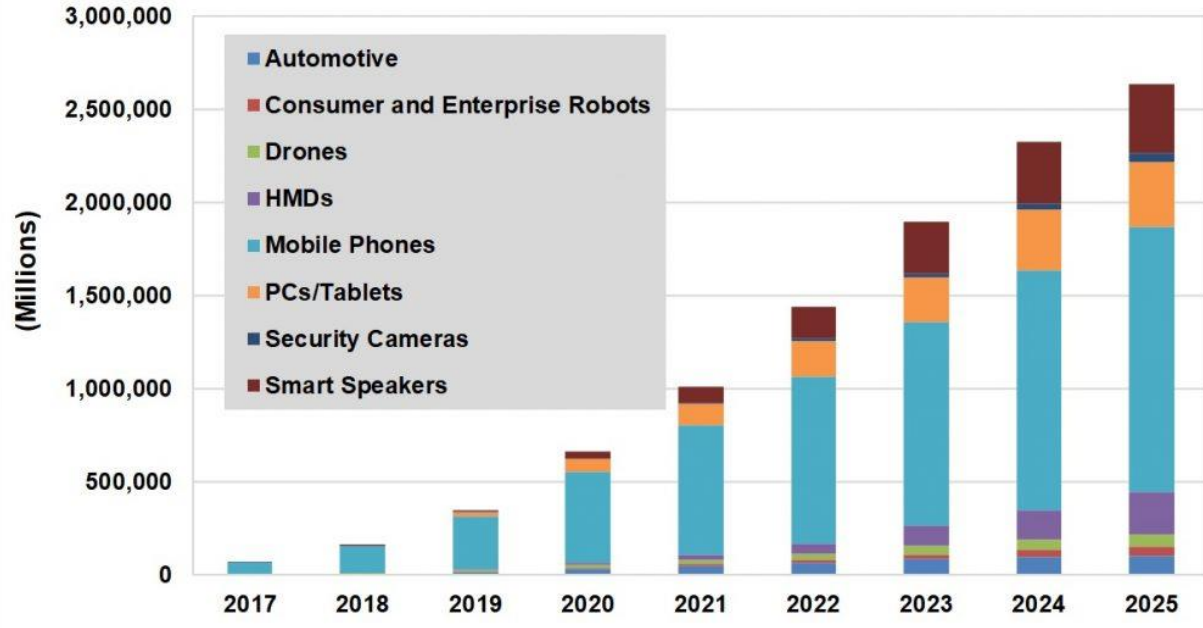


Latency < 5ms (automated driving, IoT, gaming)  
 Latency < 10ms (augmented reality, medical applications)  
 Latency < 30ms (video analytics)

Fadahunsi, Olamilekan & Maheswaran, Muthucumar. (2019). Locality sensitive request distribution for fog and cloud servers. Service Oriented Computing and Applications. 13.



AI Edge Device Shipments by Device Category, World Markets: 2017-2025



Source: Tractica

- / Smart (health, transportation, agriculture, home, environment), surveillance systems, ...
- / «... intelligent systems that can adapt to changing environments and learn from experience without being explicitly programmed » [1]
- / 80% of enterprise IoT include projects AI components in 2022 [2]

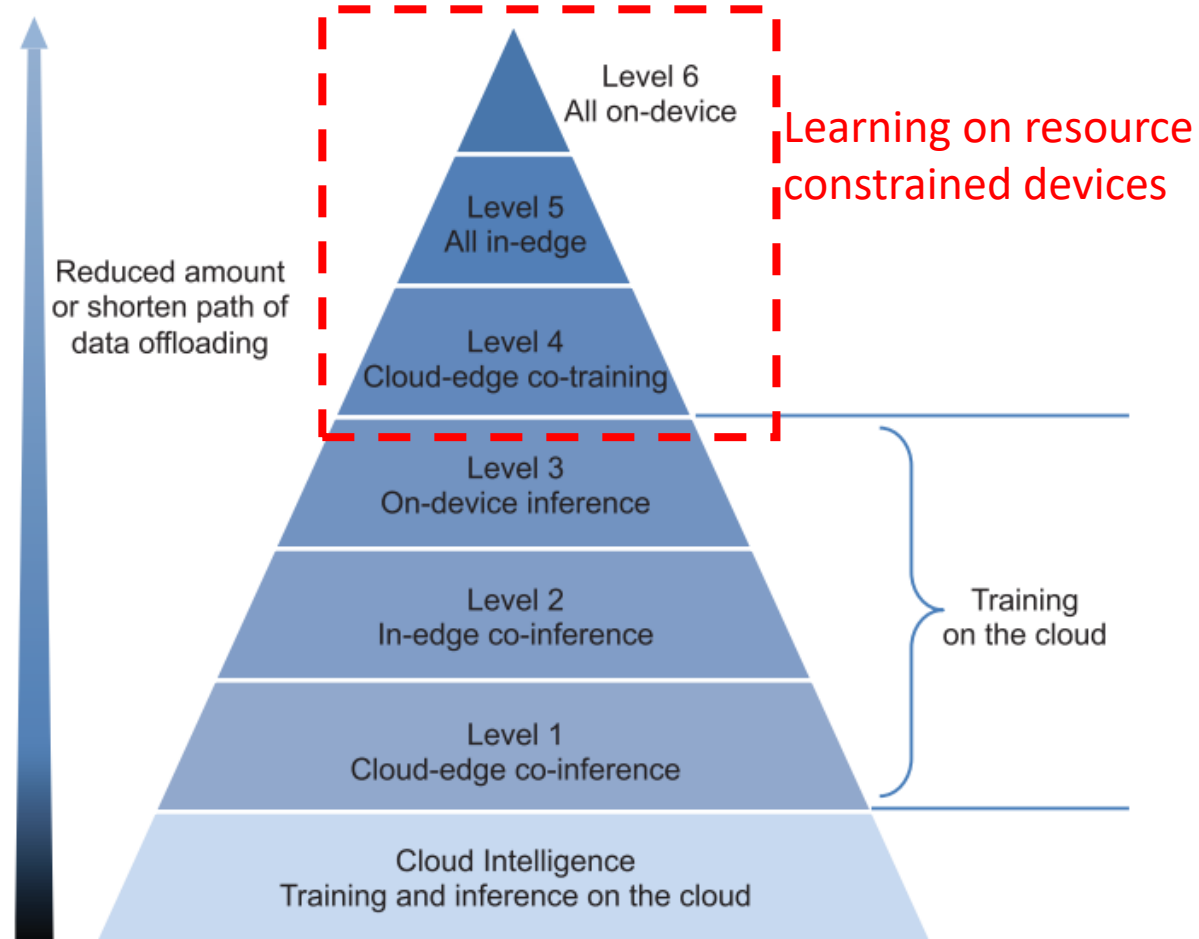
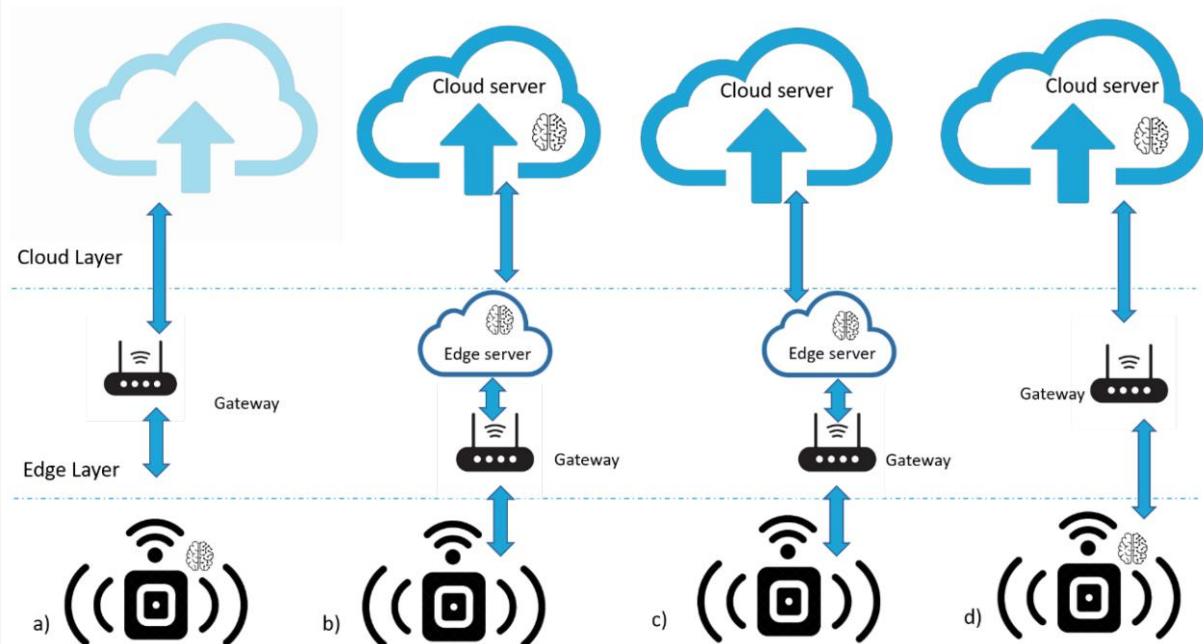
[1] A.C. Chen Liu, O. M. K. Law, J. Liao, J. Y. Chen, A. J. En Hsieh, C. h. Hsieh, Traffic safety systems edge ai computing, in IEEE/ACM Symposium on Edge Computing SEC, 2021.  
 [2] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," in *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738-1762, Aug. 2019,

# 1. Edge intelligence (EI or Edge AI) = Edge computing + AI

/ Edge intelligence:

- / «providing reliable and real-time intelligent services on the network's edge» [1],
- / « ... derives from the concept of combining edge computing and ML » [2]

/ Both learning and inference can be done at different levels



[1] Mohammad Yahya Akhlaqi, Zurina Binti Mohd Hanapi, Task offloading paradigm in mobile edge computing-current issues, adopted approaches, and future directions, Journal of Network and Computer Applications, Volume 212, 2023, 103568, ISSN 1084-8045,  
 [2] Asif Mahmud Raivi, Sangman Moh, A comprehensive survey on data aggregation techniques in UAV-enabled Internet of things, Computer Science Review, Volume 50, 2023, 100599, ISSN 1574-0137,

Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1738-1762, Aug. 2019

# 1- Edge intelligence- devices and platforms

/ Edge devices:

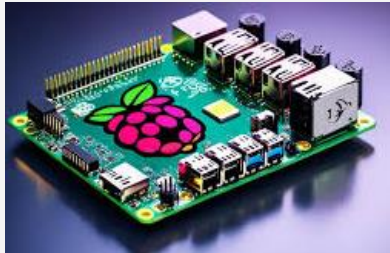
/ “resource-constrained device located at the edge of the network” [1]

/ “...ranging from credit-card-sized computer to a micro data-center server ... physical proximity ... most crucial characteristic” [2]

/ Limited resources = {processing, memory, network, storage}

[1] R. Singh, S. S. Gill, Edge AI: A survey, Internet of Things and Cyber-Physical Systems, Volume 3, 2023, Pages 71-92, ISSN 2667-3452

[2] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," in *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738-1762, Aug. 2019,



Raspberry Pi



Google Coral dev

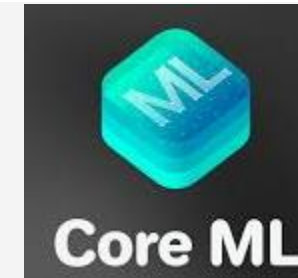


BeagleBone AI



Arduino Nano 33 BLE Sense

Jouini, O.; Sethom, K.; Namoun, A.; Aljohani, N.; Alanazi, M.H.; Alanazi, M.N. A Survey of Machine Learning in Edge Computing: Techniques, Frameworks, Applications, Issues, and Research Directions. *Technologies* 2024, 12, 81  
Jagreet Kaur Gill, Edge AI Applications and How does it work ?  
06 September 2024, <https://www.xenonstack.com/blog/edge-ai-applications>



NVIDIA Jetson



## 2. Problem statement and general pattern

When applying machine learning on the edge ...



We need:

- ✓ ↗ Accuracy
- ✓ ↘ Latency
- ✓ ↘ network traffic

Edge intelligent devices

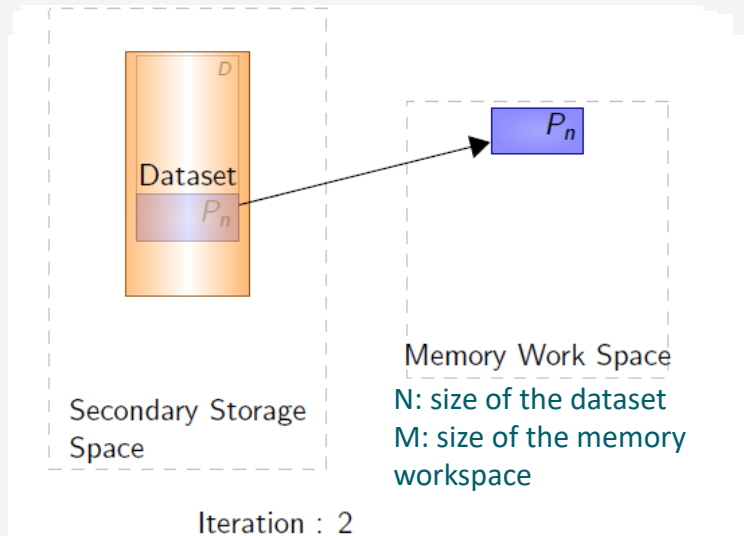
Device constraints:

- ↘ Performance
- ↘ Mem/Storage
- ↘ Energy

Source: <https://www.wordslingersok.com/2017/09/caught-between-a-rock-and-a-hard-place/>

# 2. Problem statement and general pattern -2-

- / Several ML algorithms → go through all (or most) of the dataset during the learning process
- / When **the dataset size > memory workspace** → I/O swapping issues ↗↗↗



**Each dataset loading « read all... » = a lot of swapping operations (depending on the dataset to memory proportion).**

**Very poor temporal locality**

For → some initial state ... until something converges

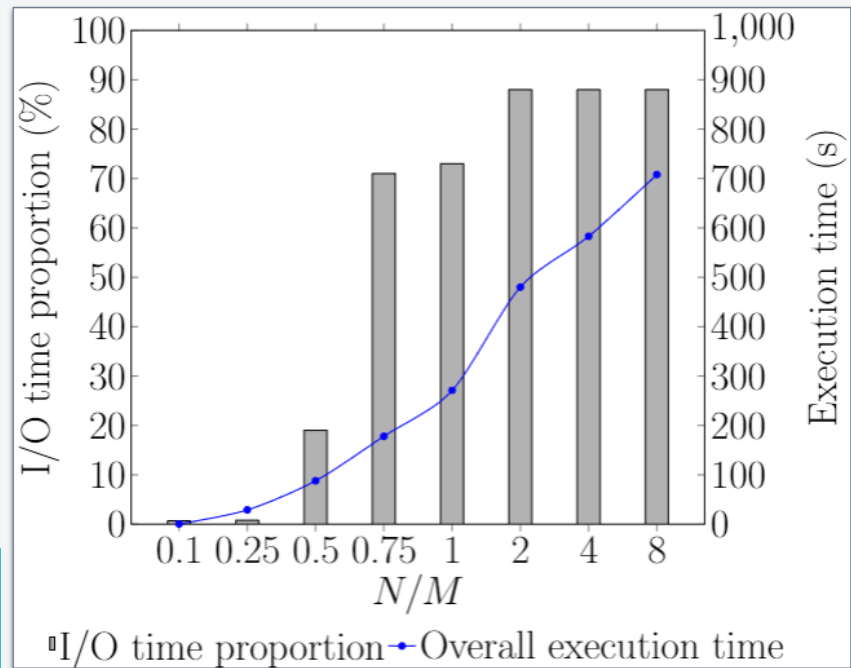
**read all (or at least a large part of) the dataset**

