Learning and Nonlinear Models - Journal of the Brazilian Society on Computational Intelligence (SBIC), Vol. 14, Iss. 2, pp. 36-54, 2016

A Review on Evolving Interval and Fuzzy Granular Systems

Daniel Leite
Department of Engineering, Federal University of Lavras
daniel.leite@deg.ufla.br

Pyramo Costa Jr.
Graduate Program in Electrical Engineering, Pontifical Catholic University of Minas Gerais
pyramo@pucminas.br

Fernando Gomide
School of Electrical and Computer Engineering, University of Campinas
gomide@dca.fee.unicamp.br

Abstract – This article provides definitions and principles of granular computing and discusses the generation and online adaptation of rule-based models from data streams. Essential notions of interval analysis and fuzzy sets are addressed from the granular computing point of view. The article also covers different types of aggregation operators which perform information fusion by gathering large volumes of dissimilar information into a more compact form. We briefly summarize the main historical landmarks of evolving intelligent systems leading to the state of the art. Evolving granular systems extend evolving intelligent systems allowing data, variables and parameters to be granules (intervals and fuzzy sets). The aim of the evolution of granular systems is to fit the information carried by data streams from time-varying processes into rule-based models and, at the same time, provide granular approximation of functions and linguistic description of the system behavior.
Keywords – Granular computing, evolving intelligent systems, fuzzy systems, interval mathematics.

1. FOUNDATIONS OF GRANULAR COMPUTING

Theories and methodologies that make use of granules to solve problems featured by supplying huge amounts of data, information, and knowledge label a new area of multi-disciplinary study called Granular Computing [1–6]. Granular computing as a paradigm of information processing spotlights multiple levels of data detailing to often provide useful abstractions and approximate solutions to hard real-world problems [7–10].

Granular information systems have appeared under different names in related fields such as interval analysis, fuzzy and rough sets, divide and conquer, quotient space theory, information fusion, and others (see [4]). Elementary processing units in granular systems are referred to as information granules. An information granule is defined as a clump of entities that may originate at the numeric (singular) or granular level and are arranged together due to their similarity, proximity, indistinguishability, or coherency. The goal of a granule is to catch the very essence of the overall data in a concise and explainable manner [1] [9]; it defines a subset of a universal set and conveys an internal representation. Granules may be interpreted from two points of view: from the perspective of uncertainty theory, they are units lacking precise knowledge; from that of knowledge engineering, they are units of elementary knowledge.

Granular computing is intended to identify manifestations of granules from moving back and forth among granularities to yield more or less differentiation. Too much detail is wasteful whereas too little renders a system useless. In general, there is no universal level of granularity of information: the size of granules is problem-oriented and user-dependent. Information granularity is defined as the extent to which a larger and more complex system is broken down into smaller and simpler parts. We can quantify the granularity of a granule, for example, by counting its number of elements. The more elements are located in a granule, the lower is its granularity, and the higher is its generality [11]. High granularities can produce substantial computational overhead for data storage. In excess, granularities and granules bring undesirable scalability issues such as incapacity to satisfy the required throughput. The granularity of information that is explicitly inbuilt into granules provides useful features in information systems modeling such as transparency and flexibility.

Let the result of data granulation be designated as a granular structure. A granular structure is a family of granules which, when considered together, reassemble the more complex original system. Handling a complex phenomenon by means of granular structures allows us to arrive at meaningful solutions. Based on some carefully chosen granularity, granular computing systems attempt to solve a problem by isolating its loosely connected sub-problems and handling them on an individual basis.

Granules of multiple sizes are related to the depth of penetration that characterizes a system. A coarse granular structure contains fewer number of granules compared to a fine granular structure. This can be stated more precisely as follows. A coarse granular system regards a small amount of large granules usually characterized by low precision and high interpretability. A fine granular system regards a large amount of small granules, high precision, and limited interpretation. Low-level refined granules provide details about the system functionality. More abstract, high-level granules are easier to manage and interpret, but may lose important minutiae.

Input and output data sets generate input and output granular structures, respectively, which should be somehow connected. We name the correspondence between input and output granular structures as granular mapping. A granular mapping is defined over information granules lying in an input space and maps them into a collection of granules expressed in some output space. Granular mappings can be encountered quite frequently in rule-based systems, where the mapping is given as If-Then statements [7, 9].

