Computer Science and Information Systems 19(1):435–453 https://doi.org/10.2298/CSIS210314062D

An Effective Method for Determining Consensus in Large Collectives *

Dai Tho Dang[1,2]**, Thanh Ngo Nguyen[3], and Dosam Hwang[1]**

[1] Department of Computer Engineering,
Yeungnam University, Gyeongsan 38541, Republic of Korea
daithodang@ynu.ac.kr, dshwang@yu.ac.kr
[2] Vietnam - Korea University of Information and Communication Technology,
The University of Danang, Danang, Vietnam
ddtho@vku.udn.vn
[3] Department of Applied Informatics, Faculty of Computer Science and Management,
Wrocław University of Science and Technology, 50-370 Wrocław, Poland
thanh-ngo.nguyen@pwr.edu.pl

Abstract. Nowadays, using the consensus of collectives for solving problems plays an essential role in our lives. The rapid development of information technology has facilitated the collection of distributed knowledge from autonomous sources to find solutions to problems. Consequently, the size of collectives has increased rapidly. Determining consensus for a large collective is very time-consuming and expensive. Thus, this study proposes a vertical partition method (VPM) to find consensus in large collectives. In the VPM, the primary collective is first vertically partitioned into small parts. Then, a consensus-based algorithm is used to determine the consensus for each smaller part. Finally, the consensus of the collective is determined based on the consensuses of the smaller parts. The study demonstrates, both theoretically and experimentally, that the computational complexity of the VPM is lower than 57.1% that of the basic consensus method (BCM). This ratio reduces quickly if the number of smaller parts reduces.

Keywords: large collective, consensus, algorithm, computational complexity.

1. Introduction

Rapid development in information technology has facilitated the use of distributed knowledge from autonomous sources to find solutions to problems [1]. One such example is social networks. Social media platforms, such as Twitter, Facebook, Instagram, and Wikipedia, have revolutionized communication among individuals, groups, and organizations. Exploiting the data generated from social network sites is helpful for both individuals and organizations, such as businesses for marketing, sales, customer support, and public relations. One example of knowledge created by collectives of users is Wikipedia.

* This is an extended version of the article titled "A New Approach to Determine 2-Optimality Consensus for Collectives". In: Fujita H., Fournier-Viger P., Ali M., Sasaki J. (eds) Trends in Artificial Intelligence Theory and Applications. Artificial Intelligence Practices. IEA/AIE 2020. Lecture Notes in Computer Science, vol 12144, Springer.

** Corresponding authors

It is currently the most extensive online encyclopedia collection, with over 54 million articles available in more than 312 languages. Data from social media are considered sources of knowledge [2], and organizations and individuals are increasingly looking for ways to benefit from the collective intelligence of these sources [3]. Another example is Internet of Things (IoT). It has given rise to large amounts of continuous data collected from the physical world [4], [5]. IoT has pervasively penetrated most areas of human life, such as homes, cities, industry, organizations, agriculture, hospitals, and healthcare [6], [7], [8]. Its applications collect data for their aims, such as decision making, system performance boosting, optimal management of resources [9]. This leads to the continuous growth of collectives [10].

The rapid development of other fields has also contributed to the increase in the size of collectives; one such field is biology, where technological advances have allowed researchers to gather unprecedented amounts of data. The amount of biological data is rapidly increasing. Over the last decade, the amount of produced data has doubled almost every seven months [11]. Advances in computational sciences and communication technologies have allowed biologists to share data [12].

Consensus determination has a significant role in computer science, automatic control, social sciences, and biology [13], [14], [15], [16]. Consensus determination is based on collective members' knowledge states. However, the knowledge states in a collective are often inconsistent; thus, consensus determination is complex [17]. The Consensus method is an efficient tool to solve this problem [18].

Consensus determination is an NP-hard problem [16], [18], [19], and many heuristic algorithms have been used to find consensus for different knowledge structures [18], [20]. The complexity of most such algorithms is $O(n^2)$ or larger [16], [18], [19]. For large collectives, determining consensus is very time-consuming and expensive. This study considers determining consensus for large collectives.

This study is an expanded version of our earlier conference paper [21]. In that paper, we proposed an algorithm for determining the 2-Optimality consensus for a large binary collective, the vertical partition method (VPM). First, this method vertically divides the collective into many small parts. Second, it uses a brute-force algorithm to determine the optimal consensuses of these parts. Finally, these consensuses are used to determine the consensus of the whole collective. The approach reduces the time complexity of the brute-force algorithm, and the optimal consensuses of the smaller parts can be used to find consensus in a collective. An experiment showed that the VPM is 99.94% and 99.89% faster if we vertically partition the collective into three and two parts. However, this was only a case study with a binary collective and brute-force algorithm. The two most fundamental problems of the VPM have not been solved. The first is the computing of the computational complexity of the VPM. The second is proving the efficiency of the VPM-for determining consensus in large collectives in general. In this study, we deal with these two problems. The contributions of this study are as follows:

- We propose the VPM and develop a general mathematical model for the VPM.
- The computational complexity of the VPM is computed as a function of the collective sizes of the smaller parts.
- We prove that the computational complexity of the VPM is lower than 57.1% that of the BCM. This ratio reduces quickly if the number of smaller parts reduces.
- The efficiency of the VPM was measured experimentally through a case study.

The remainder of this paper is organized as the following. We present some related concepts of this study in Section 2. In Section 3, the VPM is described. The computational complexity of the VPM is calculated in Section 4. The capability of the VPM is demonstrated in Section 5. In Section 6, we investigate the efficiency of the VPM through a case study. Finally, conclusions and future work are shown in Section 7.

2. Related works

Nowadays, collective intelligence is attracting researchers from many fields, such as biology [13], computer science [22], and automatic control [23].

In computer science, the consensus problem has been investigated in distributed computing [13], multi-agent systems [25], [26], IoT [27], etc. In recent years, collective intelligence has become a promising research area, attracting increasing interest from researchers and organizations. Axiomatic, optimization, and constructive methods have been used to address the consensus problem.

The axiomatic method was first proposed by K. Arrow under seven conditions [27]. It employs simple structures, such as partial order linear order. Nguyen introduced a set of ten postulates for consensus choice functions [17]. However, no consensus choice functions satisfy all postulates concurrently. The postulates 1-Optimality and 2-Optimality have an important role because if one consensus satisfies one of these two postulates, it will satisfy most of the others.

