TiAcc: Triangle-inequality based Hardware Accelerator for K-means on FPGAs

Yuke Wang, Boyuan Feng, Gushu Li,

Georgios Tzimpragos, Lei Deng, Yuan Xie, Yufei Ding

University of California, Santa Barbara

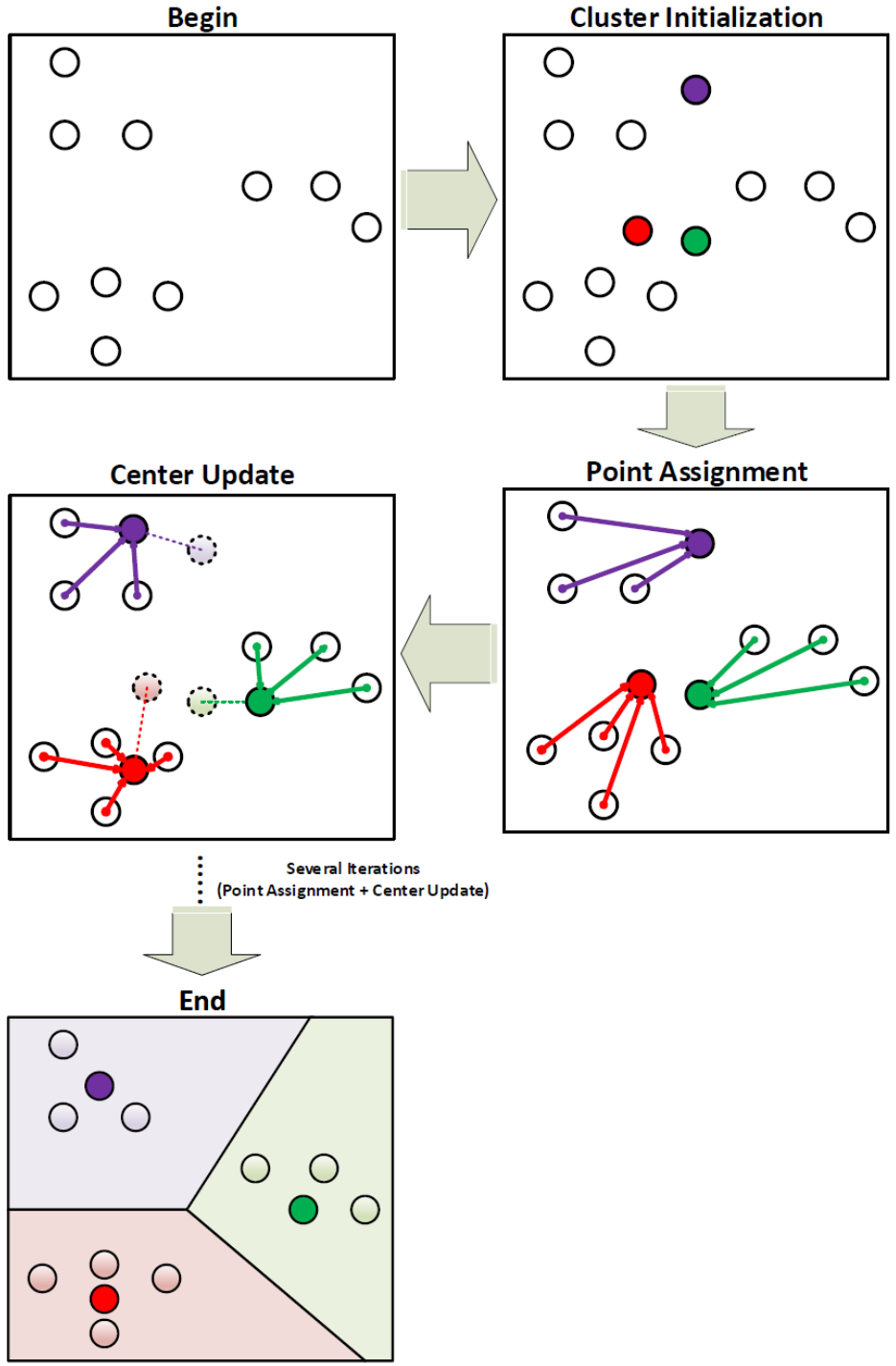
Yuke Wang is a third-year PhD in the department of computer science at the UCSB.

K-means Algorithm

- Unsupervised Learning.
  Unlabeled data clustering
  Image segmentation
  Feature learning

- Important but time-consuming on real-world datasets.