In granular computing, everything, including data, variables and parameters, is allowed to be granular. In general, inaccurate measurements and perception-based information are granular, for example: 'x is small', 'approximately 90', 'temperature is high', '$[20, 25]$', 'probability is high'. In this sense, a granular system provides NL-capability [12], that is, capability to operate on information described in Natural Language. NL-capability is important because much of human knowledge is described in natural language. Imprecision of human sensory organs and brain is passed on to natural language [13]. More specifically, when a proposition expressed in a natural language is represented as a system of generalized constraints [14], it is, in effect, a granular system. Computation with information described in natural language ultimately reduces to computation with granular values.

Computing with granules brings together existing formalisms of interval analysis, fuzzy sets, rough sets, etc. under one roof. In spite of several visible distinct underpinnings of these theories, they exhibit fundamental synergies, which are exploited in the granular computing framework [3].

1.1 Interval Analysis

Interval analysis is a branch of mathematics that provides reliable numerical tools for problem solving; it treats an interval both as a set and as a number [15–20]. While arithmetic performs operations on numbers, interval arithmetic performs operations on intervals. Generally speaking, intervals are instances of granules. Granular computing materializes in the framework of interval analysis and provides features for interpretability.

Interval analysis is a theory oriented toward computational implementation because it supports the development of interval-based granular algorithms [21–23]. These algorithms are mainly designed to automatically provide rigorous bounds on approximation errors, rounding errors, and propagated uncertainties in initial data. This is of utmost importance because modeling of complex systems must compromise complexity and precision. Operations involving imprecise objects must consider the nature of the imprecision.

The main concern of the interval analysis is to provide a guaranteed approximation of the set of solutions of a problem. 'Guaranteed' in this context means that outer approximations (enclosure) of intervals can always be obtained and, moreover, be made as precise as desired when further information yields intervals of narrower width. Intervals acknowledge limited precision by associating with a variable of the model under investigation a set of reals as possible values. For ease of storage and fast computation, these sets are restricted to intervals [24]. Essentials of interval theory, which form a background of fundamentals for interval granular computing, are summarized next.

1.1.1 Interval Vectors

An interval I is a closed bounded set of real numbers

$$[l, L] \quad = \quad \{x : l \leq x \leq L\}, \tag{1}$$

where l and L denote its endpoints. An n-dimensional interval vector is an ordered n-tuple of intervals (I_1, ..., I_j, ..., I_n). If I is, e.g., a two-dimensional interval vector, then $I = (I_1, I_2)$ for some $I_1 = [l_1, L_1]$ and $I_2 = [l_2, L_2]$.

Set-theoretic operations of intersection, $\cap$, and union, $\cup$, are applicable to intervals. The intersection of two intervals, I^1 and I^2, is empty, $I^1 \cap I^2 = \emptyset$, if either $l^1 > L^2$ or $L^1 < l^2$. This indicates that I^1 and I^2 have no common points. Otherwise, the intersection of I^1 and I^2 is again an interval:

$$I^1 \cap I^2 \quad = \quad [max(l^1, l^2),\ min(L^1, L^2)]. \tag{2}$$

The intersection of interval vectors is empty if the intersection of any of their items is empty. Otherwise, for $I^1 = (I^1_1, ..., I^1_j, ..., I^1_n)$ and I^2 = (I^2_1, ..., I^2_j, ..., I^2_n) we have:

$$I^1 \cap I^2 \quad = \quad (I^1_1 \cap I^2_1, ..., I^1_j \cap I^2_j, ..., I^1_n \cap I^2_n). \tag{3}$$

If two intervals have nonempty intersection, then their union,

$$I^1 \cup I^2 \quad = \quad [min(l^1, l^2),\ max(L^1, L^2)], \tag{4}$$

is an interval. Disconnected sets must not be expressed as a single interval.

The convex hull of two interval vectors, I^1 and I^2, namely $ch(I^1, I^2)$, is the smallest interval vector containing all their elements. Then,

$$ch(I_j^1, I_j^2) \quad = \quad [min(l_j^1, l_j^2),\ max(L_j^1, L_j^2)],\ j = 1, ..., n. \tag{5}$$

Hull computation is an efficient procedure to combine sets independently of their connection. It follows that $I^1 \cup I^2 \subseteq ch(I^1, I^2)$ for any I^1 and I^2.