**perform some (more or less complex) processing**

**update the model**

**tightly coupled instructions**

Example with Random forests on a BB board

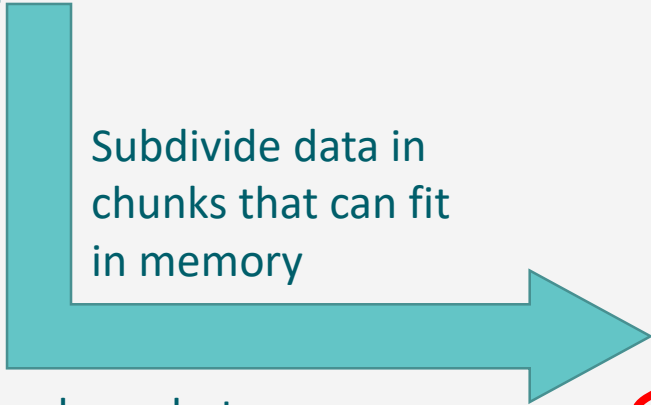


# 2. Problem statement and general pattern

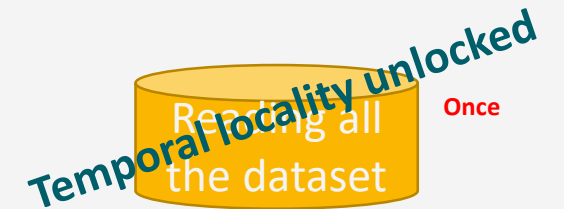
## Proposed solution pattern – Divide and conquer



For → some initial state ... until something converges  
read the whole (or at least a large part of the) dataset  
Performing some processing  
update the model



For each chunk of data  
read one chunk of dataset



For → some initial state ... until something converges  
Performing processing on data of the chunk  
update the the model



Key method that depends on the case study

Relax the dependency between full data reading and processing (no need to read ALL the data to compute)

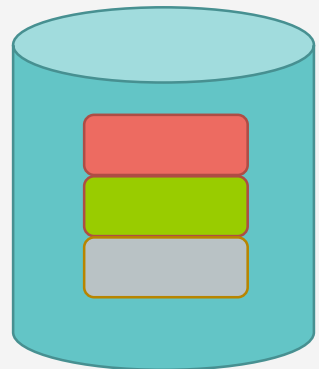
## 2. Problem statement and general pattern

### Basic solution pattern

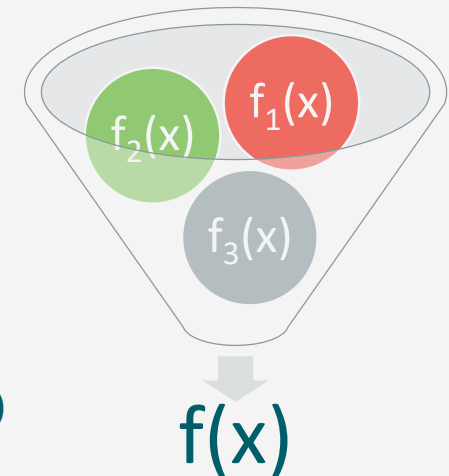
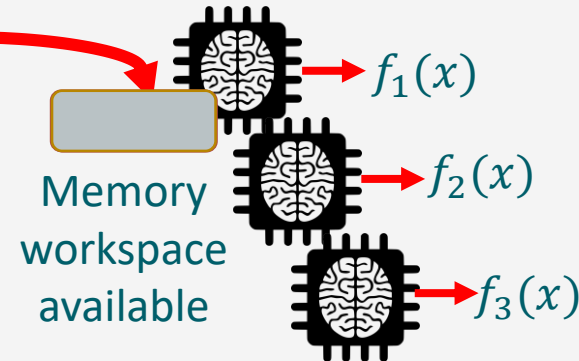
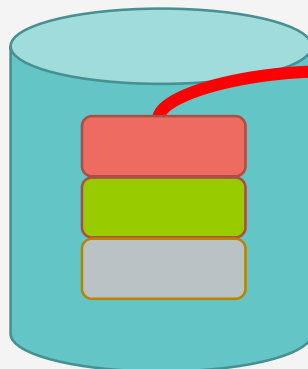
Subdivide  
the dataset

Process  
each chunk  
separatly

Aggregate  
the models



Memory  
workspace  
available



# Case study 1: K-means

2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)

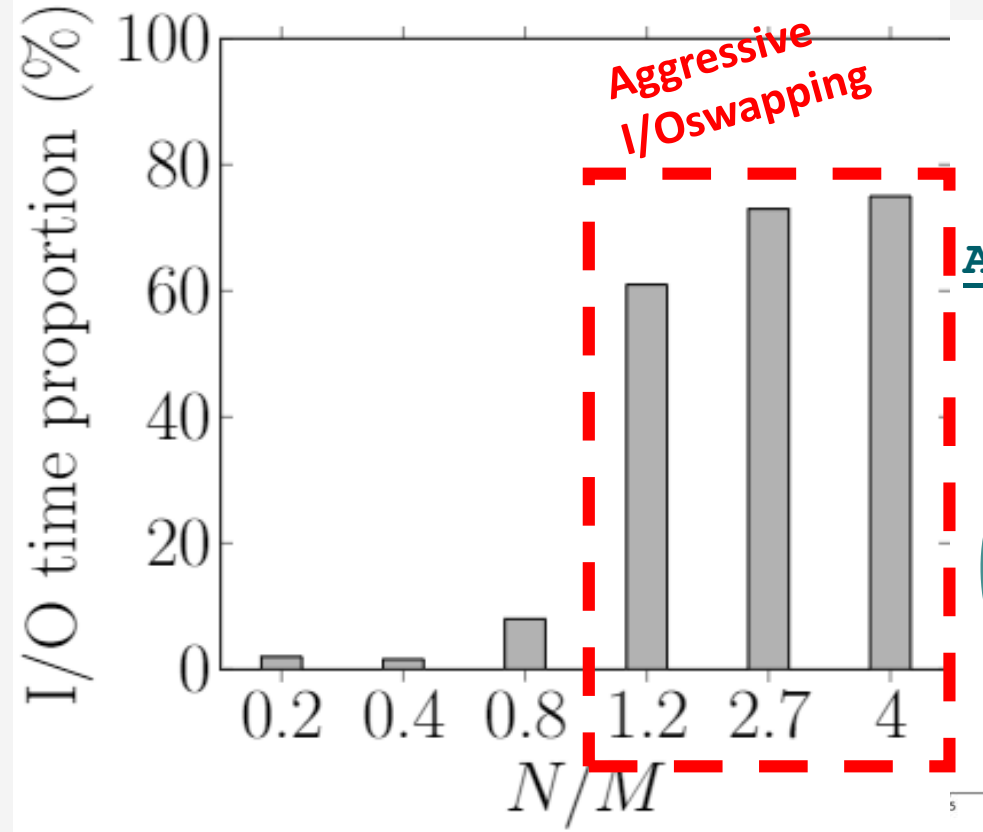
## *K*-MLIO: Enabling *K*-Means for Large Data-sets and Memory Constrained Embedded Systems

Camélia Slimani  
*Univ Brest*  
*Lab-STICC, CNRS, UMR 6285*  
F-29200 Brest, France  
Camelia.Slimani@univ-brest.fr

Stéphane Rubini  
*Univ Brest*  
*Lab-STICC, CNRS, UMR 6285*  
F-29200 Brest, France  
rubini@univ-brest.fr

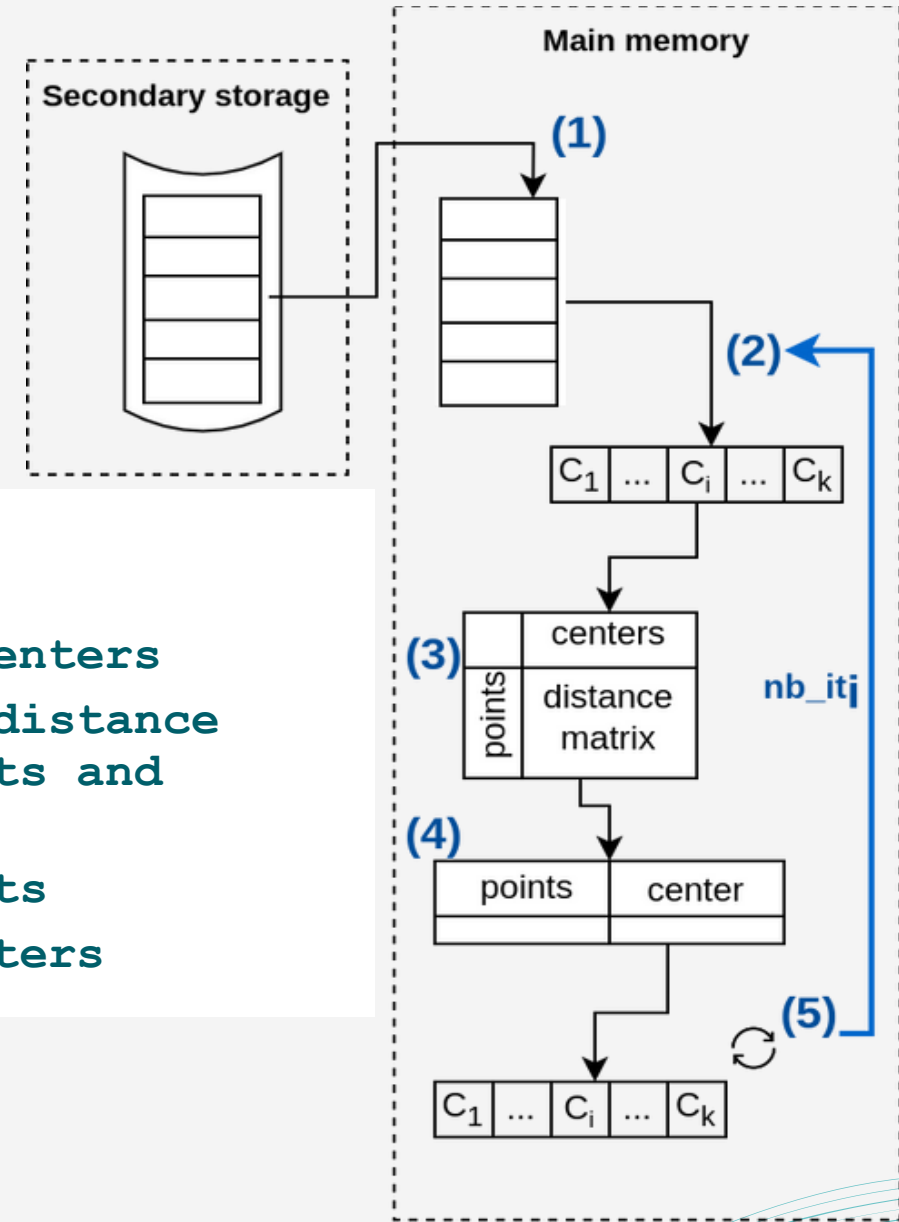
Jalil Boukhobza  
*Univ Brest*  
*Lab-STICC, CNRS, UMR 6285*  
F-29200 Brest, France  
boukhobza@univ-brest.fr

# K-means algorithm



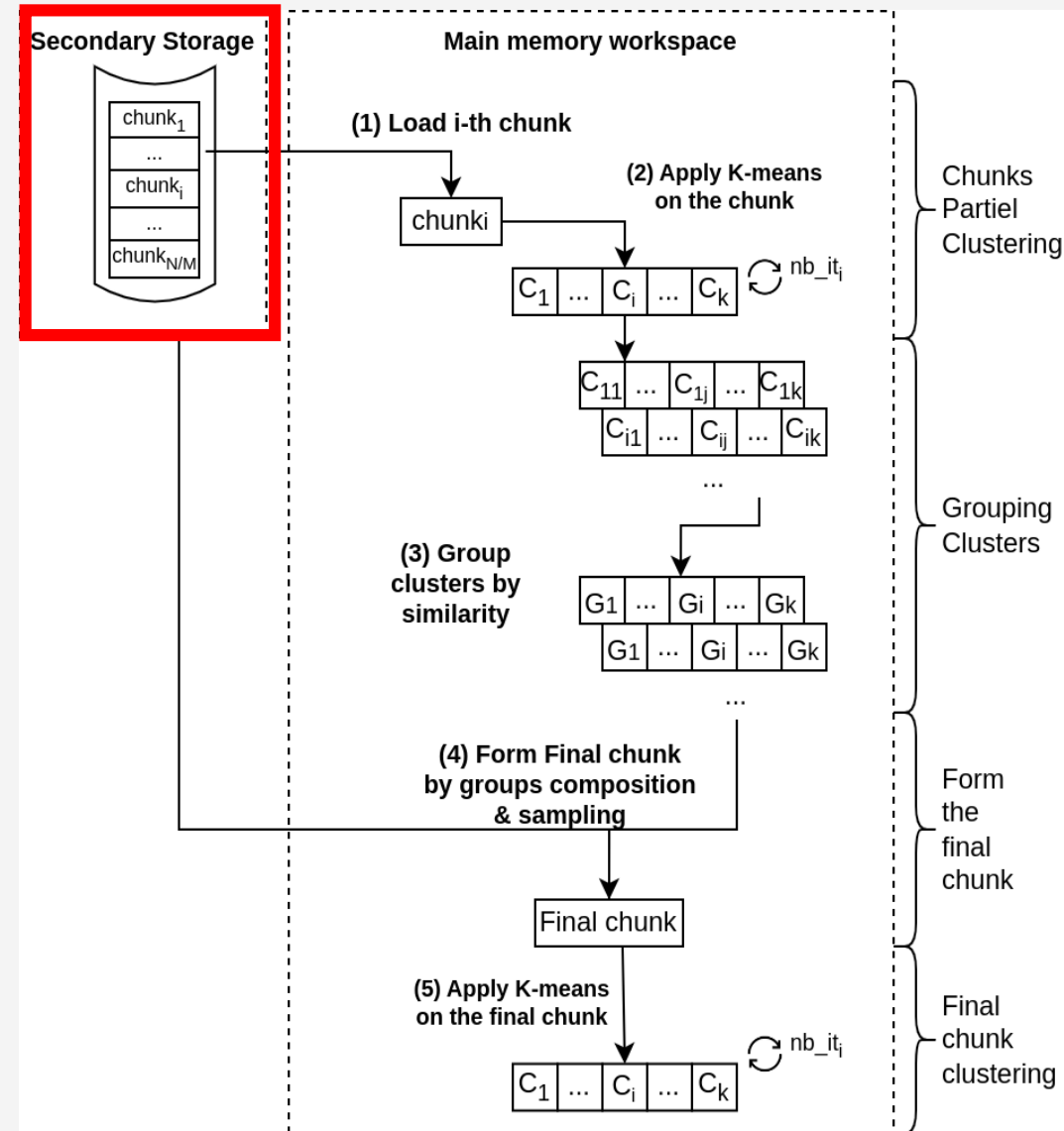
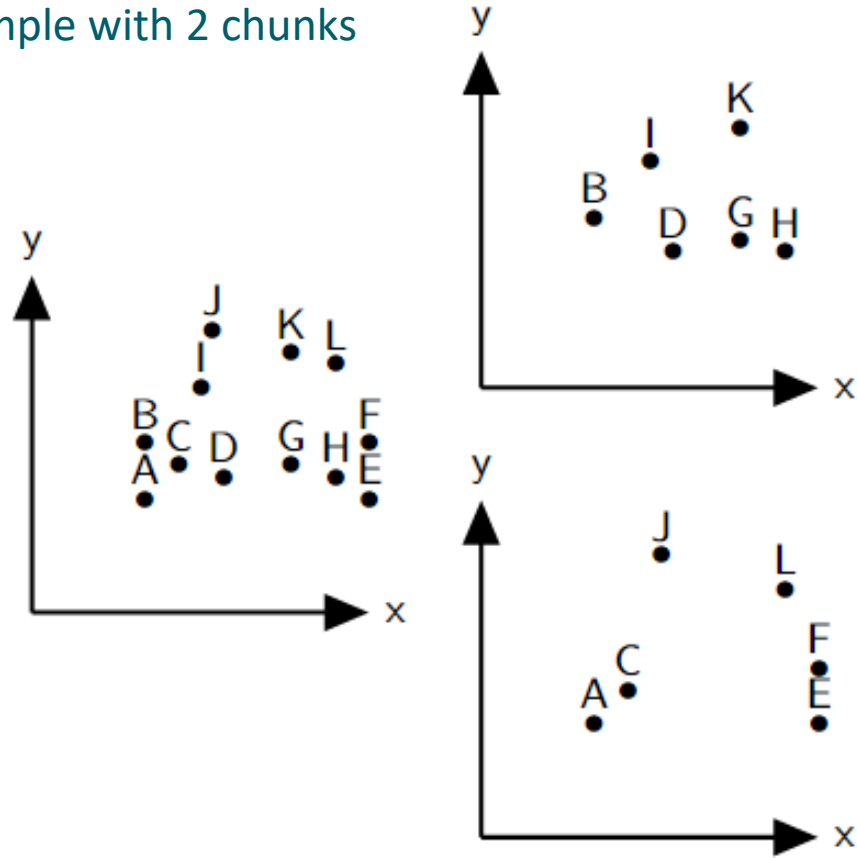
## Algorithm