- Unsatisfactory performance.
  $O(N \times K \times D)$
  N: number of points
  K: number of clusters
  D: dimensionality



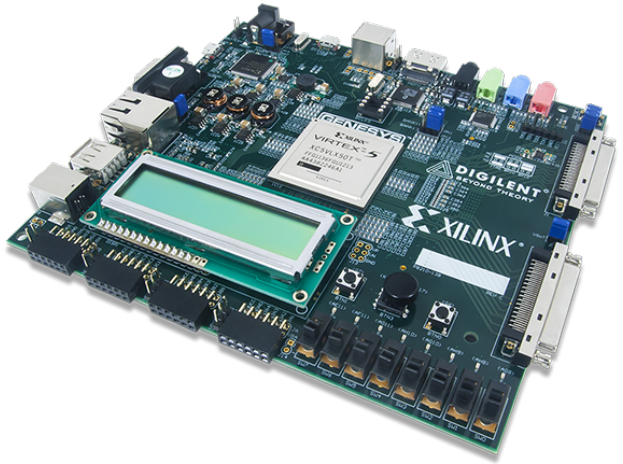
Optimization Exploration

Algorithm-level Optimization

Reduce redundant computations, such as triangle-inequality based filtering and KD-tree based methods.

Hardware-level Implementation

- Modern general purpose processor such as CPU, GPU or hardware accelerators such as FPGA and ASIC.
- FPGA solution: Energy efficiency and developing flexibility.



Credit: Google Image

Shortcomings of Previous Work

- Mostly count on hardware-level design and optimizations.
- Built and optimized for specific datasets, lacking configurability and adaptability.
- Evaluated on high-end FPGAs or simulated under ideal on-chip resource assumption.

Focus

- ☑ It must be able to leverage both algorithm-level optimization and hardware-level design.
- ☑ It must have the adaptability for handling datasets of varying size and dimensionality.
- ☑ It must have the flexibility to be implemented on the majority of off-the-shelf FPGA.

Contributions

- ➢ A novel multi-level triangle-inequality based filtering.
- ➢ A highly parameterized FPGA design.
- ➢ A data batch streaming approach.
- ➢ A pipeline decoupling technique.
- ➢ Low-cost commodity-level FPGA, such as Pynq.