The constructive method solves consensus problems based on the structure of elements and the relation between elements. The relation between elements may be a distance function or preference relation between elements. Many structures of elements have been investigated, such as n-tree [13], ordered partitions [20], disjunction and conjunction Structures [29], binary vectors [30], and ontology [31], [32]

The optimization approach defines consensus choice functions, which are usually based on optimality rules. Optimality rules include the global optimality rule, Condorcet's optimality rule, and maximal similarity rules [18].

Let *U* denote a finite set of objects that represent all potential knowledge states of the same subject. Symbol 2^U denotes the powerset of *U*, which includes the set of all subsets of *U*. Let $\prod_k(U)$ be a set of all k-element subsets of set *U* for $k \in \mathcal{N}$ (where $\mathcal{N}$ is the set of natural numbers), and let

$$\prod(U) = \bigcup_{k\in\mathcal{N}} \prod_k(U)$$

A set $X \in \prod(U)$ is called a collective. The macrostructure of the set *U* is a distance function $d : U \times U \to [0, 1]$ that satisfies the nonnegative, reflexive, and symmetrical conditions. Pair *(U,d)* is called the distance space [18].

For a given collective $X \in \prod(U)$, the consensus of *X* is found by:

- Postulate 1-Optimality if: $d(x^*, X) = min_{y\in U} d(y, X)$
- Postulate 2-Optimality if: $d^2(x^*, X) = min_{y\in U} d^2(y, X)$

where x^* is the consensus of X, $d(x^*, X)$ is the sum of the distances from x^* to collective members, $d^2(x^*, X)$ is the sum of the squared distances from x^* to collective members.

The postulates 1-Optimality and 2-Optimality have an important role in finding consensus. Determining consensuses that meet one of the two postulates are often NP-hard problems [16], [18], [19]. For example, the Kemeny ranking is an NP-hard problem, even for only four votes [14], [33]. Heuristic algorithms have been applied for this task. Over 104 algorithms and combinations have been introduced [14], and their complexities are often $O(m^2)$ or larger.

Consensus determination of large collectives is widespread in medicine and bioinformatics. Many consensus problems must be solved in these two fields, such as gene prediction, protein structure prediction, and disease-related gene ranking. One example is the consensus ranking. A large collective of gene lists of regulation, expression, correlation, interaction can be extracted from data mining results, such as disease-related genes and protein-protein interactions, and disease-related genes. Thus, it is important to rank such data. Given m rankings of n elements, the complexities of the algorithms are $O(n^3m)$, $O(mn + n^2)$, and $O(n^2m)$ [34]. The second example is determining consensus for DNA structure. In [35], algorithms were introduced to determine the 2-Optimality consensus for this structure. The last example is the multiple structure alignment problem. The complexity of the best algorithm to solve this problem is $O(n^2k^2)$, where k is the maximum length of n proteins [36].

For group decision making (GDM) problems, many consensus algorithms have been proposed for various knowledge structures. Many algorithms have been introduced for hesitant fuzzy linguistic structures. In [37], the authors proposed a new method for measuring the difference between two hesitant fuzzy linguistic term sets. Based on this measure, an algorithm was proposed to resolve the hesitant linguistic GDM problem's consensus problem. This algorithm obtains optimally adjusted individual opinions in hesitant linguistic GDM. Its computational complexity is $O(mn^2)$, where n is the number of experts, and m is the number of alternatives to be assessed. In [38], Wu and Xu first defined a new consistency measure. A new algorithm was then presented to improve the consistency index for a given hesitant fuzzy linguistic preference relation. It has a computational complexity of $O(mn^2)$. In [39], the concept of a possibility distribution was introduced. The authors proposed some aggregation operators, such as the hesitant fuzzy linguistic weighted average operator and the hesitant fuzzy linguistic ordered weighted average operator, based on the possibility distributions. A consensus measure was then defined, and a consensus reaching process was presented. The complexity of this algorithm is $O(n^2)$.

The consensus problem has also been of interest in economic [40], [41], [42]. Algorithms for investment strategy design for a multiagent system that supports investment decisions on the stock market were presented in [41]. Based on decisions generated by agents, the supervisor agent uses a consensus method to generate a satisfactory rate of return and reduce the level of risk associated with investing in a financial instrument. The complexity of this algorithm is $O(nm^2)$, where n is the size of the set of decisions and m is the number of decision elements.

3. Vertical Partition Method (VPM)

The basic consensus method (BCM) directly determines consensus based on the primary collective *X* [15]. In other words, it determines consensus based on the knowledge states of all members in the collective X. If the collective size is large, the VPM is often very time-consuming and expensive.

Instead of using the algorithm to determine the consensus based on the collective X as the BCM, the VPM applies the algorithm for smaller parts of the collective X to reduce the computational complexity. First, the primary collective is vertically partitioned into small parts. Then, a consensus-based algorithm is applied to determine consensus for each smaller part. Finally, the consensus of the collective X is determined based on the consensuses of the smaller parts. The procedure of the VPM is illustrated in Fig. 1.

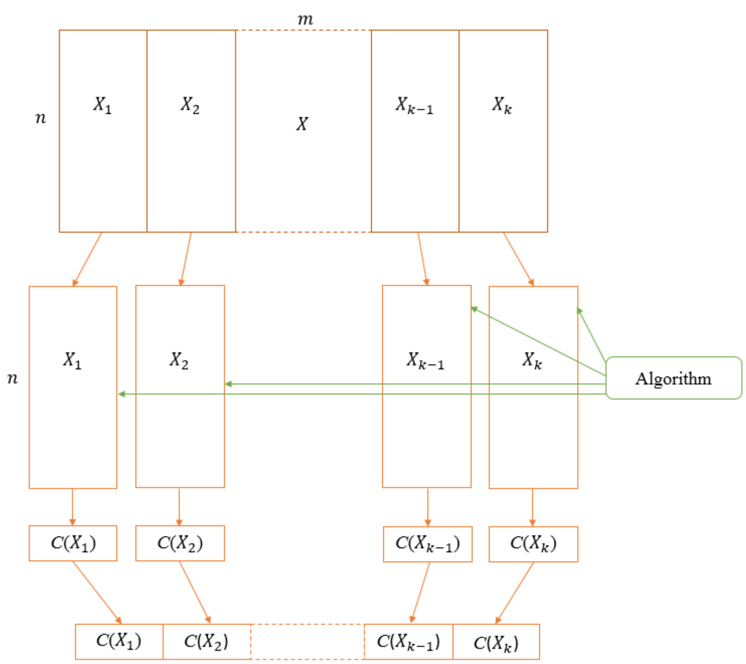

Fig. 1. Schema of the VPM.

