Applied Intelligence
https://doi.org/10.1007/s10489-021-02253-1

MapReduce based parallel fuzzy-rough attribute reduction using discernibility matrix

Pandu Sowkuntla[1] · **P. S. V. S. Sai Prasad**[1]

Accepted: 31 January 2021

Abstract

Fuzzy-rough set theory is an efficient method for attribute reduction. It can effectively handle the imprecision and uncertainty of the data in the attribute reduction. Despite its efficacy, current approaches to fuzzy-rough attribute reduction are not efficient for the processing of large data sets due to the requirement of higher space complexities. A limited number of accelerators and parallel/distributed approaches have been proposed for fuzzy-rough attribute reduction in large data sets. However, all of these approaches are dependency measure based methods in which fuzzy similarity matrices are used for performing attribute reduction. Alternative discernibility matrix based attribute reduction methods are found to have less space requirements and more amicable to parallelization in building parallel/distributed algorithms. This paper therefore introduces a fuzzy discernibility matrix-based attribute reduction accelerator (DARA) to accelerate the attribute reduction. DARA is used to build a sequential approach and the corresponding parallel/distributed approach for attribute reduction in large data sets. The proposed approaches are compared to the existing state-of-the-art approaches with a systematic experimental analysis to assess computational efficiency. The experimental study, along with theoretical validation, shows that the proposed approaches are effective and perform better than the current approaches.

Keywords Accelerator · Apache spark · Attribute reduction · Discernibility matrix · Fuzzy-rough sets · MapReduce

1 Introduction

In 1982, Z. Pawlak [1] introduced the rough set theory as a soft computing model. Since then it has been used as an effective method for feature subset selection or attribute reduction (also known as reduct computation). Feature subset selection is a widely used preprocessing step for machine learning, data mining, and pattern recognition [2, 3]. Classical rough set model uses crisp equivalence classes in attribute reduction. As a consequence, it is only applicable to perform the attribute reduction in categorical data sets. For attribute reduction in numeric data sets, this classical model requires the discretization of numerical attributes. Discretization, however, causes a major loss of information [4, 5]. Therefore, various fuzzy-rough set models [6–9] have been proposed to handle different types of attributes. Dubois and Prade [6] proposed a fuzzy-rough set model and the generalised version of this model is given by Radzikowska et al. in [7]. These fuzzy-rough set models overcome the limitations of classical rough sets and are applicable to numerical data [3, 10, 11].

In classical rough set theory, reduct computation is primarily done in two approaches: *dependency measure* approach and *discernibility matrix* approach [12]. Using these methods, several attribute reduction algorithms are developed [13–19]. These methods are generalised to the fuzzy-rough set model, and a number of fuzzy-rough attribute reduction algorithms [3, 5, 10, 11, 20–25] are proposed for numerical data. Some of these existing classical and fuzzy-rough attribute reduction algorithms are integrated as *"RoughSets"* package [26] in *R system* which is a software environment for statistical computing.

In addition to the algorithms listed above, various methods have been proposed to further improve the efficiency of fuzzy-rough attribute reduction [27–31]. Sai Prasad et al. [27] improved the MQRA (Modified Quick Reduct Algorithm) [10] to IMQRA (Improved MQRA)

✉ P. S. V. S. Sai Prasad
saics@uohyd.ernet.in

Pandu Sowkuntla
pandu.sowkuntla@uohyd.ac.in

[1] School of Computer and Information Sciences, University of Hyderabad, Hyderabad-500046, Telangana, India

Published online: 24 April 2021

by incorporating a simplified computational model, and by removing the objects of absolute positive region. In [29], Jensen et al. developed the nnFRFS (nearest neighbour Fuzzy Rough Feature Selection) algorithm and the nnFDM (nearest neighbour Fuzzy Discernibility Matrix) algorithm for scalable fuzzy-rough feature selection. In nnFRFS, fuzzy-rough membership degree is determined only for the objects that are closest neighbours. Thus the number of calculations in the algorithm is greatly decreased. Similarly, in nnFDM, the matrix is constructed only for the objects that are nearest neighbours. Jinkun Chen et al. [31] developed an approach for fuzzy-rough attribute reduction based on graphs. They demonstrated in this approach that the attribute reduction is equivalent to finding the minimal traversal of a derivative hyper graph. In [28], Qian et al. developed an accelerator called forward approximation (FA-FPR) to improve the process of fuzzy-rough attribute reduction. The experimental findings have shown that the FA-FPR is much faster than its predecessors. Later, Peng Ni et al. [30] developed a positive region based attribute reduction accelerator (PARA) that outperformed FA-FPR especially for the data sets having large object space. PARA is developed by removing object pairs that have been discerned in the process of attribute reduction. Through carefully studying all these existing methods, it can be observed that each approach accelerates the attribute reduction process by ignoring or removing the objects that are no longer useful or cause redundant computations. Notice that all of these current methods are sequential approaches. Even with the accelerators, the memory requirements of attribute reduction in fuzzy-rough sets restrict the applicability to small data sets, requiring parallel/distributed solutions.

From the review of literature, it is found that there exist many parallel/ distributed approaches primarily based on the MapReduce paradigm [32] for attribute reduction using the classical rough set model to handle large categorical data sets [33–38]. However there are currently a limited number of parallel/distributed approaches available in the literature for fuzzy-rough attribute reduction in large numeric data sets [39–43]. In [39], an algorithm MR_FRDM_SBE is developed based on the discernibility matrix. This algorithm uses sequential backward elimination (SBE) strategy for reduct generation, where the reduct is initialised to all the attributes, and redundant attributes are removed one by one. Alternative to SBE strategy is sequential forward selection (SFS) where the reduct is initialised to empty set, and the attributes are incrementally added. In the literature [12], it can be found that the SFS strategy is more effective than the SBE strategy because it facilitates utilisation of accelerator in the attribute reduction. In [40], an algorithm MR_IMQRA is developed based on dependency measure approach that uses a vertical distribution of the input data to the nodes of the cluster. MR_IMQRA has been found to be more effective for larger attribute space data sets with moderate object space. In [42], Qinghua Hu et al. developed a MapReduce based multi modality attribute reduction method based on multi kernel fuzzy rough sets model. This method has a presumption that entire data set to be available at each node of the cluster, and in every partition, computation with respect to a subset of objects is involved. This assumption of availability of data set at each node could hinder the scalability of the approach. L. Kong et al. [41], developed a distributed fuzzy rough set (DFRS) method for attribute reduction in cloud computing. The DFRS methodology involves parallel computation of reducts on overlapping subsets of given data set, and union of the individual solutions obtained becomes the solution of the approach. This approach has the advantage of avoiding intermediate data transfer across the nodes but has overheads with respect to computations over overlapping subsets. In [43], W. Ding et al. developed a Multi granulation Consensus Fuzzy-Rough Attribute Reduction Algorithm (MCFR). This algorithm is capable of handling granular and structurally-complex large attributes to find the attribute reduction sets. Notice that the approaches in [39, 40], and the approach in [42] are MapReduce based approaches, and the approaches in [41, 43] are non-MapReduce based parallel/distributed approaches.

Except a few discernibility matrix based methods [29, 39], most of the scalable methods either sequential [27–30] or parallel/distributed [40–43] are developed based on the dependency measure approach. As specified in [12, 28, 29], in the dependency measure based approach of attribute reduction, for each attribute, a similarity matrix is constructed that contains the similarity measure of each pair of objects in the data set. Furthermore, in each iteration, the similarity matrices should be constructed for different subsets of attributes. If a data set contains $|U|$ objects and $|A|$ conditional attributes, then the memory utilization for constructing similarity matrices is $\mathcal{O}(|A| * |U|^2)$. In the discernibility matrix based approach, a matrix is constructed for each pair of objects in the data set. And each entry in the matrix contains discernible value for each attribute between a pair of objects. Since the discernibility matrix is symmetric, only the upper diagonal or lower diagonal entries are computed. And the entries are computed between the objects that are from different decision classes. Because of the above reasons, the memory utilization of the discernibility matrix based approach is at most $\mathcal{O}(|A| * ((|U|^2/2) - |U|))$. As a result, a substantial decrease in memory usage is achieved in the discernibility matrix based approach relative to the dependency measure based approach.

From the current literature, it is observed that for scalable fuzzy-rough attribute reduction, the discernibility

matrix based accelerators and the corresponding parallel/distributed approaches do not exist. This has inspired us to build ways to fill the current research gap. We therefore propose a discernibility matrix based accelerator for scalable fuzzy-rough attribute reduction. This accelerator is used to build a sequential approach and the corresponding parallel/distributed approach. In summary, our contributions in this paper include the following:

1) We introduce a discernibility matrix based attribute reduction accelerator (DARA). Based on this accelerator, we develop a sequential algorithm IFDMFS (Improved Fuzzy Discernibility Matrix based Feature Selection) for scalable attribute reduction.
2) For enhancing the scalability further, a parallel algorithm MR_IFDMFS (MapReduce based parallel/distributed version of IFDMFS) is developed.

The relevance and limitations of the both sequential and parallel/distributed approaches are proved through extensive experimental analysis along with theoretical validation.

The rest of this paper is organized as follows. The theoretical background of fuzzy-rough sets, and details of the Apache Spark are given in Section 2. In Section 3, the existing fuzzy discernibility matrix based attribute reduction is discussed. In Section 4, the proposed DARA, and the sequential algorithm using DARA are discussed. In Section 5, the proposed parallel approach is provided. The experimental analysis is given in Section 6. And, finally, the conclusion is given in Section 7.

2 Preliminaries

In this section, we discuss the principles of fuzzy-rough set theory on the basis of [5–8]. In addition, an overview of the Apache Spark MapReduce framework is provided.

2.1 Fuzzy-rough sets

Let $DS = (U, A \cup \{d\})$ be a decision system. Here, U denotes the set of objects, A denotes the set of numeric conditional attributes, and 'd' is the categorical decision attribute. The core principle for the reduction of attributes using a classical rough set model is the *indiscernibility relation* [1], which is an equivalence relation. The idea of indiscernibility relation is generalised by using a *fuzzy similarity relation* [6] in fuzzy-rough set model.

Definition 1 For a given decision system $DS = (U, A \cup \{d\})$, a fuzzy similarity relation SIM_a is a fuzzy relation on U using the knowledge of the attribute $a \in A$. The similarity relation satisfies the following conditions.

1) Reflexivity: $\forall x \in U, \mu_{SIM_a}(x, x) = 1$
2) Symmetry: $\forall x, x' \in U, \mu_{SIM_a}(x, x') = \mu_{SIM_a}(x', x)$
3) T-transitivity: $\forall x, y, z \in U, \mu_{SIM_a}(x, z) \geq T(\mu_{SIM_a}(x, y), \mu_{SIM_a}(y, z))$

Here, $\mu_{SIM_a}(x, x')$ denotes the similarity measure between the objects x and x' of the attribute a. And, T is a fuzzy $T - norm$ [44] which is an associative aggregation operator $T : [0, 1] \times [0, 1] \rightarrow [0, 1]$. Note that if the fuzzy similarity relation meets the first and second conditions of the Definition 1 (Reflexivity and Symmetry), the relation is assumed to be a *fuzzy tolerance relation*. And if the relation satisfies all three conditions, the relation is considered a *fuzzy T-equivalence relation*.