If $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$ are interval vectors, then

$$I^1 \subseteq I^2 \quad \text{if and only if} \quad I_j^1 \subseteq I_j^2,\ j = 1, ..., n. \tag{6}$$

We denote the width of an interval vector, namely $wdt(I)$, as the length of its largest side:

$$wdt(I) \quad = \quad max(wdt(I_1), ..., wdt(I_j), ..., wdt(I_n)), \tag{7}$$

where,

$$wdt(I_j) \quad = \quad L_j - l_j,\ j = 1, ..., n. \tag{8}$$

Finally, it is worth defining the midpoint of an interval I:

$$mp(I) \quad = \quad \frac{l + L}{2}. \tag{9}$$

Analogously, if $I = (I_1, ..., I_j, ..., I_n)$ is an interval vector, then:

$$mp(I) \quad = \quad (mp(I_1), ..., mp(I_j), ..., mp(I_n)). \tag{10}$$

1.1.2 Interval Arithmetic

Operations on real numbers can be extended to intervals. Interval arithmetic treats intervals as numbers: adding, subtracting, multiplying, and dividing them.

The rules for interval addition and subtraction are:

$$I^1 + I^2 \quad = \quad [l^1, L^1] + [l^2, L^2] = [l^1 + l^2, L^1 + L^2], \tag{11}$$
$$I^1 - I^2 \quad = \quad [l^1, L^1] - [l^2, L^2] = [l^1 - L^2, L^1 - l^2]. \tag{12}$$

Operations of addition and subtraction for interval vectors are understood to be component-wise. For two interval vectors, $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$, we have

$$I^1 + I^2 \quad = \quad (I_1^1 + I_1^2,\ ...,\ I_j^1 + I_j^2,\ ...,\ I_n^1 + I_n^2), \tag{13}$$
$$I^1 - I^2 \quad = \quad (I_1^1 - I_1^2,\ ...,\ I_j^1 - I_j^2,\ ...,\ I_n^1 - I_n^2). \tag{14}$$

For the product of two independent intervals, I^1 and I^2, we get

$$I^1 I^2 \quad = \quad \{x^1 x^2 : x^1 \in I^1, x^2 \in I^2\}. \tag{15}$$

Clearly, the result is again an interval, say I^3, whose endpoints are

$$[l^3, L^3] = [min(l^1 l^2,\ l^1 L^2,\ L^1 l^2,\ L^1 L^2),\ max(l^1 l^2,\ l^1 L^2,\ L^1 l^2,\ L^1 L^2)]. \tag{16}$$

The reciprocal of an interval I yields:

$$1/I \quad = \quad \{1/x : x \in I\}. \tag{17}$$

If I is an interval not containing the number 0, then $1/I = [1/L, 1/l]$ if $l > 0$; or $1/I = [1/l, 1/L]$ if $L < 0$. In case I contains 0 so that $l \leq 0 \leq L$, then the set is unbounded and cannot be represented as an interval whose endpoints are real numbers. For the quotient of two intervals, we have:

$$I^1/I^2 = I^1(1/I^2) = \{x^1/x^2 : x^1 \in I^1, x^2 \in I^2\}. \quad (18)$$

I^1/I^2 is again an interval if 0 is not contained in I^2. I^1 and I^2 are independent.

The product and quotient operations for interval numbers hold for interval vectors. For two interval vectors, $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$, it follows that:

$$I^1 I^2 = (I_1^1 I_1^2,\ ...,\ I_j^1 I_j^2,\ ...,\ I_n^1 I_n^2), \quad (19)$$

$$I^1/I^2 = (I_1^1/I_1^2,\ ...,\ I_j^1/I_j^2,\ ...,\ I_n^1/I_n^2). \quad (20)$$

1.1.3 Distance Between Intervals

A metric to measure the distance between two intervals, I^1 and I^2, is:

$$d(I^1, I^2) = max(|l^1 - l^2|, |L^1 - L^2|). \quad (21)$$

With this metric, the correspondence between the interval number system and the real number system, $[x, x] \leftrightarrow x$, holds [25]. The metric $d(.)$ preserves the distance between the corresponding items. We have that

$$d([x^1, x^1], [x^2, x^2]) = max(|x^1 - x^2|, |x^1 - x^2|) = |x^1 - x^2| \quad (22)$$

for any x^1 and x^2. The real line is isometrically embedded into the metric space of intervals [25].