Let a large collective X contain n members, where the length of each member is m. The VPM with k parts to determine consensus for the collective X is described as follows:

- **Step 1**: Use the vertical partition to divide the collective X into k disjointed parts $X_1, X_2, ..., X_k$ that satisfy the following:

$$U_1 \cup U_2 \cup \ldots \cup U_k = X$$

$$U_1 \cap U_2 \cap \ldots \cap U_k = \emptyset$$

$$|length(X_i) - length(X_j)| = 1 \text{ or } |length(X_i) - length(X_j)| = 0$$

for $1 \leq i, j \leq k$.

- **Step 2**: Determine consensuses for $X_1, X_2, \ldots, X_k$ as $C(X_1)$, $C(X_2), \ldots, C(X_k)$, respectively.
- **Step 3**: Determine consensus $C(X)$ by combining $C(X_1), C(X_2), ..., C(X_k)$ sequentially:

$$C(X) = C(X_1)C(X_2)...C(X_k)$$

Note that the number of smaller parts k is a natural number that satisfies:

$$2 \leq k \leq \lfloor \frac{m}{2} \rfloor \tag{1}$$

Under this condition, the VPM is very general and flexible.

4. Computational Complexity of the VPM

Let $CVPM(m, m_1, m_2, ...m_k)$ represent the computational complexity of the VPM, where $m, m_1, m_2, \ldots, m_k$ are the lengths of $X, X_1, X_2, \ldots, X_k$, respectively. We can calculate $CVPM(m, m_1, m_2, ...m_k)$ based on the computational complexity of the steps.

Let $O(g(m))$ represent the computational complexity of partitioning the collective X into smaller parts, $O(f(l))$ represent the computational complexity of determining consensus for a smaller part with length l, and $O(h(m))$ represent the computational complexity of generating consensus for the collective X by combining the consensuses of parts $X_1, X_2, \ldots, X_k$. The computation of $CVPM(m, m_1, m_2, ...m_k)$ is detailed as follows:

- In step 1, the collective X with the length of m is vertically partitioned into k smaller parts $X_1, X_2, \ldots, X_k$. The computational complexity of this task is $O(g(m))$.
- In step 2, the complexity of finding the consensuses of k smaller parts X_i $(i = \overline{1, k})$ is computed as the following:

$$O(f(m_1)) + O(f(m_2)) + ... + O(f(m_k))$$

  The difference between the lengths of members of any two smaller parts is not larger than 1. The length of the smaller parts X_i $(i = \overline{1, k})$ are $\lfloor \frac{m}{k} \rfloor$ or $\lfloor \frac{m}{k} \rfloor + 1$. The number of smaller parts with the length $\lfloor \frac{m}{k} \rfloor$ is $k - (m - k \times \lfloor \frac{m}{k} \rfloor)$, and the number of smaller parts with the length $\lfloor \frac{m}{k} \rfloor + 1$ is $m - k \times \lfloor \frac{m}{k} \rfloor$. We have

$$O(f(m_1)) + O(f(m_2)) + ... + O(f(m_k))$$

$$= (k - (m - k \times \lfloor \frac{m}{k} \rfloor) \times O(f(\lfloor \frac{m}{k} \rfloor)) + (m - k \times \lfloor \frac{m}{k} \rfloor \times O(f(\lfloor \frac{m}{k} \rfloor) + 1)$$

- In step 3, the complexity of generating consensus for the collective X by combining the consensuses of $X_1, X_2, ..., X_k$ is $O(h(m))$.

Thus

$$CVPM(m, m_1, m_2, ...m_k) = O(g(m)) + (k - (m - k \times \lfloor \frac{m}{k} \rfloor) \times O(f(\lfloor \frac{m}{k} \rfloor))$$

$$+(m - k \times \lfloor \frac{m}{k} \rfloor \times O(f(\lfloor \frac{m}{k} \rfloor) + 1) + O(h(m))$$

$O(g(m)) = O(m)$ and $O(h(m)) = O(m)$ are linear functions; thus, in the case of large collective, do not consider them:

$$CVPM(m, m_1, m_2, ...m_k) = (k - (m - k \times \lfloor \frac{m}{k} \rfloor) \times O(f(\lfloor \frac{m}{k} \rfloor))$$

$$+(m - k \times \lfloor \frac{m}{k} \rfloor \times O(f(\lfloor \frac{m}{k} \rfloor) + 1) \tag{2}$$

5. Efficiency of the VPM

The efficiency of the VPM is measured by comparing its computational complexity with that of the BCM. Denoting $p = \lfloor \frac{m}{k} \rfloor$, we have $m = kp + r$ $(0 \leq r < k)$ where r is the remainder in the division of m by k. Thus, $X_1, X_2, \ldots, X_k$ include:

- $(k - r)$ parts have p columns;
- r parts have $(p + 1)$ columns.

We have

$$CVPM(m, m_1, m_2, ...m_k) = (k - r) \times O(f(p)) + r \times O(f(p + 1)) \tag{3}$$

Because $2 \leq k \leq \lfloor \frac{m}{2} \rfloor$ (from (1)) and $p = \lfloor \frac{m}{k} \rfloor$, we have

$$p \geq 2 \tag{4}$$

The BCM directly calculates consensus based on all knowledge states of X. By $CBCM(m)$ we denote the computational complexity of the BCM. We have

$$CBCM(m) = O(f(m)) \tag{5}$$

Theorem 1. *If the computational complexity of the BCM is $O(m^2)$, we have*

$$CBCM > 1.75 \times CVPM$$

Proof.

The algorithm determining consensus has quadratic computational complexity.

From (4), we have

$$CVPM = (k - r)p^2 + r(p + 1)^2$$

$$CVPM = kp^2 + 2pr + 1 \tag{6}$$

From (5), we get

$$CBCM = m^2 = (kp + r)^2$$

$$CBCM = k^2p^2 + 2kpr + r^2 \tag{7}$$

From (6) and (7), we have

$$\frac{CBCM}{CVPM} = \frac{k^2p^2 + 2kpr + r^2}{kp^2 + 2pr + 1} = \frac{k(kp^2 + 2pr + 1) - (k - r^2)}{kp^2 + 2pr + 1}$$

$$= \frac{k(kp^2 + 2pr + 1)}{kp^2 + 2pr + 1} - \frac{k - r^2}{kp^2 + 2pr + 1}$$

$$= k - \frac{k - r^2}{kp^2 + 2pr + 1} > k - \frac{k}{kp^2 + 2pr + 1} > k - \frac{k}{kp^2} = k - \frac{1}{p^2}$$

Thus

$$\frac{CBCM}{CVPM} > k - \frac{k}{p^2} \tag{8}$$

From (1) and (4), we have $k \geq 2$ and $p \geq 2$. From (8), we get

$$\frac{CBCM}{CVPM} > k - \frac{1}{p^2} \geq 2 - \frac{1}{2^2} = 1.75$$

Or

$$CBCM > 1.75 \times CPVM$$

From (8), we can see that $\frac{CBCM}{CVPM}$ increases quickly if k increases. It reaches $\frac{m}{2^{m-1}}$ when $k = \lfloor \frac{m}{2} \rfloor$.