Overview

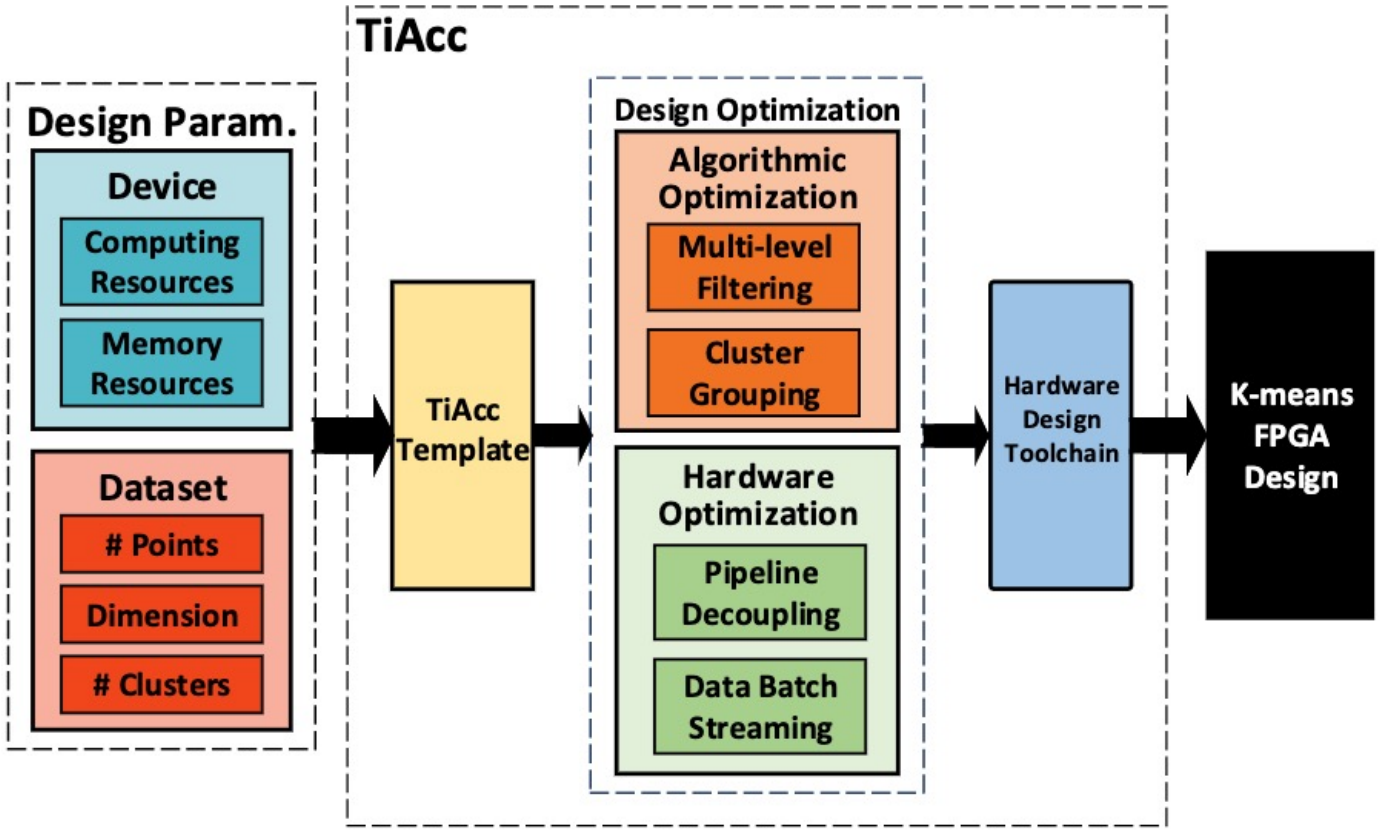


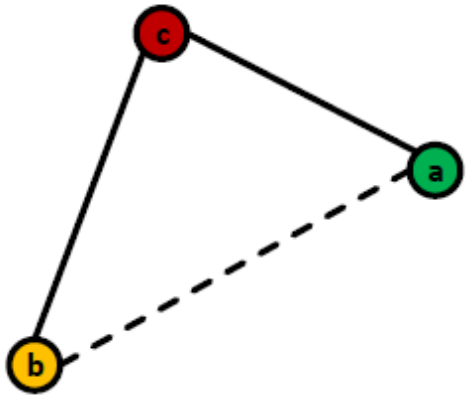
Fig. 1: TiAcc Design Workflow.

Background

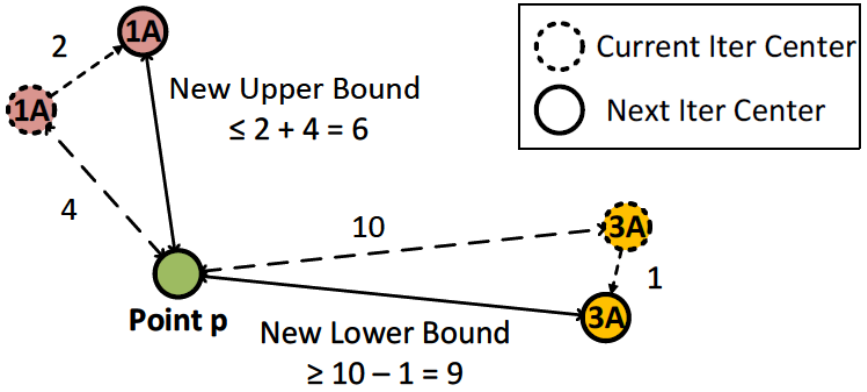
Triangle Inequality

$$lb(a,b) = d(a,c) - d(b,c)$$

$$ub(a,b) = d(a,c) + d(b,c)$$



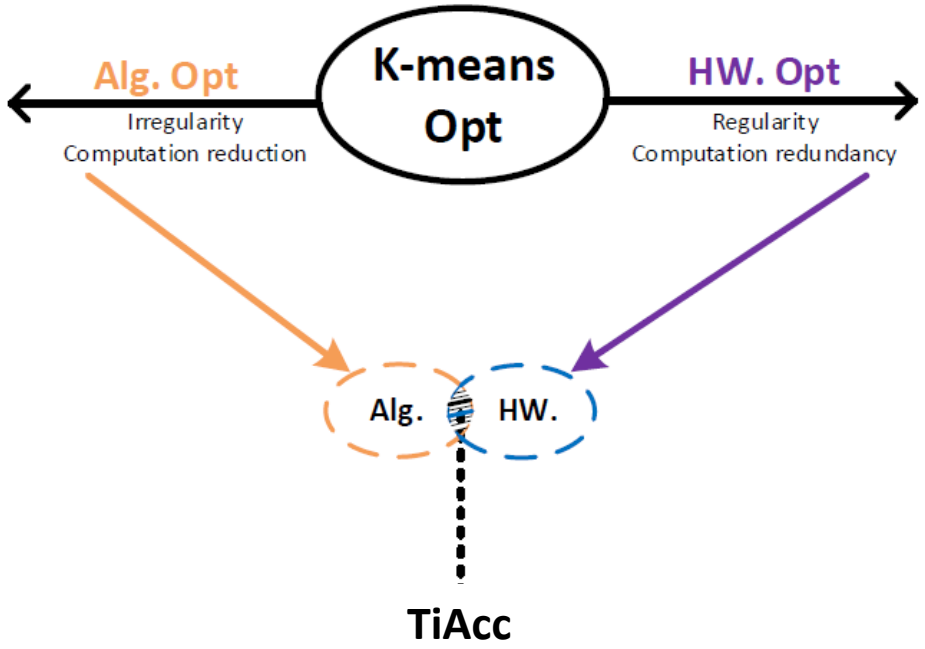
Elkan's K-means



Shortcomings of Elkan's K-means

- ❖ Extra computations required.
- ❖ Extra memory space required.
- ❖ Calculation irregularity increased.