The distance between two interval vectors, $I^1 = (I_1^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_n^2)$,

$$d(I^1, I^2) = (max(|l_1^1 - l_1^2|, |L_1^1 - L_1^2|),\ ...,\ max(|l_n^1 - l_n^2|, |L_n^1 - L_n^2|)), \quad (23)$$

is an interval vector. Sometimes, we are more interested in a number to represent the overall distance between interval vectors. A measure for the overall distance between two interval vectors, I^1 and I^2, is

$$D(I^1, I^2) = max(d(I^1, I^2)). \quad (24)$$

1.1.4 Interval Functions

Consider a real-valued function $f(x)$ and a corresponding interval-valued function $f(I)$. $f(I)$ is a united extension of $f(x)$ if $f(I) = f(x)$ for any value of $x \in I$. If the parameters of $f(I)$ are degenerated, then $f(I)$ is a degenerated interval equal to $f(x)$. Formally, the image of an interval I under a real mapping f is

$$f(I) = \{f(x) : x \in I\}. \quad (25)$$

The image of a specified n-dimensional vector I admitting a multivariable real function f is:

$$f(I_1, ..., I_j, ..., I_n) = \{f(x_1, ..., x_j, ..., x_n) : x_j \in I_j\ \forall j\}. \quad (26)$$

Generally, the image of an interval through f is not a box (see Fig. 1) and it may be difficult to obtain in closed form. In practice, $f(I)$ can be approximated by an inclusion function $F(I)$, which is a box in the range of f if f is continuous.

An interval function F from $\mathbb{IR}^n$ to $\mathbb{IR}^m$ is called an interval inclusion function of f if

$$f(I) \subseteq F(I)\ \forall I \in \mathbb{IR}^n. \quad (27)$$

Inclusion functions are not unique and they depend on how we choose F. An inclusion function is optimal if $F(I)$ is the interval hull of $f(I)$. In other words, the optimal interval inclusion function for $f(I)$ is the smallest box $F^*(I)$ that contains $f(I)$. Figure 1 illustrates the idea. $F^*(I)$ is unique.

In particular, for degenerated intervals I, it follows that:

$$F(I) = f(I) = F^*(I). \quad (28)$$

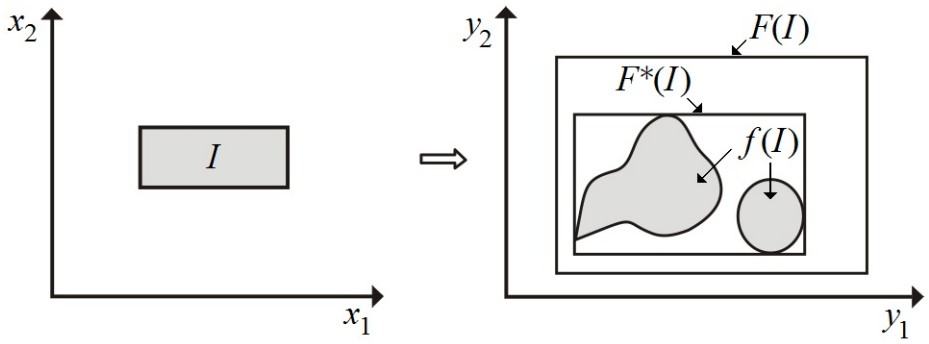

Figura 1: Image f of box I and inclusion functions F and F^*

Consider f monotonically increasing in $I = [l, L]$. Then, assuming continuity or upper semicontinuity of f, we can obtain $f(I)$ using:

$$f(I) \quad = \quad [f(l), f(L)]. \tag{29}$$

Consequently,

$$f(x) \subseteq [f(l), f(L)] \; \forall x \in I. \tag{30}$$

With monotonic decreasing functions, we order the resulting endpoints properly. In these cases $f(I) = [f(L), f(l)]$, i.e. strict inclusion relationship holds.

Nonmonotonic functions could be monotonic under endpoint constraint. For example, $f(I) = sin(I)$ is not monotonic in general but defining $I = [-\Pi/2, \Pi/2]$, then $f(I)$ is monotonic and $f(I) = sin(I) = [sin(l), sin(L)]$.

