Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Efficient composing rough approximations for distributed data☆

Shaoyong Li [a,*], Zhiyong Hong [a], Tianrui Li [b,c,**]

[a] Faculty of Intelligent Manufacturing, Wuyi University, Jiangmen, 529020, China
[b] School of Information Science and Technology, Southwest Jiaotong University, Chengdu, 611756, China
[c] Sichuan Provincial Key Laboratory of Higher Education of Cloud Computing and Intelligent Technique, Chengdu, 611756, China

ARTICLE INFO

Article history:
Received 12 July 2018
Received in revised form 30 May 2019
Accepted 1 June 2019
Available online xxxx

Keywords:
Distributed data
Data mining
Composing rough approximations
Matrix-based rough set approach

ABSTRACT

With the development of network and sensor technologies, distributed information processing has been involved in many scientific research fields. Rough set theory has been proved to be a powerful tool to deal with uncertain, inconsistent and fuzzy information. In rough set methodologies, computing rough approximations is a key and time-consuming step for attribute reduction and rule extraction. In this paper, we introduce a matrix-based rough set approach for processing data distributed across different sites. The approach can efficiently compose rough approximations of global concepts in the distributed information system by utilizing the existing rough approximations of local concepts in subsystems. Some numerical examples are employed to verify the feasibility of the approach. The corresponding algorithm is developed and the experimental estimations show that our algorithm outperforms its counterpart on the computational time.

1. Introduction

With the rapid development of Internet and sensor technologies, a large amount of data is collected from different sources for a variety of scientific research related to data mining and knowledge discovery. In many real-time decision-making solutions, data that they depended on was collected from different geographical location sites. It is urgent to study the approaches for mining distributed data to many scientific research fields related to data mining. In recent years, the scholars had been obtained some achievements in processing distributed information such as energy resource estimation [1], health condition assessment of structures [2] and nutritional assessment [3], etc.

The frameworks of processing distributed data were classified two types as centralized or decentralized [4]. The approaches based on centralized framework require all data being collected in a central node and then process them as a whole. The approached based on decentralized framework allow each node to process locally collected data and then send the locally knowledge to a central processing node for fusion.

During the fusion on the central processing node, how to process the conflict and inconsistency of locally knowledge is a key problem. Rough Set Theory (RST) is one of main theories in Granular Computing [5,6]. As a kind of powerful mathematics tool for processing uncertain, inconsistent and fuzzy information, RST [7] has been successfully applied in many scientific research fields related to data mining and knowledge discovery [8–11] such as evaluation of clean energy development levels [12], breast cancer prediction [13], water eutrophication assessment [14] and risk assessment [15]. Skowron et al. introduced the strategies of granular composition and decomposition based on RST [16,17]. In order to use RST in big data mining, some scholars further investigated the parallel strategies of granular composition and decomposition [18,19]. The strategies of granular composition and decomposition based on RST may be applied in conflict resolution for those existing results. Hence, RST may be applied in distributed information processing.

Computation of rough approximations is a key and most time-consuming step in problem solving by rough sets methodologies. Accelerating computation of rough approximations has been paid attentions in RST research society [18,20–30]. Dynamic data processing is a hot issue on RST research in recent years [20–23]. Many scholars were interesting in researching the incremental approaches for updating rough approximations such as: Wang et al. proposed an approach for updating approximations of Dominance-based Rough Set Approach when attributes and its' values varying in an order information system [20]. Hu et al. introduced an approach for dynamically updating approximations of fuzzy probabilistic rough sets over two universes when

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.06.001.

* Corresponding author.
** Corresponding author at: School of Information Science and Technology, Southwest Jiaotong University, Chengdu, 611756, China.
E-mail addresses: meterer@163.com (S. Li), hongmr@wyu.edu.cn (Z. Hong), trli@swjtu.edu.cn (T. Li).

https://doi.org/10.1016/j.knosys.2019.06.001
0950-7051/

the object set varying which involves the adding and deleting objects [21]. Hang et al. provided an approach for dynamically updating approximations of fuzzy concepts when objects and features varying simultaneously in the fuzzy decision systems [22]. Luo et al. attempted to apply Dominance-based Rough Set Approach in processing hierarchical attribute value and then provided an approach and corresponding algorithm for updating rough approximations [23].

Feature extraction (attribute reduction) is an important mission in data mining research. RST has been widely applied in attribute reduction [10,31]. Recently, Qian et al. pointed out that many existing heuristic attribute reduction algorithms based on RST are expensive in computational time-consuming. For this shortcoming, they proposed an approach for accelerating computation of rough approximations in order to reduce the time taken by attribute reduction [24]. Following that, they considered the challenges to apply classical rough set theory in big data processing such as limited labeled property of big data, computational inefficiency and over-fitting in attribute reduction. To address these challenges, they introduced a theoretic framework called local rough set, and developed a series of corresponding concept approximation and attribute reduction algorithms with linear time complexity [25].

To apply RST in Big Data processing, some parallelizable rough sets approach for computing rough approximations and the corresponding parallel algorithms have been proposed and developed, respectively [18,26–28]. In Big Data processing, it is common to find data partitioned vertically or horizontally, and distributed across different sites [32]. For data partitioned vertically, Zhang et al. proposed a parallelizable approach for computing rough approximations in big data processing with Map-Reduce technique [26] and Li et al. introduced several strategies of composing and decomposing granules in Dominance-based Rough Set Approach and developed the corresponding algorithms for computing approximations in parallel on the multi-core platform [18]. For data partitioned horizontally, Zhang et al. combined matrix operation in rough approximations computing in order to process incomplete information in parallel [27]. Following that, they also developed a parallel algorithm based on the Multi-GPU platform for computing rough approximations under a composite binary relation [28].

Based on the above discussion, there are a lot of rough set approaches that may be applied in dynamic information processing, attribute reduction and big data processing. However, these approaches cannot be applied in composing global rough approximations in distributed information processing directly. In many real-time applications, it is required to process data that was collected from different sites [33,34]. The decentralized framework allows each node processing data collected locally, but it need to investigate how to eliminate the conflict and inconsistency among local knowledge during the fusion. RST has been applied in the update and composition of knowledge by decomposition, composition and transform of information granular [6], then it is a candidate tool for processing distributed data. In this paper, we introduce an rough set approach that may compose the global rough approximations by using the existing local ones. In order to eliminate the conflict and inconsistency among local rough approximations, we decompose and reconfigure the computational process of matrix-based rough approximations firstly. Then we investigate a strategy to compose the global rough approximations by using the existing ones and some necessary computations or operations. Some numerical examples illustrate the feasibility of the approach. The performance of the approach and the main influence factors are demonstrated by using four UCI data sets [35]. Hence, the approach may be a candidate tool to form global knowledge in the fusion stage of decentralized data mining.

The remainder of the paper is organized as follows. We present some concepts and notations of RST and matrix-based rough set approach in Section 2. In Section 3, an approach for composing rough approximations in the distributed information system is introduced. An algorithm for efficiently composing rough approximations is developed in Section 4. Section 5 shows the experimental estimations with data sets from UCI. This paper ends with conclusions and the further research topics in Section 6.

2. Preliminaries

In this section, we briefly review some concepts and notations of RST and matrix-based rough set approach which can be found in [7,36–38].

2.1. An information system and rough approximations

In RST, an information system is presented as a quadruple $S = (U, A, V, f)$, where U is a non-empty finite set of objects, called as universe; A is the union of conditional attribute set C and decision attribute set D, and $C \cap D = \emptyset$; $V = \bigcup_{a \in A} V_a$ is the union of attribute domains, where V_a is the domain of the attribute a; f is an information function and is presented as $f : U \times A \to V$, which means that $\forall a \in A$ and $\forall x \in U$, there exists $f(x, a) \in V_a$, where $f(x, a)$ is the value of the object x w.r.t. the attribute a.