TiAcc Design



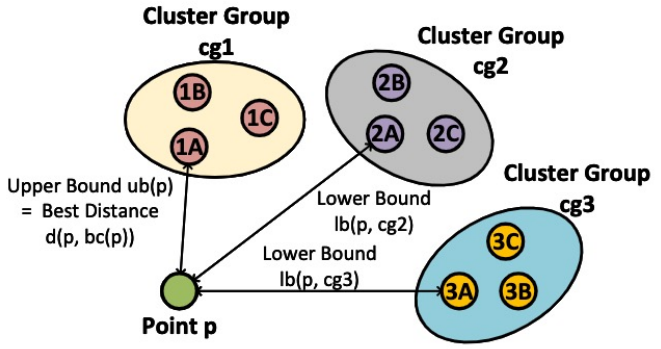
Algorithmic Optimization

Algorithm 1: TiAcc K-means

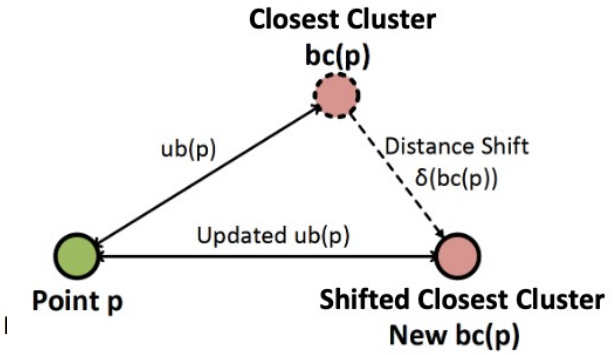
input : Point set, P = $\{p_1, p_2, \ldots, p_n\}$
output: Point label, $PL = \{pl_1, pl_2, \ldots, pl_n\}$

```
1  Initialize C = {c_1, c_2, ..., c_k} ;
2  CG = {cg_1, cg_2, ..., cg_{k_c}} = clusterGroup(C);
3  disBound = disComp(P, C, CG) ;
4  while TiAcc K-means not converge do
5  |   P_pass = pointFilter(P, disBound) ;
6  |   CG_pass = groupFilter(P_pass, disBound, CG) ;
7  |   foreach p in P_pass do
8  |   |   assignedLabel[p] = argmin(p, CG_pass[p]) ;
9  |   end
10 |   updateCenter(C, assignedLabel);
11 end
```

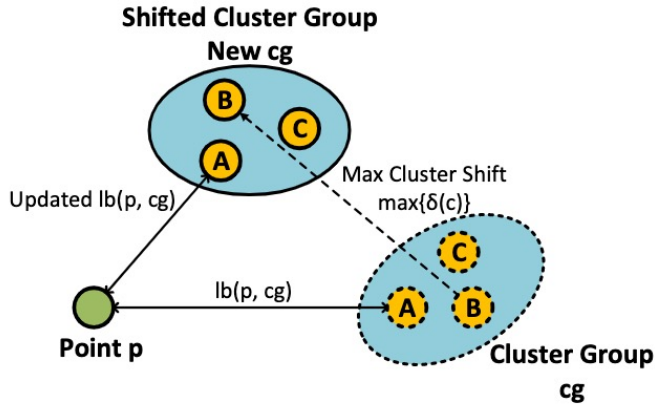
Filtering Step



(a) Point-cluster Relationship.

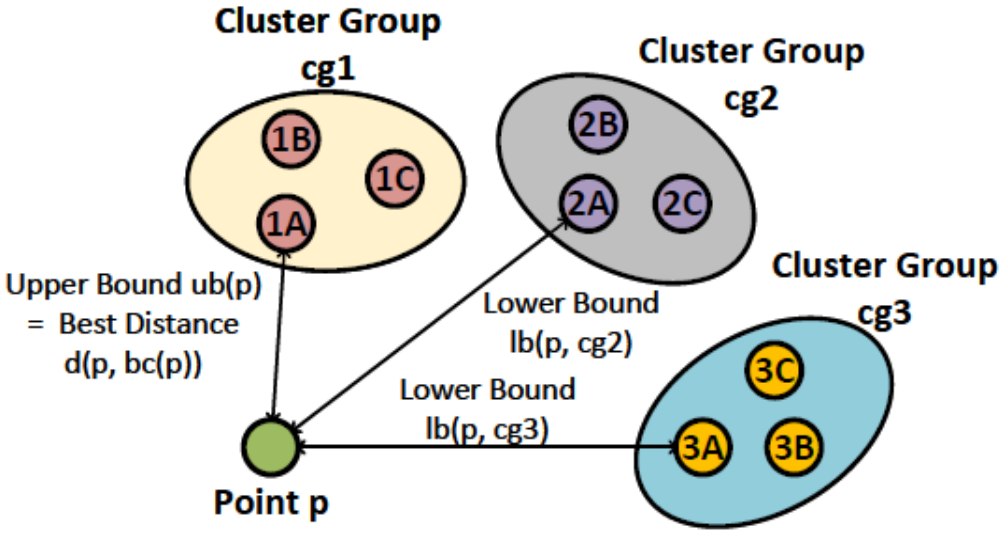


(b) $ub(p)$=$ub(p)$+$\delta(bc(p))$.