The similarity measure of an object pair of the fuzzy similarity relation can be computed by using various approaches [5]. In the proposed work, the fuzzy similarity measure is calculated using the following method.

$$\mu_{SIM_a}(x, x') = max\left(min\left(\frac{a(x) - a(x') + \sigma(a)}{\sigma(a)}, \frac{a(x') - a(x) + \sigma(a)}{\sigma(a)}\right), 0\right) \quad (1)$$

Here, $\mu_{SIM_a}(x, x')$ is the measure of degree to which the objects x and x' are similar for attribute a, and $\mu_{SIM_a}(x, x') \in [0, 1]$. The notation $a(x)$ denotes the value of the object x for attribute a. And, the notation $\sigma(a)$ represents the standard deviation of the attribute a. If an attribute a is categorical (qualitative), the classical indiscernibility relation is adopted and the $\mu_{SIM_a}(x, x')$ measure is given as follows.

$$\mu_{SIM_a}(x, x') = \begin{cases} 0 \ if \ a(x) \neq a(x') \\ 1 \ if \ a(x) = a(x') \end{cases} \quad (2)$$

For a subset of attributes $P \subseteq A$, the fuzzy similarity relation is expanded by using the $T - norm$ (T) specified below.

$$\mu_{SIM_P}(x, x') = \underset{a \in P}{T}(\mu_{SIM_a}(x, x')) \ \forall x, x' \in U \quad (3)$$

Since the proposed method is the discernibility matrix based approach for fuzzy-rough attribute reduction, here the details about the reduct computation in the dependency measure approach are ignored. The principles of the existing discernibility matrix based fuzzy-rough reduct computation are provided in Section 3.

2.2 Apache spark

MapReduce [32] is a run time framework for the scalable execution of parallel/distributed algorithms. The framework has three phases: *map, shuffle and sort,* and *reduce*. In the processing of huge data on large clusters of computational nodes, the framework coordinates these three

phases. Some of the existing MapReduce frameworks are: Hadoop, Apache Spark, Twister, etc. The Apache Spark [45] framework has been selected for the implementation of the proposed approaches because of its special features [46, 47].

Apache Spark is an in-memory MapReduce framework that can execute iterative algorithms and has reliable fault tolerance. It provides programming APIs in Scala, Java, R, and Python languages. *RDD (Resilient Distributed Data set)* is the primary data structure used in the Apache Spark. RDD contains the collection of data items distributed across the nodes that can be manipulated in parallel. Operations performed on RDD can be classified into following two categories.

- *Transformation:* It is a function that generates new RDD from the existing one. Some of the frequently used transformation operations in the proposed approaches are: *map(), mapPartitions(),* and *reduceByKey()*.
- *Action:* It is a function that produces actual results from the given RDD. The action operations that are frequently used in the proposed approaches are: *reduce(), collect(),* and *count()*.

3 Fuzzy discernibility matrix based attribute reduction

Three approaches to fuzzy-rough attribute reduction were proposed by Jensen et al. [5]. The approaches are: (i) *Fuzzy Lower approximation based Feature Selection (FLFS)*, (ii) *Fuzzy Boundary region-based Feature Selection (FBFS)*, and (iii) *Fuzzy Discernibility Matrix-based Feature Selection (FDMFS)*. As the proposed method is based on the discernibility matrix approach, this section provides the details of the FDMFS approach. And the algorithm of FDMFS is given at the end of the section. Table 1 shows the tiny decision system, used to illustrate the basic concepts and the proposed approaches.

3.1 Fuzzy discernibility

The discernibility relation is determined with the complement to the indiscernibility relation in the classical rough set theory model. In the same way, the discernibility relation ($DIS_a(x, x')$) in fuzzy-rough set model is obtained by performing fuzzy negation on the fuzzy similarity between the two objects x and x' as given below.

$$\mu_{DIS_a}(x, x') = N(\mu_{SIM_a}(x, x')) \tag{4}$$

Here, N represents the fuzzy negation, $\mu_{SIM_a}(x, x')$ is fuzzy similarity measure and $\mu_{DIS_a}(x, x')$ is fuzzy dissimilarity (fuzzy discernibility) measure between the objects x and x' based on the knowledge of attribute a. In the proposed works and in literature [5, 10], the *standard fuzzy negation* is considered for N and is given below.

$$N(\mu_{SIM_a}(x, x')) = 1 - \mu_{SIM_a}(x, x') \tag{5}$$

From the above equation, the fuzzy discernibility measure $\mu_{DIS_a}(x, x') \in [0, 1]$.

Example 1 For the decision system given in Table 1, if $\mu_{SIM_b}(x_1, x_2) = 0.54$ is the fuzzy similarity measure based on (1), then by using (4) the fuzzy discernibility measure is given as $\mu_{DIS_b}(x_1, x_2) = 1 - 0.54 = 0.46$.

3.2 Fuzzy discernibility matrix (FDM)

The fuzzy discernibility relation is represented in a fuzzy discernibility matrix (FDM). Each entry (clause) in the matrix is a vector that includes a fuzzy discernibility measure of each attribute of A. The FDM contains the entries between the objects of different decision classes. And the rest of the entries are empty. Therefore, the matrix is a *decision relative fuzzy discernibility matrix*. In the matrix, an entry $C_{xx'}$ between the objects $x, x' \in U$ is given as,

$$C_{xx'} = \begin{cases} < v_1, v_2, ..v_i, ..v_{|A|} >, & if\ d(x) \neq d(x') \\ \emptyset & otherwise \end{cases} \tag{6}$$

Here, in an entry $C_{xx'}$ each value $v_i = \mu_{DIS_i}(x, x')\ \forall i \in A$.

Example 2 From the decision system given in Table 1, using (1) and (4), the fuzzy discernibility measures for all the conditional attributes between the objects x_1 and x_2 are computed as, $\mu_{DIS_a}(x_1, x_2) = 1.0$, $\mu_{DIS_b}(x_1, x_2) = 0.46$, $\mu_{DIS_c}(x_1, x_2) = 0.11$, $\mu_{DIS_d}(x_1, x_2) = 0.14$ and $\mu_{DIS_e}(x_1, x_2) = 1.0$. From (6), an entry $C_{x_1x_2}$ in FDM is represented as $C_{x_1x_2} =< 1.0, 0.46, 0.11, 0.14, 1.0 >$.

The discernibility measure for a subset of attributes $P \subseteq A$ is calculated from discernibility measure of each individual attribute using the following definition.

Definition 2 In a given decision system $DS = (U, A \cup \{d\})$, the discernibility measure or *satisfiability* (SAT) for a subset of attributes $P \subseteq A$ in an entry $C_{xx'}$ of FDM is given by,

$$SAT_P(C_{xx'}) = \bigcup_{a \in P} \{\mu_{DIS_a}(x, x')\} \tag{7}$$

In (7), the fuzzy union is computed by using a specified fuzzy $S - norm$ ($T - conorm$) [5]. Here, $S - norm$ is an aggregation operator $\perp : [0, 1] \times [0, 1] \rightarrow [0, 1]$, and for

Table 1 An example decision system

	Attributes ($A \cup \{d\}$)					
Objects (U)	a	b	c	d	e	f
x_1	3.59	3.52	1.86	0.76	6.30	1
x_2	1.97	4.31	1.57	0.54	3.69	2
x_3	1.51	0.85	7.54	2.31	2.09	2
x_4	2.16	0.50	3.80	1.79	2.12	1
x_5	3.73	1.17	5.68	4.34	3.37	1

any fuzzy values of p, q, r, and t, it satisfies the following conditions.

1) Identity: $\perp(p, 0) = \perp(0, p) = p$
2) Commutativity: $\perp(p, q) = \perp(q, p)$
3) Associativity: $\perp(p, \perp(q, r)) = \perp(\perp(p, q), r)$
4) Monotonicity: $\perp(p, q) \leq \perp(r, t)\ if\ p \leq r\ and\ q \leq t$

In the implementation of the proposed algorithms, we use the *Lukasiewicz T-conorm* or *S-norm* ($\perp(x, y) = min(1, x + y)$) [5] to evaluate the discernibility measure between the set of attributes as given in (7).

Example 3 By continuing Example 2, if we select subset $P = \{b, c\}$, the resultant satisfiability of the entry $C_{x_1x_2}$ is $SAT_P(C_{x_1x_2}) = \perp(0.46, 0.11) = min(1, 0.46 + 0.11) = 0.57$.

In the FDM, the satisfiability of all the entries for a subset of attributes $P \subseteq A$ is computed as,

$$SAT(P) = \frac{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_P(C_{xx'})}{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_A(C_{xx'})} \tag{8}$$

By using (8), the fuzzy-rough reduct is defined as given below.

Definition 3 For a given decision system $DS = (U, A \cup \{d\})$, the fuzzy-rough reduct R is a minimal subset of the conditional attribute set A ($R \subseteq A$) such that,

$$SAT(R) = SAT(A)\ and\ SAT(R') < SAT(A)\ for\ any\ R' \subset R \tag{9}$$

Thus, the attributes set $R \subseteq A$ is said to be fuzzy-rough reduct, if and only if R is a minimal subset of A satisfying $SAT(R) = 1$.

3.3 Reduct computation using FDM (FDMFS algorithm)

In [5], the authors provided the methodology to compute the reduct by using an FDM. It is referred to as FDMFS (FDM based Feature Selection). Here, we are providing the methodology in the form of an algorithm given in Algorithm 1.

Algorithm 1 FDMFS algorithm

Input: 1. Fuzzy discernibility matrix: FDM
2. S-norm: $\perp$
Output: Reduct R

```
1  R = ∅, SAT(R) = 0.0 // Initialize reduct and
       satisfiability of reduct
2  SAT_A = Σ_{C_xx' ∈ FDM ∧ C_xx' ≠ ∅} SAT_A(C_xx')
3  repeat
4      Temp = R
5      for each a ∈ (A − R) do
6          Compute
           SAT(R ∪ {a}) = (Σ_{C_xx' ∈ FDM ∧ C_xx' ≠ ∅} SAT_{(R∪{a})}(C_xx')) / SAT_A
7          if (SAT(R ∪ {a}) > SAT(Temp)) then
8              Temp = R ∪ {a}
9              SAT(Temp) = SAT(R ∪ {a})
10         end
11     end
12     R = Temp
13 until (SAT(R) == 1)
14 return R
```

The FDMFS algorithm starts with the initialization of the current reduct R to the empty set ($\emptyset$). At each iteration of the algorithm, the next best attribute a^{best} to be added to the reduct R is computed by using the following equation derived from (8).

$$SAT(R \cup \{a^{best}\}) = \max_{a \in (A-R)} (SAT(R \cup \{a\})) \tag{10}$$

The denominator value of $SAT(R \cup \{a\})$ is a normalising factor that is calculated as SAT_A and shown in second line of Algorithm 1. The next best attribute a^{best} is determined using (10) and added to the reduct set R. This procedure is repeated until $SAT(R)$ becomes 1. When the $SAT(R)$ reaches 1, the algorithm returns reduct set R and terminates.

4 Discernibility matrix based attribute reduction accelerator (DARA)

The idea behind introducing DARA is explored in this section. *SAT-region removal* which is the main feature of

DARA is presented. And a sequential IFDMFS (Improved Fuzzy Discernibility Matrix-based Feature Selection) algorithm is proposed based on the DARA. The IFDMFS is an improved version of FDMFS [5].

4.1 Motivation

From the analysis of literature, it can be observed that the number of objects contained in the data set prevent the scalability of the fuzzy-rough attribute reduction in massive data sets. From the algorithms of [27, 28, 30], it is noticed that, in each iteration of the reduct computation, the algorithms removed the objects which are no longer useful in future computations. Therefore, the basic principle of any accelerator of fuzzy-rough attribute reduction is to remove the redundant objects in the data set. In the proposed works, the removal of the redundant objects is done in two stages. In the first stage, by constructing a decision relative FDM (given in Section 3.2), the computations are restricted to the objects which are from different decision classes. In the second stage, we incorporate the SAT-region removal feature into the proposed algorithm. This feature acts as an accelerator, and is called DARA. This accelerator limit the computations to the subset of entries of FDM in each iteration of reduct computation to avoid the redundant computations. The following theorem explores the redundant computations involved in an iteration of the existing algorithm to select an attribute to the reduct set.

Theorem 1 *In a given decision system* $DS = (U, A \cup \{d\})$, *consider an attribute set* $R \subseteq A$, *if* $C_{xx'}$ *is an entry of FDM for which* $SAT_R(C_{xx'}) = SAT_A(C_{xx'})$ *is satisfied, then* $\forall R' \supseteq R$,

$$SAT_{R'}(C_{xx'}) = SAT_R(C_{xx'})$$

Where SAT value of an entry $C_{xx'}$ *is computed using a given fuzzy* $S - norm : \perp$.