- 1) Loading the data
- 2) Initializing the centers
- 3) Calculating the distance matrix between points and centers
- 4) Reassigning points
- 5) Updating the centers



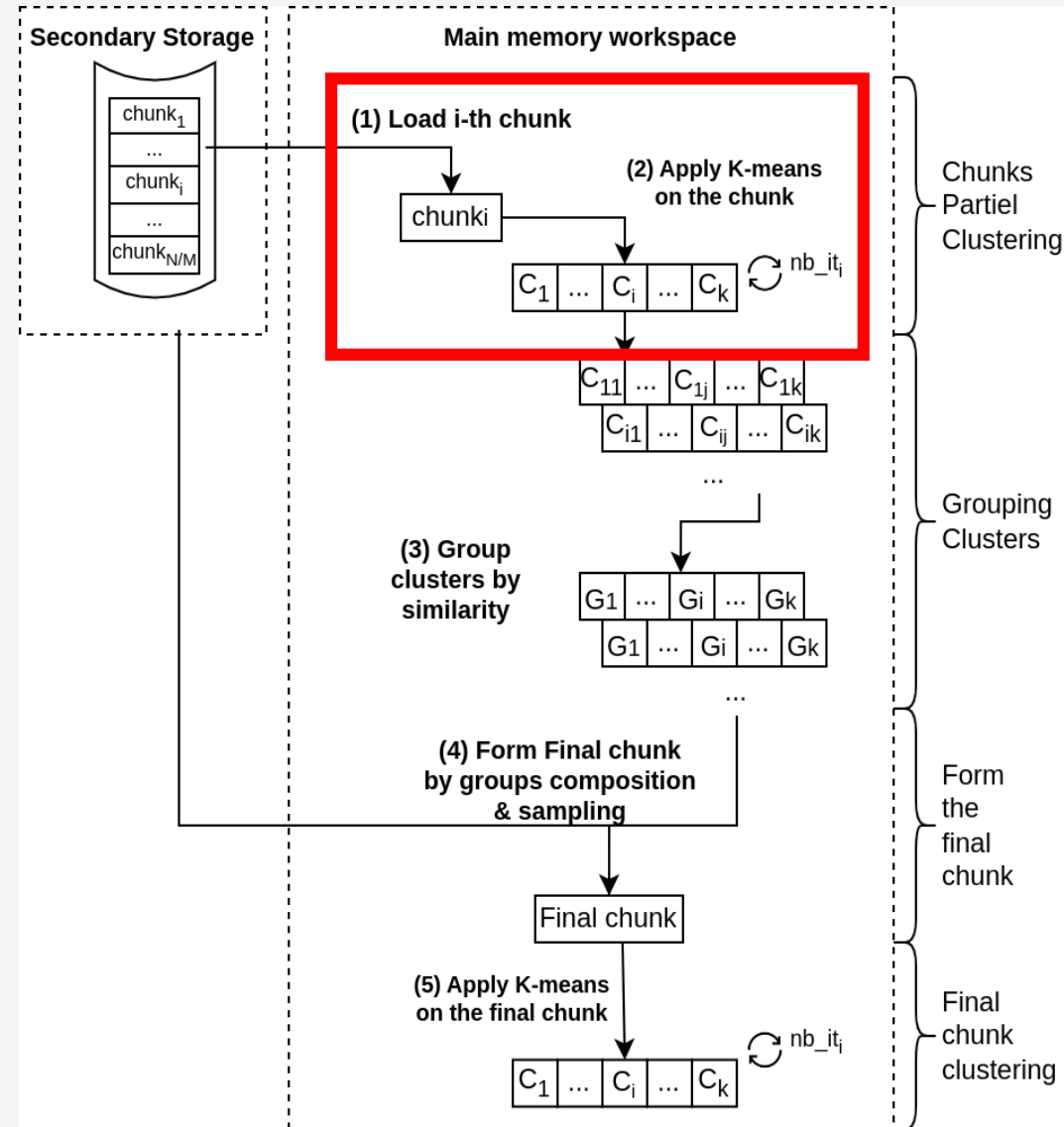
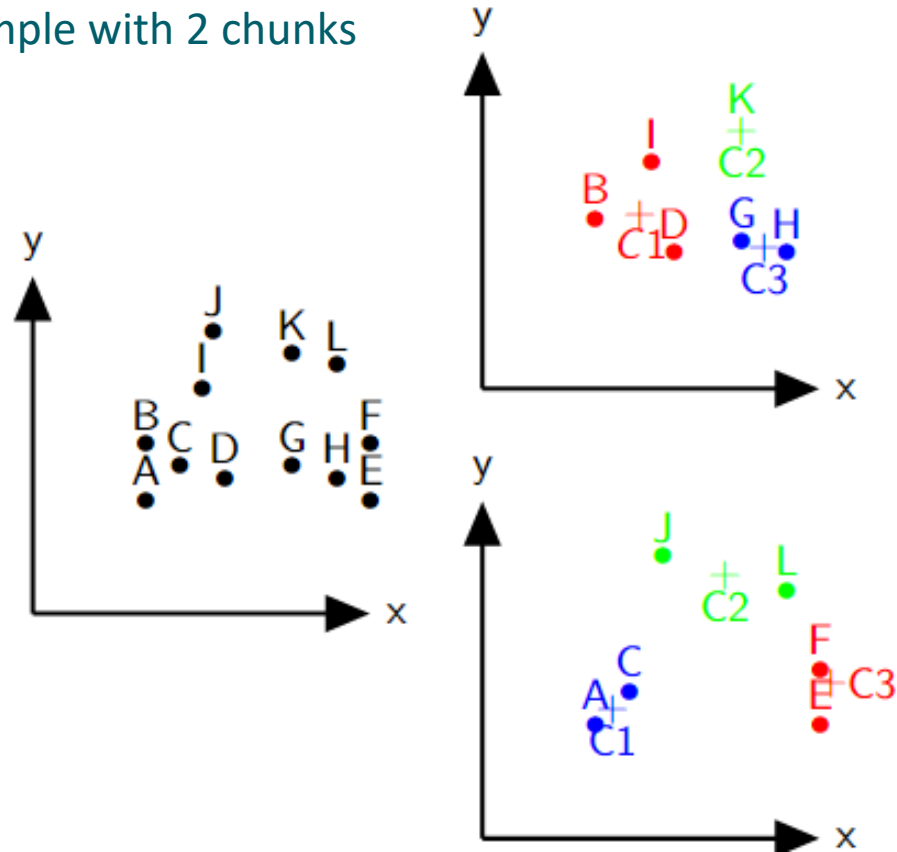
# K-MLIO ( K-means with low I/O Overhead )

Example with 2 chunks



# K-MLIO ( K-means with low I/O Overhead )

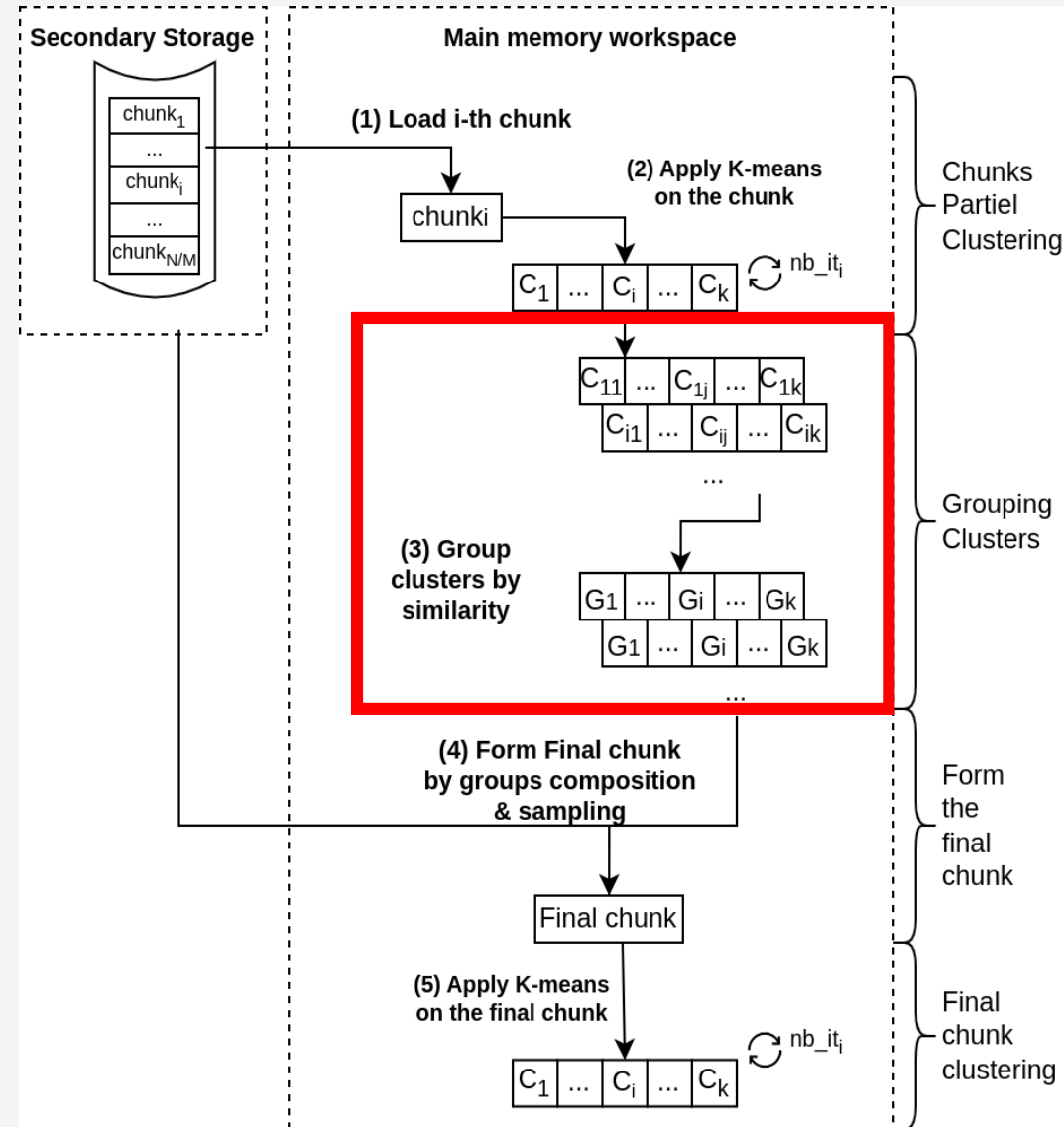
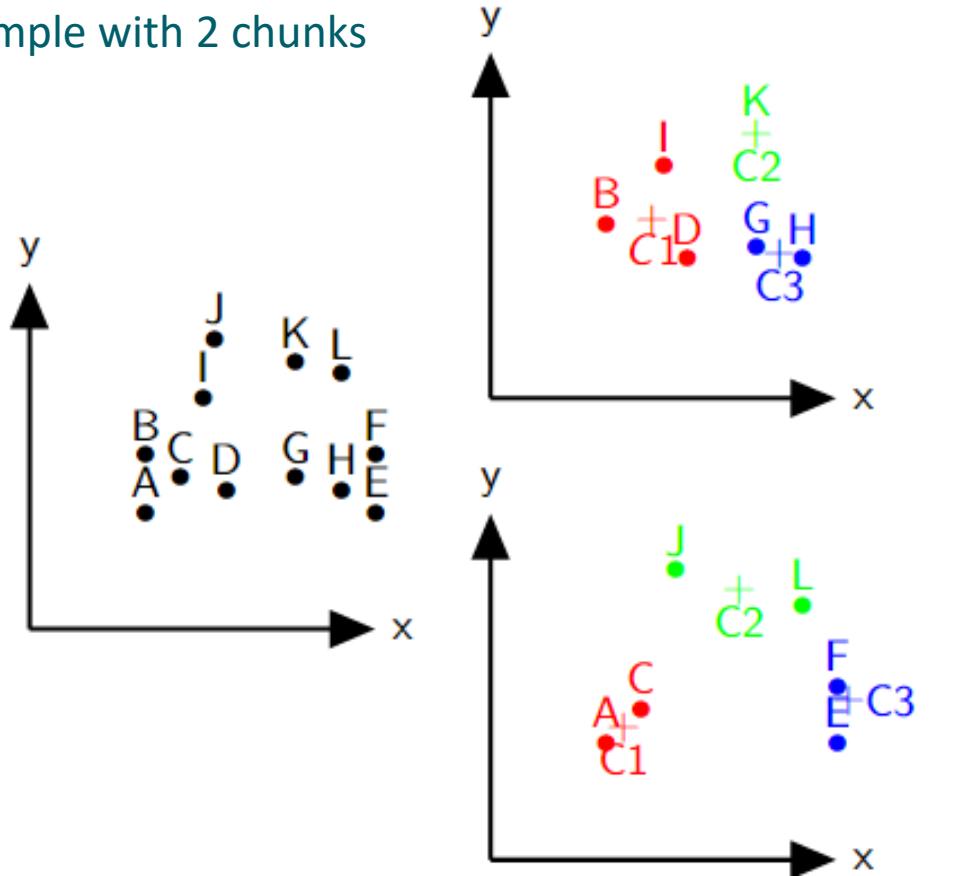
Example with 2 chunks