Theorem 2. *If the computational complexity of the BCM is higher than $O(m^2)$, we have*

$$CBCM > 1.75 \times CVPM$$

Proof.

Let us consider the case that the computational complexity of the BCM is (m^3).

From (3), we have

$$CVPM = (k - r)p^3 + r(p + 1)^3 \tag{9}$$

From (5), we have

$$CBCM = m^3 \tag{10}$$

From Theorem 1, we have

$$1.75 \times ((k - r)p^2 + r(p + 1)^2) < m^2 \tag{11}$$

Multiply both sides of (11) by m, we get

$$1.75 \times ((k - r)p^2 + r(p + 1)^2)m < m^3$$

Or

$$1.75 \times ((k-r)p^2m + r(p+1)^2m) < m^3 \tag{12}$$

Let us consider the left-hand side of the inequality (12). Because $m = kp + r$ and $k > r \geq 0$, we have $m \geq kp$.

Thus

$$1.75 \times ((k-r)p^2m + r(p+1)^2m) \geq 1.75 \times ((k-r)p^2(kp) + r(p+1)^2(kp)) \tag{13}$$

Because $k \geq 2$ and $p \geq 2$ (from (1) and (4)), then $kp > p+1$. We have

$$1.75 \times ((k-r)p^2(kp) + r(p+1)^2(kp)) = 1.75 \times (k(k-r)p^3 + r(p+1)^2(kp))$$
$$\gg 1.75 \times ((k-r)p^3 + r(p+1)^3) \tag{14}$$

From (13) and (14), we get

$$1.75 \times ((k-r)p^2 + r(p+1)^2)m \gg 1.75 \times ((k-r)p^3 + r(p+1)^3) \tag{15}$$

From (12) and (15), we have

$$1.75 \times ((k-r)p^3 + r(p+1)^3) \ll m^3$$

Or

$$1.75 \times CVPM \ll CBCM$$

We proved that Theorem 2 is true if the computational complexity of the BCM is $O(m^3)$.

Assume that $1.75 \times CVPM \ll CBCM$ with the complexity of the BCM is $O(m^t)$ for $t > 3$. We have

$$CBCM = m^t \tag{16}$$

$$CPVM = (k-r)p^t + r(p+1)^t \tag{17}$$

$$1.75 \times ((k-r)p^t + r(p+1)^t) < m^t \tag{18}$$

We need to prove $1.75 \times CVPM \ll CBCM$ with the complexity of the BCM is $O(m^{t+1})$. In other words, we need prove that

$$1.75 \times ((k-r)p^{t+1} + r(p+1)^{t+1})m < m^{t+1} \tag{19}$$

Multiply both sides of (18) by m, we get

$$1.75 \times ((k-r)p^t + r(p+1)^t)m < m^{t+1}$$

Or

$$1.75 \times ((k-r)p^tm + r(p+1)^tm) < m^{t+1} \tag{20}$$

Let us consider the left-hand side of the inequality (20). Because $m = kp + r$ and $k > r \geq 0$, we have $m \geq kp$. Thus

$$1.75 \times ((k-r)p^t m + r(p+1)^t m) \geq 1.75 \times ((k-r)p^t (kp) + r(p+1)^t (kp)) \quad (21)$$

Because $k \geq 2$ and $p \geq 2$ (from (1) and (4)), we have $kp > p + 1$. We have

$$1.75 \times ((k-r)p^t (kp) + r(p+1)^t (kp)) = 1.75 \times (k(k-r)p^{t+1}) + r(p+1)^t (kp)$$
$$\gg 1.75 \times ((k-r)p^{t+1} + r(p+1)^{t+1}) \quad (22)$$

From (21) and (22), we obtain

$$1.75 \times ((k-r)p^t m + r(p+1)^t m) \gg 1.75 \times ((k-r)p^{t+1} + r(p+1)^{t+1}) \quad (23)$$

From (20) and (23), we have

$$1.75 \times ((k-r)p^{t+1} + r(p+1)^{t+1}) \ll m^{t+1}$$

It means that (19) was proved.

Theorem 3. *The computational complexity of the BCM is $O(m^t n^w)$. If $t \geq 2$, for any $w \geq 0$, we have*

$$CBCM > 1.75 \times CVPM$$

Proof.

From (3), we have

$$CVPM = (k-r)p^t n^w + r(p+1)^t n^w \quad (24)$$

From (5), we have

$$CBCM = m^t n^w \quad (25)$$

Thus

$$\frac{CBCM}{CVPM} = \frac{m^t n^w}{(k-r)p^t n^w + r(p+1)^t n^w}$$
$$= \frac{m^t n^w}{n^w \times ((k-r)p^t + r(p+1)^t)}$$
$$= \frac{m^t}{(k-r)p^t + r(p+1)^t}$$

From Theorem (1) and Theorem (2), we get

$$= \frac{m^t}{((k-r)p^t + r(p+1)^t)} > 1.75$$

Or

$$CBCM > 1.75 \times CPVM$$

6. Application of the PVM

This section examines the efficiency of the VPM through a case study. Determining the consensus for a binary collective is an NP-hard problem; applying the VPM can efficiently deal with this situation.

Set U is described as $U = \{u_1, u_2, ..., u_q\}$ where each element is a binary vector of length m. The size of U is 2^m. Each set $X \in \prod(U)$ is a collective that is represented as

$$X = \{x_1, x_2, ..., x_n\}$$

where each element x_i is a binary vector for $1 \leq i \leq n$. Each element $x_i \in X$ is represented as

$$x_i = (x_i^1, x_i^2, \ldots, x_i^m), x_i^j = \{0, 1\}, 1 \leq j \leq m.$$

The brute-force algorithm is used to find the optimal consensus for collectives containing binary vectors. This algorithm is unfeasible because its computational complexity is $O(n2^m)$. In this study, the VPM using the brute-force algorithm with two and three parts is investigated.

6.1. Algorithms

TwP algorithm

In this algorithm, the collective X is vertically partitioned into two parts: X_1 and X_2.

