PCM- oMaRS Algorithm: Parallel computation of Median- Omniscient Maximal Reduction Steps

Ammar Suhail Balouch[1]

Associate Prof. Dr. Eng.
Database Research Group
Department of Computer Science
University of Rostock

Abstract: The goal of a distributed computation algorithm is to determine the result of a function of numerical elements, which are distributed in n multi sets.It is known that computation of holistic aggregation functions on distributed multi sets indeed requires more work than non holistic aggregation functions. But with this article we will prove that the computation of a holistic function, which named exact median, can be computed efficiently by providing both a candidate finding and a deterministic location algorithms which computes the position of exact median, dispelling the misconception that solving distributed median computation through parallel aggregation is infeasible. Some of most important part in Big Data field is to evaluate massive data values. A special case in this field is the calculation of k^{th} smallest values (specially the median) of distributed multi sets containing enormous data. Many approximation algorithms and algorithms with iterative or recursive steps of determination of median give solutions for the computation of median. But firstly sometime approximate value is dangerous for some data evaluation projects or researchs and secondly with other algorithms, the data blocking time is too long through the iteration or the recursionbetweenglobal node and local nodes. This article focuses on a solution that gives a best effectively computation for this problem named PCM-oMaRS algorithm. The PCM- oMaRS algorithm guarantees the maximal reduction steps of the computation of the exact median in distributed multi sets and proves that we can compute the exact median effectively without needing the usage of recursive or iterative methods at the global communication level, which reducesthe blocking timemaximally. This algorithm provides more efficient execution not only in distributed multi sets even in local multi set with enormous data.

Keywords: Median, Parallel Computation, Algorithm, Big Data, Evaluation, Analysis, Distributed Multi sets

I. INTRODUCTION

The parallel execution of aggregations is an interesting area. It be growing recently and plays important roles, thanks emerging application fields such as, e.g. data mining or sensor networks. The goal of distributed aggregation is to compute an aggregation function on a multi set of distributed values.Typical aggregation functions are max, sum, count, average, median, variance, k[th] smallest, or largest values (median is a special case of k[th] smallest selection with $k = \frac{n}{2}$), or combinations thereof.

Depending on the database community classifications there are three categories of aggregation functions: distributive (max, min, sum, count …), algebraic (plus, minus, average, variance …), and holistic (median, k[th] smallest or largest value …). Combinations of these functions are believed to support a wide range of reasonable aggregation queries. The distributed computations of holistic functions are the most problematic cases, in especially of distributed enormous data sets. Such data is like streaming data from network-sensors.

In this article we shed a new light on the problem of distributed computation of exact median for general *n* distributed multi sets. In particular, we prove that the median can be calculate in more effective steps than the previous algorithms. Actually with our computation of the median in distributed multi sets we dare to say that this computation is no longer referred to as iterative or recursive strategy at the global level. To the best of our knowledge, there is no other algorithm solve the problem of computation of the exact median in distributed multi sets with steps without usage of iterations or recursions that have to do with the communications between the nodes of networks. Therefore we present a novel algorithm PCM- oMaRS that solves this problem without such step and blocks determinate data only in one time by one step, actually by the last step of our algorithm, and if it is necessary.

[1] *Corresponding Author: ammar.balouch2@uni-rostock.de*

This algorithm consists of three major phases and depends on mathematical definition of median. We prove too that this algorithm is more efficient executed at both the global with distributed multi sets and local level with multi set containging enormous numeric values.

This article is organized as follows: In section 2 we list related works with a short summary. PCM- oMaRS algorithm is presented in section 3. The distributed mathematical definition of computation of median and of PCM-oMaRS algorithm are to find in section 4 and at the end we summarize the most important points of the article in section 5.

II. RELATED WORK

Actually the research of distributed algorithm to determine the median is for more than 40 years active. This problem has attracted many researchers. Blum et al. [1] developed in 1973 the first linear algorithm to find the median. In 1976 developed Schönhage et al. [2] another algorithm which performs fewer comparisons on the worst case.

Rodeh [3] in 1982 considered on the problem of computing the median ofa bag of $2n$ numbers by using communicating processes, each having some of the numbers in its local memory. The memories are assumed to be disjoint. This algorithm described the distibutive median problem as a seried of transformations. Its algorithm is optiomal up to a constant. Marber et al. [4] in 1985 considered the problem of selecting the k'th largest element in a set of n elements distributed arbitrarily among the processors of a Shout-Echo network. In 1987 developed Chin et al.