An interval function $f(I)$ is inclusion isotonic when for any interval vectors, I^1 and I^2,

$$\text{if } I^1 \subset I^2, \text{ then } f(I^1) \subset f(I^2). \tag{31}$$

Finite interval arithmetic [19] is inclusion isotonic. Let $\bullet$ denote the operations of addition, subtraction, multiplication and division, thus

$$I^1 \bullet I^2 \subset I^3 \bullet I^4 \tag{32}$$

holds whenever $I^1 \subset I^3$ and $I^2 \subset I^4$. Interval enclosures are inclusion isotonic interval extensions of real-valued continuous functions.

An interval function $f(I) \in \mathbb{IR}$ is called 'thin' when it involves only degenerate interval parameters or, equivalently, singular parameters. For instance,

$$f(I) = a_0 + \sum_{j=1}^{n} a_j I_j \tag{33}$$

is thin for $(a_0, ..., a_n)$ degenerated intervals. When an interval function involves at least one interval parameter of nonzero width, it is called 'thick'. Interval granular models may contain thin and thick interval functions.

Interval analysis goes far beyond what has been covered in this section. For instance, we do not address interval statistics [26], intervals in fuzzy set theory [27], interval integration [25], complex interval arithmetic [28], but the essential concepts to support the development of interval granular models.

1.2 From Interval Analysis to Fuzzy Set Theory

While interval analysis arose out of a need to analyze error and uncertainty on digital computers [18], fuzzy set theory arose from a need of more complete and inclusive mathematical models of uncertainty [29]. Relationships between fuzzy set theory and interval mathematics have been reported by Lodwick [30].

Fuzzy arithmetic [31] is defined by means of the extension principle for fuzzy sets [32] [29]. The extension principle for fuzzy sets is the united extension in the interval analysis terminology when the fuzzy set is restricted to be an interval [30]. When

intervals and fuzzy sets are non-interactive, arithmetic on alpha level sets is a united extension arithmetic. Both concepts are related fundamentally through what is known as set functions [33].

From the point of view of intervals as sets, interval analysis can be considered as a subset of the fuzzy set theory. For instance, an interval $[l, L]$ is a trapezoidal fuzzy set $[l, \lambda, \Lambda, L]$ where $l = \lambda$ and $\Lambda = L$ [34].

Fuzzy interval analysis [35] and interval type-2 fuzzy logic systems [36] [37] are explicit examples of joint efforts between fuzzy set theory and interval analysis to overcome the difficulties of uncertainty modeling.

Interval analysis and fuzzy set theory are instances of practical frameworks used to represent granular information and construct granular mappings. Conceptually, intervals and fuzzy sets are different ways to model imprecise quantities and capture our inherent notion of approximate numbers. 'Above 100' and 'around 1.5 and 1.7' are instances of intervals whereas 'approximately 100' and 'around 1.6' are instances of fuzzy sets.

A striking difference between intervals and fuzzy sets comes from the idea of partial membership intrinsic to fuzzy sets. Whenever interval quantification becomes too restrictive, fuzzy sets provide an important feature of describing information granules whose constituting elements may belong only partially. Fuzzy sets prevent defining hard borders between full belongingness and full exclusion by means of smooth transition boundaries. Granules formalized in the language of fuzzy sets support a vast array of human-centric pursuits [11].

1.3 Fuzzy Sets

Fuzzy sets [29,32] constitute one of the most influential notions in science and engineering. A fuzzy set captures in a granular way the essential in which much of physical phenomena is observed and described. Fuzzy information granulation underlies the basic concepts of linguistic variables, fuzzy rules, and fuzzy rule base [11]. In fuzzy set theory, objects, variables and concepts are a matter of degree. In particular, fuzzy information granulation allows both, incorporation of domain knowledge and knowledge discovery from data.

Fuzzy sets extend the notion of set by assigning to each element of a reference set a value representing its degree of membership in the fuzzy set. Membership values correspond to the degree the element is similar with typical elements representing the concept associated with the fuzzy set. This characteristic of fuzzy sets facilitates the management of the uncertainty carried by such elements.

Concepts and definitions related to fuzzy sets which are useful for granular modeling and computing are summarized in the next sections.