$\forall x \in U$, the available information of x can be described by a vector as

$$V_A(x) = \begin{bmatrix} f(x, a_1) & f(x, a_2) & \cdots & f(x, a_m) \end{bmatrix}. \tag{1}$$

$\forall B \subseteq C$, an indiscernibility relation R w.r.t. the attribute set B can be defined as

$$R = \{(x, y) \in U \times U | f(x, a) = f(y, a), \forall a \in B\}. \tag{2}$$

The universe U is divided by the indiscernibility relation R into a family of equivalence classes $U/R = \{[x]_R | x \in U\}$, where $[x]_R = \{y \in U | (x, y) \in R\}$. $\forall X \subseteq U$, the lower and upper approximations of X w.r.t. the indiscernibility relation R are defined as

$$\underline{R}(X) = \{x \in U | [x]_R \subseteq X\} \quad and \quad \overline{R}(X) = \{x \in U | [x]_R \cap X \neq \emptyset\}, \tag{3}$$

respectively. The pair $[\underline{R}(X), \overline{R}(X)]$ is called as Pawlak's rough set of X w.r.t. the attribute set B. The sets $Bn(X) = \overline{R}(X) - \underline{R}(X)$, $Pos_R(X) = \underline{R}(X)$ and $Neg_R(X) = U - \overline{R}(X)$ are called as the boundary, positive region and negative region of X w.r.t. the attribute set B, respectively.

2.2. Matrix-based rough set approach

In matrix-based rough set approach, $\forall X \subseteq U$ is described by a n-col vector as

$$G(X)_{n \times 1} = \begin{bmatrix} g_1 & g_2 & \cdots & g_n \end{bmatrix}^T, \tag{4}$$

where T denotes the transpose operation. $n = |U|$ and $|\bullet|$ presents the cardinality of $\bullet$. g_i is a characterize function as

$$g_i = \begin{cases} 1 & , \quad x_i \in X \\ 0 & , \quad x_i \notin X \end{cases}. \tag{5}$$

The indiscernibility relation R is presented as a matrix $M_{n \times n} = (m_{ij})_{n \times n}$, where

$$m_{ij} = \begin{cases} 1 & , \quad (x_i, x_j) \in R \\ 0 & , \quad (x_i, x_j) \notin R \end{cases}. \tag{6}$$

$\Lambda_{n \times n}$ is an induced diagonal matrix from $M_{n \times n}$ and defined as

$$\Lambda_{n \times n} = diag(\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \ldots, \frac{1}{\lambda_n}), \tag{7}$$

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.

where $\lambda_i = \sum_{j=1}^{n} m_{ij}$, $1 \le i \le n$. It is obvious that $\lambda_i = |[x_i]_R|$. Then $\Lambda_{n\times n}$ can also be defined as

$$\Lambda_{n\times n} = diag(\frac{1}{|[x_1]_R|}, \frac{1}{|[x_2]_R|}, \ldots, \frac{1}{|[x_n]_R|}). \tag{8}$$

$H(X)_{n\times 1}$ is a n-col basic vector and defined as

$$H(X)_{n\times 1} = \begin{bmatrix} h_1 & h_2 & \cdots & h_n \end{bmatrix}^T = \Lambda_{n\times n} \cdot \Omega_{n\times 1} \tag{9}$$

where $\Omega_{n\times 1} = M_{n\times n} \cdot G(X)_{n\times 1}$ and $\cdot$ is dot product. Then

$$h_i = \frac{\sum_{j=1}^{n} m_{ij} g_j}{\sum_{j=1}^{n} m_{ij}} = \frac{\sum_{j=1}^{n} m_{ij} g_j}{\lambda_i} = \frac{|[x_i]_R \cap X|}{|[x_i]_R|}. \tag{10}$$

Obviously, h_i indicates the rough membership degree [36] of the object x_i in X and $0 \le h_i \le 1$. Four cut matrices of $H(X)_{n\times 1}$ were defined in [38] as follows:

(1) $H(X)_{n\times 1}^{[\mu,\nu]} = (h^{[\mu,\nu]})_{n\times 1}$, where if $\mu \le h_i \le \nu$ then $h^{[\mu,\nu]} = 1$, otherwise $h^{[\mu,\nu]} = 0$;
(2) $H(X)_{n\times 1}^{(\mu,\nu]} = (h^{(\mu,\nu]})_{n\times 1}$, where if $\mu < h_i \le \nu$ then $h^{(\mu,\nu]} = 1$, otherwise $h^{(\mu,\nu]} = 0$;
(3) $H(X)_{n\times 1}^{[\mu,\nu)} = (h^{[\mu,\nu)})_{n\times 1}$, where if $\mu \le h_i < \nu$ then $h^{[\mu,\nu)} = 1$, otherwise $h^{[\mu,\nu)} = 0$;
(4) $H(X)_{n\times 1}^{(\mu,\nu)} = (h^{(\mu,\nu)})_{n\times 1}$, where if $\mu < h_i < \nu$ then $h^{(\mu,\nu)} = 1$, otherwise $h^{(\mu,\nu)} = 0$.

In matrix-based rough set approach, the basic vectors of the lower and upper approximations of X w.r.t. the attribute set B can be defined respectively by two cut matrices of $H(X)_{n\times 1}$ as follows,

$$G(\underline{R}(X)) = H(X)_{n\times 1}^{[1,1]} \quad and \quad G(\overline{R}(X)) = H(X)_{n\times 1}^{(0,1]}. \tag{11}$$

Meanwhile, the basic vectors of boundary, positive and negative regions of X w.r.t. the attribute set B can be defined respectively as follows, $G(Bn_R(X)) = H(X)_{n\times 1}^{(0,1)}$, $G(Pos_R(X)) = H(X)_{n\times 1}^{[1,1]}$ and $G(Neg_R(X)) = H(X)_{n\times 1}^{[0,0]}$.

3. Composition of rough approximations of global concepts

A distributed information system can be viewed as a family of geographically distributed locations information systems and is presented as $DS = (U, A, V, f) = \bigcup_{i=1}^{m} S_i$, where $S_i = (U_i, A_i, V_i, f_i)$ and $i = 1, \ldots, m$. The information system S_i is regraded as a subsystem of DS. For a distributed information system and its subsystems, the following items hold.

- For any two subsystems S_i and S_j, if $i \ne j$, then their universes are independence of each other, i.e., $U_i \cap U_j = \emptyset$;
- $U = \bigcup_{i=1}^{m} U_i$;
- $A = A_i$;
- $V = \bigcup_{i=1}^{m} V_i$;
- $f = f_i$;
- Let $X_i \subseteq U_i$ and $X \subseteq U$, then $X = \bigcup_{i=1}^{m} X_i$, where X_i is a local concept in the subsystem while X is a global concept in the distributed information system.

Example 1. A distributed information system DS is shown in Table 1, in which there are three subsystems S_1, S_2 and S_3. The universe of DS is $U = U_1 \cup U_2 \cup U_3$, in where $U_1 = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, $U_2 = \{x_7, x_8, x_9, x_{10}\}$ and $U_3 = \{x_{11}, x_{12}, x_{13}, x_{14}, x_{15}\}$ are three universes of S_1, S_2 and S_3, respectively. The attribute set $A = C \cup D$, in where the condition attribute set $C = \{a_1, a_2, a_3\}$ and the decision attribute set $D = \{d\}$.

Obviously, $U_1 \cap U_2 = U_1 \cap U_3 = U_2 \cap U_3 = \emptyset$.

Table 1
A distribution information table.

U_1	a_1	a_2	a_3	d	U_2	a_1	a_2	a_3	d	U_3	a_1	a_2	a_3	d
x_1	2	2	1	1	x_7	1	1	2	1	x_{11}	2	2	1	0
x_2	2	2	1	0	x_8	2	3	1	0	x_{12}	1	1	2	1
x_3	1	2	2	0	x_9	2	2	1	1	x_{13}	1	2	1	1
x_4	1	1	2	1	x_{10}	2	2	1	1	x_{14}	2	3	1	0
x_5	2	2	1	1						x_{15}	2	2	1	1
x_6	1	1	2	1										

Definition 1. Let $DS = (U, A, V, f) = (\bigcup_{i=1}^{m} U_i, A, \bigcup_{i=1}^{m} V_i, f)$ be a distributed information system. $\forall X \subseteq U$ can be described as a n-col vector as

$$G(X) = \begin{bmatrix} G(X_1) & G(X_2) & \cdots & G(X_m) \end{bmatrix}^T, \tag{12}$$

where $G(X_i)$ is the vector description of X_i and $n = |U|$.