[5] an improved algorithm for finding the median distributively. He embedded in the first part of its algorithm the Rodeh's algorithm, in the second part of its algorithm reduced the problem size by one qurter with three messages instead of reducing the problem size by half with two messages. With the third part of its algorithm resloved the problem of choosing the initiator. Santoro et al. [6] minimized with its algorithm the communication activities among the processors and considered the distributed k-selection problem.

[7] shows the existence of small core-sets for the problems of computing k-median and k-means clustering for points in low dimension and get an (1+ε)-approximation. Kuhn et al. in 2008 [8] presented a k-selection algorithm and proved that distibuted selection indeed requires more work than other aggregation function. In this article has shown that the k^{th} smallest element can be computed efficiently by providing both a randomized and a deterministic k-selection algorithm.

[9] considered the k-median clustering on stream data arriving at distributed sites which communicate through a routing tree.They proposed a suite of algorithms for computing (1+ε)-approximation k-median clustering over distributed data streams. The algorithms are able to reduce the data transmission to a small fraction of the orginal data. In [10], the author shows a unified framework for constructing core-sets and approximate clustering for such general sets of functions. In [11] has shown an algorithm that produces (1+ε)α- approximation, using any α-approximation non-distributed algorithm as a subroutine, with total communication cost.

There are many others works in this field, [12] [13] [14] [15] [16]. All of these researches have used the iteration, recursion or approximation in their steps. For these works we did not have different between approximation and non approximation algorithms. We consider us in our algorithm on avoiding the iteration and recursion steps for computing the exact median (Non approximation).

III. PCM- OMARS ALGORITHM

In this section we illustrate firstly the mechanism of PCM- oMaRS algorithm, secondly present its two sub algorithms, Candidate-Finding and Exact-Median-Computing algorithms and thirdly give an abstract of its cases.PCM- oMaRS algorithm computes the exact median of distributed multi sets without usage of iterative or recursive steps. This algorithm calculates a temorary median, then computes the position of the exact median at least getting the value of exact median from the node in the computed position. An abstract of this idea is illustrated in the following.

3.1. Illustration of PCM- oMaRS Algorithm Mechanism

Figure 1 shows an abstract of the mechanism of our algorithmcleared by one sequence. In this figure we can see that the finding of position of exact median depending on the value of sclenD and of sgenD in which will be known to which direction must the position be moved from the position of the temporary median to achieve the position of exact median.

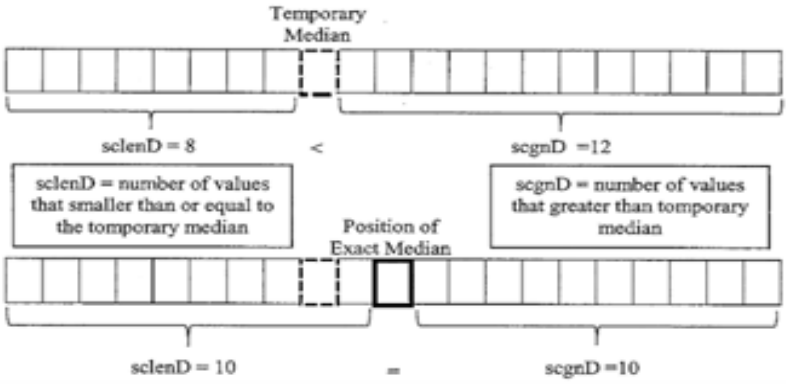

Fig1. *Abstract of PCM- oMaRS algorithm mechanism*

3.2. Algorithm I: Candidate-Finding Algorithm

Instead to apply a randomized algorithm, we use this algorithm to select the temporary median. The temporary median in this algorithm is the median of all minimum, maximum and median values of all multi sets.

1- For each *Di* in *Pi*;*i=1,…,n*

Get ***MinDi***, ***MaxDi***, ***MedDi***;

2- Ordering all ***MinDi***, ***MaxDi, MedDi***

Ord(***MinDi***, ***MaxDi, MedDi***);

3- Calculate **MedT**:

Median of Ord(***MinDi***, ***MaxDi, MedDi***);

If the size of the sequence*Ord*is an EVEN number then the temorary median value is the first/second value of the two middle values of *Ord* instead to compute the average of both middle values. That means,$MedT$ must stay one of the existing values of all multi datasets and not a new calculated value.By step 1 we require the minimum and maximum values of each data set because with these values we can achieve better assessment of a temporary median, which we need to compute the exact median in the following steps and plays an important role with the necessary amount of caclucation values. Now we continue with the Exact-Median-Computing Algorithm.