Proof For any two fuzzy values p, q, where $0 \leq p \leq 1$ and $0 \leq q \leq 1$, from identity property of $S - norm$, we have $\perp(p, 0) = p$, and $\perp(0, q) = q$. Similarly, from monotonicity property of $S - norm$, we have $\perp(p, q) \leq \perp(r, t)$ whenever $p \leq r, q \leq t$ for any $0 \leq r, t \leq 1$. Using identity property $\perp(0, q) = q$, and as $0 \leq p, q \leq q$, we have $q = \perp(0, q) \leq \perp(p, q)$,

$$\therefore q \leq \perp(p, q) \tag{11}$$

Similarly,

$$p \leq \perp(p, q) \tag{12}$$

From (7), for an entry $C_{xx'}$ in the discernibility matrix, the satisfiability of a subset of attributes $R \subseteq A$ is given by,

$$SAT_R(C_{xx'}) = \bigcup_{att \in R} \{\mu_{DIS_{att}}(C_{xx'})\}$$

For any attribute $a \in (A - R)$, let R' be $R \cup \{a\}$ then

$$\begin{aligned} SAT_{R\cup\{a\}}(C_{xx'}) &= \bigcup_{att\in(R\cup\{a\})} \{\mu_{DIS_{att}}(C_{xx'})\} \\ &= \mathop{\perp}_{att\in(R\cup\{a\})} (\mu_{DIS_{att}}(C_{xx'})) \\ &= \perp(\mathop{\perp}_{att\in R} (\mu_{DIS_{att}}(C_{xx'})), \mu_{DIS_a}(C_{xx'})) \\ &= \perp(SAT_R(C_{xx'}), \mu_{DIS_a}(C_{xx'})) \end{aligned}$$

From (11) and (12),

$$\begin{aligned} SAT_R(C_{xx'}) &\leq \perp(SAT_R(C_{xx'}), \mu_{DIS_a}(C_{xx'})) \\ &= SAT_{R\cup\{a\}}(C_{xx'}) \\ &= SAT_{R'}(C_{xx'}) \end{aligned}$$

For any $R' \supseteq R$, the same argument can be successfully applied for each addition of attribute in $R' - R$. For any $R' \supseteq R$, we have,

$$SAT_R(C_{xx'}) \leq SAT_{R'}(C_{xx'})$$

And, since $R \subseteq R' \subseteq A$, we have,

$$SAT_R(C_{xx'}) \leq SAT_{R'}(C_{xx'}) \leq SAT_A(C_{xx'})$$

But it is given that, $SAT_R(C_{xx'}) = SAT_A(C_{xx'})$

$$\therefore SAT_R(C_{xx'}) = SAT_{R'}(C_{xx'})$$

Hence proved. □

4.2 SAT-region removal as an accelerator

From Theorem 1, it is established that if an entry $(C_{xx'})$ reaches its maximum SAT value (i.e., $SAT_R(C_{xx'}) = SAT_A(C_{xx'})$) for the attributes subset $R \subseteq A$, then calculating SAT value for the same entry in selecting the next attribute to R becomes redundant computation. Therefore, for a given set of attributes $R \subseteq A$, the FDM is divided into two non overlapping sets: FDM_F(R) (FDM Fulfilled) and FDM_UF(R) (FDM Unfulfilled). And they are defined below.

Definition 4 In a given decision system $DS = (U, A \cup \{d\})$, let R be a set of attributes such that $R \subseteq A$, and $FDM\_F(R)$ be a set, where it contains the entries of FDM as given below,

$$FDM\_F(R) = \{C_{xx'} \in FDM | SAT_R(C_{xx'}) = SAT_A(C_{xx'})\}$$

From the above Definition 4, $FDM\_F(R)$ includes all the entries of FDM, which satisfy the condition

$SAT_R(C_{xx'}) = SAT_A(C_{xx'})$. That is, $FDM\_F(R)$ contains the subset of entries of FDM, which have already reached the maximum SAT value.

Definition 5 In a given decision system $DS = (U, A \cup \{d\})$, let R be a set of attributes such that $R \subseteq A$. And, let $FDM\_UF(R)$ be a set that contains the entries of FDM as given below,

$$FDM\_UF(R) = \{C_{xx'} \in FDM | SAT_R(C_{xx'}) < SAT_A(C_{xx'})\}$$

Definition 5 states that, $FDM\_UF(R)$ includes all the entries of FDM, which strictly satisfy the condition $SAT_R(C_{xx'}) < SAT_A(C_{xx'})$. That is, $FDM\_UF(R)$ contains the subset of entries of FDM, which have not yet reached the maximum SAT value.

Theorem 2 *The selection of next best attribute a^{best} from $(A - R)$ into reduct R in algorithm FDMFS can be equivalently performed by restricting the computations to only the entries of $FDM\_UF(R)$.*

Proof In the FDMFS algorithm, the next best attribute a^{best} from $(A - R)$ is selected based on the criteria given below.

$$SAT(R \cup \{a^{best}\}) = \max_{a \in (A-R)} (SAT(R \cup \{a\}))$$

$$\text{Here, } SAT(R \cup \{a\}) = \frac{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_{(R \cup \{a\})}(C_{xx'})}{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_A(C_{xx'})}$$

$$= \frac{\sum_{C_{xx'} \in FDM\_F(R) \wedge C_{xx'} \neq \emptyset} SAT_R(C_{xx'}) + \sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SAT_{(R \cup \{a\})}(C_{xx'})}{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_A(C_{xx'})}$$

$$(\because FDM = FDM\_F(R) \cup FDM\_UF(R))$$

It is observed that, the expression $\sum_{C_{xx'} \in FDM\_F(R) \wedge C_{xx'} \neq \emptyset} \{SAT_R(C_{xx'})\}$ and the expression $\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} \{SAT_A(C_{xx'})\}$ are independent of attribute a, $\forall a \in (A - R)$. Therefore, for the next best attribute a^{best}, we have,

$$SAT(R \cup \{a^{best}\}) = \max_{a \in (A-R)} (SAT(R \cup \{a\}))$$

$$= \max_{a \in (A-R)} \left(\frac{\sum_{C_{xx'} \in FDM\_F(R) \wedge C_{xx'} \neq \emptyset} SAT_R(C_{xx'}) + \sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SAT_{(R \cup \{a\})}(C_{xx'})}{\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_A(C_{xx'})} \right)$$

From the above equation, it can be noticed that, the expressions $\sum_{C_{xx'} \in FDM\_F(R) \wedge C_{xx'} \neq \emptyset} SAT_R(C_{xx'})$ and $\sum_{C_{xx'} \in FDM \wedge C_{xx'} \neq \emptyset} SAT_A(C_{xx'})$ are constants in computing $SAT(R \cup \{a\})$, $\forall a \in (A - R)$.

Hence, the next best attribute selection based on all the entries of FDM using (10) is same as the next best attribute a^{best} which achieves

$$\max_{a \in (A-R)} \left(\sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SAT_{(R \cup \{a\})}(C_{xx'}) \right).$$

Therefore, the computations are performed only on $FDM\_UF(R)$ entries and not on all the entries of FDM. □

It is obvious from Theorem 2 that in the proposed algorithm, the computations are carried out only on the entries of $FDM\_UF(R)$ in each iteration of the reduct computation. In other words, the (10) in the existing FDMFS algorithm is updated to compute the next best attribute a^{best}. The modified equation is given below.

$$SATUF(R \cup \{a^{best}\}) = \max_{a \in (A-R)} (SATUF(R \cup \{a\})) \quad (13)$$

Here $SATUF(R \cup \{a\}) = \sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SAT_{(R \cup \{a\})}(C_{xx'})$. The SAT-region removal feature is derived from Theorem 2, and is defined below.

Definition 6 For a given decision system $DS = (U, A \cup \{d\})$, in the FDMFS algorithm, let initial reduct $R = \phi$, $FDM\_F(R) = \phi$, and initial $FDM\_UF(R) = \{1, 2, ...|FDM|\}$. Then after adding an attribute to the reduct set R at i^{th} iteration of the algorithm, the SAT-region removal is given by,

$$FDM\_UF(R^{i+1}) = FDM\_UF(R^i) - FDM\_F(R^{i+1})$$

Since $FDM\_F(R)$ includes the entries that have reached maximum SAT value, the set of entries of $FDM\_F(R)$ is known to be SAT-region. Thus, the removal of the entries of $FDM\_F(R)$ from $FDM\_UF(R)$ is referred to as SAT-region removal. For each iteration of the proposed algorithm, the removal of the SAT-region is done such that the reduct computation is accelerated. Therefore, SAT-region removal serves as an accelerator, and as it is based on the discernibility matrix, is referred to as DARA. The proposed IFDMFS algorithm given in Algorithm 2 incorporates DARA.

4.3 IFDMFS algorithm

The proposed sequential IFDMFS algorithm is developed based on the following Corollary 1, which is derived from Theorem 2 and Definition 6.

Corollary 1 *For a given decision system $DS = (U, A \cup \{d\})$, if $R^0 = \phi \subseteq R^1 \subseteq R^2 ... \subseteq R^i \subseteq ... \subseteq R^n \subseteq A$, where R^i is attributes selected into reduct R by the i^{th} iteration of the proposed algorithm, which computes the final reduct R^n in n iterations, then the fuzzy discernibility matrix*

$$FDM = FDM\_UF(R^0) \supseteq FDM\_UF(R^1) \supseteq ...$$
$$\supseteq FDM\_UF(R^i) \supseteq ... \supseteq FDM\_UF(R^n) = \phi$$

Procedure of IFDMFS algorithm is given in Algorithm 2. Initially the FDM for the given decision system is constructed based on the procedure given in Section 3.2. In the process of reduct computation, IFDMFS algorithm starts its first iteration by initializing the reduct R as an empty set ($\emptyset$), and the satisfiability value is initialized as $SATUF(R) = 0.0$. The set which contains the entries with maximum SAT value is initialized to $FDM\_F(R) = \emptyset$, and the set which has the entries that have not yet reached maximum SAT value is $FDM\_UF(R) = \{1, 2, ...|FDM|\}$.

The next best attribute a^{best} is computed by using (13), and the attribute is added to the reduct R. The entries which have reached maximum SAT value are added to the set $FDM\_F(R)$, and the SAT-region removal is performed. In the subsequent iteration, the next best attribute is computed by using only the entries of $FDM\_UF(R)$ after the SAT-region removal process is completed. This procedure is repeated until $FDM\_UF(R)$ becomes empty ($\emptyset$). From Corollary 1, it is to be noted that, once $FDM\_UF(R)$ becomes empty ($\emptyset$), then $FDM\_F(R)$ set contains all the entries of FDM. That is, all the entries of FDM have fulfilled the satisfiability, and $SAT(R) = 1$ (i.e., $SAT(R) = SAT(A)$). Hence the algorithm returns the reduct R, and terminates.