Example 2 (*Continuation of Example 1*)**.** Let $X_1 = \{x_1, x_4, x_5, x_6\}$, $X_2 = \{x_7, x_9, x_{10}\}$ and $X_3 = \{x_{12}, x_{13}, x_{15}\}$. We give the vector descriptions of X_1, X_2 and X_3 by Eq. (4) as follows.

$$G(X_1) = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T$$
$$G(X_2) = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}^T$$
$$G(X_3) = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix}^T$$

Then the vector description of $X = \bigcup_{i=1}^{3} X_i$ is

$$G(X) = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}^T.$$

In a distributed information system, $G(X)$ is a vector description of global concept X and $G(X_i)$ is a vector description of local concept X_i. $G(X)$ is composed by all $G(X_i)$ and may be regarded as a higher level vector than $G(X_i)$.

Let DM be a matrix of indiscernibility relation on universe U in the distributed information system DS. Analogously, M^{ii} is the matrix of indiscernibility relation on universe U_i in the subsystem S_i. Then DM can be composed by combining all matrices of indiscernibility relations of subsystems as well as the dependency matrices among subsystems. The matrix of indiscernibility relation of subsystem can be computed by Eq. (6). In the following, we give a definition of the dependency matrices between any two subsystems.

Definition 2. Let $S_i = (U_i, A, V_i, f)$ and $S_j = (U_i, A, V_j, f)$ be two subsystems of the distributed information system DS, where $A = C \cup D$. Let $n_i = |U_i|$, $n_j = |U_j|$, $x_l \in U_i$ and $x_k \in U_j$, where $l = 1, \ldots, n_i$ and $k = 1, \ldots, n_j$. $\forall B \subseteq C$, the dependency matrices between S_i and S_j are defined as follows:

- $M_{n_i\times n_j}^{ij} = (m_{lk}^{ij})_{n_i\times n_j}$ is the matrix of S_i depending on S_j, where

$$m_{lk}^{ij} = \begin{cases} 1, & V_B(x_l) = V_B(x_k) \\ 0, & otherwise \end{cases}, \tag{13}$$

- $M_{n_j\times n_i}^{ji} = (m_{kl}^{ji})_{n_j\times n_i}$ is the matrix of S_i depended by S_j, where

$$m_{kl}^{ji} = \begin{cases} 1, & V_B(x_k) = V_B(x_l) \\ 0, & otherwise \end{cases}. \tag{14}$$

$V_B(x_l) = V_B(x_k)$ or $V_B(x_k) = V_B(x_l)$ mean that x_l is indiscernible with x_k w.r.t. the attribute set B.

Obviously, $M_{n_i\times n_j}^{ij} = M_{n_j\times n_i}^{ji}{}^T$. The matrix of indiscernibility relation M in the distributed information system DS can be

composed by

$$M_{n\times n}=\begin{bmatrix} M^{11}_{n_1\times n_1} & \cdots & M^{1m}_{n_1\times n_m}\\ \vdots & \ddots & \vdots \\ M^{m1}_{n_m\times n_1} & \cdots & M^{mm}_{n_m\times n_m}\end{bmatrix}. \tag{15}$$

Example 3 (*Continuation of Example 1*)**.** The matrices of indiscernibility relation of subsystems S_1, S_2 and S_3 can be computed by Eq. (6), the results are as follows:

$$M^{11}_{n_1\times n_1}=\begin{bmatrix}1&1&0&0&1&0\\1&1&0&0&1&0\\0&0&1&0&0&0\\0&0&0&1&0&1\\1&1&0&0&1&0\\0&0&0&1&0&1\end{bmatrix},\ M^{22}_{n_2\times n_2}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&1\\1&0&1&1\end{bmatrix}$$

and $M^{33}_{n_3\times n_3}=\begin{bmatrix}1&0&0&0&1\\0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\1&0&0&0&1\end{bmatrix}$.

The dependency matrices among S_1, S_2 and S_3 can be computed by Definition 2. The results are as follows:

$$M^{12}_{n_1\times n_2}=\begin{bmatrix}0&0&1&1\\0&0&1&1\\0&0&0&0\\1&0&0&0\\0&0&1&1\\1&0&0&0\end{bmatrix}\qquad M^{13}_{n_1\times n_3}=\begin{bmatrix}1&0&0&0&1\\1&0&0&0&1\\0&0&0&0&0\\0&1&0&0&0\\1&0&0&0&1\\0&1&0&0&0\end{bmatrix}$$

$$M^{23}_{n_2\times n_3}=\begin{bmatrix}0&1&0&0&0\\0&0&0&1&0\\1&0&0&0&1\\1&0&0&0&1\end{bmatrix}$$

$$M^{21}_{n_2\times n_1}=\begin{bmatrix}0&0&0&1&0&1\\0&0&0&0&0&0\\1&1&0&0&1&0\\1&1&0&0&1&0\end{bmatrix}$$

$$M^{13}_{n_1\times n_3}=\begin{bmatrix}1&1&0&0&1&0\\0&0&0&1&0&1\\0&0&0&0&0&0\\0&0&0&0&0&0\\1&1&0&0&1&0\end{bmatrix}$$

$$M^{23}_{n_2\times n_3}=\begin{bmatrix}0&0&1&1\\1&0&0&0\\0&0&0&0\\0&1&0&0\\0&0&1&1\end{bmatrix}$$

If $\forall m^{ij}_{lk}=0$ in $M^{ij}_{n_i\times n_j}$, then S_i is not depending on S_j. Accordingly, $\forall m^{ji}_{kl}=0$ in $M^{ji}_{n_j\times n_i}$ and S_i is not depended by S_j. $\forall M^{ij}_{n_i\times n_j}$, $\exists m^{ij}_{lk}=0$, then all subsystems do not depend on each other. For this case, we can get rough approximations of global concept in the distributed information systems by uniting all rough approximations of local concept in the subsystems directly as follows.

$$G(\underline{R}(X))=\begin{bmatrix}G(\underline{R}(X_1)) & G(\underline{R}(X_2)) & \cdots & G(\underline{R}(X_m))\end{bmatrix}^T \tag{16}$$

$$G(\overline{R}(X))=\begin{bmatrix}G(\overline{R}(X_1)) & G(\overline{R}(X_2)) & \cdots & G(\overline{R}(X_m))\end{bmatrix}^T \tag{17}$$

Eqs. (16) and (17) can also be written as follows:

$$H(X)^{[1,1]}_{n\times 1}=\begin{bmatrix}H(X_1)^{[1,1]}_{n_1\times 1} & H(X_2)^{[1,1]}_{n_2\times 1} & \cdots & H(X_m)^{[1,1]}_{n_m\times 1}\end{bmatrix}^T \tag{18}$$

$$H(X)^{[0,1]}_{n\times 1}=\begin{bmatrix}H(X_1)^{[0,1]}_{n_1\times 1} & H(X_2)^{[0,1]}_{n_2\times 1} & \cdots & H(X_m)^{[0,1]}_{n_m\times 1}\end{bmatrix}^T \tag{19}$$

In the following, we discuss how to compose the rough approximations of global concept in the distributed information under the case when there are at least two subsystems depending on each other. Based on the matrix $M^{ij}_{n_i\times n_j}$, we may define a n_i-col vector $\Gamma^{ij}_{n_i\times 1}=(\gamma^{ij}_l)_{n_i\times 1}$ to describe the degree of each object in S_i depending on S_j, where $\gamma^{ij}_l=\sum_{k=1}^{n_j} m^{ij}_{lk}$. Then we use a n_i-col vector $\delta\Gamma^{i}_{n_i\times 1}$ to present the degree of each object in S_i depending on subsystems except for itself. $\delta\Gamma^{i}_{n_i\times 1}$ is called as a dependency vector of subsystem S_i.

$$\delta\Gamma^{i}_{n_i\times 1}=\sum_{j\neq i}^{m}\Gamma^{ij}_{n_i\times 1}=(\delta\gamma^{ij}_l)_{n_i\times 1}, \tag{20}$$

where $\delta\gamma^{ij}_l=\sum_{j\neq i}^{m}\gamma^{ij}_l$.