3.3. Algorithm II: Exact-Median-Computing Algorithm

After getting a temporary median, we start to determine the position of the exact median. Forthat the PCM- oMaRS algorithms send for all multi sets to send back:

- Number of values that greater than the temporary median$MedT$
- Number of values that smaller or equal than the temporary median$MedT$

The algorithm summerizes these values in two parameters **scgnD** and **sclenD** respectivelly.

4- For each Di do ;i=*1,…,n*:

Calculate **cgnDi**:
Count all numbers > MedT;
Calculate **scgnD**:
sum (cgnDi);
Calculate **clenDi**:
Count all numbers ≤ MedT
Calculate **sclenD**:
sum (clenDi)-1;

Now we determine the position of the exact median of all values in all distributed multi sets.

5- **If** scgnD < sclenD **then**

Calculate **MedLP** = ((sclenD - scgnD)/2);

Move position of Median **to left in MedLP** Postions;

Get **maximum MedLPlargest** numbers from each Di **smaller or equal than** MedT, differencing{MedT};

Ordering these values **descending** in **LtD;**

Get **MedE** = the number in LtD[**MedLP**];

else If scgnD > sclenD **then**

Calculate **MedRP** = (scgnD – sclenD)/2;

Move position of Median **to rightMedRP** Postions;

Get **maximum MedRP smallest** numbers From each Di **greater than** MedT;

Ordering these **ascending** in **GtD**;

Get **MedE** = the number in GtD[**MedRP**];

elseMedE = **MedT**;

In this step we can find that the best case is if $scgnD = sclenD$, we don't need to do anything more in this case. But in other cases in this step we reached too the maximum reduction of computation steps and maximal reduction of values, which are needed to calculate the exact median through calculating the position MedLP / MedRP and then getting maximum MedLP largest / MedRP smallest numbers smaller / greater than the temporary median, only here in this time point the blocking of required data will be executed, when it is necessary.

PCM- oMaRS algorithm is concerned many cases. An abstract of these cases is represented in the following section.

3.4. PCM-oMaRS Cases Map

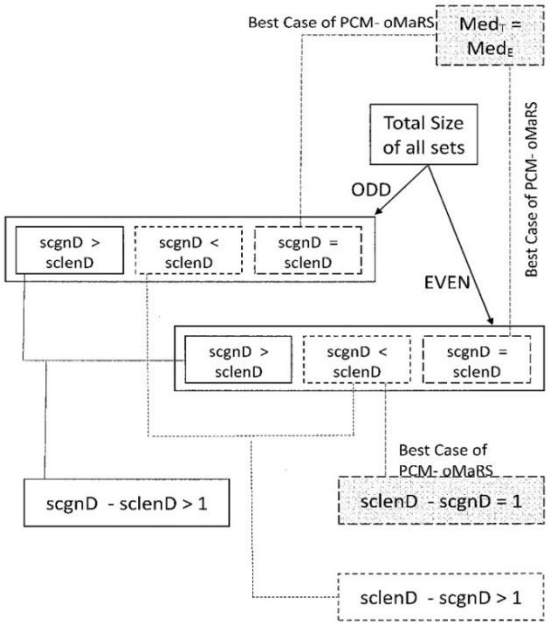

Fig2. Abstract map of PCM-oMaRS cases

In this figure, it was clarified that the best case of our algorithm is executable by the case that the number of values that greater than the temporary median equal to the number of values that smaller\equal than\to the temporary median for both cases of total size of all multi sets. If the toal size of all mulit sets is an even number and the

difference equal to 1 then this case is too a best case of PCM- oMaRS algorithm because in this case the temporary median is one of the two middle values of exact median.In these cases, the PCM-oMaRS algorithm does not need more to apply algorithm II completely, because the temporary median in this case is the required exact median. In other words that means, after computing the position of the exact median we do not need to apply any operation anymore to achieve the required result. We are working on increasing the cases of best case in the framework of the optimization of our algorithm.