- X_1 has n vectors, the length of vectors is $\lfloor \frac{m}{2} \rfloor$.
- X_2 has n vectors, the length of vectors is $m - \lfloor \frac{m}{2} \rfloor$.

The brute-force algorithm is used to determine the 2-Optimality consensus for X_1 and X_2. Then, the 2-Optimality consensus of the collective X is determined. The TwP algorithm is represented as follows.

Algorithm 1. TwP

Input: Collective $X = \{x_1, x_2, ..., x_n\}$
Output: 2-Optimality consensus x^* of the collective X
BEGIN
1. Vertically partition the collective X into two parts: X_1, X_2;
2. $C(X_1) = \text{brute} - \text{force}(X_1)$;
3. $C(X_2) = \text{brute} - \text{force}(X_2)$;
4. $x^* = concat(C(X_1), C(X_2)$;

END

ThP algorithm

In the ThP algorithm, the collective X is vertically partitioned into three parts: X_1, X_2, and X_3. Note that the difference between the lengths of any two smaller parts is equal to 0 or 1. The brute-force algorithm is used todetermine the 2-Optimality consensus for X_1, X_2, and X_3. Finally, the 2-Optimality consensus of the collective X is determined.

This algorithm is presented as the followings.

Algorithm 2. ThP
Input: Collective $X = \{x_1, x_2, \dots, x_n\}$
Output: 2-Optimality consensus x^* of the collective X
BEGIN
1. Vertically partition the collective X into two parts: X_1, X_2, X_3 ;
2. $C(X_1) = \text{brute} - \text{force}(X_1)$;
3. $C(X_2) = \text{brute} - \text{force}(X_2)$;
4. $C(X_3) = \text{brute} - \text{force}(X_3)$;
5. $x^* = concat(C(X_1), C(X_2), C(X_3)$;
END

6.2. Experiments and Evaluation

The TwP and ThP algorithms are the VPM using the brute-force algorithm. This section estimates the ability of the TwP and ThP algorithms by experiments. The two algorithms are examined both running time and consensus quality. We compare these two algorithms to the basic heuristic and brute-force algorithms. The reason is that the basic heuristic algorithm is the most common algorithm to find consensus for binary collectives, and the brute-force algorithm is used to develop the TwP and ThP algorithms.

The significant level α is chosen as 0.05. Consensus quality of a heuristic algorithms is calculated as follows:

$$CQ = 1 - \frac{|d^2(x^*, X) - d^2(x_{opt}, X)|}{d^2(x_{opt}, X)}$$

where x^* is the 2-Optimality consensus found by the heuristic algorithm, and x_{opt} the optimal consensus found by the brute-force algorithm.

Consensus quality

The following experiment aims to evaluate the consensus quality of the algorithms TwP and ThP. A dataset with 26 collectives is created randomly. Each collective includes 650 elements, and the element length is 22.

We run the basic heuristic, TwP, and ThP algorithms on the dataset. It generates three consensus quality samples of the basic heuristic, TwP, and ThP algorithms. The samples are represented in Table 1. In Fig.5., red, green, and black columns describes consensus quality for the TwP, ThP, and basic heuristic algorithms, respectively.

The boxplots of these consensus quality samples are described in Fig.6. The medians of the TwP, ThP, and basic heuristic algorithms' consensus quality are 0.99925, 0.99780, and 0.96590, respectively. The consensus quality sample of the basic heuristic algorithm has the lowest level of closeness with each other.

We need to determine the distribution of these samples. The null hypothesis H_0 for this test is that the consensus quality sample is normally distributed. The Shapiro-Wilk test is applied to find distributions of these samples. The *p-value* of the TwP algorithm's consensus quality sample is 0.0002. Because *p-value* $< \alpha$, H_0 is rejected. It indicates that the consensus quality sample of the TwP algorithm is not normally distributed.

The similarity, *p-values* of the consensus quality samples of the algorithms ThP and basic heuristic are less than the significant level (*p-value*=0.03077 and *p-value*=0.000002 for the consensus quality sample of the ThP and basic heuristic algorithms, respectively).

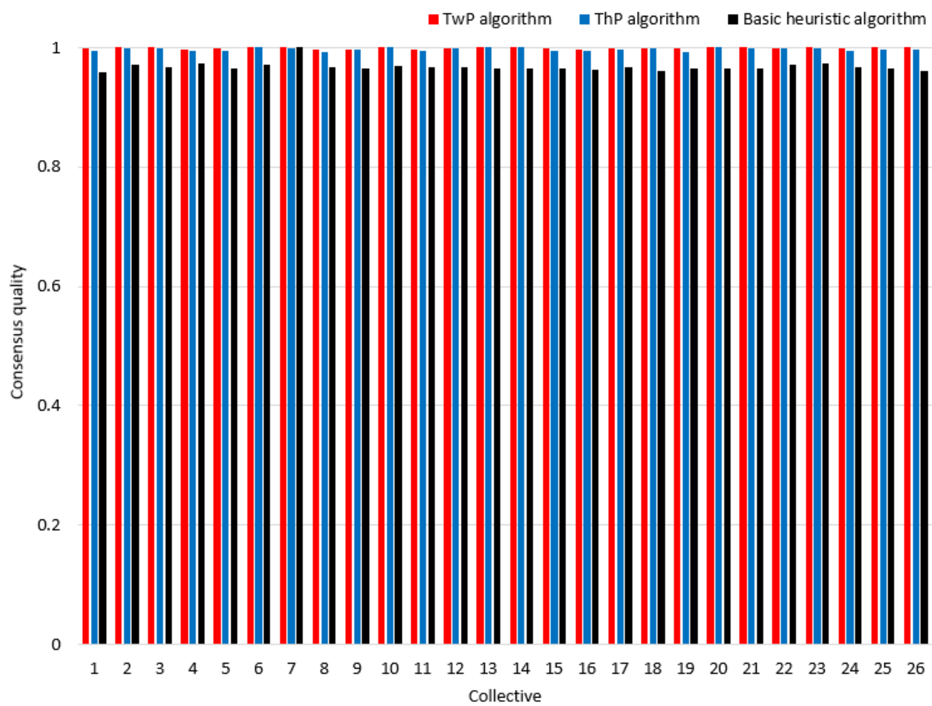

Fig. 2. Consensus quality of the algorithms TwP, ThP, and basic heuristic.

