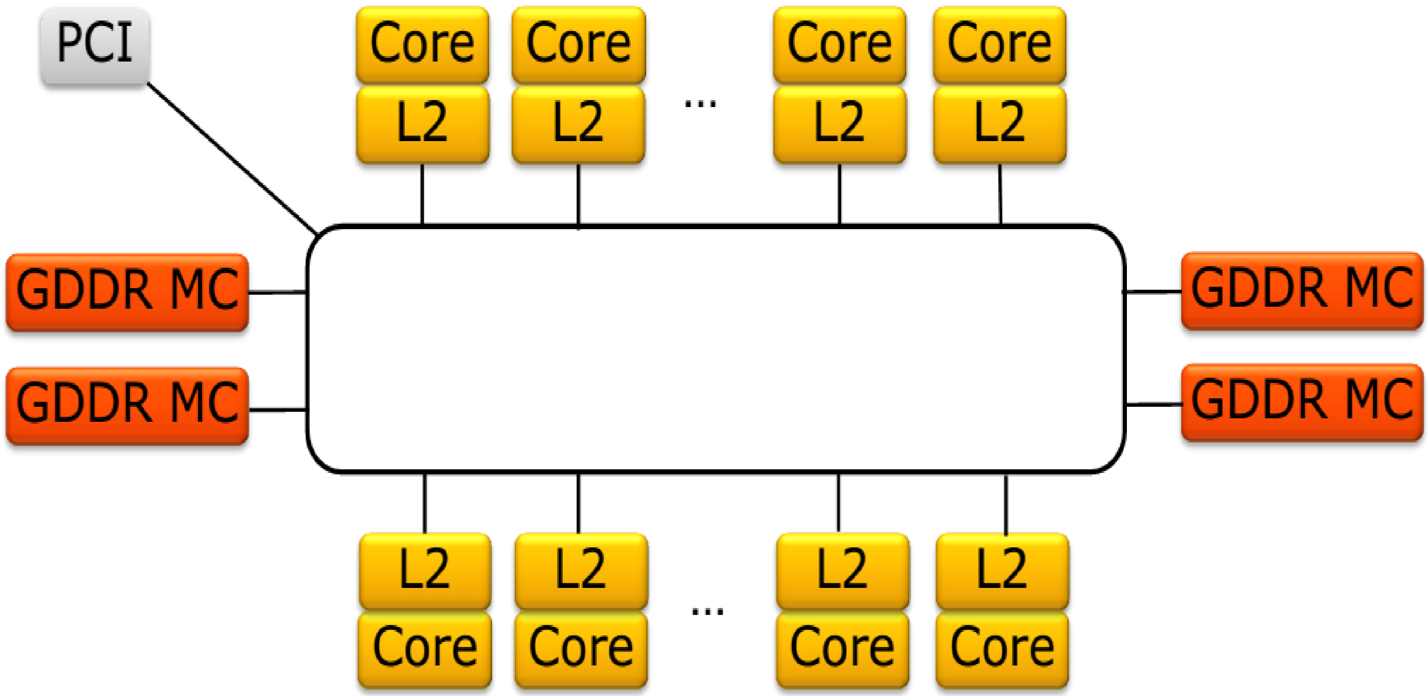
Application of Information and Communication Technologies AICT2015.
Russia, Rostov-on-Don, 14-16 October 2015

Accelerating Medoids-based Clustering with the Intel Many Integrated Core Architecture