In matrix-based rough set approach, the induced diagonal matrix based by the matrix description of binary relation is one of main components in computing rough approximations. By Eq. (7), λ^i_l of the object x_l w.r.t. the matrix of indiscernibility relation M in the distributed information system DS can be written as

$$\lambda^i_l=\sum_{j=1}^{m}\gamma^{ij}_l=\gamma^{ii}_l+\delta\gamma^{ij}_l. \tag{21}$$

Obviously, λ^i_l can be divided into two parts, γ^{ii}_l and $\delta\gamma^{ij}_l$. γ^{ii}_l is the cardinality of the equivalence class of the object $x_l\in U_i$ in the subsystem S_i while $\delta\gamma^{ij}_l$ is the number of indiscernible objects in the other subsystems.

Example 4 (*Continuation of Example 3*)**.** The dependency vectors of S_1, S_2 and S_3 are as follows:

$$\delta\Gamma^1_{6\times 1}=\begin{bmatrix}4&4&0&2&4&2\end{bmatrix}^T,\ \delta\Gamma^2_{4\times 1}=\begin{bmatrix}3&1&5&5\end{bmatrix}^T,$$
$$\delta\Gamma^3_{5\times 1}=\begin{bmatrix}5&3&0&1&5\end{bmatrix}^T.$$

The vector $\Omega_{n\times 1}$ in [38] gave the cardinalities of intersections of concept and equivalence classes. To consider the interconnections of subsystems in a distributed information system, we use a n_i-col vector $\Omega^{ij}_{n_i\times 1}$ to present the cardinalities of intersections of $G(X_j)_{n_j\times 1}$ and $M^{ij}_{n_i\times n_j}(l,:)$, where $M^{ij}_{n_i\times n_j}(l,:)$ is the lth row of matrix $M^{ij}_{n_i\times n_j}$ and $l=1,\ldots,n_i$. Then, we have

$$\Omega^{ij}_{n_i\times 1}=M^{ij}_{n_i\times n_j}\cdot G(X_j)_{n_j\times 1}=(\omega^{ij}_l)_{n_i\times 1},$$

where $G(X_j)_{n_j\times 1}=(g^j_k)_{n_j\times 1}$, $\omega^{ij}_l=\sum_{l=1}^{n_j} m^{ij}_{lk}g^j_k$ and $k=1,\ldots,n_j$. Then, the vector $\Omega_{n\times 1}$ in the distributed information system can be composed as

$$\Omega_{n\times 1}=\begin{bmatrix}\Omega^1_{n_1\times 1} & \Omega^2_{n_2\times 1} & \cdots & \Omega^m_{n_m\times 1}\end{bmatrix}^T \tag{22}$$

where $n=\sum_{i=1}^{m}n_i$ and $\Omega^i_{n_i\times 1}=(\omega^i_l)_{n_i\times 1}=\sum_{j=1}^{m}\Omega^{ij}_{n_i\times 1}$. Let $\delta\Omega^{ii}_{n_i\times 1}=\sum_{j=1,j\neq i}^{m}\Omega^{ij}_{n_i\times 1}$, then

$$\Omega^{i}_{n_i\times 1}=\Omega^{ii}_{n_i\times 1}+\delta\Omega^{ii}_{n_i\times 1}, \tag{23}$$

where $\Omega^{ii}_{n_i\times 1}=(\omega^{ii}_l)_{n_i\times 1}$, $\delta\Omega^{ii}_{n_i\times 1}=(\delta\omega^{ii}_l)_{n_i\times 1}$ and $\delta\omega^{ii}_l=\sum_{j=1,j\neq i}^{m}\omega^{ij}_l$.

Example 5 (*Continuation of Example 3*)**.** For $i=1,2$ and 3, the vectors $\delta\Omega^{ii}_{n_i\times 1}$ are as follows:

$$\delta\Omega^{11}_{6\times 1}=\begin{bmatrix}3&3&0&2&3&2\end{bmatrix}^T,\ \delta\Omega^{22}_{4\times 1}=\begin{bmatrix}3&0&3&3\end{bmatrix}^T$$
$$\text{and } \delta\Omega^{33}_{5\times 1}=\begin{bmatrix}4&3&0&0&4\end{bmatrix}^T.$$

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.

Let $X = \sum_{i=1}^{m} X_i$ be a concept in the distributed information system DS. The basic vector $H(X)_{n\times 1}$ of the concept X can be composed by

$$H(X)_{n\times 1} = \begin{bmatrix} H(X_1)_{n_1\times 1} & H(X_2)_{n_2\times 1} & \cdots & H(X_m)_{n_m\times 1} \end{bmatrix}^T, \quad (24)$$

where $n = \sum_{i=1}^{m} n_i$, $H(X_i)_{n_i\times 1} = (h_l^i)_{n_i\times 1}$ and $h_l^i = \frac{\omega_l^i}{\lambda_l^i}$. Obviously, we can get $\omega_l^i = \omega_l^{ii} + \delta\omega_l^{ii}$ from Eq. (23). Combined with Eq. (21), we have

$$h_l^i = \frac{\omega_l^i}{\lambda_l^i} = \frac{\omega_l^{ii} + \delta\omega_l^{ii}}{\gamma_l^{ii} + \delta\gamma_l^{ii}}. \quad (25)$$

Assume that $h_l^{ii} = \frac{\omega_l^{ii}}{\gamma_l^{ii}}$ and $H(X_i)^i_{n_i\times 1} = (h_l^{ii})_{n_i\times 1}$. $H(X_i)^i_{n_i\times 1}$ is the basic vector of the concept X_i in the subsystem S_i.

Proposition 1. *The following items hold.*

(1) $\delta\omega_l^{ii} \le \delta\gamma_l^{ii}$;
(2) If $\delta\omega_l^{ii} < \delta\gamma_l^{ii}$, *then that*

(1) if $h_l^{ii} > 0$, *then* $h_l^i < h_l^{ii}$;
(2) if $h_l^{ii} = 0$ *and* $\delta\omega_l^{ii} = 0$, *then* $h_l^i = h_l^{ii}$;
(3) if $h_l^{ii} = 0$ *and* $\delta\omega_l^{ii} \neq 0$, *then* $1 > h_l^i > h_l^{ii}$;

(3) If $\delta\omega_l^{ii} = \delta\gamma_l^{ii} = 0$, *then* $h_l^i = h_l^{ii}$;
(4) If $\delta\omega_l^{ii} = \delta\gamma_l^{ii} \neq 0$, *then that*

(1) if $h_l^{ii} = 1$, *then* $h_l^i = h_l^{ii} = 1$;
(2) if $h_l^{ii} < 1$, *then* $1 > h_l^i > h_l^{ii}$.

Proof. (1) $\because \delta\omega_l^{ii} = \sum_{j=1,j\neq i}^{m} \omega_l^{ij} = \sum_{j=1,j\neq i}^{m}\sum_{k=1}^{n_j} m_{lk}^{ij} g_k^j \le \sum_{j=1,j\neq i}^{m}\sum_{k=1}^{n_j} m_{lk}^{ij} = \sum_{j=1,j\neq i}^{m} \gamma_l^{ij} = \delta\gamma_l^{ii}$. $\therefore \delta\omega_l^{ii} \le \delta\gamma_l^{ii}$ holds. The items (2), (3) and (4) can be proved easily and the proofs are omitted. □

Let $0 \le \mu \le \nu \le 1$, we use a n_i-col vector $\delta H(X_i)^{i[\mu,\nu]}_{n_i\times 1} = (\delta h_l^{ii[\mu,\nu]})_{n_i\times 1}$ to present the degree of $H(X_i)_{n_i\times 1}$ depending on subsystems except for the subsystem S_i. Here we define

$$\delta h_l^{ii[\mu,\nu]} = \begin{cases} 1, & \delta\gamma_l^{ii} = 0 \\ 1, & \mu \le \delta h_l^{ii} \le \nu \\ 0, & Otherwise \end{cases}. \quad (26)$$

Proposition 2. *Let* $H(X_i)^{[\mu,\nu]}_{n_i\times 1}$ *and* $H(X_i)^{i[\mu,\nu]}_{n_i\times 1}$ *are cut matrices of* $H(X_i)_{n_i\times 1}$ *and* $H(X_i)^i_{n_i\times 1}$, *respectively.* $*$ *is an operator of Hadamard product [39], then the following equation holds when* $\mu = \nu = 0$ *or* $\mu = \nu = 1$.