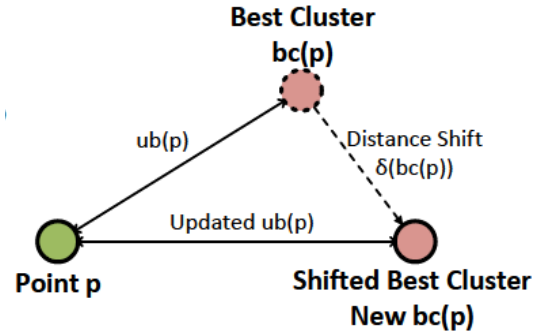
(c) $lb(p, cg)$=$lb(p, cg)$-$max\ \delta(c)$.

Cluster Grouping



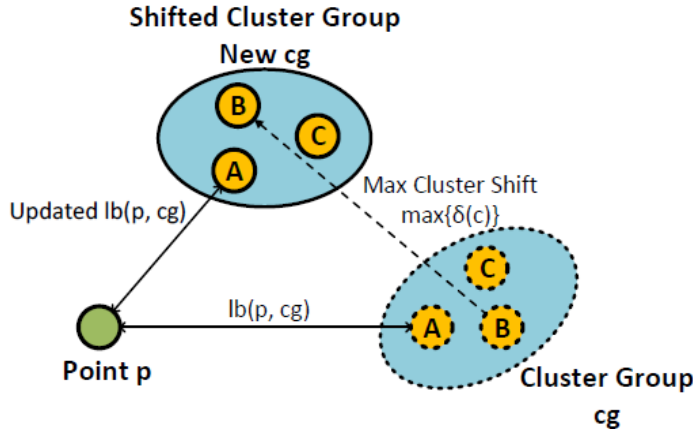
- Group the initial clusters by classic K-means.
- Fewer distance lower bounds.
- Less computation and memory overhead.

Distance Bounds



Upper bound update

The distance upper bound of the point will be updated if the best cluster of point has shifted since the last iteration.



Lower bound update

The distance lower bound between a point and cluster group will be updated based on the maximum distance shifts of the clusters inside the cluster group