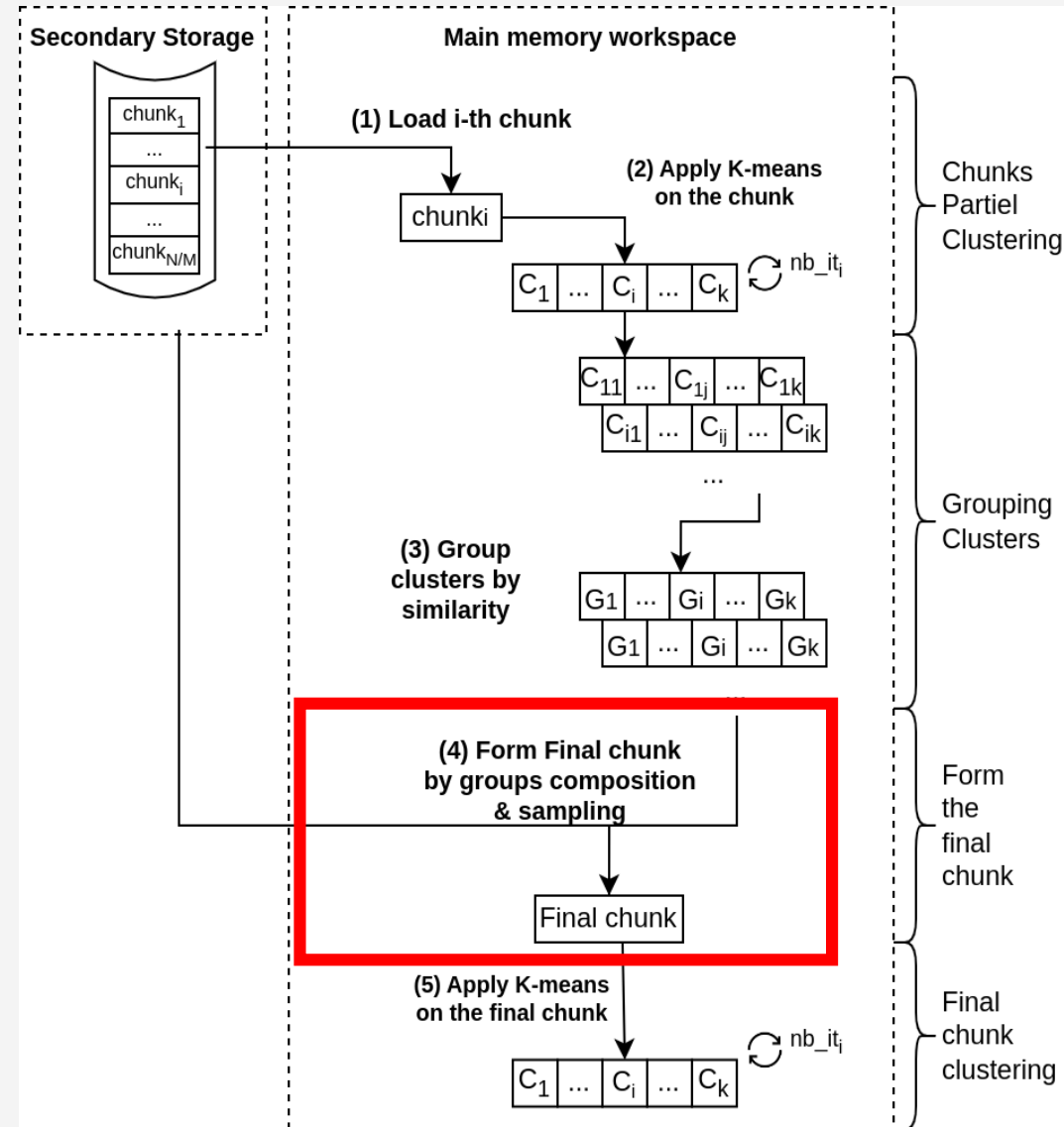
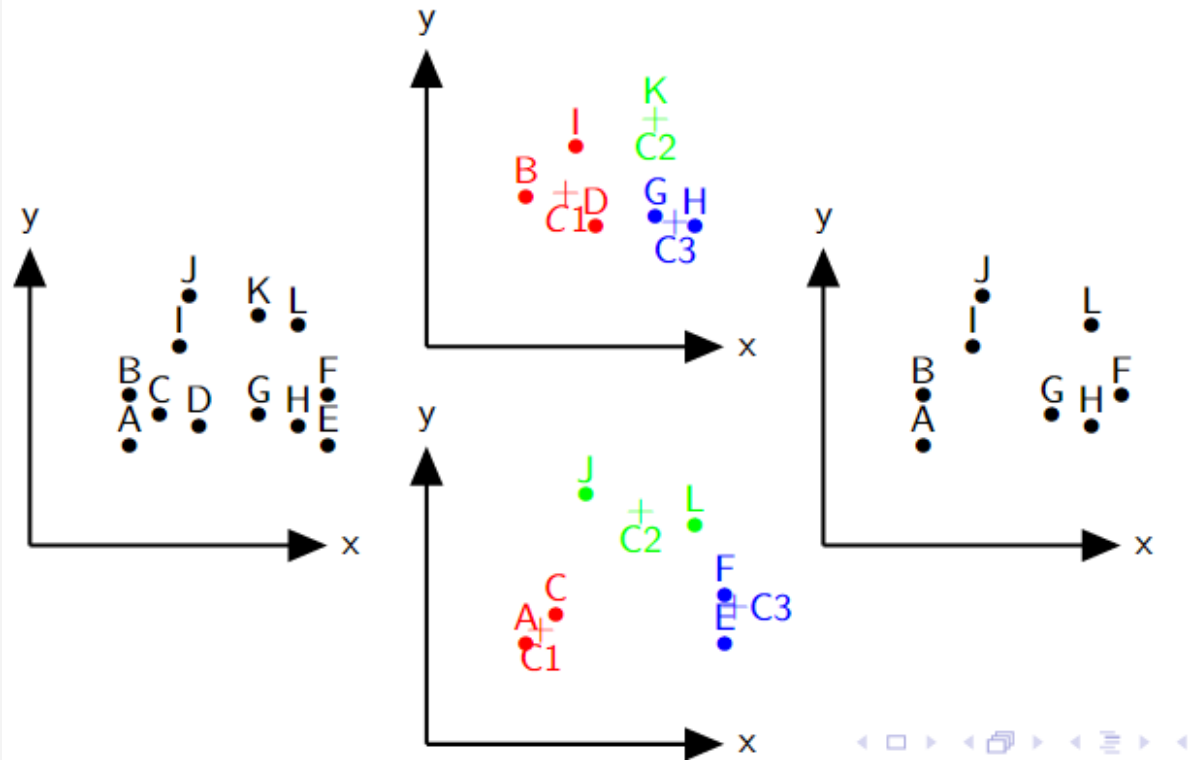
# K-MLIO ( K-means with low I/O Overhead )

Example with 2 chunks



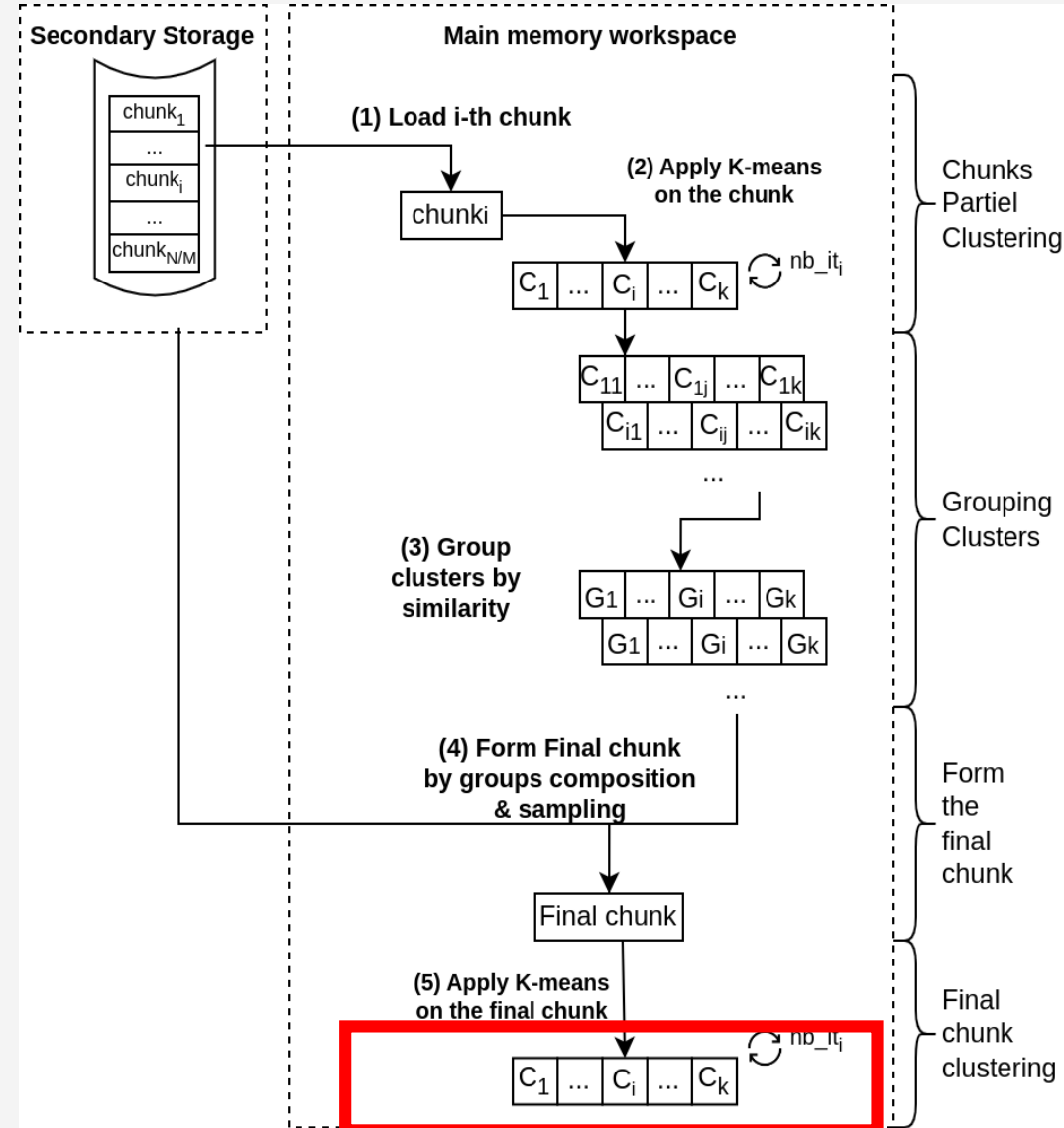
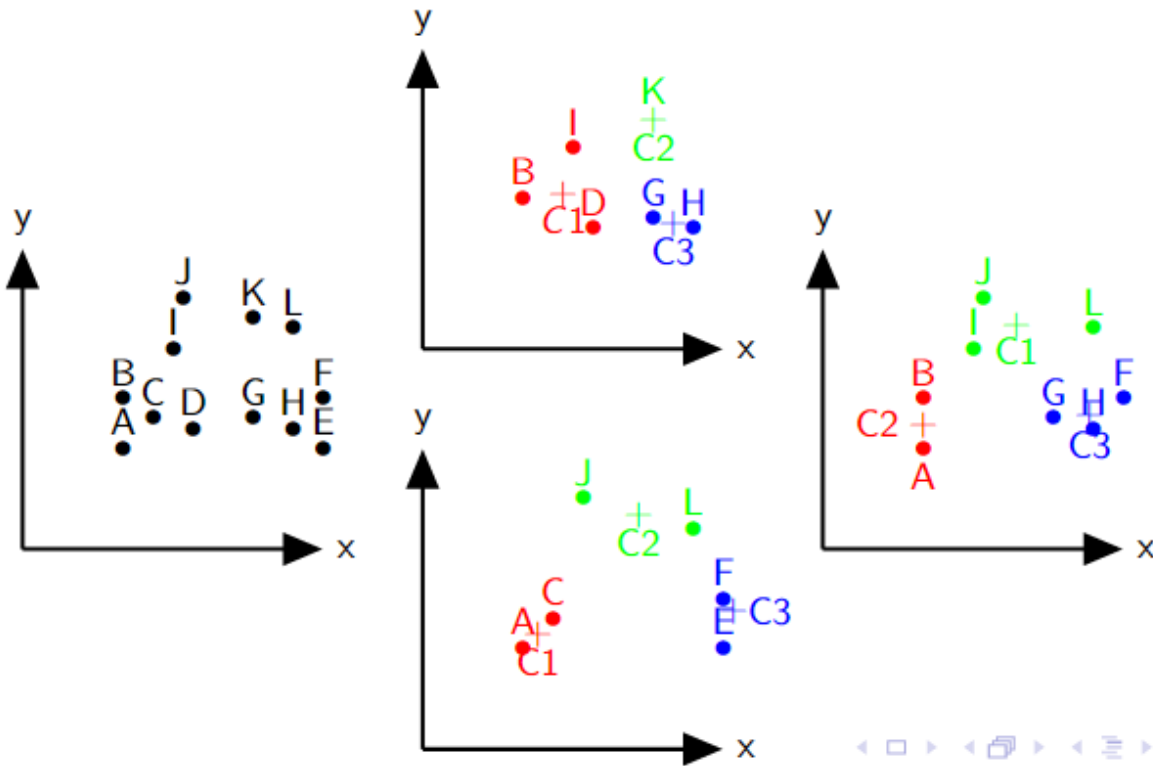
# K-MLIO ( K-means with low I/O Overhead )

Example with 2 chunks



# K-MLIO ( K-means with low I/O Overhead )

Example with 2 chunks



# Some results

- / Beaglebone black board (512 MB of RAM and 2GB of swap area)
- / ClusterGeneration R package



N/M	I/O time (%)		K-MLIO I/O time reduction
	K-means	K-MLIO	
1.5	67	3.5	94%
2.7	73	3.5	95%
4	75	3	96%

Table: Réduction du temps d'E/S de K-MLIO

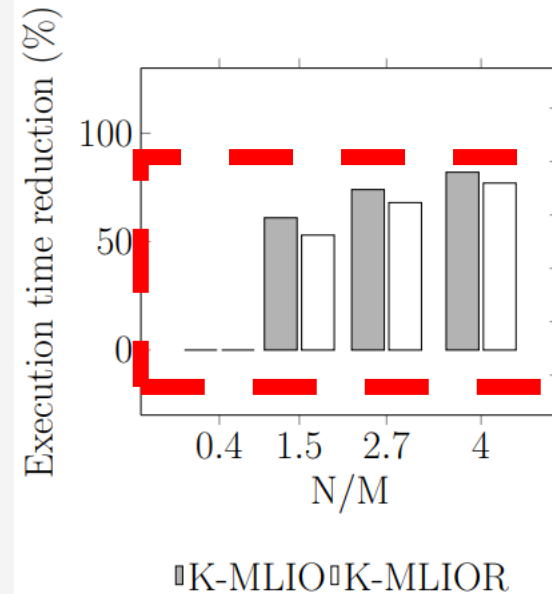
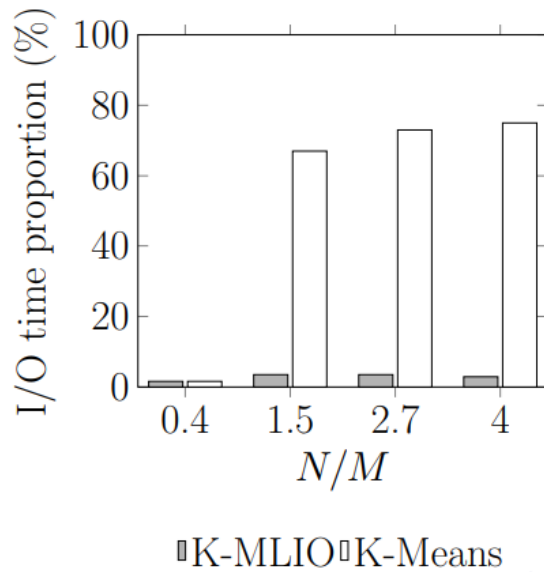


Figure: Medium Separation

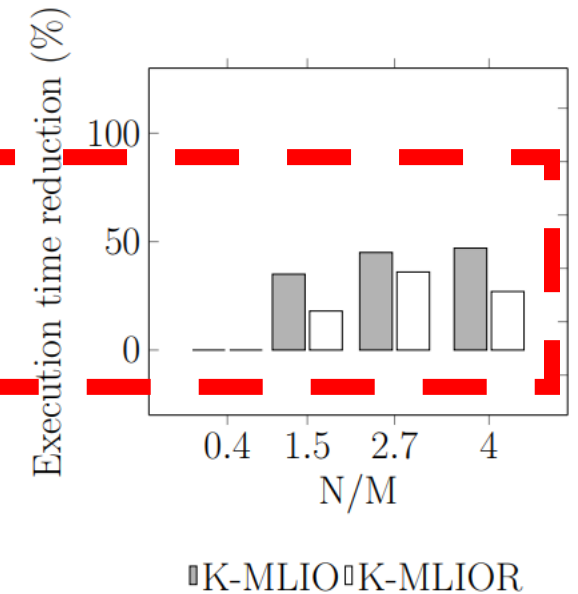


Figure: High separation

## Case study 2: Gaussian Mixture Model

### PIGMMaLION: a Partial Incremental Gaussian Mixture Model with a Low I/O Design

Meriem Bouzouad<sup>1,2</sup>, Yasmine Benhamadi<sup>1,2</sup>, Camélia Slimani<sup>1</sup>, Jalil Boukhobza<sup>1</sup>

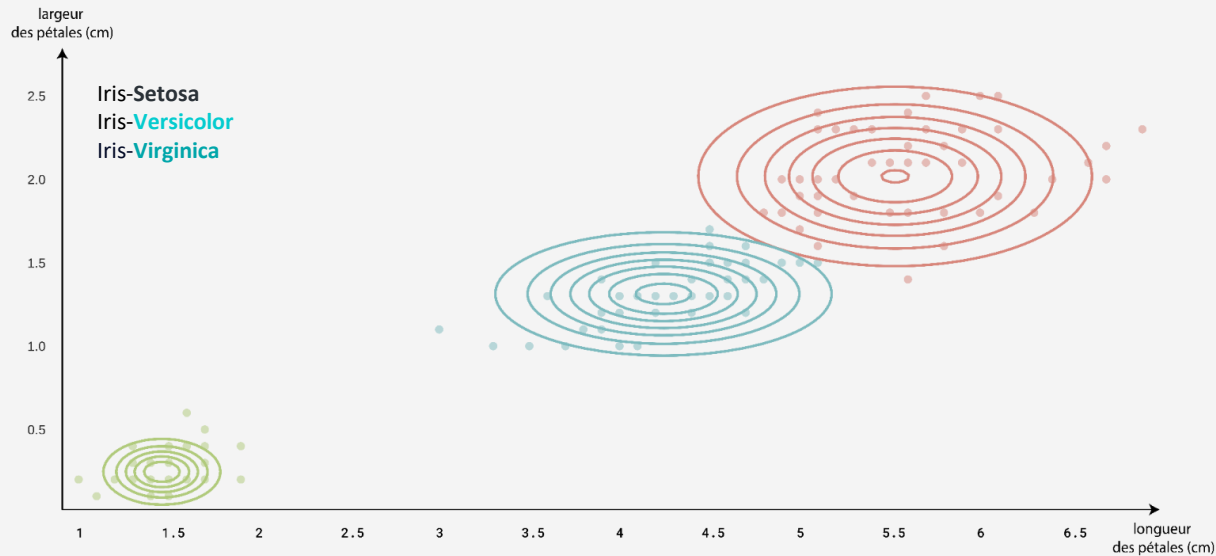
<sup>1</sup> ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285, Brest, France

<sup>2</sup> École Nationale Supérieure D'Informatique (ESI), Algiers, Algeria

{im\_bouzouad, iy\_benhamadi}@esi.dz, {camelia.slimani, jalil.boukhobza}@ensta-bretagne.fr

# Context : Gaussian Mixture Models (GMM)

/ Probabilistic Machine Learning Models used for **clustering** data and **density estimation**.