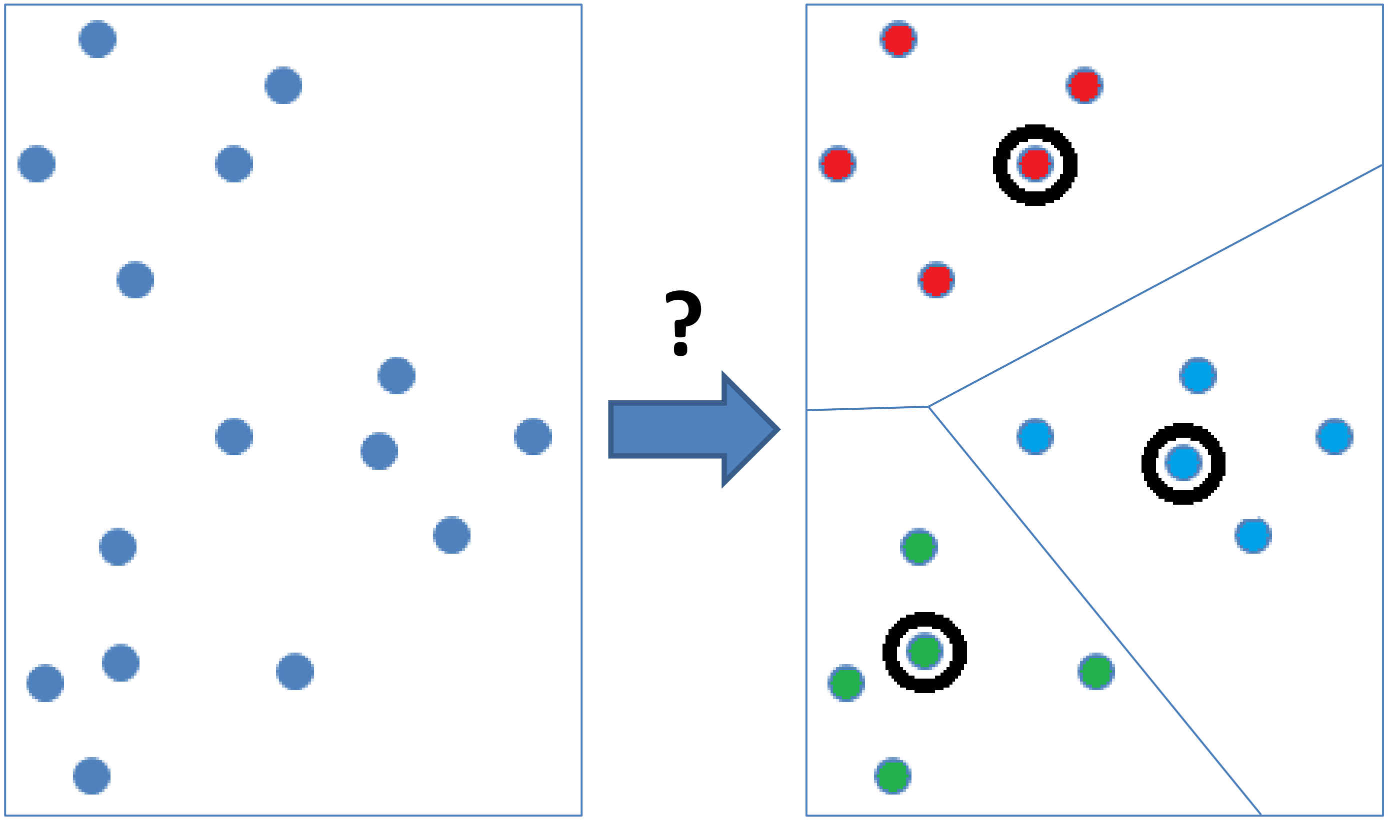
Timofey Rechkalov, Mikhail Zymbler
South Ural State University, Russia

The reported study was supported by
"Participant of Youth Scientific Innovational Competition" UMNIK program.

Intel Xeon Phi



Partitioning Clustering



PAM properties

- *PAM algorithm (Partitioning Around Medoids)* – partitioning clustering algorithm which selects cluster centers among clustered objects
- Such objects called *medoids*
- Iteration time complexity is $O(k(n\text{-}k)^2)$, where
  - n is the number of clustered objects
  - k is the number of clusters