Architecture Design

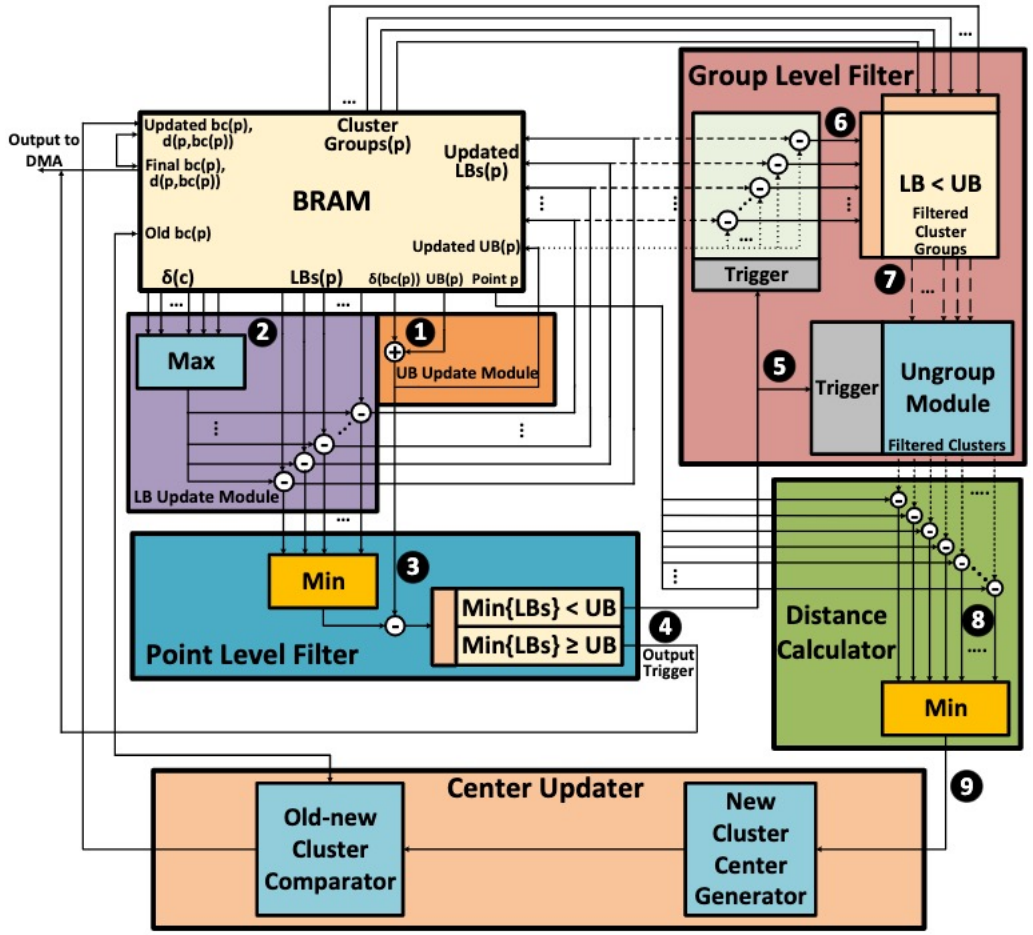
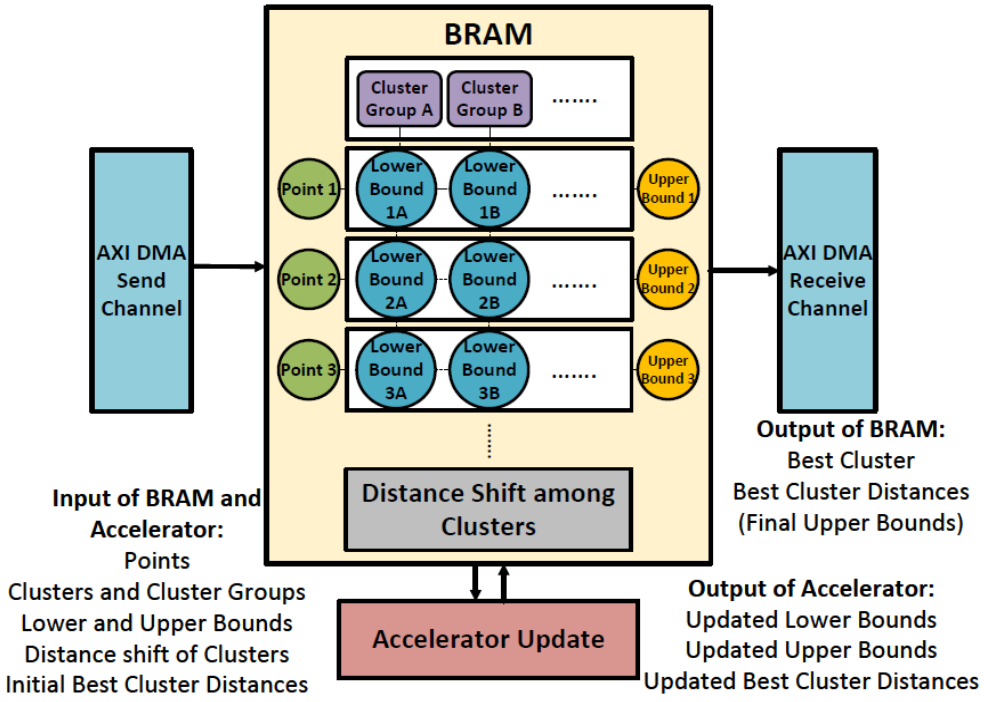


Fig. 4: TiAcc Architecture Design.