IV. MATHEMATICAL FORM OFPCM- OMARS

Before we present the mathematical definition of PCM- oMaRS algorithm, we give firstly the definition of a median of a multi set D and secondly continue with definition of a median of distributed multi sets depending of the idea of PCM- oMaRS.

4.1. Median of a Multi Set

Let $D = (x_1, x_2, \dots, x_n, \dots, x_m)$ be a sorted multi set containing numerical values and A, B, C, E be sub multi sets of D. If its size is an ODD number then the definition of the median of this multi set is shown as following:

$$Med(D) = \mu$$

Where

$$\mu \in D \ \wedge \ |C \cup (A \backslash \{\mu\})| = |B| \, ;$$

$$B = \{x_j \, | x_j \in D \ \wedge x_j > \mu \, ; j = 1, \dots, m \} \ \wedge$$

$$\wedge A = \ \{x_i \, | x_i \in D \ \wedge x_i \leq \mu \, ; i = 1, \dots, m\} \wedge$$

$$\wedge C = (A(\mu) = |\{x_l \mid x_l \in D \ \wedge x_l = \mu \, ; l = 1, \dots, m\}| - 1)$$

Note:$A(\mu) = n$ gives us how many duplicat (n) of μ allowed in the set A. E.g. $C = (A(5) = 3) = \{5,5,5\}$.

If its size is an EVEN number then the definition of the median of this multi set is shown as following:

$$Med(D) = \frac{\mu_1 + \mu_2}{2}$$

Where

$$\mu_1, \mu_2 \in D \wedge \ |C \cup (A \backslash \{\mu_1\})| = |E \cup (B \backslash \{\mu_2\})| \, ;$$
$$A = \ \{x_i \, | x_i \in D \ \wedge x_i > \mu \, ; i = 1, \dots, m\} \wedge$$

$$\wedge C = (A(\mu_1) = |\{x_l \, | x_l \in D \ \wedge \ x_l = \mu_1 \, ; l = 1, \dots, m\}| - 1) \wedge$$

$$\wedge B = \{x_j \, | x_j \in D \ \wedge x_j \geq \mu_2 \, ; j = 1, \dots, m \} \ \wedge$$

$$\wedge E = (B(\mu_2) = |\{x_k \, | x_k \in D \ \wedge \ x_k = \mu_2 \, ; k = 1, \dots, m\}| - 1)$$

4.2. Median of Multi Sets

Now let us distribute values in different ordered multi sets, as following:

$$D_i = \left(x_1^i, x_2^i, \dots, x_n^i, \dots, x_m^i\right); i = 1, \dots, p$$

If the size of all distributed multi sets is an ODD number then the **exact median Med_E** of these distributed multi sets is defined as following:

$$Med_E = \left\{ \mu \mid \mu \in \left\{ \bigcup_{i=1}^{p} D_i \right\} \wedge |C \cup (LS \backslash \{\mu\})| = |RS| \right\}$$

Where LS is the multiset of all values of each distributed multi sets smaller or equal than the median

$$LS = lsD_1 \cup lsD_2 \cup \dots \cup lsD_p$$

$$C = (LS(\mu) = |A| - 1);$$

$$A = \left\{ x_l \mid x_l \in \left\{ \bigcup_{i=1}^{p} D_i \right\} \wedge x_l = \mu \, ; l = 1, \dots, m \right\}$$

Where

$$lsD_i = \left\{ x_j^i \mid x_j^i \in D_i \wedge x_j^i \leq \mu \wedge j = 1, \dots, m \right\};\ i = 1, \dots, p$$

And RS is the set of all values greater than the median

$$RS = rsD_1 \cup rsD_2 \cup \dots \cup rsD_p$$

Where

$$rsD_i = \left\{ x_k^i \mid x_k^i \in D_i \wedge x_k^i > \mu \wedge k = 1, \dots, m \right\};\ i = 1, \dots, p$$

Then the size of LS and RS is defined as following

$$|C \cup (LS \backslash \{\mu\})| = |lsD_1| + |lsD_2| + \cdots + \left|lsD_p\right| - 1$$

$$|rs| = |rsD_1| + |rsD_2| + \cdots + \left|rsD_p\right|$$

Respectively. It is simple now to find that if $RS > LS$ then

$$|RS| = \sum_{i=1}^{p} |D_i| - |C \cup (LS \backslash \{\mu\})|$$

And if $RS < LS$ then

$$|C \cup (LS \backslash \{\mu\})| = \sum_{i=1}^{p} |D_i| - |RS|$$

Let $n_{rs} = |RS|$and $n_{ls} = |C \cup (LS \backslash \{\mu\})|$ for further use.