Algorithm 2 Sequential IFDMFS algorithm

Input: 1. Input file: data set $DS = (U, A \cup \{d\})$
2. Fuzzy similarity relation: SIM
3. Fuzzy negation: N, and S-norm: $\perp$

Output: Reduct R

1 Construct FDM for DS // Construct FDM using the procedure given in Section 3.2
2 $R = \emptyset$, $SATUF(R) = 0.0$ // Initialize reduct and satisfiability of reduct
3 $FDM\_F(R) = \emptyset$, $FDM\_UF(R) = \{1, 2, ...|FDM|\}$
4 **repeat**
/* ============= Phase 1: Computation of the best attribute ============= */
5 $Temp = R$
6 **for** *each* $a \in (A - R)$ **do**
7 Compute $SATUF(R \cup \{a\}) = \sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SATUF_{(R \cup \{a\})}(C_{xx'})$
8 **if** $(SATUF(R \cup \{a\}) > SATUF(Temp))$ **then**
9 $Temp = R \cup \{a\}$
10 $SATUF(Temp) = SATUF(R \cup \{a\})$
11 **end**
12 **end**
13 $R = Temp$
/* ================== Phase 2: SAT-region removal ================== */
14 **for** *each* $C_{xx'} \in FDM\_UF(R)$ **do**
15 **if** $(SATUF_R(C_{xx'}) == SATUF_A(C_{xx'}))$ **then**
16 $FDM\_F(R) = FDM\_F(R) \cup C_{xx'}$
17 **end**
18 **end**
19 $FDM\_UF(R) = FDM\_UF(R) - FDM\_F(R)$
20 **until** $(FDM\_UF(R) == \emptyset)$
21 return R

4.3.1 Complexity analysis of IFDMFS algorithm

For a given decision system, let $|U|$ denotes the number of objects and $|A|$ denotes the number of conditional attributes. If the IFDMFS algorithm gets FDM as the input, then in the worst case (when the reduct set R is equal to all the attributes set A), the algorithm has to perform $\mathcal{O}(|A|^2)$ number of SAT evaluations for finding the reduct. And these evaluations are performed against the FDM of size $\mathcal{O}(|U|^2)$. Thus, the complexity of the proposed IFDMFS algorithm becomes $\mathcal{O}((|A|^2 * |U|^2)$. But the practical time and space complexities of the algorithm are much smaller, since the FDM is symmetric and decision relative. In other words, the construction of the FDM is done either for lower diagonal or upper diagonal entries and are formed only for the objects that are belonging to different decision classes. Thus if the decision attribute has n decision classes with the cardinalities of $c_1, c_2, ...c_n$, then the number of entries in the FDM are reduced to $E = \sum_{i=1}^{n} c_i * (\sum_{j=i+1}^{n} c_j)$ which is much smaller than $|U|^2$. In addition, the size of the FDM is reduced with DARA in each iteration that contributes to reduction in the time and space complexities as the iterations of the algorithm progress. Hence the theoretical time and space complexities of IFDMFS are $\mathcal{O}(|A|^2 * |U|^2)$ and $\mathcal{O}(|A| * E)$ respectively, whereas the exact time complexity is much smaller.

4.3.2 Illustrative example

This example is based on the decision system shown in Table 1 and is meant to illustrate how the IFDMFS algorithm works. For the given decision system, the FDM must be built on the basis of the fuzzy discernibility shown in (4) using the standard fuzzy negation in (5) and the fuzzy similarity in (1). Each entry $C_{xx'}$ of the FDM is formed on the basis of (6). The computed entries of FDM are given below.

$C_{x_1x_2} :< 1.0, 0.46, 0.11, 0.14, 1.0 >$, $C_{x_1x_3} :< 1.0, 1.0, 1.0, 1.0, 1.0 >$,
$C_{x_2x_4} :< 0.19, 1.0, 0.88, 0.82, 0.91 >$, $C_{x_2x_5} :< 0.64, 0.20, 1.0, 0.34, 0.02 >$,
$C_{x_3x_4} :< 1.0, 1.0, 1.0, 1.0, 0.19 >$, $C_{x_3x_5} :< 1.0, 0.18, 0.73, 1.0, 0.74 >$

Initial reduct $R = \emptyset$, satisfiability of reduct $SATUF(R) = 0.0$, $FDM\_F(R) = \emptyset$, and $FDM\_UF(R) = \{C_{x_1x_2}, C_{x_1x_3}, C_{x_2x_4}, C_{x_2x_5}, C_{x_3x_4}, C_{x_3x_5}\}$

First iteration Using the line number: 7 in Algorithm 2, the satisfiability values of individual attributes for all the entries in the FDM are computed and given below.

$SATUF(\{a\}) = 4.83$, $SATUF(\{b\}) = 3.84$, $SATUF(\{c\}) = 4.72$,
$SATUF(\{d\}) = 4.30$, and $SATUF(\{e\}) = 3.86$

Based on (13), the best attribute $a^{best} = \{a\}$ is selected and added to reduct set, thus $R = \{a\}$. After the computation of the best attribute, the SAT-region gets the entries which are satisfying the condition in line number: 15 of Algorithm 2. Note that, here $R = \{a\}$, and $A = \{a, b, c, d, e\}$. Using (7), the $SATUF_A(C_{xx'})$ value is computed, where $SATUF_A(C_{xx'}) = \bigcup_{a \in A}\{\mu_{DIS_a}(C_{xx'})\}$. Thus, the entries $\{C_{x_1x_2}, C_{x_1x_3}, C_{x_3x_4}, C_{x_3x_5}\}$ go into SAT-region. After removal of SAT-region from $FDM\_UF(R)$, we get $FDM\_UF(R) = \{C_{x_2x_4}, C_{x_2x_5}\}$, and $FDM\_F(R) = \{C_{x_1x_2}, C_{x_1x_3}, C_{x_3x_4}, C_{x_3x_5}\}$.

Second iteration After the first iteration, algorithm gets $R = \{a\}$, $FDM\_UF(R) = \{C_{x_2x_4}, C_{x_2x_5}\}$, and $FDM\_F(R) = \{C_{x_1x_2}, C_{x_1x_3}, C_{x_3x_4}, C_{x_3x_5}\}$. Now, the entries of $FDM\_F(R)$ in FDM are ignored and the entries of $FDM\_UF(R) = \{C_{x_2x_4}, C_{x_2x_5}\}$ are considered in the computation of the next best attribute a^{best}. So that, the redundant computations on $FDM\_F(R)$ entries are avoided which leads to a lot of reduction in computations. This reduction in computations in each iteration accelerate the attribute reduction process. Now, the satisfiability values of attributes of $(A - R)$ for all the entries in the $FDM\_UF(R)$ are computed and given below.

$SATUF(R \cup \{b\}) = 1.85$, $SATUF(R \cup \{c\}) = 2.0$, $SATUF(R \cup \{d\}) = 1.98$, and $SATUF(R \cup \{e\}) = 1.66$

The best attribute $a^{best} = \{c\}$ is selected and added to reduct set, thus $R = \{a, c\}$. The entries $C_{x_2x_4}$ and $C_{x_2x_5}$ satisfy the condition in line number: 15, hence they go into SAT-region, and are removed from $FDM\_UF(R)$ and added to $FDM\_F(R)$. Now, the set $FDM\_UF(R)$ becomes empty ($\emptyset$), then algorithm returns reduct set $R = \{a, c\}$ and terminates.

5 MapReduce based parallel IFDMFS (MR_IFDMFS)

The fuzzy-rough attribute reduction in parallel approach is performed in two steps. In the first step, DFDM (Distributed FDM) is constructed, and in the second step, the reduct is computed using DFDM. In this section, both the steps are discussed in detail. The equivalence of the reduct generated by both sequential (IFDMFS) and parallel (MR_IFDMFS) algorithms is discussed at the end of this section.

5.1 Distributed fuzzy discernibility matrix (DFDM)

In our previous work [39], we developed a MapReduce based parallel MR_FRDM_SBE algorithm, which utilizes the DFDM in reduct computation. The procedure for the construction of DFDM in [39] is formulated as Algorithm 3 and Algorithm 4 to complete and assist the readability of the proposed work.

Definition 7 For a given decision system $DS = (U, A \cup \{d\})$, let $DS = \bigcup_{i=1}^{p} DS^i$, where $DS^i = (U^i, A \cup \{d\})$ is i^{th} data partition and satisfies (i) $U = \bigcup_{i=1}^{p} U^i$, (ii) $U^i \cap U^j = \emptyset$, $\forall i, j \in \{1, 2, ...p\}$ *and* $i \neq j$, where p is the number of data partitions.

Algorithm 3 Construction of DFDM

Input: 1. Input file: data set $DS = (U, A \cup \{d\})$
2. Fuzzy similarity relation: SIM
3. Fuzzy negation: N, and S-norm: $\perp$
Output: Distributed Fuzzy Discernibility Matrix (DFDM) as RDD

```
1  DS_RDD = readASRDD(DS) // Read the input data set as an RDD
2  DFDM = ∅ // Initialize DFDM as empty
3  for each decisionVal ∈ decisionClasses do
4      newDS_dclass = DS_RDD.filter(decisionVal == d).collect()
5      DS_RDD = DS_RDD.filter(decisionVal != d)
6      Broadcast(newDS_dclass)
7
8      dmRDD = DS_RDD.mapPartitions(data => {
9      dMat = ∅
10     for x = data do
11         for x' = newDS_dclass do
12             dmEntry = DFDMEntry(x, x', N, SIM, ⊥)
13         end
14     end
15     if dmEntry.maxDissVal ≠ 0 then
16         dMat = dMat.union(dmEntry)
17     end
18     dMat })
19
20     DFDM = DFDM.union(dmRDD)
21 end
22 return DFDM
```

Algorithm 4 Computation of DFDM entry: *DFDMEntry*

Input: 1. x and x' are two objects in DS of different decision classes.
2. Fuzzy negation: N, Fuzzy similarity relation: SIM
3. S-norm: $\perp$
Output: $entry =< DissValues, maxDissVal >$

```
1  entry.DissValues = N(μ_SIM_a(x, x')), ∀a ∈ A
2  entry.maxDissVal = ⊥(entry.DissValues)
3  return entry
```

As given in Definition 7, the input data set DS is horizontally partitioned by MapReduce framework, and the data partitions are distributed to the nodes of the cluster. Here, each data partition which is a sub-table of the form $DS^i = (U^i, A \cup \{d\})$ is assigned to a mapper of the cluster. The complexity involved in the construction of DFDM for large data sets is handled in two steps. In the first step, as the discernibility matrix is symmetric, we restrict the creation of DFDM for lower diagonal entries. And in the second step,

since the discernibility matrix is decision relative, we can only compute the entry $C_{xx'}$ (clause) of DFDM for the pair of objects x and x' which are from different decision classes.

The algorithm to construct the DFDM is given in Algorithm 3, and it is written in pseudo-Spark's API. Along with input data table DS, the algorithm also requires the inputs of fuzzy negation (N), fuzzy similarity relation (SIM), and $S-norm$. Here $S-norm$ is used to find μ_{DIS} value for multiple attributes. From the input data table DS_RDD, the objects of a particular decision class are filtered into $newDS\_dclass$ and they are removed from DS_RDD. Depending on the broadcast size, the objects of $newDS\_dclass$ are broadcasted as bulk or in chunks. All these broadcasted objects are compared with other decision class objects in DS_RDD using $mapPartitions()$ to form the new entries for DFDM. The formation of new entry is done by using (1), (2) and (4). Each entry is inserted into $dmRDD$, and then this $dmRDD$ is finally added to RDD of $DFDM$ using a $union$ operation. This above procedure is repeated to construct DFDM for all decision classes, except for the final decision class.

From Algorithm 4, it can be observed that, each entry has two parts: $DissValues$ and $maxDissVal$. The variable $DissValues$ represents the vector of fuzzy dissimilarity values of the entry $C_{xx'}$ of x and x' objects over all conditional attributes A. And, the variable $maxDissVal$ represents $SAT_A(C_{xx'})$ value in (7).

5.2 Parallel reduct computation using DFDM

After the construction of DFDM, the next step in fuzzy-rough attribute reduction is the computation of reduct using DFDM. Like in the sequential approach, in the parallel approach also, the reduct is computed in two phases: (i) computation of the best attribute, and (ii) SAT-region removal. The MapReduce based parallel/distributed algorithms for reduct computation are given in the form of a driver (master), mapper, and reducer. The driver algorithm MR_IFDMFS: driver() is given in Algorithm 5, the mapper algorithm MR_IFDMFS: map() is given in Algorithm 6, and the reducer algorithm MR_IFDMFS: reduce() is given in Algorithm 7. All the algorithms are written using pseudo-Spark's API for better readability. Since the proposed method is a parallel/distributed approach, the constructed DFDM is distributed to the nodes of the cluster as defined below.