$$H(X_i)^{[\mu,\nu]}_{n_i\times 1} = H(X_i)^{i[\mu,\nu]}_{n_i\times 1} * \delta H(X_i)^{i[\mu,\nu]}_{n_i\times 1}$$

Proof. Let $\mu = \nu = 1$, we have $h_l^{i[1,1]} = (\frac{\omega_l^{ii} + \delta\omega_l^{ii}}{\gamma_l^{ii} + \delta\gamma_l^{ii}})^{i[1,1]}$ by Eq. (25). According to Eq. (26), there is $\delta h_l^{ii[1,1]} = 1$ if $\delta\omega_l^{ii} = \delta\gamma_l^{ii}$. If $\delta\omega_l^{ii} = \delta\gamma_l^{ii}$, we have $h_l^{i[1,1]} = h_l^{ii[1,1]} = h_l^{ii[1,1]} \times \delta h_l^{ii[1,1]}$. If $\delta\omega_l^{ii} < \delta\gamma_l^{ii}$, there are $\delta h_l^{ii[1,1]} = 0$ and $h_l^{i[1,1]} = \delta h_l^{ii[1,1]} = h_l^{ii[1,1]} \times \delta h_l^{ii[1,1]}$. Then, $h_l^{i[1,1]} = h_l^{ii[1,1]} \times \delta h_l^{ii[1,1]}$ holds for $\delta\omega_l^{ii} \le \delta\gamma_l^{ii}$. $\therefore H(X_i)^{[1,1]}_{n_i\times 1} = H(X_i)^{i[1,1]}_{n_i\times 1} * \delta H(X_i)^{i[1,1]}_{n_i\times 1}$ holds. Analogously, $H(X_i)^{[0,0]}_{n_i\times 1} = H(X_i)^{i[0,0]}_{n_i\times 1} * \delta H(X_i)^{i[0,0]}_{n_i\times 1}$ can be proved. Therefore, $H(X_i)^{[\mu,\nu]}_{n_i\times 1} = H(X_i)^{i[\mu,\nu]}_{n_i\times 1} * \delta H(X_i)^{i[\mu,\nu]}_{n_i\times 1}$ holds for $\mu = \nu = 0$ or $\mu = \nu = 1$. □

By Proposition 2, we can compose the matrix descriptions of lower and upper approximations of rough sets, respectively. The following Proposition may be used to compose the matrix descriptions of boundary regions of rough sets.

Proposition 3. $H(X_i)^{(\mu,\nu)}_{n_i\times 1} = H(X_i)^{i(\mu,\nu)}_{n_i\times 1} + \delta H(X_i)^{i(\mu,\nu)}_{n_i\times 1}$, *where* $\delta H(X_i)^{i(\mu,\nu)}_{n_i\times 1} = (H(X_i)^{i[\nu,1]}_{n_i\times 1} - H(X_i)^{[\nu,1]}_{n_i\times 1}) + (H(X_i)^{i[0,\mu]}_{n_i\times 1} - H(X_i)^{[0,\mu]}_{n_i\times 1})$.

Proof. By matrix descriptions of positive, boundary and negative regions, $h_l^{i(\mu,\nu)} + h_l^{i[0,\mu]} + h_l^{i[\nu,1]} = 1$. Then $h_l^{i(\mu,\nu)} = 1 - h_l^{i[0,\mu]} - h_l^{i[\nu,1]} = h_l^{ii(\mu,\nu)} + h_l^{ii[0,\mu]} + h_l^{ii[\nu,1]} - h_l^{i[0,\mu]} - h_l^{i[\nu,1]} = h_l^{ii(\mu,\nu)} + (h_l^{ii[0,\mu]} - h_l^{i[0,\mu]}) + (h_l^{ii[\nu,1]} - h_l^{i[\nu,1]})$. Hence, $H(X_i)^{(\mu,\nu)}_{n_i\times 1} = H(X_i)^{i(\mu,\nu)}_{n_i\times 1} + (H(X_i)^{i[\nu,1]}_{n_i\times 1} - H(X_i)^{[\nu,1]}_{n_i\times 1}) + (H(X_i)^{i[0,\mu]}_{n_i\times 1} - H(X_i)^{[0,\mu]}_{n_i\times 1})$. $\because \delta H(X_i)^{i(\mu,\nu)}_{n_i\times 1} = (H(X_i)^{i[\nu,1]}_{n_i\times 1} - H(X_i)^{[\nu,1]}_{n_i\times 1}) + (H(X_i)^{i[0,\mu]}_{n_i\times 1} - H(X_i)^{[0,\mu]}_{n_i\times 1})$, $\therefore H(X_i)^{(\mu,\nu)}_{n_i\times 1} = H(X_i)^{i(\mu,\nu)}_{n_i\times 1} + \delta H(X_i)^{i(\mu,\nu)}_{n_i\times 1}$ holds. □

Proposition 4. *The following items hold.*

(1) $H(X_i)^{(\mu,\nu]} = H(X_i)^{[\nu,\nu]} + H(X_i)^{(\mu,\nu)}$;
(2) $H(X_i)^{[\mu,\nu)} = H(X_i)^{[\mu,\mu]} + H(X_i)^{(\mu,\nu)}$.

Proof. The proofs of items (1) and (2) are similar to that of Proposition 3. □

Example 6 (*Continuation of Examples 4 and 5*)**.** Assume that we have obtained $H(X_i)^{i[1,1]}$ for $i = 1, 2, 3$ as follows.

$$H(X_1)^{1[1,1]} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T$$
$$H(X_2)^{2[1,1]} = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}^T$$
$$H(X_3)^{3[1,1]} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix}^T$$

The matrices $\delta H(X_i)^{i[1,1]}$ for $i = 1, 2, 3$ can be computed as follows.

$$\delta H(X_1)^{1[1,1]} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}^T$$
$$\delta H(X_2)^{2[1,1]} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$$
$$\delta H(X_3)^{3[1,1]} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix}^T$$

By Proposition 2, we can obtain matrix descriptions of lower approximations of global concept X in the distributed information system as

$$H(X)^{[1,1]} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}^T.$$

Example 6 shows that the lower approximations of global concepts in a distributed information system can be composed by the lower approximations of local concepts in its subsystems. Analogously, the upper approximations and boundary regions of global concepts can be composed by Propositions 3 and 4, respectively.

4. Algorithm for composing rough approximations of global concepts

Based on the approach of composing approximations in distributed information systems processing which was discussed in the previous section, we design a corresponding algorithm in order to demonstrate its performance and the main influence factors. The detailed design of the algorithm is shown in Algorithm 1.

Three functions are called by Algorithm 1 as follows:

- *fun-GammaOmega*
  The function of *fun-matrix* is to compute vectors $\Gamma^{ij}_{n_i\times 1}$, $\Gamma^{ji}_{n_j\times 1}$, $\Omega^{ij}_{n_i\times 1}$ and $\Omega^{ji}_{n_j\times 1}$ between subsystems S_i and S_j, respectively. The corresponding algorithm is designed in the following.

Algorithm 1: An Algorithm for composing rough approximations on the distributed information system.

Data: S_i, $H(X_i)^{[\mu,\nu]}$ and $H(X_i)^{(\mu,\nu)}$ for $i = 1, \cdots, m$.
Result: $H(X)^{[\mu,\nu]}$, $H(X)^{(\mu,\nu)}$ and $H(X)^{(\mu,\nu)}$.