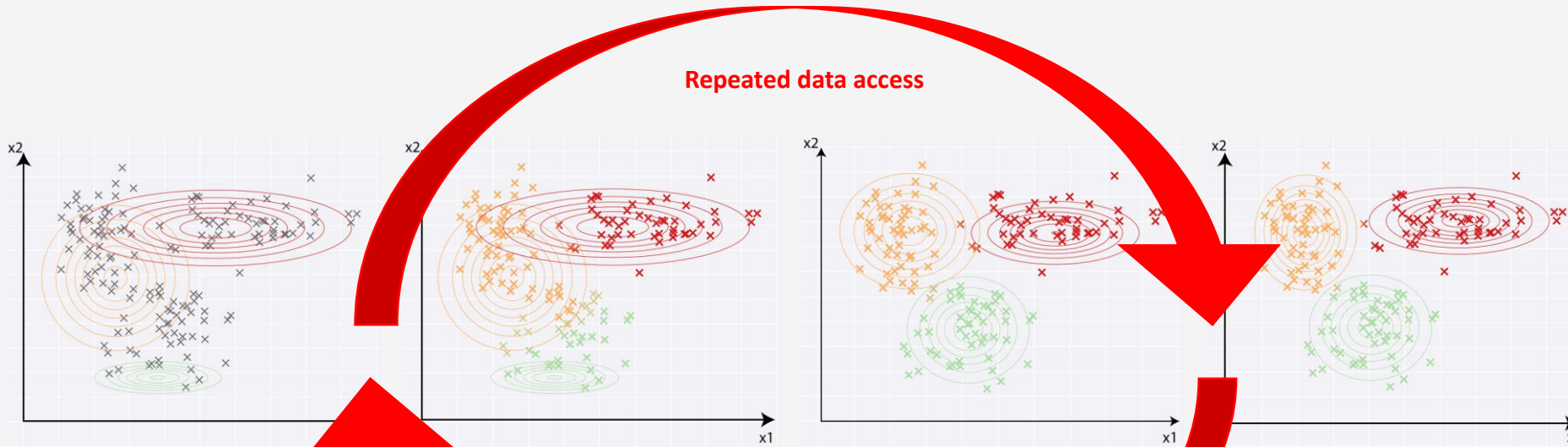
Each cluster is represented by :

- A **Gaussian density** (mean and covariance matrix)
- A **weight** : probability to be affected to this cluster

$$p(x_i | \theta_{iris}) = w_1 \cdot g(x_i | \mu_1, |\Sigma_1) + w_2 \cdot g(x_i | \mu_2, |\Sigma_2) + w_3 \cdot g(x_i | \mu_3, |\Sigma_3)$$

Reynolds, Douglas A. "Gaussian mixture models." *Encyclopedia of biometrics* 741.659-663 (2009).

# Expectation Maximization (EM) Algorithm



(1) Parameters Initialization

(2) E-Step : Expectation Calculation

(3) M-Step : Maximization

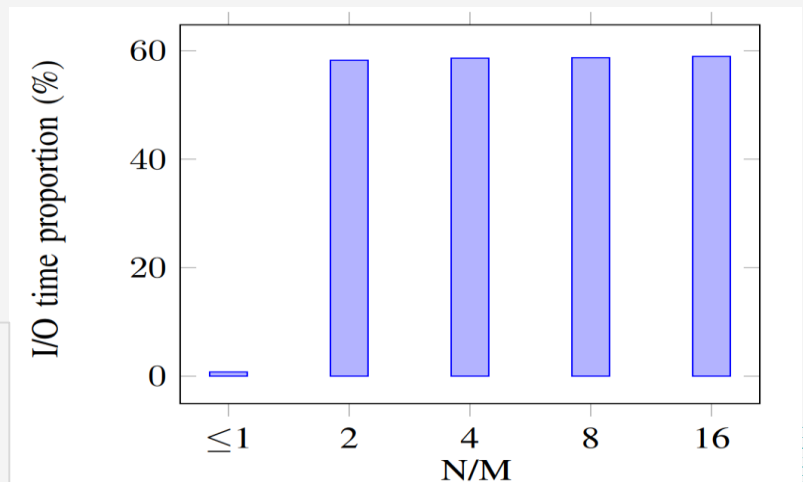
Final GMM

Calculate assignments to current clusters

Update cluster means, covariances and weights

Loop until convergence

**N** : Number of observations in the training set  
**M** : Number of observations that can be contained in main memory



# PIGMMaLION : Partial Incremental Gaussian Mixture Model with a Low I/O Design.

$$I/O_{cost\_EM} = \lceil N/M \rceil \cdot \lfloor M/B \rfloor \cdot I_{conv}$$

Depends on

Dataset size

Number of iterations

## Restrict training on a dynamic uniform subsample

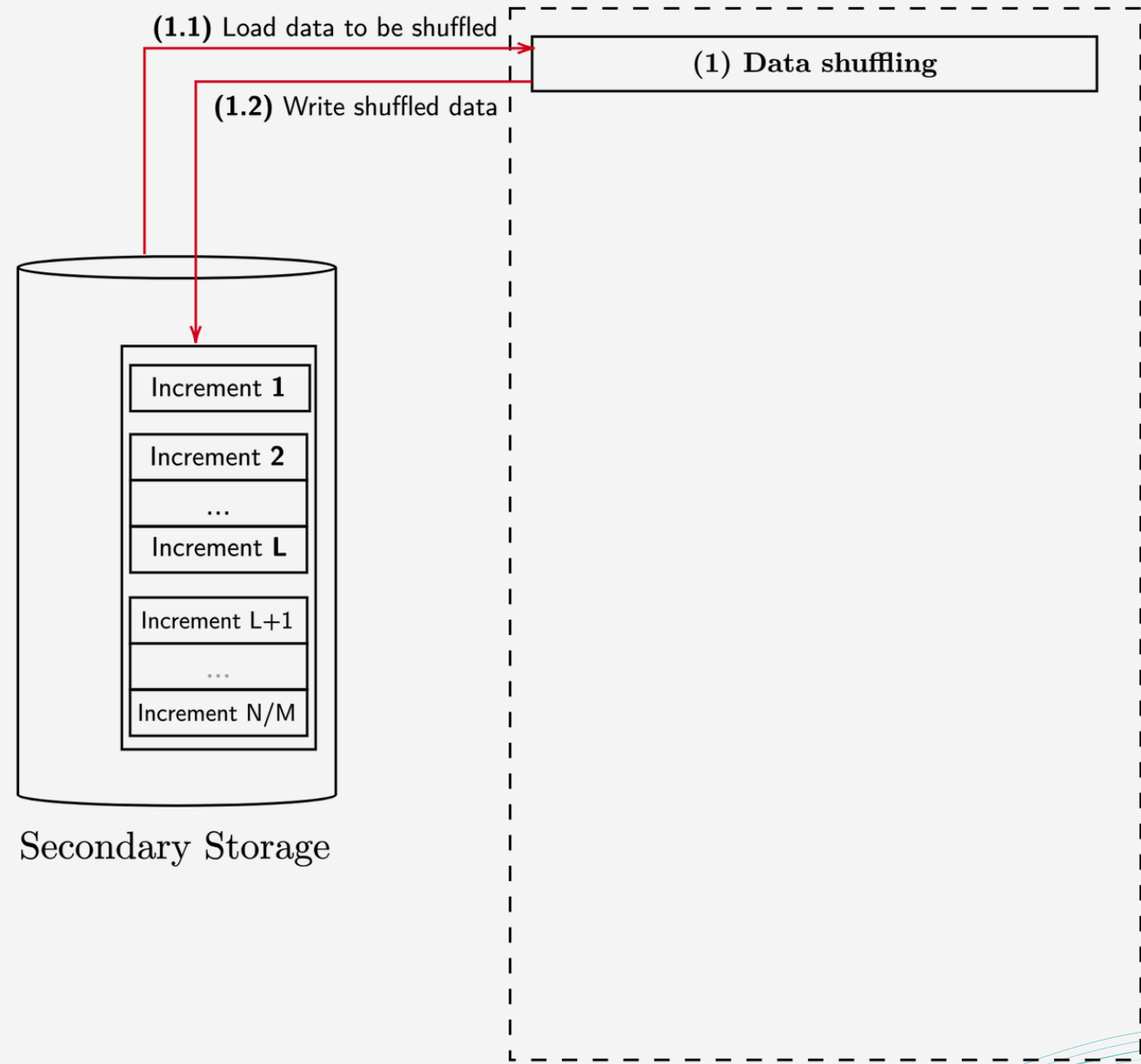
- **Infer** the sufficient subsample size to obtain a clustering quality **comparable** to the original algorithm

## Divide-and-Conquer Approach

- **Incrementally** form the GMM
- Each dataset observation needs to be loaded into main memory **only once**
- I/O cost is decorrelated from  $I_{conv}$



# PIGMMaLION design



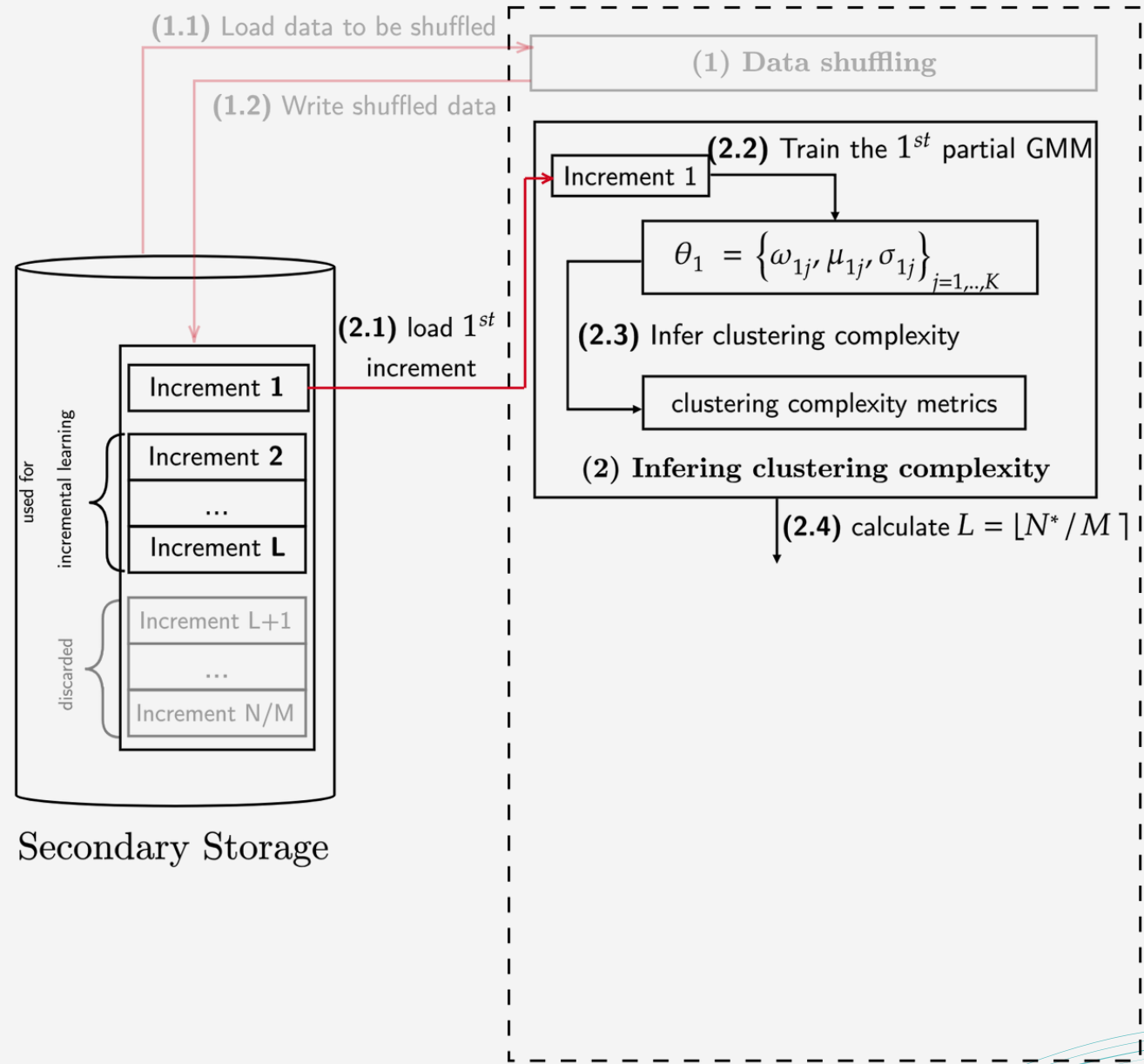
# PIGMMaLION design

## / Dynamic subsampling

- / A partial GMM is trained
- / The complexity of the clustering task is inferred based on:
  - / The uncertainty factor  $\alpha$
  - / The balance factor  $\beta$

## / $N^*$ is calculated based on clustering complexity:

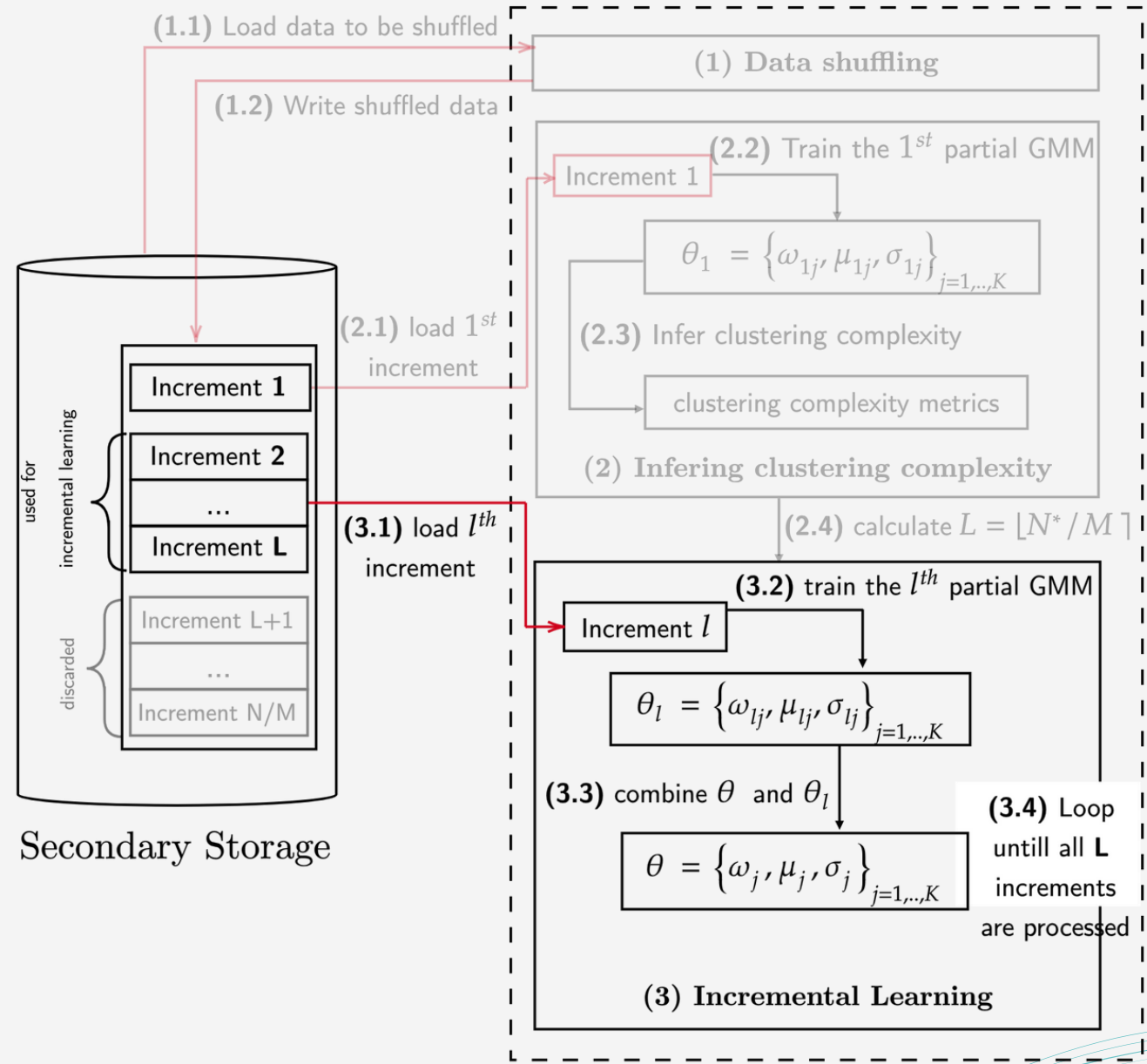
$$N^* = M + \frac{\alpha(N - M) + (1 - \beta)(N - M)}{2}$$