Definition 8 For a given decision system $DS = (U, A \cup \{d\})$, let $DFDM$ denotes distributed fuzzy discernibility matrix, then $DFDM^i$ $\{i = 1, 2, ...p\}$ denotes a sub-DFDM, and satisfies (i) $DFDM = \bigcup_{i=1}^{p} DFDM^i$, (ii) $DFDM^i \cap DFDM^j = \emptyset$, where, $i, j = 1, 2, ...p$, here p is number of partitions and $i \neq j$.

From Definition 8, each $DFDM^i$ is also called as *DFDM-split*, and each split is given to a mapper located in a node of the cluster.

5.2.1 Computation of the best attribute

Algorithm 5 MR_IFDMFS: driver()

Input: 1. Input file: data set $DS = (U, A \cup \{d\})$
2. Fuzzy similarity relation: SIM
3. Fuzzy negation: N, and S-norm: $\perp$

Output: Reduct R

1 Distribute the input data set DS into the nodes of the cluster such that each data partition becomes $DS^i = (U^i, A \cup \{d\})$, $\forall i \in \{1, 2, ...p\}$, where p is the number of data partitions in the cluster.
2 Construct $Distributed\ Fuzzy\ Discernibility\ Matrix(DFDM)$ as RDD by invoking Algorithm 3
3 Initial reduct $R = \emptyset$, $FDM\_F(R) = \emptyset$
4 **repeat**
/* ============= Phase 1: Computation of the best attribute =============== */
5 Initiate map job by invoking Algorithm 6.
$val\ SATUF_{(R\cup\{att\})}(C_{xx'})RDD = DFDM.mapPartitions(part => \{var\ mp = map()\})$
/* Each mapper returns collection of $< key, value >$ pairs for each entry $C_{xx'}$ of $DFDM^i$, where each $< key, value >=< att, SATUF_{(R\cup\{att\})}(C_{xx'}) >$, here att is an attribute, $SATUF_{(R\cup\{att\})}(C_{xx'})$ is its satisfiability value of entry $C_{xx'}$ */
6 Initiate reduce job by invoking Algorithm 7.
7 $val\ SATUF(R \cup \{attNo\})RDD = SATUF_{(R\cup\{att\})}(C_{xx'})RDD.reduceByKey((x, y) => \{var\ rp = reduce()\})$
/* Each reducer returns collection of $< key, value >=< attNo, SATUF(R \cup \{attNo\}) >$ pairs, where $attNo$ is attribute number, and $SATUF(R \cup \{attNo\})$ is satisfiability value of $attNo$ for all the entries of $DFDM$. */
8 Collect the attributes and their respective satisfiability values from the reducers. $var\ SATUF(R \cup \{attNo\}) = SATUF(R \cup \{attNo\})RDD.collect()$
9 Select the best attribute $bestAttNo$ which gets maximum satisfiability value.
10 $R = R \cup bestAttNo$
/* ====================== Phase 2: SAT-region removal ==================== */
11 Filter the entries of $DFDM$ into $FDM\_F(R)$ which satisfy $(SATUF_R(C_{xx'}) == SATUF_A(C_{xx'}))$ using map() only operation.
12 $DFDM = DFDM.filter(if(C_{xx'}\ not\ in\ FDM\_F(R)))$
13 **until** $(DFDM == \emptyset)$
14 Return R

The driver (Algorithm 5) invokes Algorithm 3 to construct DFDM. The Algorithm 3 returns the DFDM as an RDD. The driver initializes the reduct R and $FDM\_F(R)$

to an empty set ($\emptyset$) and computes the best attribute by invoking the mapper (Algorithm 6), and reducer (Algorithm 7). As mentioned earlier, each mapper gets a DFDM-split ($DFDM^i$) as input along with fuzzy $S-norm : \perp$. As shown in Algorithm 6, from each record $C_{xx'}$ (an entry) of $DFDM^i$, a set of $< key, value >$ pairs are formed $\forall att \in (A-R)$. For each $att \in (A-R)$, a $< key, value >$ pair is generated, where key is the attribute identifier (att), and the $value$ is its satisfiability value ($SATUF_{(R\cup\{att\})}(C_{xx'})$). It should be noted that to compute the $SATUF_{(R\cup\{att\})}(C_{xx'})$ value, the mapper uses fuzzy $S-norm : \perp$.

In Algorithm 7, each reducer gets a set of $< att, [SATUF_{(R\cup\{att\})}(C_{xx'})] >$ pairs as the input from all the mappers in the cluster, where $[SATUF_{(R\cup\{att\})}(C_{xx'})]$ represents the list of satisfiability values for attribute att. Based on the same key, the reducer adds the satisfiability values of each attribute received from the different mappers. This sum becomes $SATUF(R\cup\{att\})$ value, which is the satisfiability value of an attribute of all the entries in the matrix. Now, the reducer returns the $< key, value >$ pairs to the driver, where key is an attribute att, and $value$ is its $SATUF(R\cup\{att\})$ value. The driver collects all the attributes and their $SATUF(R\cup\{att\})$ values from all the reducers, and selects the best attribute ($bestAttNo$), which has the maximum $SATUF(R\cup\{att\})$ value (i.e., the attribute att, which satisfies $SATUF(R\cup\{att^{best}\}) = \max_{att\in(A-R)}(SATUF(R\cup\{att\}))$). This best attribute is added to the reduct set R.

5.2.2 Parallel SAT-region removal

SAT-region removal in the proposed parallel approach is defined below.

Definition 9 For the given decision system $DS = (U, A\cup\{d\})$, let $DS = \bigcup_{i=1}^{p} DS^i$, where $DS^i = (U^i, A\cup\{d\})$, and let $DFDM = \bigcup_{i=1}^{p} DFDM^i$, where each $DFDM^i$ is a DFDM-split. Let initial reduct $R = \phi$, $FDM\_F(R) = \phi$ in the proposed parallel algorithm, then after adding an attribute to R at each iteration of the algorithm, the SAT-region removal is given by,

$$DFDM = DFDM - FDM\_F(R)$$

After the computation of the best attribute that is to be added to the reduct R, the SAT-region removal given in Definition 9 is incorporated in the second phase of the algorithm as given in Algorithm 5. All the entries of the $DFDM$, which satisfy the condition ($SATUF_R(C_{xx'}) = SATUF_A(C_{xx'})$) are added to the set $FDM\_F(R)$. Now, the entries of $FDM\_F(R)$ are filtered out from the $DFDM$, this leads to SAT-region removal. In the driver algorithm, two phases are repeated until $DFDM$ becomes empty ($\emptyset$). If the $DFDM$ is empty, then no entries are left out in the matrix, the driver returns the reduct R, and the algorithm terminates.

Algorithm 6 MR_IFDMFS: map()

Input: 1. $DFDM^i$ is a partition of Distributed Fuzzy Discernibility Matrix. Each record of this partition read as $< key, value >=< entryNo, entryValues >$, where $entryNo$ represents $C_{xx'}$ and $entryValues$ represent vector of discernible values of all attributes. 2. S-norm: $\perp$

Output: List of $< key', value' >=< att, SATUF_{(R\cup\{att\})}(C_{xx'}) >$ pairs, here att is an attribute and $SATUF_{(R\cup\{att\})}(C_{xx'})$ is its satisfiability value of an entry $C_{xx'}$ of $DFDM^i$

```
for each C_xx' ∈ DFDM^i do
    for each att ∈ R do
        Compute ⊥_{C_xx'}(R)
    end
    for each att ∈ (A − R) do
        SATUF_{(R∪{att})}(C_xx') = ⊥(⊥_{C_xx'}(R), μ_{DIS_att}(C_xx'))
        Construct < key', value' >=< att, SATUF_{(R∪{att})}(C_xx') >
        Emit intermediate < key', value' >
    end
end
```

Algorithm 7 MR_IFDMFS: reduce()

Input: $< key, [V] >$, here, $key = attNo$ and $[V]$ is the list of satisfiability values received from the mappers

Output: $< key', value' >=< attNo, SATUF(R\cup\{attNo\}) >$

```
for each v ∈ V of key = attNo do
    SATUF(R ∪ {attNo}) = SATUF(R ∪ {attNo}) + v
end
Construct < key', value' >=< attNo, SATUF(R ∪ {attNo}) >
Emit < key', value' >
```

Theorem 3 *The reduct generated by the parallel/distributed attribute reduction algorithm is same as the reduct produced by the corresponding sequential method.*

Proof As mentioned in [12], attribute reduction involves three necessary steps: a subset of attributes generation, subset evaluation, and stopping criterion. The parallel/distributed algorithm and the corresponding sequential algorithm differ at the subset evaluation step of attribute reduction.

For the sequential method, let the decision system be $DS = (U, A\cup\{d\})$, and fuzzy discernibility matrix be FDM. The corresponding decision system and distributed FDM for parallel/distributed method are given by $DS = \bigcup_{i=1}^{p} DS^i$, and $DFDM = \bigcup_{i=1}^{p} DFDM^i$ respectively, where p is the number of

partitions. Both sequential and distributed methods differ in evaluating $SATUF(R \cup \{a^{best}\})$. From (13), for the sequential approach, we have,

$$SATUF(R \cup \{a^{best}\}) = \max_{a \in (A-R)} (SATUF(R \cup \{a\}))$$

Where,

$$SATUF(R \cup \{a\}) = \sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SATUF_{(R \cup \{a\})}(C_{xx'})$$

In parallel approach, since the $DFDM$ is distributed to the different nodes of the cluster (i.e., $DFDM = \bigcup_{i=1}^{p} DFDM^i$), the above equation is expressed as given below.

$$SATUF(R \cup \{a\}) = \sum_{i=1}^{p} \left(\sum_{C_{xx'} \in DFDM^i \wedge C_{xx'} \neq \emptyset} SATUF_{(R \cup \{a\})}(C_{xx'}) \right)$$

Here, $DFDM^i$ is a DFDM-split, and it should be noted that $DFDM$ in the parallel approach is same as $FDM\_UF(R)$ in the sequential approach after the SAT-region is removed. Therefore, in the computation of $SATUF(R \cup \{a^{best}\})$, we have,

$$\max_{a \in (A-R)} \left(\sum_{C_{xx'} \in FDM\_UF(R) \wedge C_{xx'} \neq \emptyset} SATUF_{(R \cup \{a\})}(C_{xx'}) \right) =$$

$$\max_{a \in (A-R)} \left(\sum_{i=1}^{p} \left(\sum_{C_{xx'} \in DFDM^i \wedge C_{xx'} \neq \emptyset} SATUF_{(R \cup \{a\})}(C_{xx'}) \right) \right)$$

Hence, the reduct generated by both sequential and parallel approaches is the same. □

5.2.3 Complexity analysis of MR_IFDMFS algorithm

We show the effectiveness of the MR_IFDMFS algorithm that is implemented on cluster of nodes using MapReduce model by comparing with its equivalent sequential algorithm IFDMFS that is implemented on single node. In a decision system, let $|U|$ denotes number of objects, $|A|$ denotes number of attributes. Assume that the decision system is partitioned and distributed to cluster nodes where the workload is divided equally into p number of data partitions.

From Section 4.3, the time and space complexities of the sequential IFDMFS algorithm are $\mathcal{O}(|A|^2 * |U|^2)$ and $\mathcal{O}(|A| * E)$ respectively. And due to DARA, these complexities are further reduced. In parallel MR_IFDMFS algorithm, since data is divided into p partitions, the time and space complexities are reduced to $\mathcal{O}(\frac{(|A|^2 * |U|^2)}{p})$ and $\mathcal{O}(\frac{(|A| * E)}{p})$. In the construction of DFDM, the communication cost for broadcasting a portion of the decision class objects is an overhead. And the cost of shuffle and sort phase is negligible in the reduct computation due to a small number $(p * |A|)$ of values are involved in the communication. Thus, in the MR_IFDMFS algorithm, the computing task of attribute reduction is distributed to different nodes in the form of p partitions. The computational complexity is thus greatly reduced by p times, with overhead arising due to communication costs.