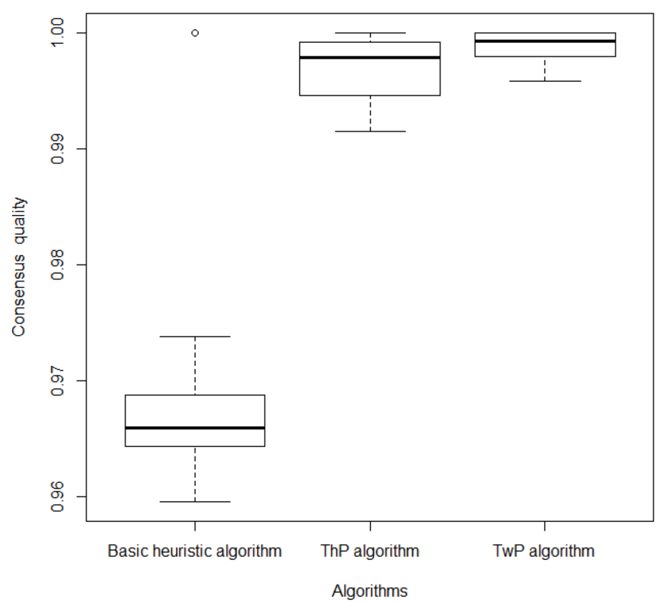

Fig. 3. The boxplots of consensus quality of the algorithms TwP, ThP, and basic heuristic.

Table 1. Consensus quality of the algorithms TwP, ThP, and basic heuristic.

Collective	TwP algorithm	ThP algorithm	Basic heuristic algorithm
1	0.9985	0.9945	0.9595
2	1.0000	0.9984	0.9718
3	1.0000	0.9981	0.9675
4	0.9968	0.9948	0.9738
5	0.9978	0.9936	0.9643
6	1.0000	1.0000	0.9716
7	1.0000	0.9993	1.0000
8	0.9958	0.9915	0.9672
9	0.9971	0.9966	0.9656
10	1.0000	1.0000	0.9687
11	0.9958	0.9935	0.9672
12	0.9996	0.9983	0.9674
13	1.0000	1.0000	0.9648
14	1.0000	1.0000	0.9643
15	0.9988	0.9946	0.9648
16	0.9960	0.9939	0.9624
17	0.9982	0.9975	0.9662
18	0.9989	0.9985	0.9616
19	0.9979	0.9926	0.9641
20	1.0000	1.0000	0.9652
21	1.0000	0.9992	0.9652
22	0.9985	0.9984	0.9715
23	1.0000	0.9986	0.9737
24	0.9981	0.9954	0.9666
25	1.0000	0.9958	0.9648
26	1.0000	0.9972	0.9611

It means that the consensus quality samples are not normally distributed. We compare these three consensus quality samples. The hypotheses are declared as follows:

- H_0: The medians of consensus quality of the algorithms TwP, ThP, and basic heuristic are equal.
- H_1: The medians of consensus quality of the algorithms TwP, ThP, and basic heuristic are not equal.

Because three samples do not come from the normal distribution, the Kruskal-Wallis test is applied to evaluate the hypotheses. We obtain *p-value*=2.7e-11. As *p-value*$<$0.05, H_0 is rejected. We can conclude that the medians of consensus quality of the TwP, ThP, and basic heuristic algorithms are not equal.

From the output of the Kruskal-Wallis test, we realize that there is a significant difference between samples. However, we do not know which pairs of samples are different. The Pairwise Wilcoxon test is used to calculate pairwise comparisons between samples with corrections for multiple testing. The *p-values* are shown for each pair in the output as follows:

– The *p-value* for the basic heuristic and ThP algorithms is 2.6e-08.

– The *p-value* for the basic heuristic and TwP algorithms is 1.4e-08.

– The *p-value* for the TwP and ThP algorithms is 0.024.

Since three *p-values* are less than 0.05, we can conclude that the difference in consensus quality between the basic heuristic algorithm and the ThP algorithm, between the basic heuristic algorithm and the TwP algorithm, between the TwP algorithm and the ThP is statistically significant.

The consensus quality of the TwP algorithm is 0.1% higher than that of the ThP algorithm and 3.4% higher than that of the basic heuristic algorithm. The consensus quality of the TwP algorithm is 3.3% higher than that of the basic heuristic algorithm.

Running time

The brute-force algorithm determines consensus based on the knowledge states of all members in the collective. The brute-force is the BCM, and the algorithms TwP and ThP are the VPM. They are developed based on the brute-force algorithm. The following experiment aims to evaluate the running time of VPM by comparing the running time of the brute-force, TwP, and ThP.

A dataset containing 15 collectives is randomly created. The vector length is 22 and collective sizes are 300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800, 850, 900, 950, and 1000. We perform the ThP, TwP, and brute-force algorithms on this dataset. Three running time samples of the three algorithms are generated. They are represented in Table 2.

Table 2. Running time of the algorithms brute-force, TwP, and ThP (seconds).

Collective size	Brute-force algorithm	TwP algorithm	ThP algorithm
300	96.810	0.111	0.029
350	107.038	0.129	0.034
400	123.702	0.148	0.038
450	138.886	0.165	0.044
500	154.490	0.181	0.046
550	170.055	0.201	0.048
600	184.172	0.221	0.054
650	199.352	0.234	0.062
700	218.713	0.254	0.069
750	233.024	0.271	0.073
800	248.737	0.297	0.078
850	264.513	0.311	0.081
900	282.291	0.325	0.086
950	294.624	0.344	0.093
1000	307.256	0.361	0.104

The Shapiro-Wilk test is applied to specify the distribution of the samples. Their *p-values* larger than α (*p-value*=0.601, *p-value* =0.7, *p-value*=0.739 for the running time sample of the algorithms brute-force, TwP, ThP, respectively). It means that these samples come from the normal distribution. The hypotheses to compare the running time of these algorithms are declared as follows:

- H_0: The means of running time of the algorithms brute-force, TwP, ThP are equal.
- H_1: The means of running time of the algorithms brute-force, TwP, ThP are not equal.

As the samples come from the normal distribution, we use the one-way ANOVA to evaluate the hypotheses. We get *p-value*=2e-16, it means that the means of running time of the brute-force, TwP, ThP algorithms are not equal.

This result indicates that some of the sample means are different. However, we do not know which pairs of samples are different. We use the Tukey HSD test for performing multiple pairwise-comparison between the means of samples. The *p-values* are shown for each pair in the output as follows:

– The *p-value* for the ThP algorithm and brute-force algorithms is 1e-12.
– The *p-value* for the TwP algorithm and brute-force algorithms is 1e-12.
– The *p-value* for the TwP algorithm and ThP algorithms is 0.99.