1 **for** $i \leftarrow 1$ **to** $m-1$ **do**
2 **for** $j \leftarrow i+1$ **to** m **do**
3 Compute $\Gamma^{ij}_{n_i\times 1}$, $\Gamma^{ji}_{n_j\times 1}$, $\Omega^{ij}_{n_i\times 1}$ and $\Omega^{ji}_{n_j\times 1}$ by calling *fun-GammaOmega* function, respectively;
4 **end**
5 **end**
6 **for** $i \leftarrow 1$ **to** m **do**
7 Compute $\delta H(X_i)^{i[\mu,\nu]}$ by calling *fun-deltavector* function;
8 **if** $\delta H(X_i)^{i[\mu,\nu]} == NULL$ **then**
9 $H(X_i)^{[\mu,\nu]} \leftarrow H(X_i)^{i[\mu,\nu]}$;
10 $H(X_i)^{(\mu,\nu)} \leftarrow H(X_i)^{i(\mu,\nu)}$;
11 $H(X_i)^{(\mu,\nu)} \leftarrow H(X_i)^{i(\mu,\nu)}$;
12 **end**
13 **else**
14 Compute $H(X_i)^{[\mu,\nu]}$ by calling *fun-HadamardProduct2* function;
15 $H(X_i)^{(\mu,\nu)} \leftarrow H(X_i)^{i(\mu,\nu)} + \delta H(X_i)^{i(\mu,\nu)}$ /* According to Proposition 3. */;
16 $H(X_i)^{(\mu,\nu)} \leftarrow H(X_i)^{[\mu,\nu]} + H(X_i)^{(\mu,\nu)}$ /* According to Proposition 4. */;
17 **end**
18 **end**
19 return $H(X)^{[\mu,\nu]}$, $H(X)^{[\mu,\nu]}$, $H(X)^{(\mu,\nu)}$ and $H(X)^{(\mu,\nu)}$;

Function fun-GammaOmega(U_x, U_y, B, $G(X_x)$, $G(X_y)$, β)

1 $count \leftarrow 0$;
2 **foreach** $u_i \in U_x$ **do**
3 **foreach** $u_j \in U_y$ **do**
4 **if** $V_B(u_i) \neq V_B(u_j)$ **then** $M(i,j) \leftarrow 0$;
5 **else**
6 $M(i,j) \leftarrow 1$;
7 $count \leftarrow count + 1$;
8 **end**
9 **end**
10 **end**
11 **if** $\frac{count}{|U_x|} < \beta$ **then** return false ;
12 **else**
13 $M^T \leftarrow transpose(M)$ /* *transpose* returns the transpose of a matrix. */;
14 $\Gamma^{ij}_{n_i\times 1} \leftarrow sum(M)$;
15 $\Gamma^{ji}_{n_j\times 1} \leftarrow sum(M^T)$ /* *sum* returns the sum of row elements of a matrix. */;
16 $\Omega^{ij}_{n_i\times 1} \leftarrow M \cdot G(X_y)$;
17 $\Omega^{ji}_{n_j\times 1} \leftarrow M^T \cdot G(X_x)$;
18 return $\Gamma^{ij}_{n_i\times 1}$, $\Gamma^{ji}_{n_j\times 1}$, $\Omega^{ij}_{n_i\times 1}$ and $\Omega^{ji}_{n_j\times 1}$.
19 **end**

In the algorithm of *fun-GammaOmega*, the input data includes $U_x, U_y, B, G(X_x), G(X_y)$ and β, where U_x and U_y are universes of subsystems S_i and S_j, respectively. B is a subset of the conditional attribute set C. $G(X_x)$ and $G(X_y)$ are concepts of subsystems S_i and S_j, respectively. β is an index to characterize the degree of subsystems S_i and S_j depending on each other. The functions of *transpose* and *sum* are called by the algorithm of *fun-GammaOmega* and can be found in GNU Octave or Matlab, respectively. The time complexity of this algorithm can be analyzed as follows. Steps 2–10 are to compute the dependency matrix between subsystems S_i and S_j. The time complexity of steps 2–10 is $O(|U_x||U_y||B|)$. The step 13 is to transpose the dependency matrix and its complexity is $O(|U_x||U_y|)$. Steps 14–15 are to compute $\Gamma^{ij}_{n_i\times 1}$, $\Gamma^{ji}_{n_j\times 1}$, $\Omega^{ij}_{n_i\times 1}$ and $\Omega^{ji}_{n_j\times 1}$, respectively. The time complexity of steps 14–17 is $O(|U_x||U_y|)$. Hence, the time complexity of *fun-GammaOmega* is $O(|U_x||U_y||B|)$.

- *fun-deltavector*
  The function of *fun-deltavector* is to compute cut matrix $\delta H(X_i)^{i[\mu,\nu]}$. The input data of *fun-deltavector* includes $\Gamma^{ij}_{n_i\times 1}$, $\Omega^{ij}_{n_i\times 1}$, μ and ν, where $j = 1, \ldots, m$. An algorithm of *fun-deltavector* is designed according to Eqs. (20), (23) and (26) in the following. Its time complexity is $O(n_i + m)$.

Function fun-deltavector($\Gamma^{ij}_{n_i\times 1}$, $\Omega^{ij}_{n_i\times 1}$, μ, ν)

1 $\delta\Gamma^{ii}_{n_i\times 1} \leftarrow (0)_{n_i\times 1}$;
2 $\delta\Omega^{ii}_{n_i\times 1} \leftarrow (0)_{n_i\times 1}$;
3 **for** $j \leftarrow 1$ *and* $j \neq i$ **to** m **do**
4 **if** $\Gamma^{ij}_{n_i\times 1} \neq NULL$ **then** $\delta\Gamma^{ii}_{n_i\times 1} \leftarrow \delta\Gamma^{ii}_{n_i\times 1} + \Gamma^{ij}_{n_i\times 1}$;
5 **if** $\Omega^{ij}_{n_i\times 1} \neq NULL$ **then** $\delta\Omega^{ii}_{n_i\times 1} \leftarrow \delta\Omega^{ii}_{n_i\times 1} + \Omega^{ij}_{n_i\times 1}$;
6 **end**
7 **for** $l \leftarrow 1$ **to** n_i **do**
8 **if** $sum(\delta\Gamma^{ii}_{n_i\times 1}) == 0$ **then** $\delta H(X_i)^{i[\mu,\nu]} \leftarrow NULL$;
9 **else**
10 **if** $\delta\omega_l^{ii} == \delta\gamma_l^{ii}$ **then** $\delta h_l^{ii[\mu,\nu]} \leftarrow 1$;
11 **else**
12 **if** $\mu \leq \frac{\delta\omega_l^{ii}}{\delta\gamma_l^{ii}} \leq \nu$ **then** $\delta h_l^{ii[\mu,\nu]} \leftarrow 1$;
13 **else** $\delta h_l^{ii[\mu,\nu]} \leftarrow 0$;
14 **end**
15 **end**
16 **end**
17 return $\delta H(X_i)^{i[\mu,\nu]}$;

- *fun-HadamardProduct2*
  The function of *fun-Hadamardproduct2* is compute the Hadamard product of two 2-dimension matrices. The input matrices $X1$ and $X2$ are the same in size. n_x and n_y are the numbers of rows and columns of them, respectively. The corresponding algorithm is designed in the following. Its time complexity is $O(n_x \cdot n_y)$.

Function fun-HadamardProduct2($X1, X2$)

1 $s_1 \leftarrow size(X1), s_2 \leftarrow size(X2)$ /* *size* returns the row and column numbers of a matrix. */;
2 **if** $s_1 \neq s_2$ **then** return false ;
3 **else**
4 **for** $i = 1$ **to** $s1(1)$ **do**
5 **for** $j = 1$ **to** $s1(2)$ **do** $Y(i,j) \leftarrow X1(i,j) * X2(i,j)$;
6 **end**
7 return Y.
8 **end**

In Algorithm 1, the steps 1–5 are to compute the vectors $\Gamma^{ij}_{n_i\times 1}$, $\Gamma^{ji}_{n_j\times 1}$, $\Omega^{ij}_{n_i\times 1}$ and $\Omega^{ji}_{n_j\times 1}$ related to the dependency matrices

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.

Table 2
The basic information of data sets.

Data sets	Original objects	Attributes	Classes	Testing objects	Attributes selected
Abalone	4177	8	29	4177	2
Car Evaluation	1728	6	4	8640	4
Libras Movement	360	91	15	3600	20
Optdigits	5620	64	9	5620	20

between subsystems S_i and S_j for $i, j = 1, \ldots .m$. The key of these steps is call the function *fun-GammaOmega*. In algorithm of *fun-GammaOmega*, β is an index to decide whether to consider the dependency between two subsystems and $\frac{count}{|U_x|}$ indicates the degree of S_i depending on S_j. In some cases scenario, it needs to consider the dependency among subsystems completely, i.e., $\beta = 0$. Then, the time complexity of the steps 1–5 is $O(\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}|U_i|\cdot|U_j|\cdot|B|)$. The step 17 is to compute cut matrix $\delta H(X_i)^{i[\mu,\nu]}$ by calling the function *fun-deltavector*. The steps 14–16 are to compose the rough approximations and boundary regions of global concepts in the distributed information system. Hence, the time complexity of Algorithm 1 is $O(\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}|U_i|\cdot|U_j|\cdot|B| + \sum_{i=1}^{m}\sum_{j=1}^{m}|U_i|) \approx O(\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}|U_i|\cdot|U_j|\cdot|B|)$.