6 Experimental analysis

We carried out the experiments in two stages to evaluate the proposed MR_IFDMFS algorithm. Since MR_IFDMFS is a MapReduce based parallel/distributed algorithm, it is implemented in Apache Spark (version: 2.3.1) with the Scala programming language (version: 2.11.4).

In the first stage of the experiments, we run the MR_IFDMFS algorithm on a node with a single core to get a pure sequential version of MR_IFDMFS that is given as IFDMFS in Algorithm 2. The comparative and classification accuracy results of the IFDMFS algorithm are presented by comparing with the existing PARA [30] algorithm, which is an accelerator for fuzzy-rough reduct computation. The PARA algorithm is implemented on C++ programming language. We obtained the source code of the PARA algorithm from the authors and conducted the experiments.

In the second stage of the experiments, the MR_IFDMFS algorithm is executed on a cluster of nodes. Computational time analysis and performance evaluation are two concerned metrics for parallel/distributed algorithms. These metrics of the proposed MR_IFDMFS algorithm are shown by comparing with the existing parallel/distributed fuzzy-rough attribute reduction algorithms: MR_FRDM_SBE [39] and DFRS [41]. The MR_FRDM_SBE algorithm is our previous work, which is implemented in Apache Spark (version: 2.3.1) with the Scala programming language (version: 2.11.4). The DFRS algorithm is a non-MapReduce parallel/distributed algorithm. The authors of DFRS provided MATLAB simulation for fuzzy-rough attribute reduction and the source code made available in GitHub repository: https://github.com/qu10wenhao/DFRS.git.

6.1 Experimental setup

The experiments of the proposed sequential IFDMFS approach and the existing PARA approach [30] are conducted on a system with Intel (R) Core (TM) i7-8700 CPU@3.20GHz processor with 12 cores and 32 GB of main memory. The system is installed with Ubuntu 18.04 LTS operating system.

The experiments of the proposed parallel/distributed approach MR_IFDMFS and existing approach MR_FRDM_SBE [39] are carried out on a 7-node cluster. In the cluster, one node is set as master (driver) as well as slave, and the rest are set as workers (slaves). The master

node uses Intel (R) Xeon (R) Silver 4110 CPU @ 2.10GHz processor with 32 cores and 64 GB of main memory. All the worker nodes use Intel (R) Core (TM) i7-8700 CPU@3.20GHz processor with 12 cores and 32 GB of main memory. All the nodes run on Ubuntu 18.04 LTS operating system and they are connected via Ethernet (with 1000 Mbps speed). Each node is installed with Java 1.8.0_171, Apache Spark 2.3.1, and Scala 2.11.4. The experiments of the existing DFRS algorithm [41] are conducted on a system with Intel (R) Core (TM) i7-8700 CPU@3.20 GHz processor having 12 cores and 32 GB of main memory. The system is installed with Ubuntu 18.04 LTS operating system and MATLAB 2017 environment. The DFRS source code is executed on this node and the results of the simulation in 7 nodes and 50 nodes are obtained.

Twelve numerical data sets are used in the experimental analysis. They are selected from the UCI Machine Learning Repository [48]. All the data sets with different sizes are chosen according to limited hardware configuration of the cluster. In the selection of these data sets, we considered the aspect of variance in sizes of object space and attribute space to illustrate the relevance and limitations of the proposed approaches. A detailed description of these data sets is given in Table 2. In the table, the data sets from the serial numbers 1 to 7 are considered as small data sets and from 8 to 12 are considered as large data sets.

6.2 Experimental results of IFDMFS

In this section, the efficiency of the proposed IFDMFS algorithm is shown by comparing its results with the PARA algorithm based on computational time, reduct size, and classification accuracy. As IFDMFS and PARA are sequential algorithms, the experiments are conducted on small data sets.

6.2.1 Comparative analysis on computational time

Reduct is computed for the given data sets using the proposed IFDMFS algorithm and the PARA algorithm. In PARA, the authors used a threshold value α ($0 \leq \alpha \leq 1$) in computing the reduct. The threshold value α is used to bridge the gap between the dependency measures of all conditional attributes and reduct attributes. In PARA, therefore the dependency measure with α is considered as the approximate dependency measure of the reduct set. But we have not used any threshold value in the proposed IFDMFS. Therefore the reduct generated by the proposed algorithm is exact. Since we do not use any threshold value, we have taken the α value as 0.0 for PARA, for appropriate comparison with IFDMFS. In addition, we also performed experiments on PARA with α value of 0.12, as suggested by the authors of PARA. The results are reported in Table 3. The running time and the reduct size of the obtained reduct on each data set are separately reported.

The following observations are made from the results:

i) The PARA algorithm with threshold value of $\alpha = 0.0$ resulted with no dimensionality reduction, and all the attributes were selected as reduct. Thus, even though the proposed IFDMFS algorithm did not use a threshold value, it is compared with the PARA that has $\alpha = 0.12$.
ii) IFDMFS algorithm achieved a minimum of 17% and a maximum of 99% computational gains over PARA ($\alpha = 0.12$) on all the data sets except Ionosphere and Shuttle. These significant results of the proposed IFDMFS algorithm over PARA is due to incorporated DARA.
iii) The existing PARA ($\alpha = 0.12$) algorithm obtained 47% computational gain over the proposed IFDMFS

Table 2 Details of different data sets

S.No	Data set	Rename	Objects	Attributes	Classes
1	Ionosphere	Ionosphere	351	34	2
2	Waveform database generator	Waveform	5000	21	3
3	Madelon	Madelon	2000	500	2
4	Statlog(Landsat Satellite)	Satimage	6435	36	7
5	Musk	Musk	6598	166	2
6	Letter recognition	Letter	20000	16	26
7	Statlog(Shuttle)	Shuttle	58000	09	7
8	Gene expression cancer RNA-Seq	Genes	801	20531	5
9	Isolet	Isolet	7797	617	26
10	Human Activities Postural Transitions	HAPT	10929	561	12
11	Diagnosis	Diagnosis	58509	48	11
12	Person	Person	154680	04	11

Table 3 Running time (Seconds) and reduct size of PARA and IFDMFS

	PARA ($\alpha = 0.0$)		PARA ($\alpha = 0.12$)		IFDMFS	
Data set	Running time	Reduct size	Running time	Reduct size	Running time	Reduct size
Ionosphere	6.74	34	1.96	16	2.06	07
Waveform	996.52	21	316.15	14	86.47	08
Madelon	> 259200	*	105341.20	138	231.55	07
Satimage	5993.86	36	345.79	12	286.84	14
Musk	92660.73	166	28531.10	71	554.51	20
Letter	6420.3	16	3558.30	15	2369.03	15
Shuttle	4721.33	09	1520.37	06	2903.80	09

*The run time of the PARA is more than three days. The results are not reported due to manual termination of the program

algorithm in the Shuttle data set which has larger object space and much smaller attribute space. This is due to the increase in the construction time of the discernibility matrix with the larger object space is not able to compensate the benefits obtained in the iterations for reduct computation as $|A|$ is much smaller.

iv) It is also noted from the results that the IFDMFS generated the reducts that are significantly smaller in size relative to the PARA ($\alpha = 0.12$) algorithm.

In summary, the comparative study of IFDMFS and PARA has shown experimentally that the proposed algorithm is useful in achieving shorter length reducts with substantial computational gains.

6.2.2 Comparative analysis on classification accuracy

Classification accuracy results using SVM (Linear Kernel) and KNN (K=3) classifiers for the proposed IFDMFS algorithm and the existing PARA algorithm on different data sets are given in Table 4. Each data set is randomly divided into training data and testing data of the former size 70% and the later size 30%. And the classification models are built based on all the attributes as well as on reduct attributes. Accuracy results of both algorithms are reported separately in Table 4. It should be recalled that the PARA with $\alpha = 0.0$ generates all attributes as reduct, the models are constructed only on the reduct of PARA with $\alpha = 0.12$. The following observations are drawn from the results.

i) For both classifiers, the proposed IFDMFS algorithm provided almost equivalent or better classification accuracy for reduced data compared to the PARA algorithm for Ionosphere, Madelon, Satimage, Letter, and Shuttle data sets. And on data sets such as Musk, and Waveform, the PARA algorithm provided better accuracies than the IFDMFS algorithm.

ii) Nearly for all data sets, both IFDMFS and PARA algorithms have produced the classification accuracy of reduct attributes that is approximately equal to the classification accuracy of unreduct attributes of both classifiers.

In Table 4, taking into account the number of selected attributes (reduct size) of both algorithms, it can be observed that the IFDMFS algorithm can achieve better classification accuracy with smaller reduct sets.

Table 4 Classification accuracy results of PARA and IFDMFS

	SVM			KNN (K=3)		
Data set	Unreduct	PARA ($\alpha = 0.12$)	IFDMFS	Unreduct	PARA ($\alpha = 0.12$)	IFDMFS
Ionosphere	92.38	92.38	91.43	86.73	78.94	81.19
Waveform	87.19	84.26	79.18	87.19	82.69	77.80
Madelon	58.33	58.33	82.67	53.00	55.33	83.50
Satimage	92.58	90.66	91.34	92.58	90.04	89.06
Musk	99.55	99.34	97.52	96.82	96.36	94.69
Letter	97.00	97.00	97.00	94.78	93.66	93.94
Shuttle	99.83	99.83	99.83	99.85	99.85	99.85

Table 5 Running time (Seconds) and reduct size results on small data sets

	7-nodes MR_IFDMFS		7-nodes MR_FRDM_SBE		7-nodes DFRS		50-nodes DFRS	
Data set	Running time	Reduct size	Running time	Reduct size	Running time	Reduct size	Running time	Reduct size
Ionosphere	1.91	07	6.46	12	1.09	32	0.08	32
Waveform	10.24	08	13.85	09	21.62	19	2.11	18
Madelon	21.70	07	130.28	13	27.19	288	3.79	288
Satimage	19.55	14	38.73	16	52.28	36	18.61	36
Musk	49.83	20	91.85	29	66.08	56	8.60	56
Letter	86.83	15	183.86	15	680.54	16	130.85	16
Shuttle	163.71	09	285.04	08	691.71	05	222.92	05

6.3 Experimental results of MR_IFDMFS

In this section, the computational efficiency and performance evaluation of the proposed parallel MR_IFDMFS algorithm is done by comparing its results with the existing parallel approaches MR_FRDM_SBE [39] and DFRS [41]. Since the proposed and existing algorithms are parallel/distributed methods, experiments are performed on all data sets given in Table 2. The results of large data sets and small data sets are reported in separate tables.

6.3.1 Comparative analysis on computational time

The proposed MR_IFDMFS algorithm is numerically compared with the existing MR_FRDM_SBE and DFRS algorithms based on the computational time and reduct size. The experiments of MR_IFDMFS and MR_FRDM_SBE algorithms are performed on 7-node cluster, and the simulation results of DFRS are obtained for 7 nodes and 50 nodes. The results of small data sets are reported in Table 5 and large data sets are given in Table 6. And Fig. 1 gives the clear view of the computational performance of MR_IFDMFS in large data sets. The observations on the results from Tables 5 and 6 are listed as follows.

i) From the comparison of the running times, we can observe that the MR_IFDMFS algorithm performed significantly better than the MR_FRDM_SBE algorithm on all the data sets except Person data set. Excluding the Person, on all the data sets, the MR_IFDMFS obtained a minimum of 26%, and a maximum of 99% of computational gains over MR_FRDM_SBE algorithm. In specific, the computational gains of MR_IFDMFS are higher for high dimensional data sets such as Madelon, Musk, Genes, Isolet and HAPT.

ii) From the comparison of the running times of MR_IFDMFS with DFRS executed on 7 nodes, it can be noticed that, except Ionosphere data set, on all the data sets the proposed algorithm performs well. Here, MR_IFDMFS algorithm's computational gain varies from 22% to 93%.

iii) Proposed MR_IFDMFS executed on 7-node cluster achieved better computational times than the DFRS algorithm for all large data sets (including Letter and Shuttle) even though DFRS is executed in 50 nodes. These notable achievements illustrate the computational efficiency of the proposed MR_IFDMFS algorithm.