The difference in running time between the TwP algorithm and the ThP algorithm is not statistically significant. The difference in running time between the brute-force algorithm and others is statistically significant. The running time of the TwP, ThP algorithms are equal to 0.01%, 0.003% that of the brute-algorithm, respectively.

7. Discussion

The basic heuristic algorithm is popular to find consensus for collectives in the literature. The consensus quality of the algorithms TwP and ThP are 3.4% and 3.3% higher than that of the basic heuristic algorithm, respectively. Besides, the VPM proved its effectiveness in running time by experiments. The TwP and ThP algorithms' running time is hugely less than that of the brute-force algorithm if the collective is only partitioned into two and three parts. The running time continuously reduces if the number of smaller parts increases, satisfying (1). The VPM is an efficient tool to deal with large collectives.

8. Conclusions

In this study, we introduced the VPM to determine large collectives. We developed a general mathematical model for the VPM. The computational complexity of the VPM is computed as a function of the collective sizes of the smaller parts. We proved that the computational complexity of the VPM is lower than 57.1% that of the BCM. This ratio reduces quickly if the number of smaller parts reduces. Besides, The efficiency of the VPM was measured experimentally through experiments.

In the future, we will investigate combining the VPM and parallel processing to increase the efficiency of the VPM.

Acknowledgments. This work was supported by the 2021 Yeungnam University Research Grant.

References

1. Nguyen N.T., Szczerbicki E., Trawiński B., Nguyen V.D.: Collective Intelligence in Information Systems. Journal of Intelligent and Fuzzy Systems 37, No. 6, 7113–7115. (2019), https://doi.org/10.3233/JIFS-179324
2. Oxley A.: Security Risks in Social Media Technologies. Elsevier (2013).
3. Hansen D.L., Shneiderman B. et al.: Analyzing Social Media Networks with NodeXL. Elsevier Inc. (2020).
4. Amin F., Choi G.S.: Hotspots Analysis Using Cyber-physical-social System for a Smart City. IEEE Access, Vol. 8, 122197-122209. (2020), https://doi.org/10.1109/ACCESS.2020.3003030

5. Asghari P., Rahmani A.M., Javadi S.: Internet of Things Applications: A Systematic Review. Computer Networks, Vol. 148, 241–261. (2019).
6. Farooq M.S., Riaz S.et al.: A Survey on the Role of IoT in Agriculture for the Implementation of Smart Farming. IEEE Access, Vol. 7, 156237–156271 (2019).
7. Verma P., Sood S.K.: Fog assisted-IoT Enabled Patient Health Monitoring in Smart Homes. IEEE Internet of Things Journal, Vol. 5, No. 3, 1789–1796 (2018).
8. Hassija V., Chamola V. et al.: A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. IEEE Access, Vol. 7, 82721–82743 (2019).
9. Sunhare P., Chowdhary R.R, Chattopadhyay M.K.: Internet of Things and Data Mining: An Application Oriented Survey. Journal of King Saud University - Computer and Information Sciences. (2020), https://doi.org/10.1016/j.jksuci.2020.07.002
10. Maleszka M., Nguyen N.T.: Integration Computing and Collective Intelligence. Expert Systems with Applications, Vol. 42, No. 1, 332–340. (2015), https://doi.org/10.1016/j.eswa.2014.07.036
11. Stephens Z.D., Lee S.Y. et al.: Big data: Astronomical or Genomical?. PLoS Biology, Vol. 13, No. 7, 1–11. (2015), https://doi.org/10.1371/journal.pbio.1002195
12. Yin Z., Lan H.: Computing Platforms for Big Biological Data Analytics: Perspectives and Challenges," Computational and Structural Biotechnology Journal, Vol. 15, 403–411. (2017).
13. Jansson J., Rajaby R., Shen C., Sung W.K.: Algorithms for the Majority Rule (+) Consensus Tree and the Frequency Difference Consensus Tree. IEEE/ACM Transactions on Computational Biology and Bioinformatics, Vol. 15, No. 1, 15–26. (2018).
14. Ali A., Meilă M.: Experiments with Kemeny ranking: What Works When?. Mathematical Social Sciences, Vol. 64, No. 1, 28–40, 2012, https://doi.org/10.1016/j.mathsocsci.2011.08.008
15. Dang D.T., Nguyen N.T., Hwang D.: Multi-Step Consensus: An Effective Approach for Determining Consensus in Large Collectives. Cybernetics and Systems, Vol. 50, No. 2, 208–229. (2019), https://doi.org/10.1080/01969722.2019.1565117
16. Badal P.S., Das A.: Efficient Algorithms Using Subiterative Convergence for Kemeny Ranking Problem. Computers and Operations Research, vol. 98, 198–210. (2018).
17. Nguyen N.T.: Processing Inconsistency of Knowledge in Determining Knowledge of a Collective. Cybernetics and Systems, Vol. 40, No.8, 670–688., (2009), https://doi.org/10.1080/01969720903294593
18. Nguyen N.T: Advanced Methods for Inconsistent Knowledge Management. London: Springer London. (2008).
19. D'Ambrosio A., Mazzeo G., Iorio C., Siciliano R.: A Differential Evolution Algorithm for Finding the Median Ranking Under the Kemeny Axiomatic Approach. Computers and Operations Research, Vol. 82, 126–138 (2017), https://doi.org/10.1016/j.cor.2017.01.017
20. Danilowicz C., Nguyen N.T.: Consensus-based Partitions in the Space of Ordered Partitions. Pattern Recognition, Vol. 21, No. 3, 269–273. (1988), https://doi.org/10.1016/0031-3203(88)90061-1
21. Dang D.T., Mazur Z., Hwang D. (2020) A New Approach to Determine 2-Optimality Consensus for Collectives. In: Fujita H., Fournier-Viger P., Ali M., Sasaki J. (eds) Trends in Artificial Intelligence Theory and Applications. Artificial Intelligence Practices. IEA/AIE 2020. Lecture Notes in Computer Science, Vol. 12144, 570-581. (2020), https://doi.org/10.1007/978-3-030-55789-8_49
22. Xiaohui C.: A study of Collective Intelligence in Multiagent Systems. University of Louisville, Kentucky, USA. (2004).
23. Meng ., Zhang H.T, Wang Z., Chen G.: Event-Triggered Control for Semiglobal Robust Consensus of a Class of Nonlinear Uncertain Multiagent Systems. IEEE Transactions on Automatic Control, Vol. 65, No. 4, 1683–1690. (2020), https://doi.org/10.1109/TAC.2019.2932752
24. Lynch N.A.: Distributed Algorithms. Morgan Kaufmann. (1996).
25. Sliwko L, Nguyen N.T.: Using Multi-agent Systems and Consensus Methods for Information Retrieval in Internet. International Journal of Intelligent Information and Database Systems, Vol. 1, No 2, 181-198. (2007), https://doi.org/10.1504/IJIIDS.2007.014949