Analogously for the mathematical definition of median of a multi set in case the size is an EVEN number, we can form the mathematical defintion of median of distributed multi sets for this case.

4.3. Mathematical Definition of PCM- oMaRS

Now to show that we can calculate the exact median with PCM-oMaRS algorithm depending on the computation of the position of exact median after finding a temporary one, we give the following mathematical proof and then we clear it by an example:

Let us have n multi sets. In the first step we have to get a temporary medina$\boldsymbol{Med_T}$ after orering each multi set local.$\boldsymbol{Med_T}$is the median of Ord-set in which contains ordering all Minimal, Maximal and Median values of each Multi set.

$$MinMaxMedD_i = \{Min_i, Max_i, Med_i | i = 1, \dots, P\}$$

Then we sort all these values in one multi set Ord.

$$Ord = \left(\bigcup_{i=1}^{p} MinMaxMedD_i \right)$$

We make the median of this sorted multi set as our temporary median $\boldsymbol{Med_T}$which locatedat the $\left(\frac{(3*P)+1}{2}\right)$th position if $(3 * P)$ is an Odd number or located at the two $\left(\frac{(3*P)}{2}\right), \left(\frac{(3*P)+2}{2}\right)$th positions if $(3 * P)$ is Even. For the last case we mean,if the size of Ord is an even number, instead to calculate the average of two middle values, we take the first or second value of the both middle values and let it be$\boldsymbol{Med_T}$**.** That means, always$Med_T \in \left\{ \bigcup_{i=1}^{p} D_i \right\}$

Now we get $\boldsymbol{LS}$ and $\boldsymbol{RS}$ depending on $\boldsymbol{Med_T}$ and calculate the size n_{ls} and n_{rs} of both multi sets $\boldsymbol{LS}$ and $\boldsymbol{RS}$ respectively as defined in section 4.2.

In the following we difference between three cases depending on$\boldsymbol{Med_T}$

The first case is if $\boldsymbol{n_{ls} < n_{rs}}$then we calculate the position P of exact median of all multi sets.

$$P = \frac{(n_{rs} - n_{ls})}{2}$$

This value tells us that the exact median exists in the right side (right side is a sequence contains values greater than the temporary median) of temporary median in $\boldsymbol{P}$ position, because the value of $\boldsymbol{n_{rs}}$ greater than the other one.

Very important advantage: Until this step of PCM- oMaRS, the algorithm does not need to block any change of the data in the local nodes. All transactions and changes of local data can be continued performed without any disadvantages of the efficiency of PCM- oMaRS algorithm.

To get the exact median we must get now **maximum Psmallest values greater than** $\boldsymbol{Med_T}$ from each sorted multi set and order them in GtD

$$GtD = \left(sPvg_1 \cup sPvg_2 \cup \ldots \cup sPvg_p\right)$$

So that

$$sPvg_i = \begin{pmatrix} x_j^i \mid x_j^i \in D_i \wedge x_j^i > Med_T \\ \wedge\, j = 1, \ldots, \mathrm{P} \end{pmatrix}; i = 1, \ldots, p$$

Important Note: Transactions and changes of data of local nodes are blocked only in this step.

Now get the exact median$Med_E$from theGtD

$$Med_E = GtD[P]$$

In the second case, we study if $\boldsymbol{n_{ls} > n_{rs}}$ (This case process is analogously for the first one) then we calculate the position P of exact median of all multi sets.

$$P = \frac{(n_{ls} - n_{rs})}{2}$$

From this value we know that the exact median exists in the left side (side contains the values smaller than or equal the temporary median) of temporary median in $\boldsymbol{P}$ position, because the value of $\boldsymbol{n_{ls}}$ greater than the other one.