Objective function

- Objective function

$$E = \sum_{j=1}^{n} \min_{1 \le i \le k} \rho(c_i, o_j).$$

, where c_i is the medoid, o_j is the clustered object, ρ is the distance metric

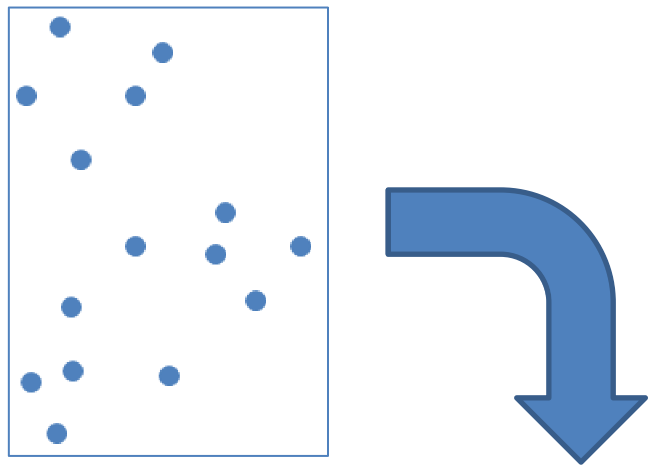
PAM pseudocode

Input: Set of objects O, number of clusters k

Output: Set of clusters C

1. Initialize C; // BUILD phase
2. **repeat** // SWAP phase
3. Find best swapping estimation T_{min};
4. Swap c_{min} and o_{min}, determined by T_{min};
5. **until** $T_{min} < 0$;