26. Qin J., Ma Q., Shi Y., Wang L.: Recent Advances in Consensus of Multi-agent Systems: A Brief Survey. IEEE Transactions on Industrial Electronics, Vol. 64, No. 6, 4972–4983. (2017), https://doi.org/10.1109/TIE.2016.2636810
27. Li S., Oikonomou G. et al.: A Distributed Consensus Algorithm for Decision Making in Service-Oriented Internet of Things. IEEE Transactions on Industrial Informatics, Vol. 10, No. 2, 1461–1468. (2014), https://doi.org/10.1109/TII.2014.2306331
28. Arrow K.J.: Social Choice and Individual Values. Wiley, New York, 1963.
29. Nguyen N.T: Processing Inconsistency of Knowledge on Semantic Level. Journal of Universal Computer Science, Vol. 11, No. 2, 285-302. (2005), https://doi.org/10.3217/jucs-011-02-0285
30. Dang D.T., Nguyen N.T., Hwang D.: A Quick Algorithm to Determine 2-Optimality Consensus for Collectives. IEEE Access, Vol. 8, 221794–221807. (2020), https://doi.org/10.1109/ACCESS.2020.3043371
31. Nguyen N.T: A Method for Ontology Conflict Resolution and Integration on Relation Level. Cybernetics and Systems, Vol. 38, No. 8, 781-797. (2007), https://doi.org/10.1080/01969720701601098
32. Pietranik M., Nguyen N.T.: A Multi-atrribute based Framework for Ontology Aligning. Neurocomputing, Vol. 146, 276-290. (2014), https://doi.org/10.1016/j.neucom.2014.03.067
33. Amodio S., Ambrosio A.D, Siciliano R.: Accurate Algorithms for Identifying the Median Ranking When Dealing with Weak and Partial Rankings under the Kemeny Axiomatic Approach. European Journal of Operational Research, Vol. 249, No. 2, 667–676. (2016), https://doi.org/10.1016/j.ejor.2015.08.048
34. Yang B.: Bioinformatics Analysis and Consensus Ranking for Biological High throughput Data. Ph.D. Dissertation, University of Paris 11. (2015).
35. Dang D.T., Phan H.T., Nguyen N.T., Hwang D. (2021) Determining 2-Optimality Consensus for DNA Structure. In: Fujita H., Selamat A., Lin J.CW., Ali M. (eds) Advances and Trends in Artificial Intelligence. Artificial Intelligence Practices. IEA/AIE 2021. Lecture Notes in Computer Science, vol 12798, 427-438. (2021), https://doi.org/10.1007/978-3-030-79457-6_36
36. Ilinkin I., Ye J., Janardan R.: Multiple Structure Alignment and Consensus Identification for Proteins. BMC Bioinform., Vol. 11, No. 1, 71-80. (2010).
37. Dong Y., ChenX., Herrera F.: Minimizing Adjusted Simple Terms in The Consensus Reaching Process With Hesitant Linguistic Assessments in Group Decision Making. Information Sciences, Vol. 297, 95–117. (2015), https://doi.org/10.1016/j.ins.2014.11.011
38. Wu Z. Xu J.: Managing Consistency and Consensus in Group Decision Making with Hesitant Fuzzy Linguistic Preference Relations. Omega, Vol. 65, 28–40. (2016), https://doi.org/10.1016/j.omega.2015.12.005
39. Wu Z., Xu J.: Possibility Distribution-Based Approach for MAGDM With Hesitant Fuzzy Linguistic Information. IEEE Transactions on Cybernetics, Vol. 46, No. 3, 694–705. (2016).
40. Duong T.H, Nguyen N.T. et al.: A Collaborative Algorithm for Semantic Video Annotation Using a Consensus-based Social Network Analysis. Expert Systems With Applications, Vol. 42, No. 1, 246-258. (2015), https://doi.org/10.1016/j.eswa.2017.01.012
41. Radojičić, D., Radojičić, N., Kredatus, S.: A Multicriteria Optimization Approach for the Stock Market Feature Selection. Computer Science and Information Systems, Vol. 18, No. 3, 749–769. (2021), https://doi.org/doi.org/10.2298/CSIS200326044R
42. Sobieska-karpinska J., Hernes M.: Consensus Determining Algorithm in Multiagent Decision Support System with Taking into Consideration Improving Agent's Knowledge. Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), 1035–1040. (2012).

Dai Tho Dang received the M.S degree in computer science from the University of Nice Sophia Antipolis, Nice, France, and the Ph.D. degree in computer science from

Yeungnam University, the Republic of Korea. He is currently working as a lecturer at Vietnam-Korea University of Infomation and Communication Technology, The University of Danang. His research interests include collective intelligence, algorithm, consensus theory, inconsistent knowledge processing. He is a reviewer for journals *IEEE Transactions on Cybernetics*, *Artificial Intelligence Review*, *Applied Intelligence*.

Thanh Ngo Nguyen received the M.Sc. degree in computer science from Le Quy Don Technical University, Hanoi, Vietnam, in 2013. He is currently pursuing the Ph.D. degree with the Faculty of Information and Communication Technology, Wroclaw University of Science and Technology, Poland. He is currently a Research and Teaching Assistant with the Faculty of Information and Communication Technology, Wroclaw University of Science and Technology. His research interests include association rules and pattern mining.

Dosam Hwang received the Ph.D. degree in Kyoto University, Kyoto, Japan. He is a full professor of the Department of Computer Engineering at Yeungnam University in Korea, whose research interests mainly include Natural Language Processing, Ontology, Knowledge Engineering, Information Retrieval and Machine translation. He has also held a position as a principal researcher at Korea Institute of Science and Technology (KIST) and has also been a visiting professor at Korea Advanced Institute of Science and Technology (KAIST). He has so far been not only a co-chair of several international conferences but also a steering committee member of ICCCI and ACIIDS, and MISSI international conferences. He has been honored as a Distinguished Researcher of KIST in 1988 by Korea's Ministry of Science and Technology (MoST) and awarded a prize for Good Conduct from Kyunghee High School in 1973. He had more than 50 publications.

Received: March 14, 2021; Accepted: August 31, 2021.