Table 6 Running time (Seconds) and reduct size results on large data sets

	7-nodes MR_IFDMFS		7-nodes MR_FRDM_SBE		7-nodes DFRS		50-nodes DFRS	
Data set	Running time	Reduct size	Running time	Reduct size	Running time	Reduct size	Running time	Reduct size
Genes	126.89	06	16742.80	10	525.31	39	268.99	97
Isolet	335.69	09	12549.32	11	3682.13	541	597.45	542
HAPT	467.70	09	8715.97	12	1093.73	347	341.14	348
Diagnosis	594.99	12	1662.36	21	8083.69	26	812.63	26
Person	1010.62	04	638.56	04	*	*	11034.15	04

*DFRS on 7 nodes could not run due to the memory overflow exception

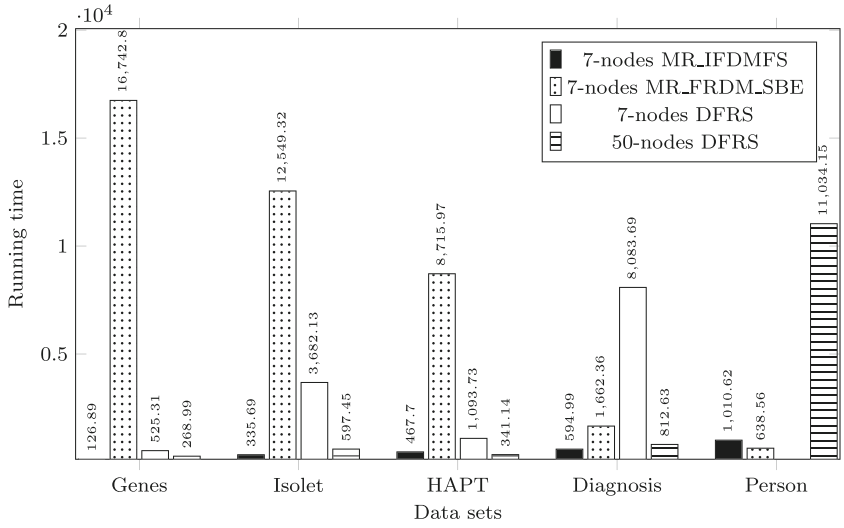

Fig. 1 Running time (Seconds) of algorithms on different data sets

iv) It is also noted from the results that for all data sets except Shuttle, the MR_IFDMFS algorithm has generated smaller reduct set than the DFRS and MR_FRDM_SBE algorithms.

v) For the Person data set, the DFRS algorithm on 7 nodes could not run due to the memory overflow exception.

vi) Results in Tables 3 and 5 established that parallelization in MR_IFDMFS achieved significant computational gains over IFDMFS in all the data sets except Ionosphere. In Ionosphere data set, MR_IFDMFS algorithm incurs almost similar computational time. This may be due to the communication costs in cluster

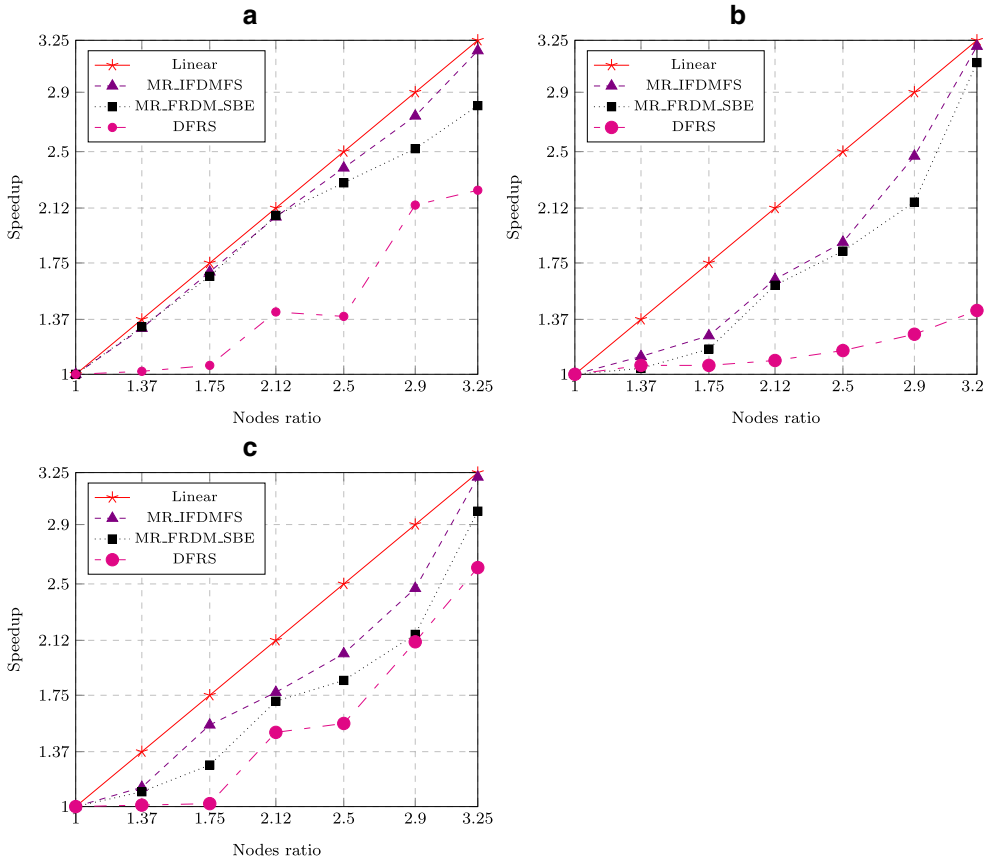

Fig. 2 Speedup results of three algorithms on different data sets. **a** Genes. **b** Diagnosis. **c** HAPT

shadowing the advantages of parallelization in view of the size of the data set being very small.

The significant computational gain achieved by MR_IFDMFS algorithm in almost all the data sets strongly establishes the role of the proposed accelerator DARA in imparting space reduction as the algorithm progresses and there by aiding in reduction of computational time.

6.3.2 Performance evaluation

Speedup and sizeup are scalability metrics for parallel systems. These metrics are used to measure the efficiency of parallel algorithms. Using these metrics, the efficiency of the proposed MR_IFDMFS algorithm is evaluated and compared to current MR_FRDM_SBE and DFRS algorithms on various data sets. Three data sets are chosen such that the first data set has large attribute space, the second has large object space, and the third data set has large object space as well as attribute space. Separate figures are given to show the speedup and sizeup performance results of the algorithms on three data sets.

Speedup evaluation In a parallel system, the speedup is measured by using the formula: $Speedup(n) = \frac{Time_1}{Time_n}$. Here, $Time_1$ is the execution time on one node, and $Time_n$ is the execution time on n-nodes. The speedup is measured by adding the number of nodes in the parallel system while keeping the data set constant. The linear speedup is ideal for a parallel system, but it is not easy to achieve. This is because when the cluster becomes larger, the movement of data and the communication cost increase across the nodes of the cluster. The speedup of the proposed algorithm has been evaluated on the data sets with different nodes from 1 to 7. From the experimental setup given in Section 6.1, it can be observed that the master node has 32 cores and the remaining slave nodes have 12 cores each. Since we have set the master node also as a slave, the number of cores is mismatched with other slave nodes. Owing to this mismatch, in finding the speedup metric of the system, we took nodes ratio based on the number of cores in the node. Figure 2 shows the speedup results of different data sets with a different ratio of nodes (number of cores) in the cluster.

From the findings in Fig. 2, it can be observed that, on all the data sets, the speedup of the proposed MR_IFDMFS is improved with an increase in the number of cores. And MR_IFDMFS efficiency is slightly higher than current MR_FRDM_SBE and DFRS algorithms. The plots of both the proposed MR_IFDMFS and the current

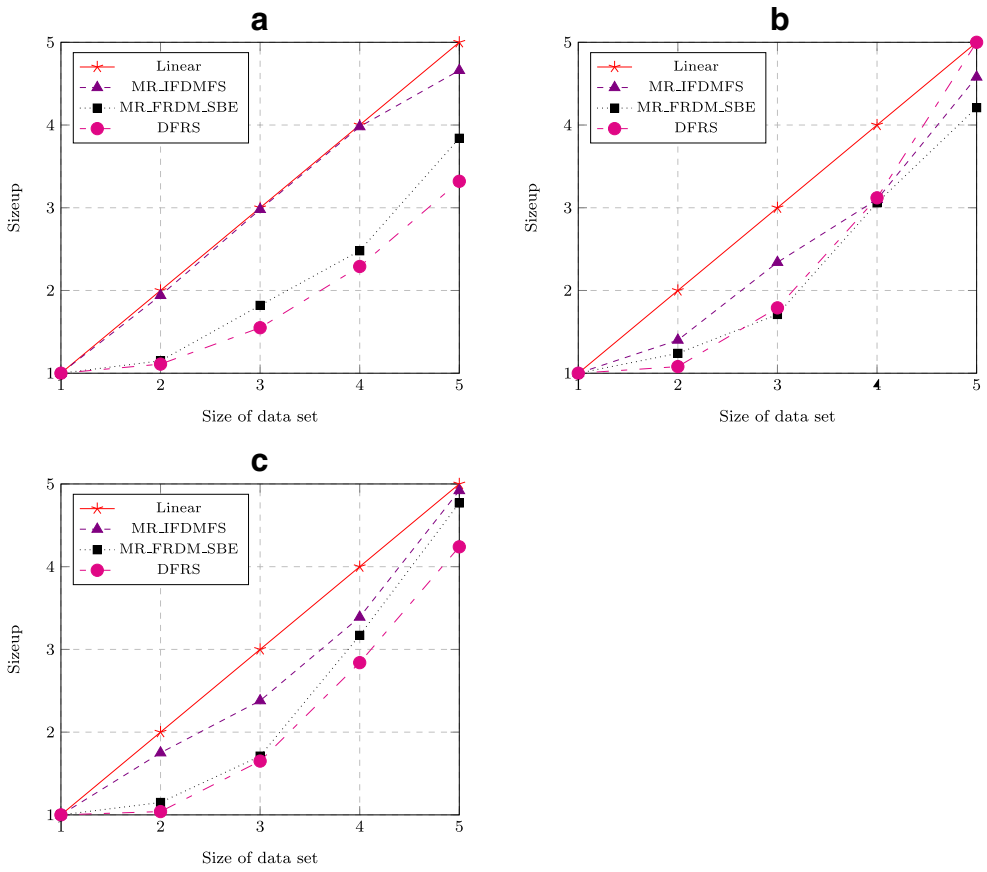

Fig. 3 Sizeup results of three algorithms on different data sets. **a** Genes. **b** Diagnosis. **c** HAPT

MR_FRDM_SBE algorithms are closer to the linear plot and both of these algorithms are outperforming the DFRS algorithm.

Sizeup evaluation Sizeup metric gives the time it spends on a parallel system when the data set size is *n-times* larger than the size of the original data set. It is defined by using the formula: $Sizeup(n) = \frac{Time_n}{Time_1}$. Here, $Time_n$ is the running time for processing n-times of data, and $Time_1$ is the running time for processing the data. To find the sizeup metric, we kept the number of nodes unchanged with seven nodes, and changed the size of the data set as 20%, 40%, 60%, 80%, and 100% of objects in the original data set. Figure 3 shows the sizeup performance results of three algorithms on different data sets.

In Fig. 3, for all the data sets, the sizeup performance of the proposed algorithm increased when the size of the data set was increased. The proposed MR_IFDMFS algorithm is also producing better sizeup performance than the existing MR_FRDM_SBE and DFRS algorithms for all the data sets.