5. Experimental estimation

We experimented our approach on a personal computer with Debian 9 GNU/Linux and Intel(R) Core(TM) i5 CPU and 4 GB memory. The Algorithm 1 and SACAM [38] were coded in GNU Octave. We downloaded four data sets from machine learning data repository, University of California at Irvine [35]. Their basic information are listed in Table 2. The object numbers of the data sets *Car Evaluation* and *Libras Movement* were too small to test, hence we replicated them five-fold and tenfold in our experiments, respectively.

In order to estimate the performance of Algorithm 1 and SACAM, we designed three groups of experiments as follows: the same size of subsystems, the different size of subsystems and the different number of subsystems.

- The same size of subsystems
  We randomly partitioned four testing data sets listed in Table 2 into 15 same size subsets, respectively. These four testing data sets were regarded as four distributed data sets. Each subset was regarded as an information subsystem during the experiment. At first, we computed the rough approximations and boundary regions of local concepts on each of subsystems by SACAM and saved these results. Then, we performed Algorithm 1 to compose the rough approximations of global concepts on these four testing data sets. In order to estimate the performance of Algorithm 1 under different data dependency degrees, we experimented our approach when $\beta = 0$, $\beta = 0.3$ and $\beta = 0.5$, respectively. To compare the performance between Algorithm 1 and SACAM, we also compute the rough approximations of global concepts by running SACAM on the testing data sets. The computational time taken by Algorithm 1 and SACAM were shown in Fig. 1.
  From Fig. 1, one can see that the computational time taken by Algorithm 1 is less than that of SACAM on these four testing data sets. On the testing data set *Libras*, the computational time taken by Algorithm 1 keeps almost unchanged for $\beta = 0$, $\beta = 0.3$ and $\beta = 0.5$. The reason is that the testing data set *Libras* was generated by replicating the original *Libras* tenfold, which leads to the high dependency degree among subsystems. For other testing data sets, the computational time taken by Algorithm 1 reduces while the β value increases.

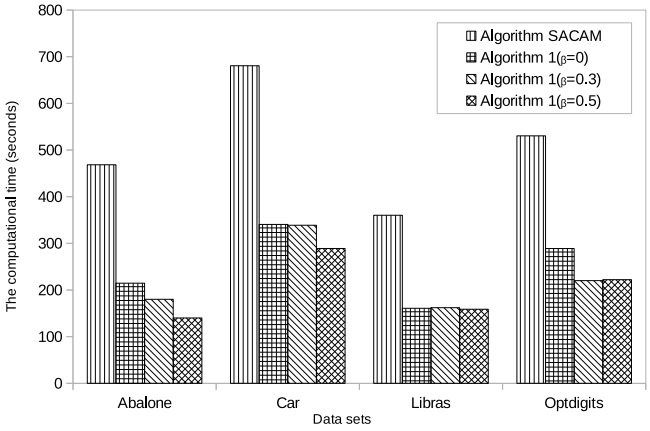

Fig. 1. Comparison of the computational time for SACAM and Algorithm 1.

- The different size of subsystems
  To study the correlation between the performance of Algorithm 1 and the distribution of data, we randomly partitioned four testing data sets listed in Table 2 into 15 subsets according to the variance of the object numbers of subsets $\sigma = 1866$, $\sigma = 4200$, $\sigma = 7466$, $\sigma = 11666$ and $\sigma = 16800$, respectively, where σ is the symbol of the variance. In this group of experiments, we do not consider the influence of the dependencies among subsystems and let $\beta = 0$. We compute the rough approximations and boundary regions of local concepts by SACAM and use Algorithm 1 to compose the rough approximations of global concepts. The computational time taken by Algorithm 1 under the different variances were shown in Fig. 2.
  From Fig. 2, we can see that the computational time taken by Algorithm 1 is a downward trend when the variance increases for these four testing data sets.
- The different number of subsystems
  This group of experiments is to estimate the correlation between the performance of Algorithm 1 and the number of subsystems. Firstly, we randomly partitioned these testing data set into 5, 10, 15, 20 and 25 same subsets in size, respectively. Then, we use Algorithm 1 to compose the rough approximations of global concepts under different number subsystems, respectively. The trends of the computational time taken by Algorithm 1 on four testing data sets are shown in Fig. 3.
  From Fig. 3, we can see that the trend of computational time taken by Algorithm 1 grows with the increasing number of subsystems for these four testing data sets. The trend of computational time taken by Algorithm 1 is consistent with its time complexity. The time complexity of Algorithm 1 is $O(\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}|U_i|\cdot|U_j|\cdot|B|)$. If the subsystems are same in size, the time complexity of Algorithm 1 is $O(\sum_{i=1}^{m-1}\frac{|U|}{m}\cdot\sum_{j=i+1}^{m}\frac{|U|}{m}\cdot|B|) = O(\frac{(m-1)^2}{m^2}\cdot|U|^2\cdot|B|)$, where $m \geq 2$. Hence, the time complexity of Algorithm 1 grows with the increasing number of the subsystems under this case.

The experimental estimations show that Algorithm 1 can reduce the computational time of computing rough approximations in distributed information systems processing than its counterpart SACAM. In addition, the performance of Algorithm 1 may be affected by the dependency degrees among subsystems, the distribution of the object numbers of subsystems and the number of subsystems.

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.

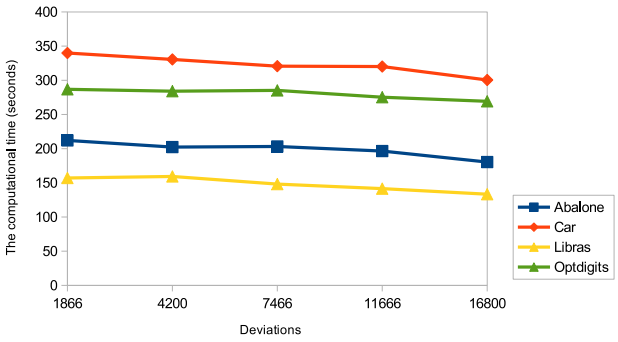

Fig. 2. The computational time taken by Algorithm 1 vs. the data distributions.

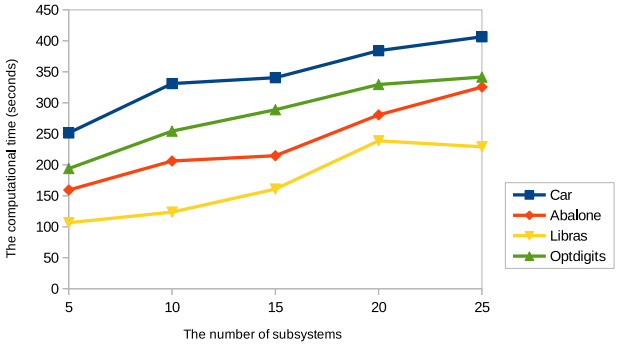

Fig. 3. The computational time taken by Algorithm 1 vs. the subsystem number.

6. Conclusion and future works

Mining data that distributed across different sites has become one of important tasks in many scientific research fields. An effective distributed data mining method can provide a timely and useful knowledge to help people develop a global decision-making solution. RST is a powerful information processing tool that can process uncertain and inconsistent information. Approximation computing is a key and time-consuming step in data mining researches based on RST. In this paper, we introduced an approach of composing the global rough approximations on the distributed information system by using the existing local ones and some necessary computations or matrix operations. With some numerical examples and experimental estimations, we have obtained the following conclusions. (1) The approach can compose rough approximations of global concepts without beginning from the scratch; (2) The approach can effectively reduce the computational time than its counterpart; (3) The performance of the approach relates to the distribution of data.