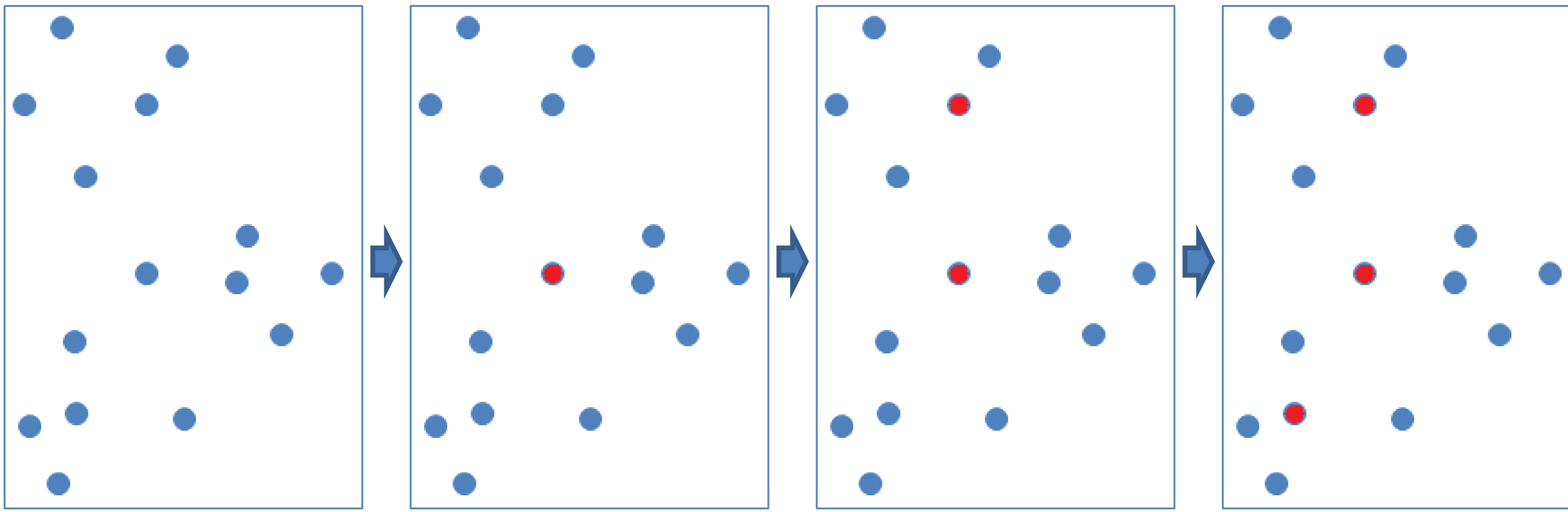
Calculating distance matrix



	o_1	o_2	o_3	…	o_n
o_1	$\rho(o_1,o_1)$	$\rho(o_1,o_2)$	$\rho(o_1,o_3)$	…	$\rho(o_1,o_n)$
o_2	$\rho(o_2,o_1)$	$\rho(o_2,o_2)$	$\rho(o_2,o_3)$	…	$\rho(o_2,o_n)$
o_3	$\rho(o_3,o_1)$	$\rho(o_3,o_2)$	$\rho(o_3,o_3)$	…	$\rho(o_3,o_n)$
…	…	…	…	…	…
o_n	$\rho(o_n,o_1)$	$\rho(o_n,o_2)$	$\rho(o_n,o_3)$	…	$\rho(o_n,o_n)$

BUILD phase

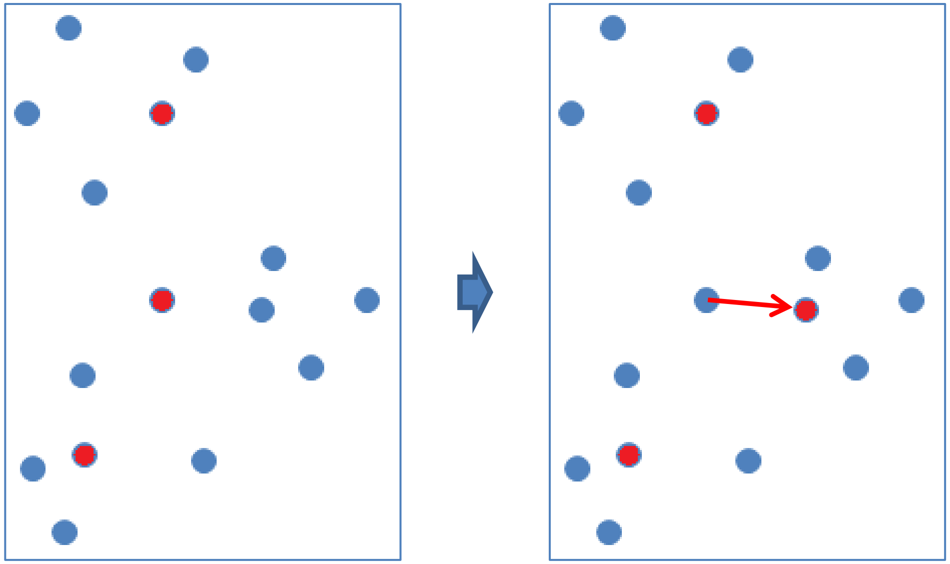
k=3



$E=\infty$ $E= 2{,}250$ $E= 1{,}578$ $E= 1{,}014$

Time complexity $O(kn^2)$

SWAP phase



E= 1,014 *E*= 0,865

Time complexity $O(k(n - k)^2)$ per iteration

Used Optimizations

- Parallelizing with OpenMP
- Loops with arithmetic operations were reorganized for vectorized execution
  - Data consists of 32 element blocks
- Tiling for better locality and cache performance

PAM implementation

Input: Set of objects *O*, number of clusters *k*

Output: Set of clusters *C*

1. M <- PrepareDistanceMatrix(O);
2. C <- BuildMedoids(M); // BUILD phase
3. **repeat** // SWAP phase
4. T_{min} <- FindBestSwap(M, C);
5. Swap c_{min} and o_{min}, determined by T_{min};
6. **until** $T_{min} < 0$;