1.3.1 Fuzzy Set Definitions

Fuzzy sets are fully characterized by their membership functions. Any function $A : X \to [0, 1]$ may serve as a membership function of fuzzy set A. Consider trapezoidal membership functions, which are piecewise linear functions described by four parameters (l, λ, Λ, L). The membership degree of an element x in the trapezoidal fuzzy set A is

$$A(x) = \begin{cases} 0, & x < l \\ \frac{x-l}{\lambda-l}, & x \in [l, \lambda[\\ 1, & x \in [\lambda, \Lambda] \\ \frac{L-x}{L-\Lambda}, & x \in]\Lambda, L] \\ 0, & x > L \end{cases} \tag{34}$$

A fuzzy set A is normal if it produces a membership degree equal to 1 for at least one element x of the universe X. Denote sup as the supremum value of A for some element x; then A is normal if

$$sup_{x \in X} A(x) = 1. \tag{35}$$

We denote support and core of a trapezoidal membership function A respectively as the set of elements of X with nonzero membership degrees in A, and the set of elements of X with membership degrees equal to 1, that is, for a trapezoidal membership function A,

$$supp(A) = \{x \in X | A(x) > 0\} = [l, L], \quad \text{and} \tag{36}$$

$$core(A) = \{x \in X | A(x) = 1\} = [\lambda, \Lambda]. \tag{37}$$

The α-cut of a fuzzy set A, A_α, is a set containing all elements of X whose membership degrees are greater than the value α. We have

$$A_\alpha = \{x \in X | A(x) > \alpha\}. \tag{38}$$

Support ($\alpha = 0$) and core ($\alpha = 1$) are boundary cases of α-level sets.

A fuzzy set is convex if for all $x^1, x^2 \in X$ and all $\kappa \in [0, 1]$ it follows that

$$A(\kappa x^1 + (1 - \kappa)x^2) \geq min(A(x^1), A(x^2)). \tag{39}$$

A fuzzy set A^1 is a subset of A^2 if and only if every element of A^1 is also an element of A^2:

$$A^1(x) \leq A^2(x), \text{ for all } x \in X. \tag{40}$$

The midpoint and width of a membership function A are, respectively:

$$mp(A) = \frac{\lambda + \Lambda}{2}, \tag{41}$$

$$wdt(A) = L - l. \tag{42}$$

Intersection and union of two fuzzy sets, say A^1 and A^2, are defined as

$$(A^1 \cap A^2)(x) = min(A^1(x), A^2(x)) \; \forall x \in X, \tag{43}$$

$$(A^1 \cup A^2)(x) = max(A^1(x), A^2(x)) \; \forall x \in X. \tag{44}$$

The convex hull of two trapezoidal fuzzy sets A^1 and A^2 is a trapezoidal fuzzy set determined as follows:

$$ch(A^1, A^2) = (min(l^1, l^2), min(\lambda^1, \lambda^2), max(\Lambda^1, \Lambda^2), max(L^1, L^2)). \tag{45}$$

1.3.2 Fuzzy Interval

Granular data may take various forms depending on how they are modeled. They can be intervals, probability distributions, rough sets, fuzzy numbers, and fuzzy intervals [38]. Fuzzy intervals and fuzzy numbers are instances of fuzzy granular data. Fuzzy data arise in the realm of expert knowledge, whenever measurements are inaccurate, variables are hard to be precisely quantified, or pre-processing steps introduce uncertainty in numerical (singular) data.

A membership function $A : X \to [0, 1]$ is upper semi-continuous if the set $\{x \in X | A(x) > \alpha\}$ is closed, that is, if the α-cuts of A are closed intervals. If the universe X is the set of real numbers and A is normal, $A(x) = 1 \; \forall x \in [\lambda, \Lambda]$, then A is a model of a fuzzy interval, with monotone increasing function ϕ_A: $[l, \lambda[\to [0, 1]$, monotone decreasing function ι_A: $]\Lambda, L] \to [0, 1]$, and zero otherwise. A fuzzy interval A has the following canonical form:

$$A(x) = \begin{cases} \phi_A, & x \in [l, \lambda[\\ 1, & x \in [\lambda, \Lambda] \\ \iota_A, & x \in]\Lambda, L] \\ 0, & \text{otherwise} \end{cases}, \tag{46}$$

where x is a real number in X. The fuzzy interval A satisfies the conditions of normality ($A(x) = 1$ for at least one $x \in X$) and convexity ($A(\kappa x^1 + (1 - \kappa)x^2) \geq min\{A(x^1), A(x^2)\}, x^1, x^2 \in X, \kappa \in [0, 1]$). If

$$\phi_A = \frac{x - l}{\lambda - l} \text{ and} \tag{47}$$

$$\iota_A = \frac{L - x}{L - \Lambda}, \tag{48}$$

then the fuzzy membership function (46) reduces to the model of a trapezoidal membership function (34). Moreover, when $\lambda = \Lambda$, then $A(x) = 1$ for one element x. In this case the corresponding fuzzy entity is called a fuzzy number [11]. Fuzzy data generalize numeric data by allowing fuzziness.