Get **maximum Pgreatest values smaller than** $\boldsymbol{Med_T}$ from each sorted multi set and order them in LtD

$$LtD = \left(gPvs_1 \cup gPvs_2 \cup \ldots \cup gPvs_p\right)$$

So that

$$gPvs_i = \begin{pmatrix} x_j^i \mid x_j^i \in D_i \wedge x_j^i \leq Med_T \\ \wedge\, j = 1, \ldots, \mathrm{P} \end{pmatrix}; i = 1, \ldots, p$$

We get the exact median from the LtD

$$Med_E = LtD[P]$$

The last case is the simplest case and the best case because if $\boldsymbol{n_{ls} = n_{rs}}$then the temporary median is the exact median$Med_E = Med_T$.

Finally we get the exact median for all values in all distributed multi sets and it is $\boldsymbol{Med_E}$. This prove is correct for the both case when the total size is odd or even, because the difference is in the computation's step of both temporary median and the position.

The PCM- oMaRS algorithm consists actually of two algorithms named **candidate-Finding** and **Exact-Median-Computing** Algorithms. The most important two steps in this algorithm are the computation of the position of exact median and its value (two values) after the finding a candidate median.

Before we present an example to clarify this mathematical definition of our algorithm, we give a proof of the validation of our algorithm.

4.4. Validation of PCM- oMaRS

To prove that the definition of exact median of distributed multi sets is correct, show the following:

We definea multi set$\boldsymbol{\rho}$which containing the first $\boldsymbol{P-1}$ elements of GtD and sort it. Then we see

$$n_{\boldsymbol{\rho}} = |\rho| = P - 1$$

Now let we have the new $RS'$and LS'**depending on** $\boldsymbol{Med_E}$and$\boldsymbol{Med_T}$, so that

$$RS' = RS \setminus \{\rho \cup \{Med_E\}\}$$
$$LS' = C \cup (LS\setminus\{\mu\}) \cup \rho \cup \{Med_T\}$$

Then

$$n_{r\acute{s}} = n_{rs} - (n_\rho + 1) = n_{rs} - ((P-1)+1)$$
$$n_{l\acute{s}} = n_{ls} + n_\rho + 1 = n_{ls} + (P-1) + 1$$

In the following we give mathematical forms to prove that the relationship$P = \frac{(n_{rs} - n_{ls})}{2}$guaranteed this validation.

$$P = \frac{(n_{rs} - n_{ls})}{2} \Rightarrow 2P = n_{rs} - n_{ls}$$
$$(P-1) + (P-1) = n_{rs} - 1 - n_{ls} - 1$$
$$n_{ls} + 1 + (P-1) = n_{rs} - 1 - (P-1)$$
$$n_{ls} + (P-1) + 1 = n_{rs} - ((P-1)+1) \Rightarrow n_{ls'} = n_{rs'}$$

The Definition of exact median of distributed multi sets is valid. These expressions are valid for the fist case in section 4.3 ($\boldsymbol{n_{ls} < n_{rs}}$). Analogically for the first case are steps calculated for the second one.

Getting the mulit set $\boldsymbol{\rho}$**,** containing the first $\boldsymbol{P-1}$ elements of LtD, so that$n_{\boldsymbol{\rho}} = P - 1$

Now we have the new $LS'$and RS'**depending on** $\boldsymbol{Med_E}$and$\boldsymbol{Med_T}$, so that

$$LS' = C \cup (LS\setminus\{\mu\}) \setminus \{\rho \cup \{Med_E\}\}$$
$$RS' = RS \cup \rho \cup \{Med_T\}$$

Then

$$n_{ls'} = n_{ls} - (\rho + 1) = n_{ls} - ((P-1)+1)$$
$$n_{rs'} = n_{rs} + n_\rho + 1 = n_{rs} + (P-1) + 1$$

We know that

$$P = \frac{(n_{ls} - n_{rs})}{2} \Rightarrow 2P = n_{ls} - n_{rs}$$
$$(P-1) + (P-1) = n_{ls} - 1 - n_{rs} - 1$$
$$n_{rs} + 1 + (P-1) = n_{ls} - 1 - (P-1)$$
$$n_{rs} + (P-1) + 1 = n_{ls} - ((P-1)+1) \Rightarrow n_{rs'} = n_{ls'}$$

The Definition of Median is in this case valid too.