In the future, we will further improve the approach to fit different data setting. In addition, we will develop the parallel algorithms for big data mining.

Acknowledgments

This work is supported by Provincial Key Platforms and Major Scientific Research Projects of Guangdong Universities, China (No. 2018KTSCX235), Jiangmen Planning Project on Basic and Theoretical Sciences, China (No. 2017JC01021) and Natural Science Foundation of Guangdong Province (No. 2016A030310003), China.

References

[1] K. Eurek, P. Sullivan, M. Gleason, D. Hettinger, D. Heimiller, A. Lopez, An improved global wind resource estimate for integrated assessment models, Energy Econ. 64 (2017) 552–567.
[2] M.H. Rafiei, H. Adeli, A novel unsupervised deep learning model for global and local health condition assessment of structures, Eng. Struct. 156 (2018) 598–607.
[3] J. da Silva Fink, E.D. de Mello, M.G. Beghetto, V.C. Luft, S.M. de Jezus Castro, P.D. de Mello, Nutritional assessment score: A new tool derived from subjective global assessment for hospitalized adults, Clin. Nutrition 37 (2) (2018) 706–711.
[4] B. Khaleghi, A. Khamis, F.O. Karray, S.N. Razavi, Multisensor data fusion: A review of the state-of-the-art, Inf. Fusion 14 (1) (2013) 28–44.
[5] L.A. Zadeh, Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems, Soft Comput. (ISSN: 1432-7643) 2 (1) (1998) 23–25.
[6] H. Fujita, A. Gaeta, V. Loia, F. Orciuoli, Resilience analysis of critical infrastructures: A cognitive approach based on granular computing, IEEE Trans. Cybern. (2018) 1–14.
[7] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (1982) 341–356.
[8] P. Mahajan, R. Kandwal, R. Vijay, Rough set approach in machine learning: A review, Int. J. Comput. Appl. 56 (10) (2012) 1–13.
[9] R. Ali, M.H. Siddiqi, S. Lee, Rough set-based approaches for discretization: a compact review, Artif. Intell. Rev. 44 (2) (2015) 235–263.
[10] Q. Zhang, Q. Xie, G. Wang, A survey on rough set theory and its applications, CAAI Trans. Intell. Technol. 1 (4) (2016) 323–333.
[11] X. Yang, T. Li, H. Fujita, D. Liu, Y. Yao, A unified model of sequential three-way decisions and multilevel incremental processing, Knowl.-Based Syst. 134 (2017) 172–188.
[12] Y. He, Y. Pang, Q. Zhang, Z. Jiao, Q. Chen, Comprehensive evaluation of regional clean energy development levels based on principal component analysis and rough set theory, Renew. Energy 122 (2018) 643–653.
[13] S.K.M. Hamouda, M.E. Wahed, R.H. Abo Alez, K. Riad, Robust breast cancer prediction system based on rough set theory at national cancer institute of Egypt, Comput. Methods Programs Biomed. 153 (2018) 259–268.
[14] H. Yan, D. Wu, Y. Huang, G. Wang, M. Shang, J. Xu, X. Shi, K. Shan, B. Zhou, Y. Zhao, Water eutrophication assessment based on rough set and multidimensional cloud model, Chem. Intell. Lab. Syst. 164 (2017) 103–112.
[15] X. He, W. Wang, X. Liu, Y. Ji, Risk assessment of communication network of power company based on rough set theory and multiclass SVM, Physics Procedia 24 (2012) 1226–1231.
[16] A. Skowron, Information granule decomposition, Fund. Inform. 47 (2001) 337–350.
[17] A. Skowron, R. Swiniarski, P. Synak, Approximation spaces and information granulation, Trans. Rough Sets III (2005) 175–189.
[18] S. Li, T. Li, Z. Zhang, H. Chen, J. Zhang, Parallel computing of approximations in dominance-based rough sets approach, Knowl.-Based Syst. 87 (2015) 102–111.
[19] J. Hu, W. Pedrycz, G. Wang, K. Wang, Rough sets in distributed decision information systems, Knowl.-Based Syst. 94 (2016) 13–22.
[20] S. Wang, T. Li, C. Luo, H. Fujita, Efficient updating rough approximations with multi-dimensional variation of ordered data, Inform. Sci. 372 (2016) 690–708.
[21] J. Hu, T. Li, C. Luo, H. Fujita, S. Li, Incremental fuzzy probabilistic rough sets over two universes, Internat. J. Approx. Reason. 81 (2017) 28–48.
[22] Y. Huang, T. Li, C. Luo, H. Fujita, S.-j. Horng, Matrix-based dynamic updating rough fuzzy approximations for data mining, Knowl.-Based Syst. 119 (2017) 273–283.
[23] C. Luo, T. Li, H. Chen, H. Fujita, Z. Yi, Incremental rough set approach for hierarchical multicriteria classification, Inform. Sci. 429 (2018) 72–87.
[24] Y. Qian, J. Liang, W. Pedrycz, C. Dang, Positive approximation: An accelerator for attribute reduction in rough set theory, Artificial Intelligence 174 (9–10) (2010) 597–618.
[25] Y. Qian, X. Liang, Q. Wang, J. Liang, B. Liu, A. Skowron, Y. Yao, J. Ma, C. Dang, Local rough set: A solution to rough data analysis in big data, Internat. J. Approx. Reason. 97 (2018) 38–63.
[26] J. Zhang, T. Li, D. Ruan, Z. Gao, C. Zhao, A parallel method for computing rough set approximations, Inform. Sci. 194 (2012) 209–223.
[27] J. Zhang, J.S. Wong, Y. Pan, T. Li, A parallel matrix-based method for computing approximations in incomplete information systems, IEEE Trans. Knowl. Data Eng. 27 (2) (2015) 326–339.
[28] J. Zhang, Y. Zhu, Y. Pan, T. Li, Efficient parallel boolean matrix based algorithms for computing composite rough set approximations, Inform. Sci. 329 (2016) 287–302.
[29] Y. Jing, T. Li, H. Fujita, Z. Yu, B. Wang, An incremental attribute reduction approach based on knowledge granularity with a multi-granulation view, Inform. Sci. 411 (2017) 23–38.
[30] G. Lang, Q. Li, M. Cai, H. Fujita, H. Zhang, Related families-based methods for updating reducts under dynamic object sets, Knowl. Inf. Syst. (2019) 1–24, http://dx.doi.org/10.1007/s10115-019-01359-w.
[31] K. Thangavel, A. Pethalakshmi, Dimensionality reduction based on rough set theory: A review, Appl. Soft Comput. 9 (1) (2009) 1–12.
[32] W. Fan, J. Li, N. Tang, W.Y. qa, Incremental detection of inconsistencies in distributed data, IEEE Trans. Knowl. Data Eng. 26 (6) (2014) 1367–1383.

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.

[33] X. Xu, J. Zhou, Y. Liu, Z. Xu, X. Zhao, Taxi-RS: Taxi-hunting recommendation system based on taxi GPS data, IEEE Trans. Intell. Transp. Syst. 16 (4) (2015) 1716–1727.

[34] Z. Shah, A.N. Mahmood, Z. Tari, A.Y. Zomaya, A technique for efficient query estimation over distributed data streams, IEEE Trans. Parallel Distrib. Syst. PP (99) (2017) 1.

[35] D. Dheeru, E. Karra Taniskidou, UCI machine learning repository (2017). URL http://archive.ics.uci.edu/ml.

[36] J. Komorowski, Z. Pawlak, L. Polkowski, A. Skowron, Rough sets: A tutorial, in: Rough Fuzzy Hybridization: A New Trend in Decision-making, 1999, pp. 3–98.

[37] G.-l. Liu, The axiomatization of the rough set upper approximation operations, Fund. Inform. 69 (2006) 331–342.

[38] J. Zhang, T. Li, D. Ruan, D. Liu, International journal of approximate reasoning rough sets based matrix approaches with dynamic attribute variation in set-valued information systems, Internat. J. Approx. Reason. (ISSN: 0888-613X) 53 (4) (2012) 620–635.

[39] C. Davis, The norm of the schur product operation, Numer. Math. (1962) 343–344.

Please cite this article as: S. Li, Z. Hong and T. Li, Efficient composing rough approximations for distributed data, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.001.