Experimental evaluation

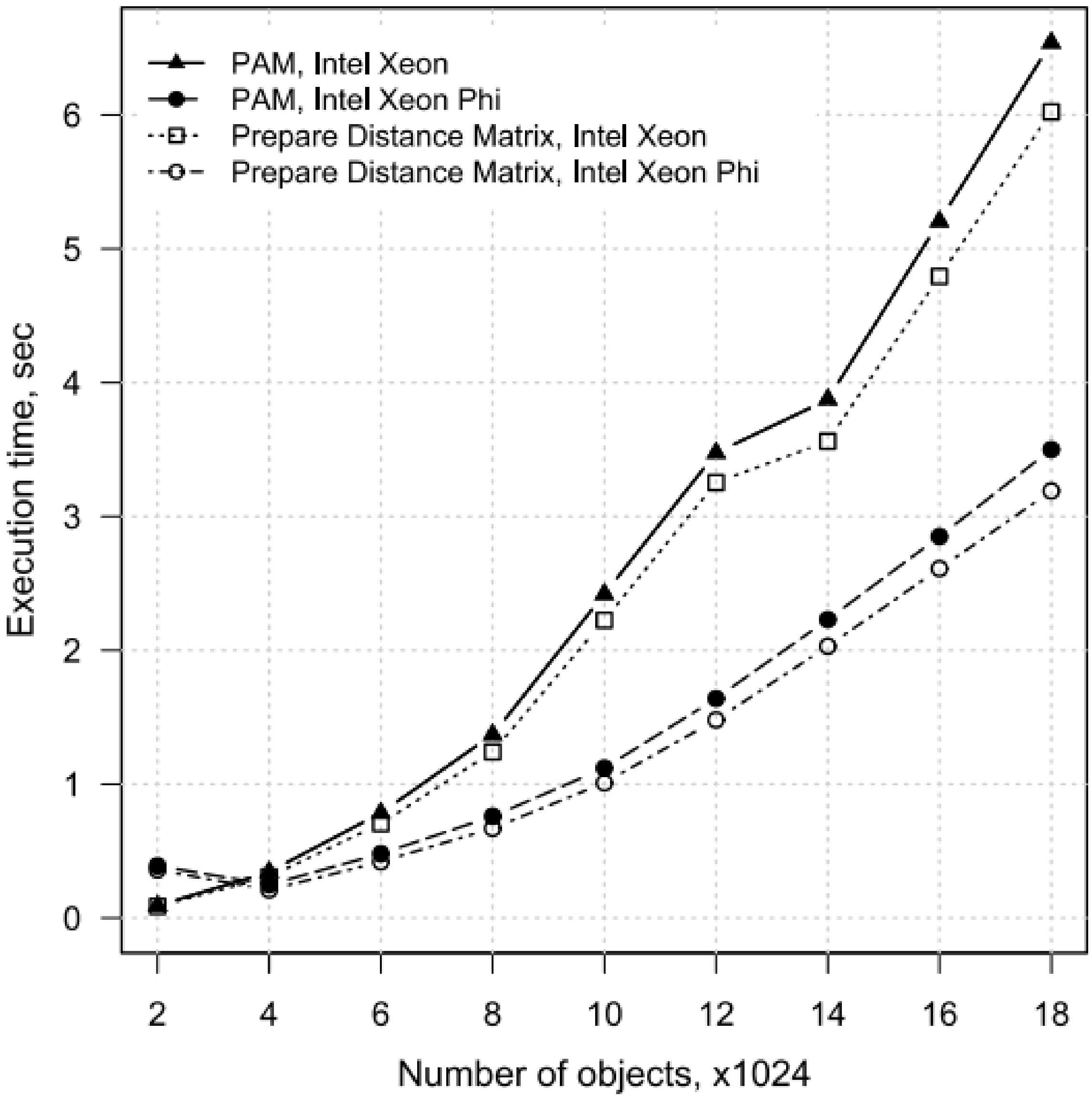
- Hardware
  - Intel Xeon Phi 60 cores
  - Intel Xeon 12 cores
- Parameters
  - Data type: float
  - Intel Xeon Phi mode: offload
- Purpose
  - Compare work time of PAM algorithm on CPU and Intel Xeon Phi