4.5. PCM- oMaRS Example

Let us have the following three multi sets after ordering:

$$D_1 = (2,7,10,13,15),$$
$$D_2 = (13,14,15,16,20,21,25),$$
$$D_3 = (2,4,11,16,70)$$

The minimal, maximal and median values for each sorted multi set is as following:

$$MinMaxMedD_1 = \{2,15,10\}$$

$$MinMaxMedD_2 = \{13,25,16\}$$

$$MinMaxMedD_3 = \{2,70,11\}$$

Then the ordered set of them is

$$Ord = (2,2,10,11,13,15,16,25,70)$$

And the median of Ord is the temporary median $\boldsymbol{Med_T}$ and equal to 13.

$$Med_T = 13$$

Now we get $\boldsymbol{LS}$ and $\boldsymbol{RS}$ depending on $\boldsymbol{Med_T}$

$$RS = rsD_1 \cup rsD_2 \cup rsD_3$$

$$rsD_1 = \{15\}, rsD_2 = \{14,15,16,20,21,25\}, rsD_3 = \{16,70\}$$

$$|rsD_1| = 1,\ |rsD_2| = 6,\ |rsD_3| = 2$$

$$\rightarrow n_{rs} = |rs| = 1 + 6 + 2 = 9$$

$$LS = lsD_1 \cup lsD_2 \cup lsD_3;$$

$$lsD_1 = \{2,7,10,13\},\ lsD_2 = \{13\},\ lsD_3 = \{2,4,11\}$$

$$|lsD_1| = 4,\ |lsD_2| = 1,\ |lsD_3| = 3$$

$$\rightarrow LS = \{2,7,10,13,13,2,4,11\}$$

But we have that

$$LS\backslash\{\mu\} = LS\backslash\{13\} = \{2,7,10,2,4,11\}$$

$$C = (LS(\mu) = 2 - 1) = (LS(13) = 1) = \{13\}$$

That means, the value 13 should be contained in the multi set LS only one time.

$$C \cup (LS\backslash\{\mu\}) = \{2,7,10,2,4,11,13\}$$

$$n_{ls} = |C \cup (LS\backslash\{\mu\})| = 7 = 4 + 1 + 3 - 1$$

Now we get that $n_{rs} = 9$ and $n_{ls} = 7$and we see that this case meets the first case in section 4.3 then we have to calculate the position P as following:

$$P = \frac{(n_{rs} - n_{ls})}{2} = \frac{(9 - 7)}{2} = 1$$

That means the exact median exists in the right side of the $\boldsymbol{Med_T}$ in 1 position.

Now we have to get smallest 1 values of each Multi set greater than 13 (Getting GtD)

$$sPvg_1 = (15),\ \ sPvg_2 = (14),\ \ sPvg_3 = (16)$$

$$\Rightarrow GtD = (14,15,16)$$

Then the exact median is

$$Med_E = GtD[1] = 14$$

Note: In step of choosing a temporary median we may remark here, if we have calculated the median of medians, we would have taken the value 11 as temporary median. This makes more effort into the calculation of the position of exact median. It would have required maximum 3 smallest values of each Multi set greater than11. That means the computation of temporary median depending on maximal, minimal and meidan values give better estimation than the median of medians technology. Now we continue with the validation of median definition for our example:

$$n_{\boldsymbol{\rho}} = |\rho| = 1 - 1 = 0$$

$$\rho = \emptyset\ ,\ \ n_\rho = 0$$

For our example:

$$R\acute{S} = \{14,15,15,16,16,20,21,25,70\} \setminus \{\emptyset \cup \{14\}\} = \{15,15,16,16,20,21,25,70\}$$

$$L\acute{S} = \{2,2,4,7,10,11,13\} \cup \emptyset \cup \{13\} = \{2,2,4,7,10,11,13,13\}$$

For our example:

$$n_{r\acute{s}} = 9 - 1 = 8\,, \qquad n_{l\acute{s}} = 7 + 1 = 8$$

That means the definition of exact median is valid in the value 14.

V. CONCLUSION

The research points of this field "median computing of distributed multi sets" divided into two main directions. The first one cares on the approximation methods. The other one focuses on the computation of the exact median with usage of iterative or recursive steps. We have shown that we can compute the exact median with clever steps depending on the calculation of the position of the exact median without needing to apply iterations or recursions at global level depending on communications to get the value of the exact median. That means, PCM-oMaRS algorithm guarantees the maximum reduction of median computation steps. Too, instead applying blocking of the required data by the beginning an execution of an algorithm, the data may be blocked only in one non iterative or recursive step with the execution of our algorithm and if it is necessary.