# PIGMMaLION design

## / Incremental learning

- / Load data one increment at a time
- / Train a partial GMMs on each loaded increment
- / Merge the partial GMMs incrementally



# Some results

Partial GMM

PIGMMaLIOn with:

$$N^* = M \text{ 1 chunk}$$

PIGMMaLIOn

PIGMMaLIOn with:

$$N^* = M + \frac{\alpha(N - M) + (1 - \beta)(N - M)}{2}$$

Incremental GMM

PIGMMaLIOn with:

$$N^* = N \text{ All the dataset}$$

Execution time  
Reduction

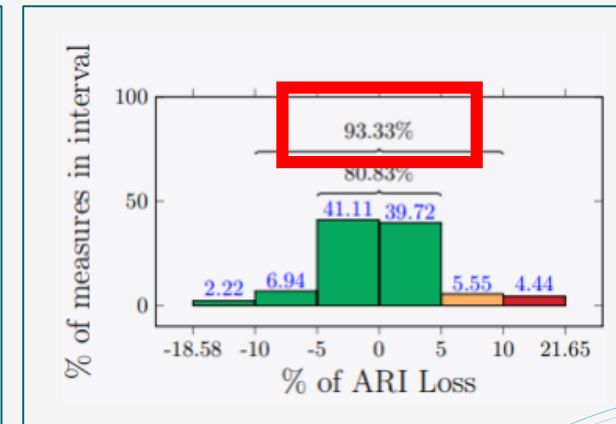
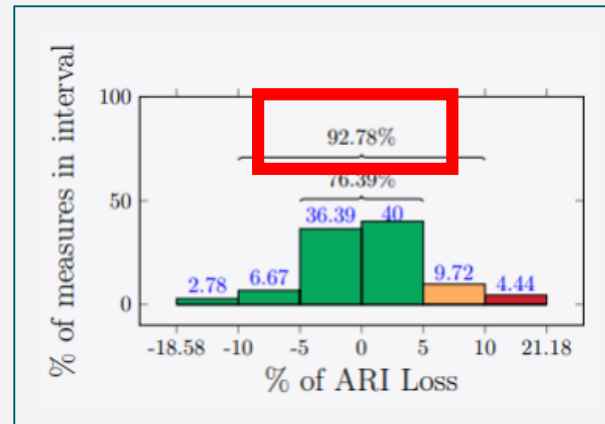
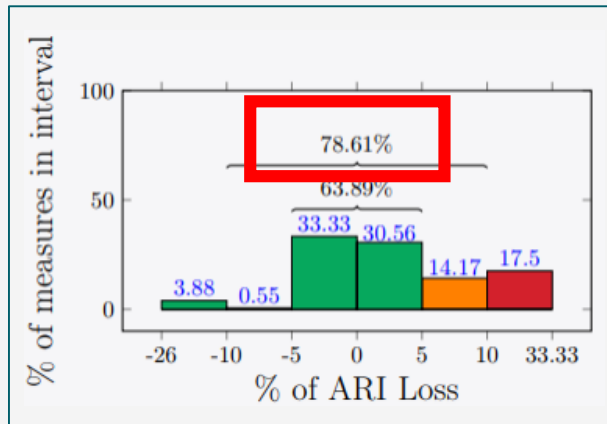
Execution time reduction Interval	Partial	PIGMMaLIOn	Incremental
< 0%	0.00%	1.94%	6.11%
[0% - 40%]	0.83%	13.33%	61.39%
[40% - 60%]	7.22%	25.00%	30.00%
> 60%	91.94%	59.72%	2.50%

Partial GMM

PIGMMaLIOn

Incremental GMM

ARI Losses








# Case study 3: Random forests

IEEE TRANSACTIONS ON COMPUTERS, VOL. 72, NO. 6, JUNE 2023

1595

## Accelerating Random Forest on Memory-Constrained Devices Through Data Storage Optimization

Camélia Slimani , Chun-Feng Wu , *Member, IEEE*, Stéphane Rubini ,  
Yuan-Hao Chang , *Senior Member, IEEE*, and Jalil Boukhobza , *Senior Member, IEEE*

# Random Forest - background

Label	Age	BMI	History (H)	Sport	sick
A	40	23	0	0	0
B	50	31	1	0	1
C	27	26	0	1	0
D	63	22	0	0	0
E	70	29	1	1	1
F	35	32	1	0	1
G	45	22	0	1	0
H	25	21	0	1	0

## Algorithm Decision Tree building method [4]

- 1: Bootstrap creation
- 2: **while** it exists an impure node  $n$  **do**
- 3:     Random creation of feature subset  $F$
- 4:     **for**  $f = 1$  to  $|F| - 1$  **do**
- 5:         Split trial of  $n$  into two children nodes according to the feature  $f$
- 6:     **end for**
- 7:     Choice of the best feature  $f^*$  and creation of the effective children nodes
- 8: **end while**

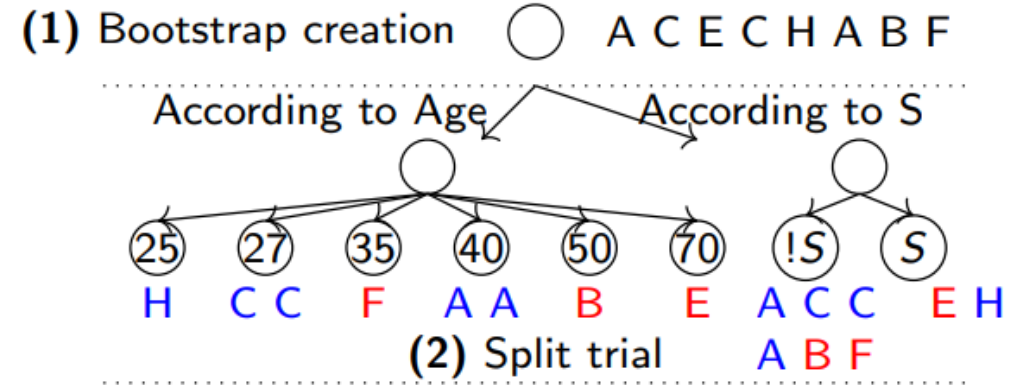
(1) Bootstrap creation ○ A C E C H A B F

# Random Forest - background

Label	Age	BMI	History (H)	Sport	sick
A	40	23	0	0	0
B	50	31	1	0	1
C	27	26	0	1	0
D	63	22	0	0	0
E	70	29	1	1	1
F	35	32	1	0	1
G	45	22	0	1	0
H	25	21	0	1	0

## Algorithm Decision Tree building method [4]

- 1: Bootstrap creation
- 2: **while** it exists an impure node  $n$  **do**
- 3:     Random creation of feature subset  $F$
- 4:     **for**  $f = 1$  to  $|F| - 1$  **do**
- 5:         Split trial of  $n$  into two children nodes according to the feature  $f$
- 6:     **end for**
- 7:     Choice of the best feature  $f^*$  and creation of the effective children nodes
- 8: **end while**

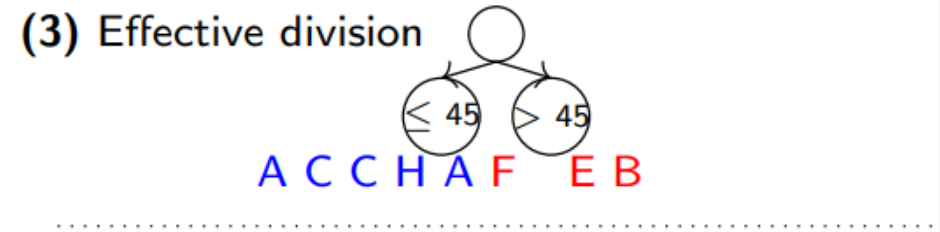
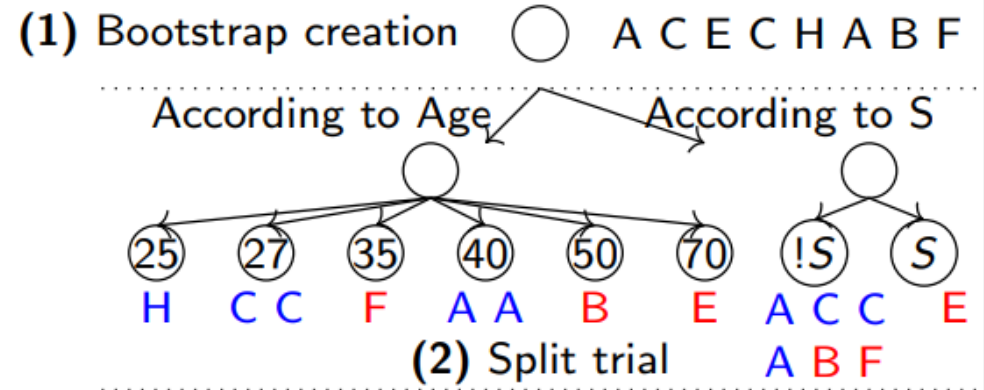


# Random Forest - background

Label	Age	BMI	History (H)	Sport	sick
A	40	23	0	0	0
B	50	31	1	0	1
C	27	26	0	1	0
D	63	22	0	0	0
E	70	29	1	1	1
F	35	32	1	0	1
G	45	22	0	1	0
H	25	21	0	1	0

## Algorithm Decision Tree building method [4]

- 1: Bootstrap creation
- 2: **while** it exists an impure node  $n$  **do**
- 3:     Random creation of feature subset  $F$
- 4:     **for**  $f = 1$  to  $|F| - 1$  **do**
- 5:         Split trial of  $n$  into two children nodes according to the feature  $f$
- 6:     **end for**
- 7:     Choice of the best feature  $f^*$  and creation of the effective children nodes
- 8: **end while**



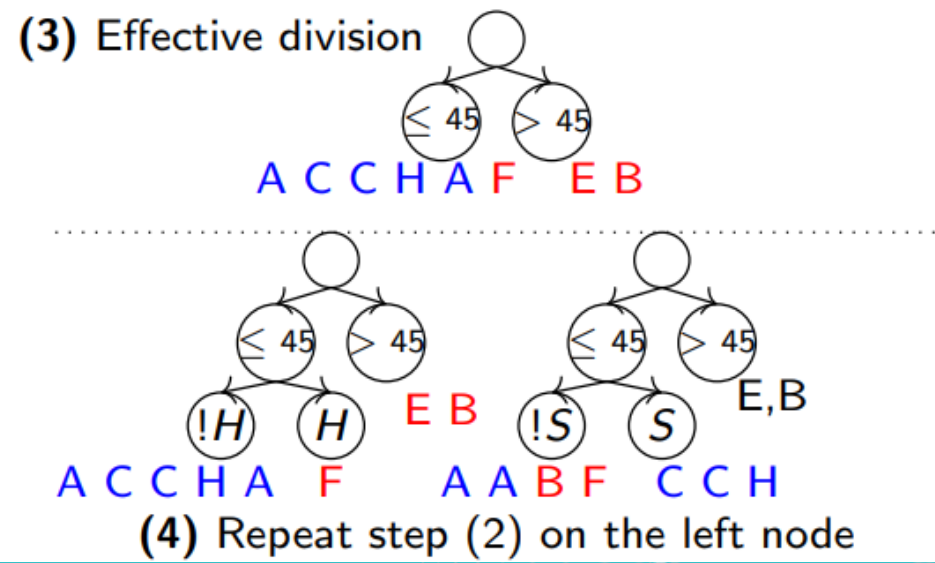
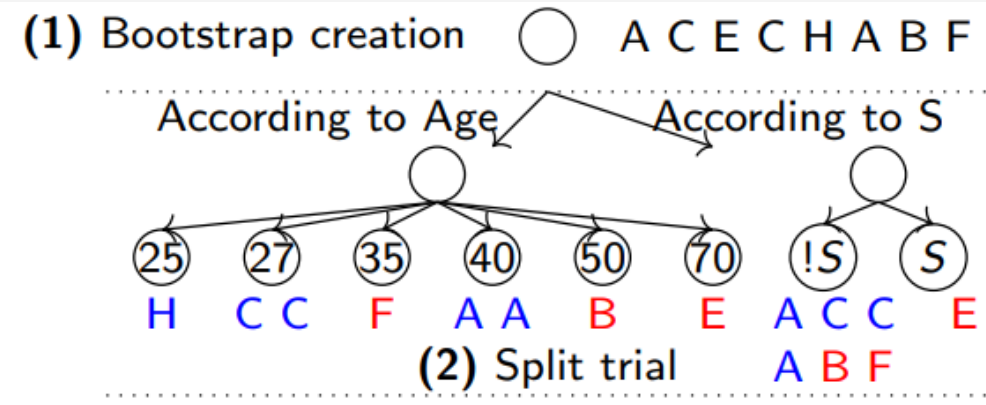


# Random Forest - background

Label	Age	BMI	History (H)	Sport	sick
A	40	23	0	0	0
B	50	31	1	0	1
C	27	26	0	1	0
D	63	22	0	0	0
E	70	29	1	1	1
F	35	32	1	0	1
G	45	22	0	1	0
H	25	21	0	1	0

## Algorithm Decision Tree building method [4]

- 1: Bootstrap creation
- 2: **while** it exists an impure node  $n$  **do**
- 3:     Random creation of feature subset  $F$
- 4:     **for**  $f = 1$  to  $|F| - 1$  **do**
- 5:         Split trial of  $n$  into two children nodes according to the feature  $f$
- 6:     **end for**
- 7:     Choice of the best feature  $f^*$  and creation of the effective children nodes
- 8: **end while**