6.3.3 Discussion

It can be observed that the experiments in the cluster for MR_IFDMFS are conducted on the data sets having a few hundred thousands of objects or a few thousands of attributes. These data sets are categorised as large due to the fact that the resulting space utilisation for the DFDM runs into several millions of entries occupying several giga bytes of memory space. This is further coupled by the overhead involved in the Apache Spark maintenance of RDDs across the transformations and for the meta data in achieving fault tolerance. In order to scale data sets to much higher sizes, more nodes need to be added to the cluster. As the cost of shuffle and sort phase of DFDM construction and distributed reduct computation are minimal, the proposed approach is scalable to very large data sets under horizontal expansion of the cluster.

7 Conclusion

In this paper, we introduced a fuzzy discernibility matrix-based accelerator. The idea behind the proposed accelerator is the removal of SAT-region. With this feature, the entries of the discernibility matrix that have reached maximum satisfiability were removed from the matrix in each iteration and the reduct computation performed on the remaining entries of the matrix. Therefore, SAT-region removal served as an accelerator and referred to as DARA. Based on DARA, a sequential IFDMFS algorithm proposed for fuzzy-rough attribute reduction. To deal with large data sets in attribute reduction, we also proposed a MapReduce based MR_IFDMFS algorithm, which is a parallel/distributed version of the IFDMFS algorithm. The experimental results have shown that the proposed algorithms IFDMFS and MR_IFDMFS performed better than the existing approaches. Extensive experimental analysis along with theoretical validation establishes the relevance and efficiency of the proposed approaches in handling large data sets for fuzzy-rough attribute reduction. In future work, we aim to investigate alternative parallel strategies to improve the space and time complexities in the creation of discernibility matrix.

Acknowledgments Authors are grateful to the reviewers for their valuable comments and suggestions. This work is supported by the Department of Science and Technology (DST), the Government of India under the ICPS project [grant number: DST/ICPS/CPS-Individual/2018/579(G) and DST/ICPS/CPS-Individual/2018/579(C)], and by the Digital India Corporation of the Ministry of Electronics and Information Technology, the Government of India under the Visvesvaraya PhD. scheme with the unique id: MEITY-PHD-1039. We would like to extend our sincere thanks to the authors of the PARA [30] algorithm for providing the source code.

Funding This work is supported by Department of Science and Technology (DST), Government of India under ICPS project [grant number: DST/ICPS/CPS-Individual/2018/579(G) and DST/ICPS/CPS-Individual/2018/579(C)], and by Digital India Corporation, a Section 8 Company of Ministry of Electronics and Information Technology, Government of India under Visvesvaraya Ph.D. scheme with the unique id: MEITY-PHD-1039.

Materials Availability Data and material are available with the authors.

Code Availability Code is available with the authors.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Pawlak Z (1982) Rough sets. Int J Comput Inform Sci 11(5):341–356
2. Yao Y, Zhao Y (2008) Attribute reduction in decision-theoretic rough set models. Inf Sci 178(17):3356–3373
3. Zhao S, Chen H, Li C, Du X, Sun H (2015) A novel approach to building a robust fuzzy rough classifier. IEEE Trans Fuzzy Syst 23(4):769–786
4. Hu Q, Yu D, Xie Z (2006) Information-preserving hybrid data reduction based on fuzzy-rough techniques. Pattern Recogn Lett 27(5):414–423
5. Jensen R, Shen Q (2009) New approaches to fuzzy-rough feature selection. IEEE Trans Fuzzy Syst 17(4):824–838
6. Dubois D, Prade H (1990) Rough fuzzy sets and fzzy rough sets. Int J Gen Syst 17(2-3):191–209
7. Radzikowska AM, Kerre EE (2002) A comparative study of fuzzy rough sets. Fuzzy Sets Syst 126(2):137–155
8. Cornelis C, Cock MD, Radzikowska AM (2008) Fuzzy rough sets: From theory into practice. In: Handbook of granular computing Wiley Ltd, pp 533–552

9. Ye J, Zhan J, Ding W, Fujita H (2021) A novel fuzzy rough set model with fuzzy neighborhood operators. Inf Sci 544:266–297
10. Cornelis C, Jensen R, Hurtado G, Śleẓak D (2010) Attribute selection with fuzzy decision reducts. Inf Sci 180(2):209–224
11. Parthaláin NM, Jensen R (2013) Unsupervised fuzzy-rough set-based dimensionality reduction. Inf Sci 229:106–121
12. Jensen R (2008) Rough set-based feature selection. In: Rough computing. IGI Global, pp 70–107
13. Wang J, Wang J (2001) Reduction algorithms based on discernibility matrix: The ordered attributes method. J Comput Sci Technol 16(6):489–504
14. Yao Y, Zhao Y (2009) Discernibility matrix simplification for constructing attribute reducts. Inf Sci 179(7):867–882
15. Sai Prasad PSVS, Rao CR (2011) Extensions to IQuickReduct. In: Lecture notes in computer science. Springer Berlin, pp 351–362
16. Janusz A, Ślezak D (2014) Rough set methods for attribute clustering and selection. Appl Artif Intell 28(3):220–242
17. Chouchoulas A, Shen Q (2001) Rough set-aided keyword reduction for text categorization. Appl Artif Intell 15(9):843–873
18. Chen Y, Liu K, Song J, Fujita H, Yang X, Qian Y (2020) Attribute group for attribute reduction. Inf Sci 535(5):64–80
19. Liu K, Yang X, Yu H, Fujita H, Chen X, Liu D (2020) Supervised information granulation strategy for attribute reduction. Int J Mach Learn Cybern, pp 1–15
20. Dai J, Hu H, Wu W-Z, Qian Y, Huang D (2018) Maximal-discernibility-pair-based approach to attribute reduction in fuzzy rough sets. IEEE Trans Fuzzy Syst 26(4):2174–2187
21. Wang C, Huang Y, Shao M, Fan X (2019) Fuzzy rough set-based attribute reduction using distance measures. Knowl-Based Syst 164:205–212
22. Wang C, Qi Y, Shao M, Hu Q, Chen D, Qian Y, Lin Y (2017) A fitting model for feature selection with fuzzy rough sets. IEEE Trans Fuzzy Syst 25(4):741–753
23. Zhang X, Mei C, Chen D, Yang Y (2018) A fuzzy rough set-based feature selection method using representative instances. Knowl-Based Syst 151:216–229
24. Tan A, Wu W-Z, Qian Y, Liang J, Chen J, Li J (2019) Intuitionistic fuzzy rough set-based granular structures and attribute subset selection. IEEE Trans Fuzzy Syst 27(3):527–539
25. Kumar A, Sai Prasad PSVS (2020) Scalable fuzzy rough set reduct computation using fuzzy min–max neural network preprocessing. IEEE Trans Fuzzy Syst 28(5):953–964
26. Riza LS, Janusz A, Bergmeir C, Cornelis C, Herrera F, Śleẓak D, Benítez JM (2014) Implementing algorithms of rough set theory and fuzzy rough set theory in the r package "RoughSets". Inf Sci 287:68–89
27. Sai Prasad PSVS, Rao CR (2014) An efficient approach for fuzzy decision reduct computation. In: Transactions on rough sets XVII. Springer Berlin, pp 82–108
28. Qian Y, Wang Q, Cheng H, Liang J, Dang C (2015) Fuzzy-rough feature selection accelerator. Fuzzy Sets Syst 258:61–78
29. Jensen R, Parthaláin NM (2015) Towards scalable fuzzy–rough feature selection. Inf Sci 323:1–15
30. Ni P, Zhao S, Wang X, Chen H, Li C (2019) PARA: A positive-region based attribute reduction accelerator. Inf Sci 503:533–550
31. Chen J, Mi J, Lin Y (2020) A graph approach for fuzzy-rough feature selection. Fuzzy Sets Syst 391:96–116
32. Dean J, Ghemawat S (2008) Mapreduce: simplified data processing on large clusters. Commun ACM 51(1):107
33. Sowkuntla P, Sai Prasad PSVS (2020) MapReduce based improved quick reduct algorithm with granular refinement using vertical partitioning scheme. Knowl-Based Syst 189:105104
34. Raza MS, Qamar U (2018) A parallel rough set based dependency calculation method for efficient feature selection. Appl Soft Comput 71:1020–1034
35. Qian J, Miao D, Zhang Z, Yue X (2014) Parallel attribute reduction algorithms using MapReduce. Inf Sci 279:671–690
36. Sai Prasad PSVS, Subrahmanyam HB, Singh PK (2016) Scalable IQRA_IG algorithm: An iterative MapReduce approach for reduct computation. In: Distributed computing and internet technology. Springer International Publishing, pp 58–69
37. Singh PK, Sai Prasad PSVS (2016) Scalable quick reduct algorithm: Iterative mapreduce approach. In: Proceedings of the 3rd IKDD conference on data science. 2016. ACM, p 25
38. Czolombitko M, Stepaniuk J (2016) Attribute reduction based on MapReduce model and discernibility measure. In: Computer information systems and industrial management. Springer International Publishing, pp 55–66
39. Pavani NL, Sowkuntla P, Rani KS, Sai Prasad PSVS (2019) Fuzzy rough discernibility matrix based feature subset selection with MapReduce. In: TENCON 2019 - 2019 IEEE region 10 conference (TENCON). IEEE, pp 389–394
40. Bandagar K, Sowkuntla P, Moiz SA, Sai Prasad PSVS (2019) MR_IMQRA: An efficient MapReduce based approach for fuzzy decision reduct computation. In: International conference on pattern recognition and machine intelligence Springer International Publishing, pp 306–316
41. Kong L, Qu W, Yu J, Zuo H, Chen G, Xiong F, Pan S, Lin S, Qiu M (2020) Distributed feature selection for big data using fuzzy rough sets. IEEE Trans Fuzzy Syst 28(5):846–857
42. Hu Q, Zhang L, Zhou Y, Pedrycz W (2018) Large-scale multimodality attribute reduction with multi-kernel fuzzy rough sets. IEEE Trans Fuzzy Syst 26(1):226–238
43. Ding W, Wang J, Wang J (2020) Multigranulation consensus fuzzy-rough based attribute reduction. Knowl-Based Syst, p 105945
44. Cock MD, Cornelis C, Kerre EE (2007) Fuzzy rough sets: The forgotten step. IEEE Trans Fuzzy Syst 15(1):121–130
45. Zaharia M, Xin RS, Wendell P, Das T, Armbrust M, Dave A, Meng X, Rosen J, Venkataraman S, Franklin MJ, et al. (2016) Apache spark: a unified engine for big data processing. Commun ACM 59(11):56–65
46. Inoubli W, Aridhi S, Mezni H, Maddouri M, Nguifo EM (2018) An experimental survey on big data frameworks. Futur Gener Comput Syst 86:546–564
47. Jakovits P, Srirama SN (2014) Evaluating mapreduce frameworks for iterative scientific computing applications. In: 2014 International conference on high performance computing & simulation (HPCS). IEEE, pp 226233
48. (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml/datasets.html

Pandu Sowkuntla received his B.Tech and M.Tech in Computer Science from Jawarharlal Nehru Technological University, Hyderabad, Telangana, India. He is currently working towards PhD degree in Computer Science from the University of Hyderabad. His current research interests include Rough Sets, Fuzzy-Rough Sets, Parallel and Distributed Computing, and Big Data Analytics.

P. S. V. S. Sai Prasad received M.Sc. degree in Mathematics and M.Tech. degree in Computer Science from Sri Sathya Sai University, Anantapur, India, in year 1997 and 2001 respectively. He received PhD degree in computer science from the University of Hyderabad, Hyderabad, India, in the year 2014.

He is currently working as Associate Professor at the School of Computer and Information Sciences, University of Hyderabad. He has published more than 30 research papers in various International Journals and Conferences. His research interests include Rough Sets, Fuzzy-Rough Sets, Soft Computing, and Big Data Analytics.