1.3.3 Similarity Between Fuzzy Sets

Consider granular data and models as fuzzy objects of trapezoidal nature. In this case, a useful similarity measure for trapezoids, say A^1 and A^2, is:

$$S(A^1, A^2) \quad = \quad 1 - \frac{|l^1 - l^2| + |\lambda^1 - \lambda^2| + |\Lambda^1 - \Lambda^2| + |L^1 - L^2|}{4}. \tag{49}$$

This measure translates the relation between the trapezoids in a number. It returns 1 for identical trapezoids (indicating the maximum degree of matching between them) and decreases linearly when A^1 and A^2 withdraw from each other. Particularly, equation (49) is a Hamming-like metric [39] where the parameters of the trapezoids are compared one by one. A thorough discussion about similarity and compatibility measures can be found in [40].

The distance between two vectors of trapezoids, say $A^1 = (A^1_1, ..., A^1_n)$ and $A^2 = (A^2_1, ..., A^2_n)$,

$$S(A^1, A^2) \quad = \quad 1 - \frac{1}{4n}\sum_{j=1}^{n}(|l^1_j - l^2_j| + |\lambda^1_j - \lambda^2_j| + |\Lambda^1_j - \Lambda^2_j| + |L^1_j - L^2_j|), \tag{50}$$

is also a number, which quantifies their relationship.

1.4 Aggregation Operators

Aggregation operators $\mathbb{C} : [0,1]^n \to [0,1]$, $n > 1$ combine input values in the unit hypercube $[0,1]^n$ into a single output value in $[0,1]$. They must satisfy two fundamental properties: (***i***) monotonicity in all arguments, i.e., given $x^1 = (x^1_1, ..., x^1_n)$ and $x^2 = (x^2_1, ..., x^2_n)$, if $x^1_j \leq x^2_j \ \forall j$ then $\mathbb{C}(x^1) \leq \mathbb{C}(x^2)$; (***ii***) boundary conditions: $\mathbb{C}(0, 0, ..., 0) = 0$ and $\mathbb{C}(1, 1, ..., 1) = 1$. Important classes of aggregation operators are summarized below. See [11] [41] for details.

1.4.1 T-norm Aggregation

T-norms (T) are commutative, associative, and monotone operators on the unit hypercube whose boundary conditions are $T(\alpha, \alpha, ..., 0) = 0$ and $T(\alpha, 1, ..., 1) = \alpha$, $\alpha \in [0,1]$. The neutral element of T-norms is $e = 1$. An example is the minimum operator:

$$T_{min}(x) = \min_{j=1,...,n} x_j, \tag{51}$$

which is the strongest T-norm because

$$T(x) \leq T_{min}(x) \ \text{ for any } x \in [0,1]^n. \tag{52}$$

The minimum is also idempotent, symmetric, and Lipschitz-continuous. Further examples of T-norms include the product,

$$T_{prod}(x) = \prod_{j=1}^{n} x_j, \tag{53}$$

and the Lukasiewicz T-norm,

$$T_L(x) = max(0, \sum_{j=1}^{n} x_j - (n-1)). \tag{54}$$

1.4.2 S-norm Aggregation

S-norms (S) are operators on the unit hypercube which are commutative, associative, and monotone. $S(\alpha, \alpha, ..., 1) = 1$ and $S(\alpha, 0, ..., 0) = \alpha$ are the boundary conditions of S-norms. It follows that $e = 0$ is the neutral element of S-norms.

S-norms are stronger than T-norms. The maximum operator:

$$S_{max}(x) = \max_{j=1,...,n} x_j, \tag{55}$$

is the weakest S-norm, that is,

$$S(x) \geq S_{max}(x) \geq T(x), \text{ for any } x \in [0,1]^n. \tag{56}$$

Other examples of S-norms include the probabilistic sum,

$$S_{prob}(x) = 1 - \prod_{j=1}^{n}(1 - x_j), \tag{57}$$

and the Lukasiewicz S-norm,

$$S_L(x) = min(1, \sum_{j=1}^{n} x_j). \tag{58}$$

The dual $\mathbb{C}_D$ of an aggregation operator $\mathbb{C}$ is

$$\mathbb{C}_D(x_1, ..., x_n) = 1 - \mathbb{C}(1 - x_1, ..., 1 - x_n). \tag{59}$$

Maximum and minimum, probabilistic sum and product, and Lukasiewicz S and T-norms are examples of self-dual aggregation operators.