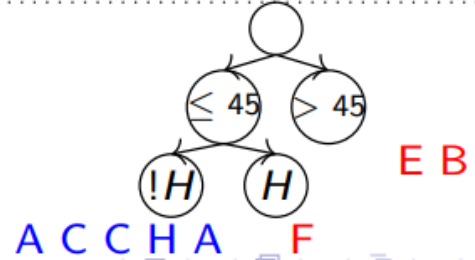
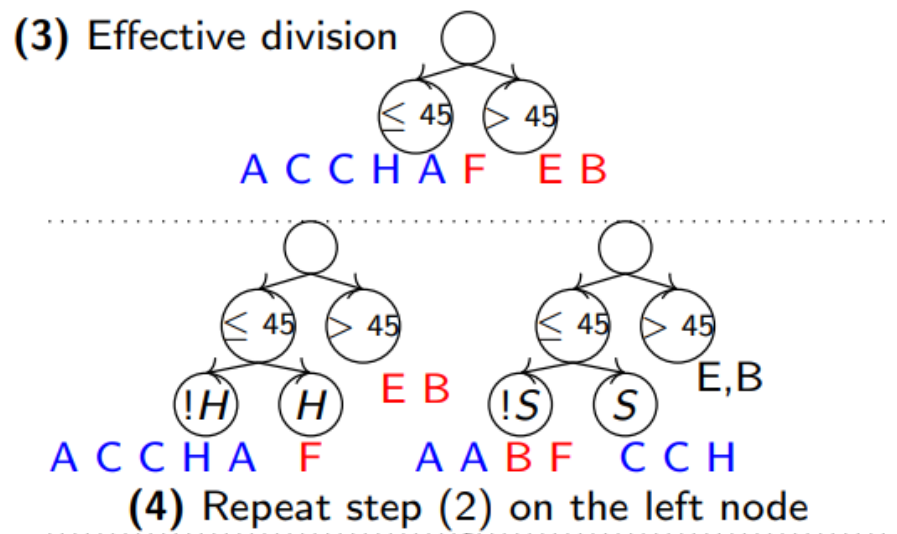
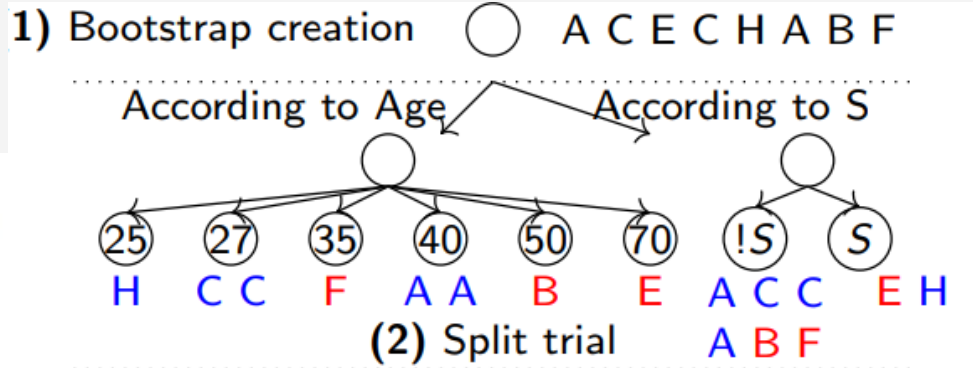


# Random Forest - background

Label	Age	BMI	History (H)	Sport	sick
A	40	23	0	0	0
B	50	31	1	0	1
C	27	26	0	1	0
D	63	22	0	0	0
E	70	29	1	1	1
F	35	32	1	0	1
G	45	22	0	1	0
H	25	21	0	1	0

## Algorithm Decision Tree building method [4]

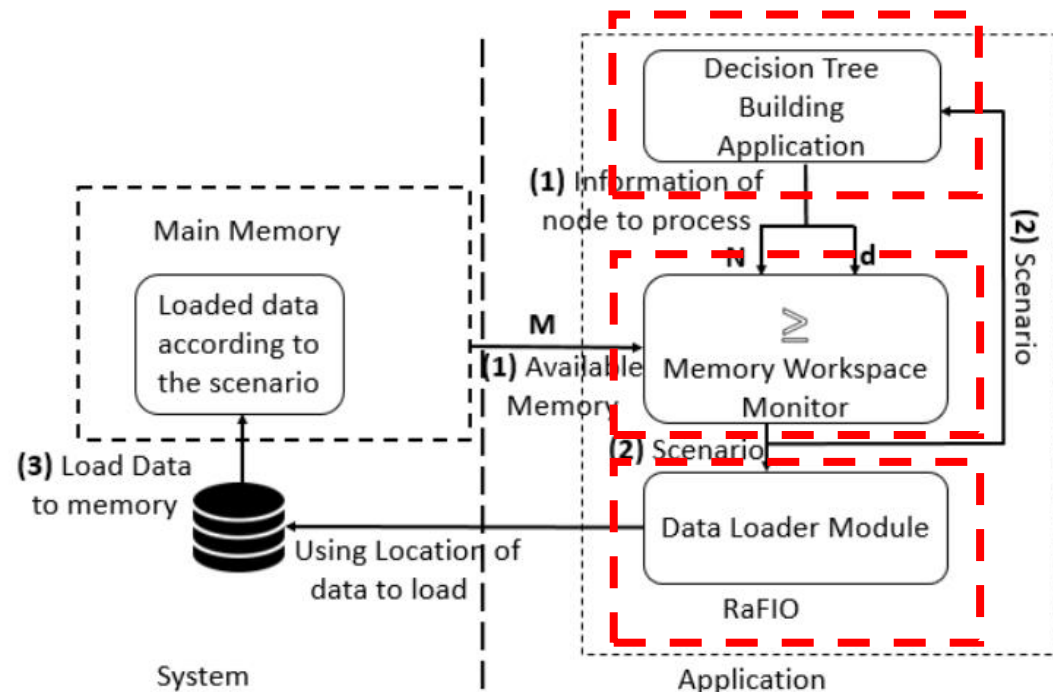
- 1: Bootstrap creation
- 2: **while** it exists an impure node  $n$  **do**
- 3:     Random creation of feature subset  $F$
- 4:     **for**  $f = 1$  to  $|F| - 1$  **do**
- 5:         Split trial of  $n$  into two children nodes according to the feature  $f$
- 6:     **end for**
- 7:     Choice of the best feature  $f^*$  and creation of the effective children nodes
- 8: **end while**



# RaFIO: Random Forest I/O-Aware algorithm

## / Key principles:

- / **On demand data loading:** Load only effectively needed data instead of the whole dataset
- / **Adaptive data loading:** Adaptively select useful data according to memory space available



## / Data volume loaded depends on the iteration.

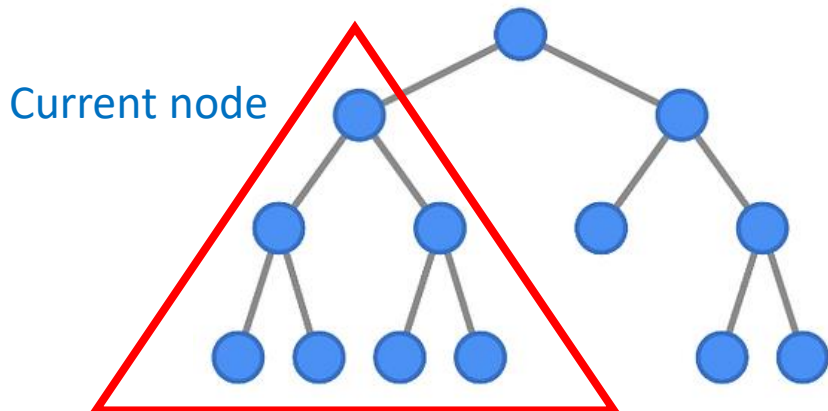
## / 3 options:

- / **Scenario 1, Full Sub-Tree Building:** all elements of the current node can fit in memory
- / **Scenario 2, Full node Building:** only features of current node can fit in memory.
- / **Scenario 3, per-Chunk split trial:** features of the current node cannot fit in the available memory → see next slide

# RaFIO: Random Forest I/O-Aware algorithm

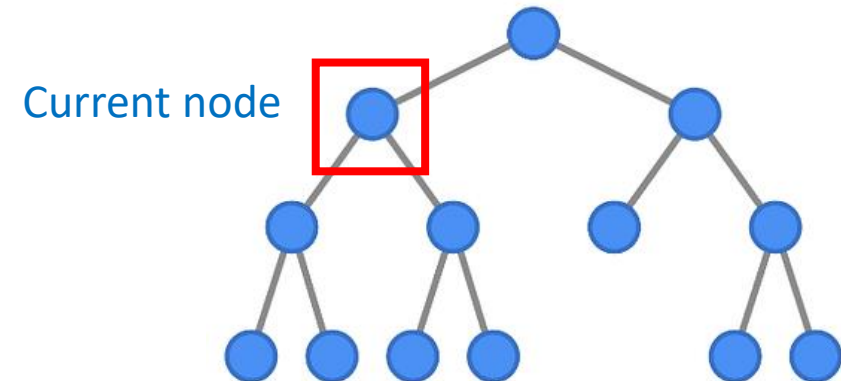
## / Scenario 1

- / Memory workspace monitor: all elements and all features  $\leq$  memory workspace
- / Data Loader module: Load all features of all elements of the node
- / Decision Tree Building module: Builds a subtree (depth first)



## / Scenario 2

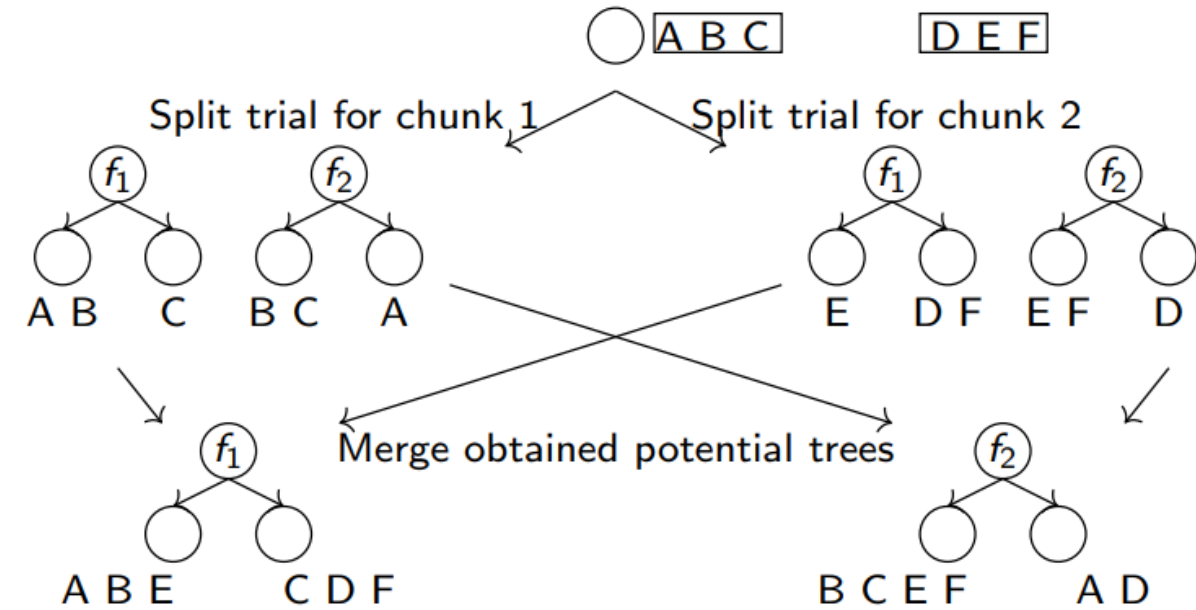
- / Memory workspace monitor: all features of elements of the node  $\leq$  memory workspace
- / Data Loader Module: loads all features of the elements of the node
- / Decision Tree Building module: Splits the current node



# RaFIO: Random Forest I/O-Aware algorithm

## / Scenario 3

- / Memory workspace monitor: features of elements of the node > memory available
- / Data Loader module: Load elements chunk by chunk
- / Decision Tree Building module:
  - / Load a chunk, i.e. a subset of elements of the node → memor
  - / Process each chunk separately
  - / Merge the sub-trees obtained from each chunk

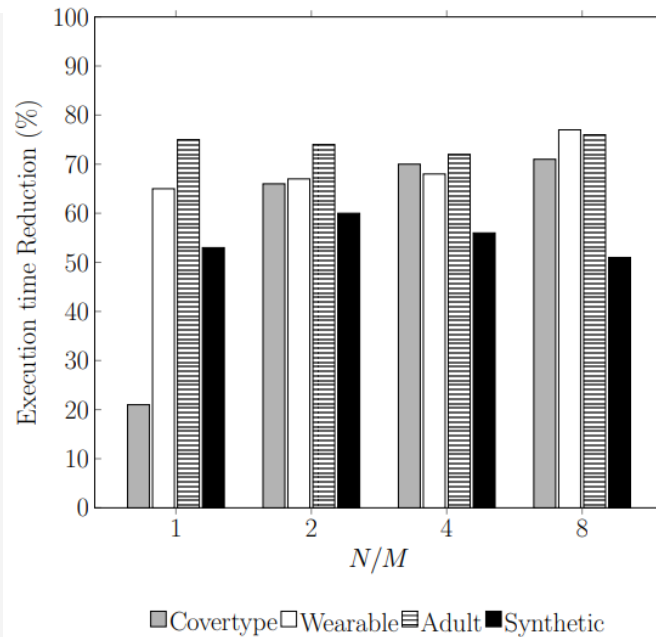


# Some results

- / Compared RaFIO to Ranger [1]
- /  $N/M = \{1, 2, 4, 8\}$ 
  - / N: Volume of data to process;
  - / M: Available memory workspace

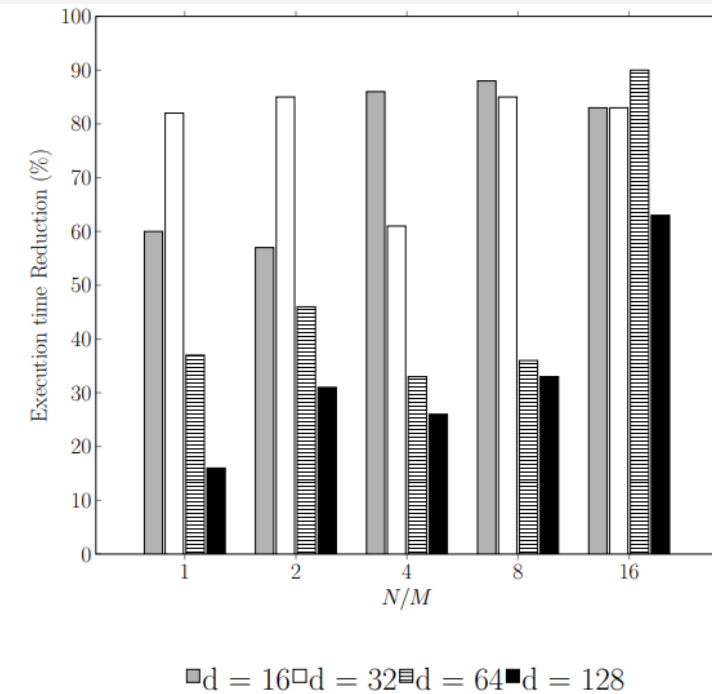
## Used Datasets

Dataset	Number of features	Number of elements	Dataset size (MB)
Coverttype	54	581012	239.36
Wearable	8	75128	4.58
Adult	14	48842	5.21
Synthetic	64	100000	48.82



RaFIO vs Ranger

Impact of the number of features



[1] Wright, Marvin & Ziegler, Andreas. (2015). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. Journal of Statistical Software. 77.