Other Optimizations



a. Data Batch Streaming

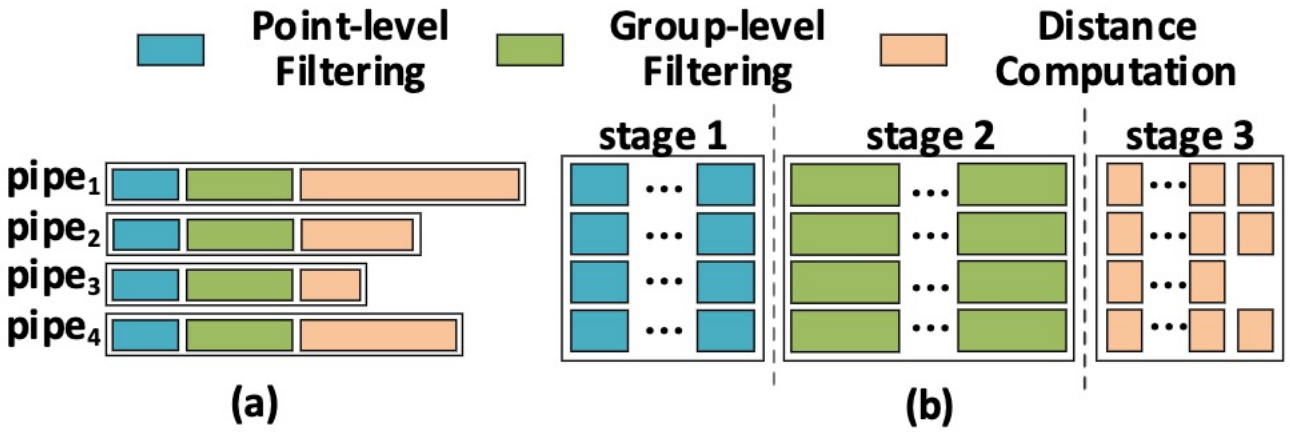


Fig. 5: (a) Conventional Pipeline; (b) Decoupled Stages.

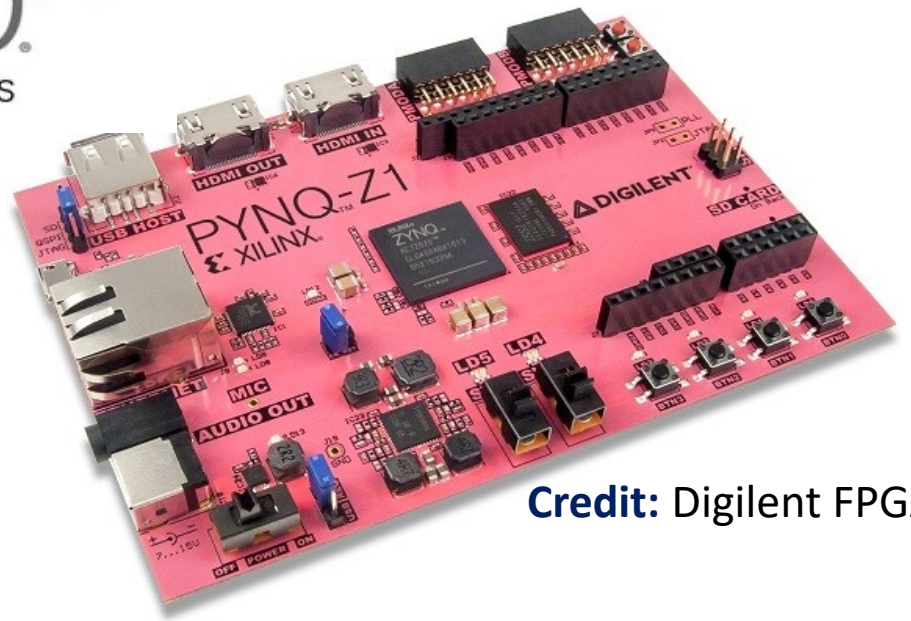
b. Pipeline Decoupling

Experiment

Experimental Setup

1) Xilinx Vivado Design Suite v2018.2.
2) Pynq-Z1
   - ARM Cortex-A9 processor (PS).
   - Artix-7 family programmable logic (PL).

Credit: Xilinx Vivado



Credit: Digilent FPGA

Datasets for Evaluation

1) Six real-life datasets from [1], Num of Points $(5{,}000 \sim 430{,}000)$ and dimensionality $(3 \sim 28)$.
2) $cluster\_size = \left\lfloor \frac{\sqrt{dataset\_size}}{t} \right\rfloor$,
3) $cluster\_group\_size = \left\lfloor \frac{cluster\_size}{10} \right\rfloor$.

Dataset	# points (n)	# dimension (d)	# clusters (k)
Parkinsons Updrs	5875	21	38
Letter Recognition	20000	16	70
Electronic Board Reading	45781	5	53
Kegg Shuffled Normal	65554	28	64
Skin NonSkin	245057	4	62
3D Spatial Network	434874	3	82

[1]: Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Experiments: Performance & Energy

TABLE II: Performance and Energy Efficiency Comparison.

Dataset.	**Time. TiAcc (s)**	**Power. TiAcc (W)**	**Speedup. *vs.* ARM**	**En.Eff. *vs.* ARM**	**Speedup. *vs.* Xeon**	**En.Eff. *vs.* Xeon**	**Speedup. *vs.* Titan Xp**	**En.Eff. *vs.* Titan Xp**
Parkinsons Updrs	0.001	5.26	28.34×	**8.57**×	6.90×	**65.95**×	3.00×	**42.43**×
Letter Recognition	0.006	4.64	13.17×	**5.42**×	1.92×	**25.10**×	0.67×	**13.33**×
Electr Board Read	0.010	4.06	9.89×	**4.99**×	1.79×	**21.53**×	0.70×	**13.88**×
KEGG Meta Net	0.016	5.52	16.53×	**4.91**×	2.27×	**27.53**×	0.31×	**6.97**×
Skin NonSkin	0.061	3.95	8.91×	**2.98**×	8.83×	**160.06**×	0.31×	**6.82**×
3D Spatial Network	0.143	4.16	6.42×	**2.28**×	7.93×	**145.14**×	0.19×	**4.31**×

Experiments: Additional Studies

TABLE III: Resource Comparison with Elkan's K-means.