Dataset properties

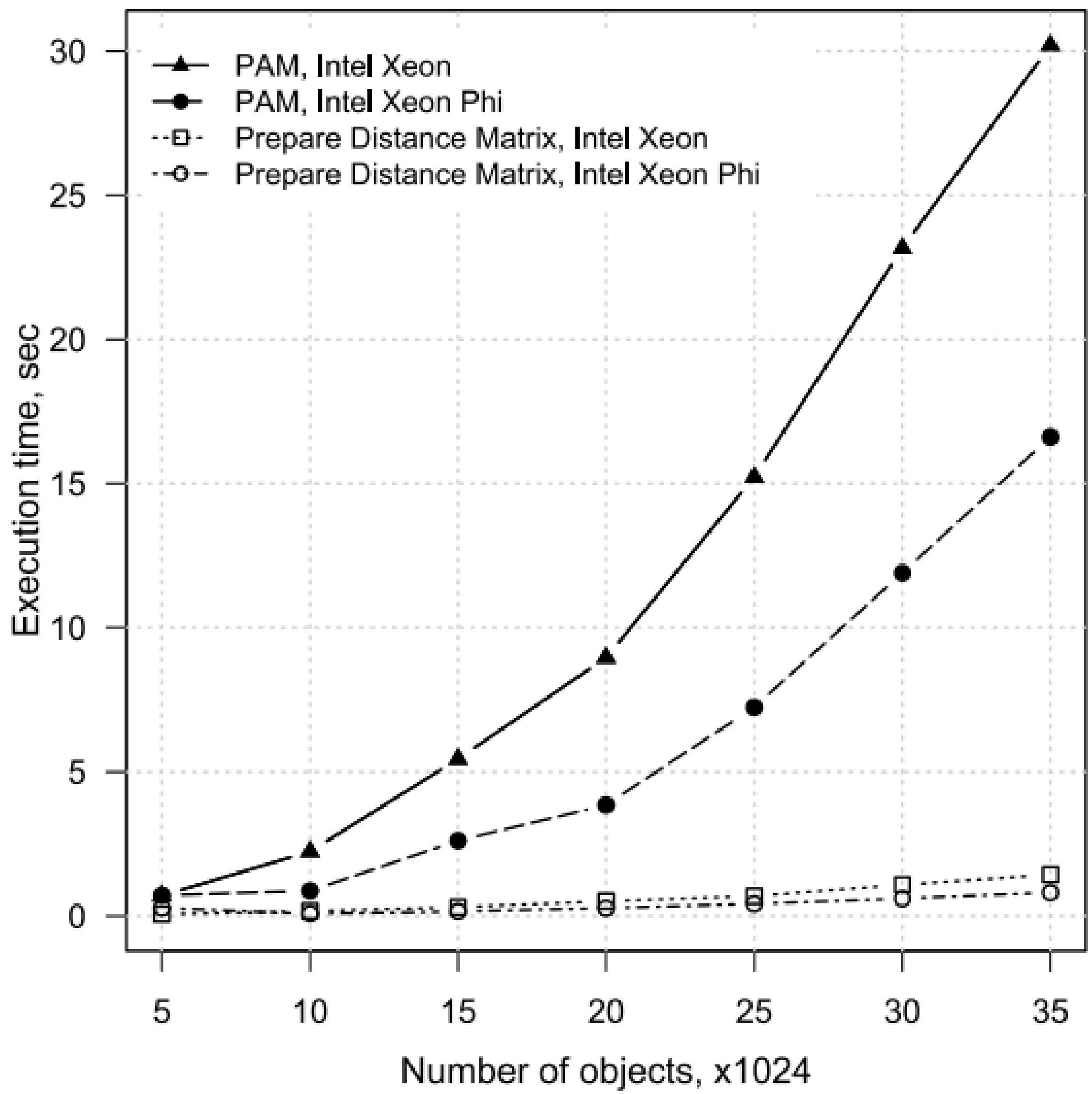
Dataset	p	k	$n, \times 2^{10}$	
			min	max
FCS Human	423	10	2	18
Corel Image Histogram	32	10	5	35

- p – size of real-valued tuple which describes clustering object
- k – the number of clusters
- n – the number of clustering objects

FCS Human evaluation



Corel Image Histogram evaluation



Conclusion

- The paper has described a parallel version of Partitioning Around Medoids clustering algorithm for the Intel Xeon Phi many-core coprocessor
  – OpenMP
  – Vectorization
  – Tiling
- Experimental results show effectiveness of suggested approach
- Experiments show that PAM performance depends on clustered data nature