In this article we would not to go on the cost of our algorithm, but we would to say that the most computation of our algorithm is calculated in the local nodes (computers), basic operations and operation with effcient complexity will be executed in the master computer (global one). That means in other word, the costs of complexity of our algorithm is computed through the common communication costs and local execution costs like all other algorithms in addition only the cost of an efficent sort algorithm in step 5. In our experiments we have proved that the execution of our algorithm can be more effective in the local execution too, if we divided the local multi set that contains enourmuous values in many local multi sets.

We have implemented this algorithm by C++ with two different input possibilities. The first one is with manually targeted inputs to test extreme cases of values distributations and the other one is random inputs to be able to check all possible cases with the passage of time. We have tested the implementation of our algorithm with more than 40000 cases, some of these depended on the manually targeted inputs and the rest were in relation to the random inputs. In each case, the number of multi sets is different, and each multi set includes many different values. Currently we are working at the optimization of this algorithm and its implementation, then we will calculate the cost of the complexity of PCM- oMaRS algorithm in details.

VI. ACKNOWLEDGMENT

We would like to thank our colleague in Database and information systems department for their notes. Moreover, we would like to express our gratitude to Mr. Prof. Riedewald, Mr. Prof. Van Bang Le and Mr. Math. Soheil Baloush for providing valuable feedback, which helped us greatly to improve the mathematical definition of our algorithm. At the same time I thank the IIE-SRF organization for its support scientists at risik and its scholarships.

REFERENCES

[1] Blum, M.; Floyd, R; Pratt, V.; Rivest, R.; Tarjan, R.: Time bounds for selection. In J. Comput. System Sci 7. 1973; S. 448-461.

[2] Schönhage, A.; Paterson, M.; Pippenger, N.: Finding the median. In J. Comput. System Sci 13. 1976; S. 184-199.

[3] Rodeh, M.: Finding the median distibutively. In J. Comput. System Sci. 24. 1982; S. 162-166.

[4] Marberg, J; Gafi, E.: An Optimal Shout-Echo Algorithm for selection in distributed Sets. In CSD-850015. 1985.

[5] Chin, F.; Ting, H.: An Improved Algorithm for Finding the median distributively. In Algorithmica 2. 1987; S. 235-249.

[6] Santoro, N.; Sidney, J. ; Sidney, S.: A distributed selection algorithm and its expected communication complexity. In Theoretical Computer Science 100. 1992; S. 185-204.

[7] Har-Peled, S.; Mazumdar, S.: On coresets for k-means and k-median clustering. In Proceedings of the Annual ACM Symposium on Theory of Computing. 2004.

[8] Zhang, Q.; Liu, J.; Wang, W. : Approximate clustering on distributed data streams. In Proceedings of the IEEE International Conference on Data Engineering. 2008. S. 1131 – 1139.

[9] Kuhn, F.; Locher, T. ; Wattenhofer, R.: Distributed Selection: A missing Piece of Data Aggregation. In Communications of The ACM 51. 2008; S. 93-99.

[10] Feldman, D.; Langberg, M. : A unified framework for approximating and clustering data. In Proceedings of the Annual ACM Symposium on Theory of Computing. 2011.

[11] Yingyu L.: Distributed k-median/k-means Clustering on General Topologies. In NIPS 2013. 2013.W.-K. Chen, *Linear Networks and Systems*. Belmont, Calif.: Wadsworth, pp. 123-135, 1993. (Book style)

[12] Jia, L.; Lin, G.; Noubir, G.; Rajaraman, R.; Sundaram, R.: Universal approximations for TSP, Steiner Tree and set cover. In 37th Annoul ACM Symposium on Theory of Computing (STOC).2005; S. 386-395.

[13] Negro, A.; Samtoro, N.; Urrutia, J.: Efficient distributed selection with bounded messages. In IEEE Transactions of Parallel and Distributed Systems.8 (4). 1997. S. 397-401.

[14] Peleg, D.: Distributed Computing: A Locality-Sensitive Approach. In SIAM Monographs on Dicrete Mathematics and Applications. 2000.

[15] Santoro, N.; Scheutzow, M.; Sidney, J. B.: On the expected complexity of distributed selection. In Journal of Parallel and distributed Computing 5 (2). 1988. S. 194-203.

[16] Yao, Y.; Gehrke, J.: The Cougar approach o in-network query processing in sensor network. In ACM SIGMOD Record, 31 (3). 2002. S. 9-18