Design	BRAM _18K	DSP48E	FF	LUT
Elkans	19	3	39453	41627
TiAcc	17	3	32488	30641

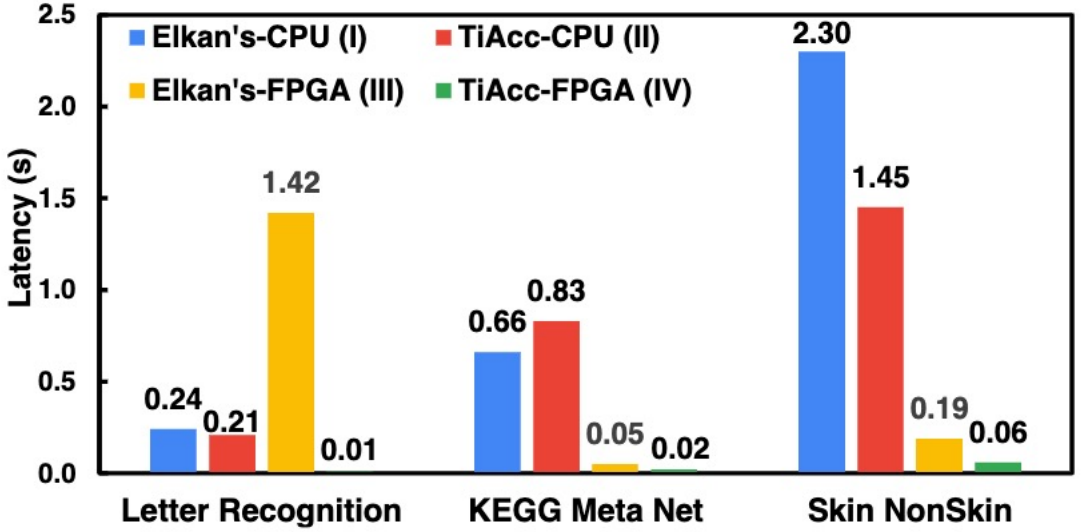


Fig. 6: Comparison with Elkan's K-means.

Experiments: Additional Studies (Con'd)

TABLE IV: Data Type Comparison.

Distance Type	Data Type	# DSP	# FF	# LUT	Percent Err.	Latency (Clock Cycles)
Euclidean Distance	I	51	281	1512	22.29%	35
	II	5	758	1370	1.62%	64
	III	16	1785	2455	0%	70
Manhattan Distance	I	0	190	981	21.67%	7
	II	2	427	2072	0.36%	56
	III	6	1231	4503	0%	69

Note: I: Fixed-point; II: Single-precision Floating-point; III: Double-precision Floating-point.

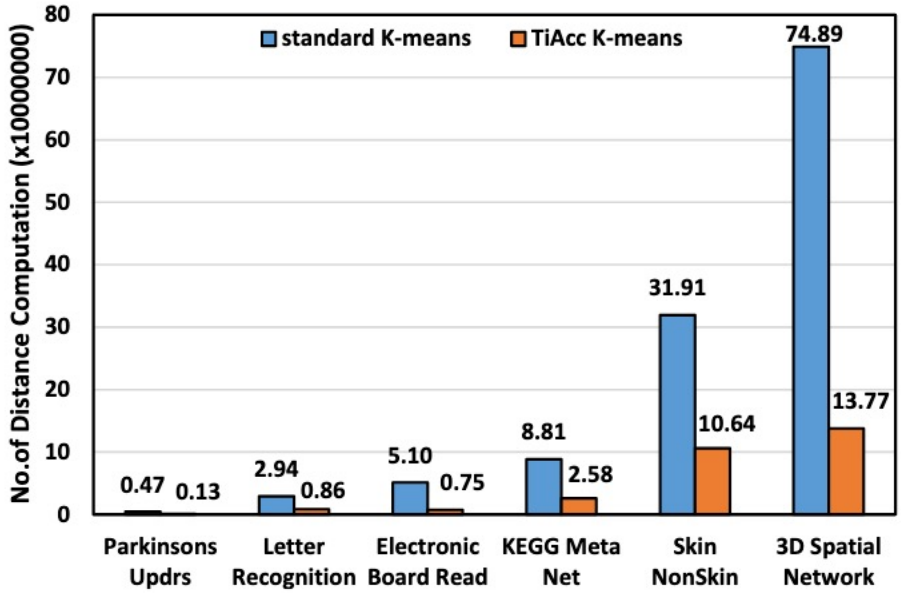


Fig. 7: Distance Computation Reduction.

Thank You

Q & A