# 4. Time limited learning and energy optimization

IEEE MASCOTS'23

## Training K-means on Embedded Devices: a Deadline-aware and Energy Efficient Design

Hafsa Kara Achira\*<sup>†</sup>, Camélia Slimani \*, Jalil Boukhobza\*

\*ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285, F-29200 Brest. France

<sup>†</sup> École Nationale Supérieure D'Informatique (ESI), Algiers, Algeria

Email: ih\_karaachira@esi.dz, camelia.slimani@ensta-bretagne.fr, jalil.boukhobza@ensta-bretagne.fr



Applied Computing Review

## Adapting Gaussian Mixture Model Training to Embedded/Edge Devices: A Low I/O, Deadline-aware and Energy Efficient Design

Meriem Bouzouad

ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285

France

im\_bouzouad@esi.dz

Camélia Slimani

ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285

France

camelia.slimani@ensta-bretagne.fr

Yasmine Benhamadi

ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285

France

iy\_benhamadi@esi.dz

Jalil Boukhobza

ENSTA Bretagne, Lab-STICC, CNRS, UMR 6285, France

Institute of Information Science, Academia Sinica, Taiwan

jalil.boukhobza@ensta-bretagne.fr

# The case of K-means: A simple time model

K-MLIO time decomposition

$$T_{K-MLIO} = \sum_{i=1}^{\lceil N/M \rceil + 1} (T_{chunk_i})$$

Partial chunk time decomposition

$$T_{chunk_i} = T_{load} + T_{init} + T_{it} \cdot it_i$$

For the final chunk

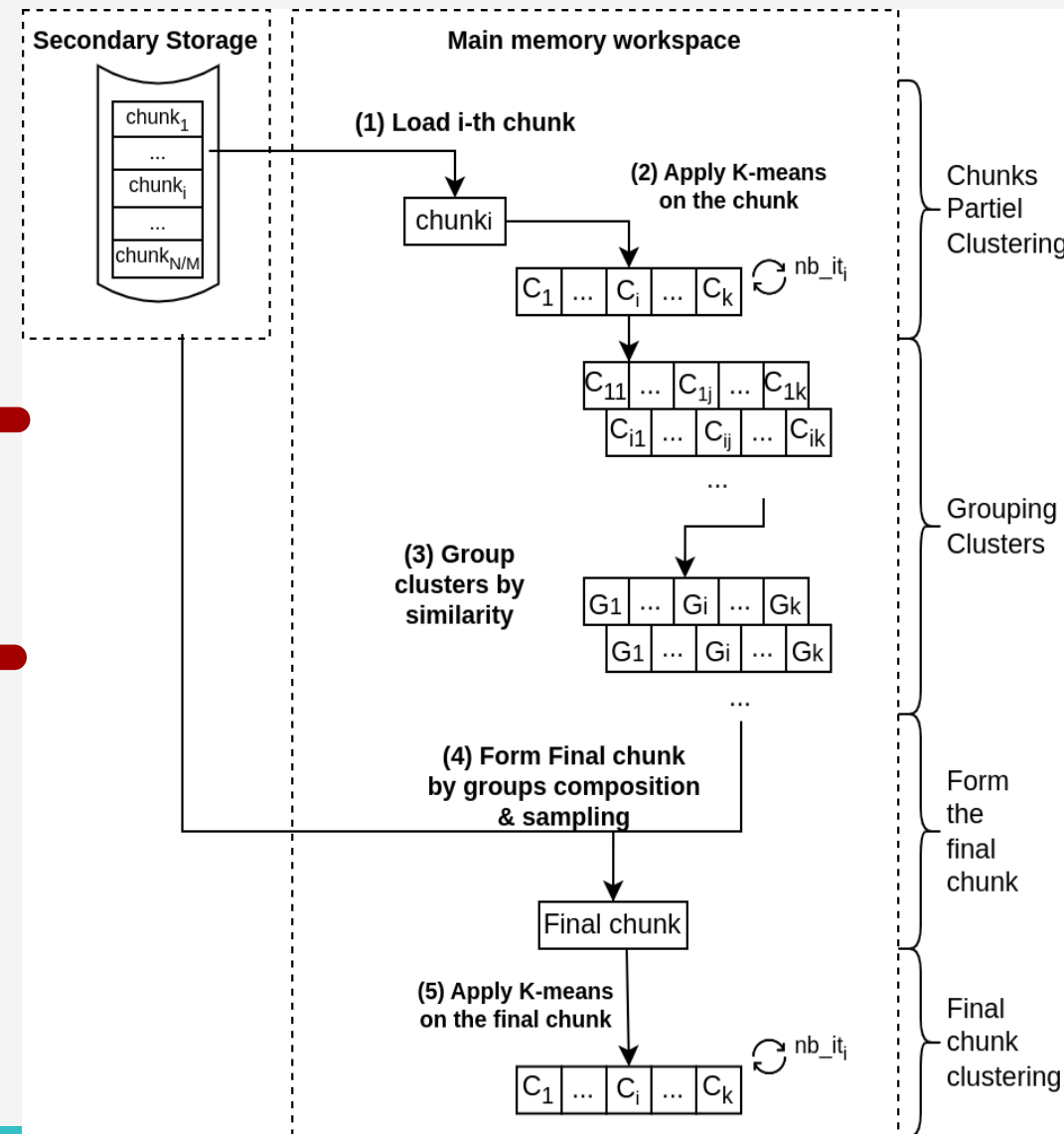
$$T_{chunk_{\lceil N/M \rceil + 1}} = T_{load} \cdot \lceil N/M \rceil + T_{init} + T_{it} \cdot it_{\lceil N/M \rceil + 1}$$

Upper-bounding

$$it_i \leq it_{max} \implies WCET_{chunk_i} = T_{load} + T_{init} + T_{it} \cdot it_{max}$$

Deduction

$$WCET_{K-MLIO} = \lceil N/M \rceil \cdot WCET_{chunk_1} + WCET_{chunk_{\lceil N/M \rceil + 1}}$$

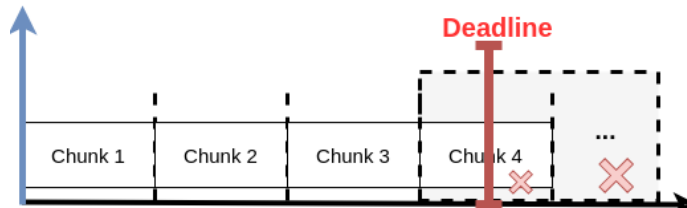




# EK-means : embedded K-means with timing and energy optimization

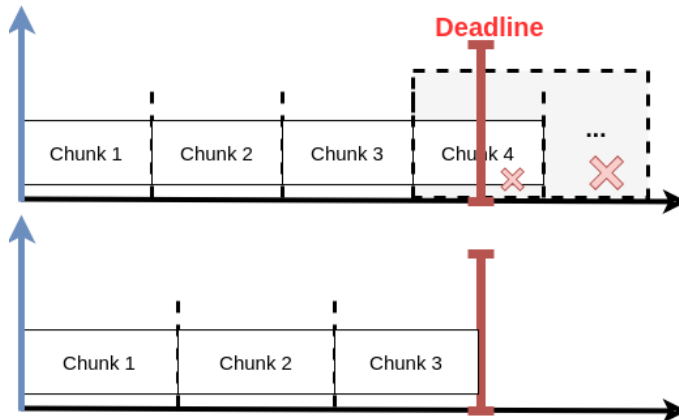
/ **Dropping a certain number of chunks** to satisfy the deadline

/ Estimate online the worst-case execution time of a chunk



/ **Applying DVFS opportunistically to reduce energy** consumption

/ Update the optimal frequency for processing the remaining chunks



# EK-means : embedded K-means with timing and energy optimization

## / Step 1: chunk processing time online analysis

/ Measure  $T_{load}$ ,  $T_{init}$ ,  $T_{it}$  with max frequency and deduce the number of cycles

## / Step 2: Estimate the number of chunks to drop (to meet the deadline)

/ According to the worst case scenario (max number of iterations to converge)

/ According to previous measured time metrics

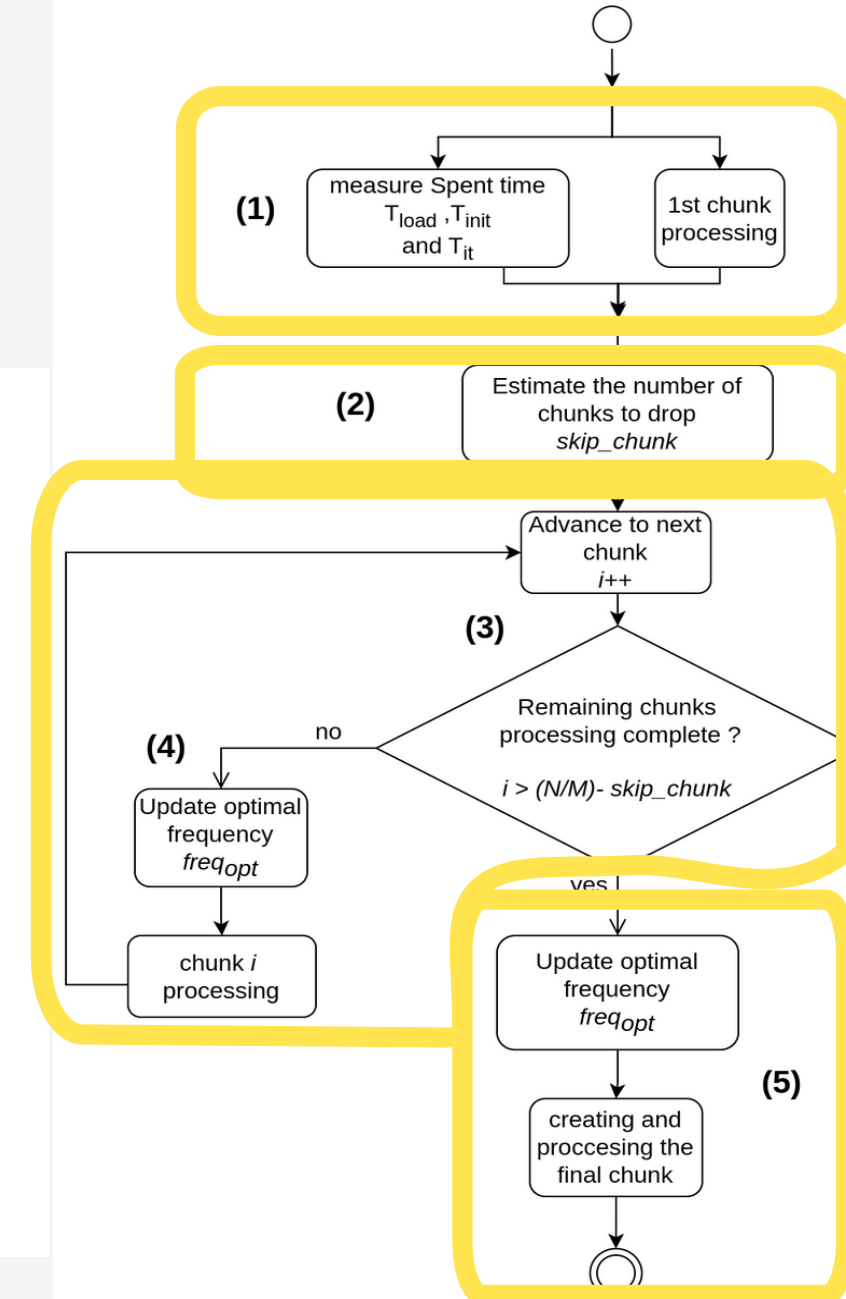
## / Step 3&4: Applying DVFS

/ Compute the slack time from previous chunks (WCET – actual time to process the chunk)

/ Find the lowest frequency that satisfies the deadline

## / Step 5: Process the final chunk

/ Select the right frequency (according to cumulated slack)



# Some results

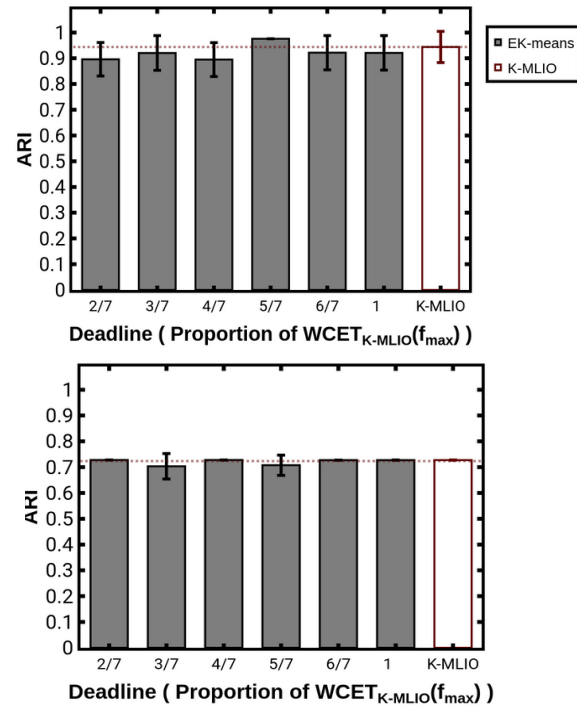
## / Deadline

Sep	$T_{deadline}$	EK-means interval (s)	$T_{deadline}$ satisfaction (%)
-0.2	300	95.82 - 250.6	100
	450	294.7 - 406.01	100
	600	424.38 - 514.37	100
	750	572.38 - 766.69	80
	900	744.19 - 899.37	100
	1050	822.63 - 957.52	100
0.0	300	55.08 - 134.99	100
	450	191.94 - 229.15	100
	600	286.99 - 337.86	100
	750	370.98 - 436.43	100
	900	482.21 - 591.01	100
	1050	578.3 - 660.63	100
0.2	300	56.67 - 248.73	100
	450	192.73 - 324.65	100
	600	247.41 - 561.85	100
	750	358.34 - 529.82	100
	900	404.72 - 821.24	100
	1050	558.44 - 812.79	100

**Deadline satisfaction in 98% of cases**

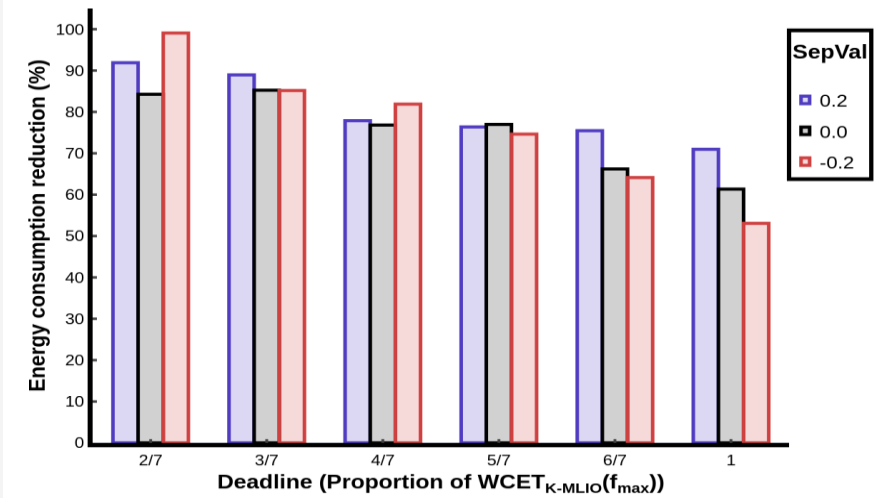
## / Accuracy

SEP = 0.2



**ARI of EK-means is 1.43% lower compared to K-MLIO**

## / Energy



**> 50% reduction on the processor dynamic energy**

# 5. Conclusions

## / Context:

- / One facet of Edge AI is the ability to learn on resource-limited edge devices to:
  - / Reduce latency
  - / Reduce the network traffic
  - / Enforce Privacy and security

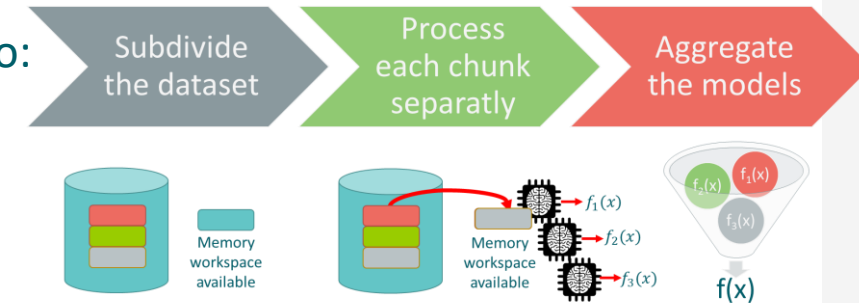
/ On commodity hardware, several frameworks can hardly be usable

/ **Problem:** Several ML algorithms have been thought with the memory large enough to contain the dataset

- / → Lag because of I/O swapping issues

## / Solution:

- / Devised a general pattern → divide (the dataset) to conquer (the I/O bottleneck)
- / Applied the optimization pattern successfully on 3 ML algorithms: K-means, GMM, Random forests
- / Reduced the I/Os by up to 95%
- / No or small loss on accuracy
- / Energy optimization



# 5. Perspectives

## Applications on the edge

- More ML algorithms to work on → Generalize the approach
- Explore deep learning algorithms
- Explore LLM deployments
- Adaptive accuracy for the learning
- ...

## System and distribution issues

- Temperature reduction issues
- Energy consumption modeling and optimization
- Design of ML libraries optimized for I/Os
- ML learning on commodity hardware (VM, containers)
- Load balancing/scheduling between edge devices
- Data and compute placement
- Carbon footprint-aware deployment
- ...

# Special thanks to ...



Hafsa Kara  
Achira



Yasmine  
Benhamadi



**You can download the  
presentation by flashing the code**



Meriem  
Bouzouad



Vincent  
Lannurien



Camélia Slimani



Stéphane  
Rubini



Chun-Feng  
Wu



Yuan-Hao  
Chang