1.4.3 Uninorm Aggregation

Uninorms (U) are bivariate, associative and symmetric operators closed under duality. Similarly as with T-norms and S-norms, associativity allows n-ary extension of uninorms. Uninorms $U : [0,1]^n \rightarrow [0,1]$ generalizes triangular norms by relaxing the assumption about the neutral element e to get values in $[0,1]$. Input values higher than e are interpreted as beneficial, a positive evidence; input values lower than e are considered detrimental, a negative evidence. Naturally, when e is equal to 0 a uninorm turns into an S-norm and when $e = 1$ the uninorm becomes a T-norm.

A commonly used family of uninorms is:

$$U(x) = \begin{cases} e \ T \ \left(\frac{x_1}{e}, ..., \frac{x_n}{e}\right) & \text{if } x \in [0,e]^n \\ (e + (1-e)) \ S \ \left(\frac{x_1 - e}{1-e}, ..., \frac{x_n - e}{1-e}\right) & \text{if } x \in [e,1]^n \\ T(x_1, ..., x_n) & \text{otherwise,} \end{cases} \tag{60}$$

where $e \neq 0$ and $e \neq 1$. Any pair of T and S norms may be used to construct a uninorm U independently of their properties or duality.

1.4.4 Averaging Aggregation

An aggregation operator $\mathbb{C}$ is averaging if for every $x \in [0,1]^n$ it is bounded by

$$T_{min}(x) \leq \mathbb{C}(x) \leq S_{max}(x). \tag{61}$$

The basic rule is that the output value cannot be lower or higher than any input value. An example of averaging operator is the arithmetic mean:

$$M(x) = \frac{1}{n} \sum_{j=1}^{n} x_j. \tag{62}$$

Averaging operators are idempotent, strictly increasing, symmetric, homogeneous, and Lipschitz continuous.

1.4.5 Compensatory T-S Aggregation

Compensatory T-S operators combine T-norms and S-norms to counterbalance their opposite effects. Contrary to uninorm aggregation, T-S aggregation is uniform in the sense that it does not depend on parts of the underlying domain.

T-S operators use both a T-norm and a S-norm and averages the two values obtained by means of a weighted quasi-arithmetic mean. The linear convex operator

$$L(x) = (1 - v)T(x_1, ..., x_n) + vS(x_1, ..., x_n), \tag{63}$$

where $v \in [0, 1]$, is an example of T-S operator of the family of weighted quasi-arithmetic means. T-S operators need not to be dual in terms of T and S. It follows that:

$$S(x) \geq L(x) \geq T(x), \text{ for any } x \in [0, 1]^n. \tag{64}$$

2. EVOLVING GRANULAR SYSTEMS

Adaptability is of paramount importance for intelligent systems. As Darwin quoted [42], it is neither the strongest nor the most intelligent that survives, but the most adaptable to change. Building adaptive models from large volumes of real-world data streams requires developing non-conventional learning algorithms able to continuously track system and environment changes. Rethinking traditional data mining and modeling techniques is primordial to support structural adaptation of information systems based on sequences of data, possibly of uncertain (granular) nature.

Because data acquisition systems and small scale computing devices became mere components of complex systems, large amounts of data have been produced uninterruptedly. Storage of large-scale data sets and offline processing are frequently impractical, especially in online applications. In addition, data from different sources may be temporally and spatially related. Online learning algorithms should capture the essential information in data streams and recursively translate it into structured knowledge. The effectiveness of data stream-oriented learning algorithms is rooted in their aptitude to evolve models from nonstationary data efficiently and quickly.

Learning system models from data streams in online mode is a challenging task for most statistical and computational intelligence methods. Adaptive - and naturally non-adaptive - learning methods face a number of drawbacks when dealing with evolving data streams including: (*i*) difficulty in choosing the model structure since data sets and related information are not available; (*ii*) forgetfulness when trying to acquire new information after concept changes; and (*iii*) limited transparency and interpretability of the resulting model. In particular, there is a need for developing recursive learning methods that explore the nature of data streams [43] and at the same time fulfill accuracy, transparency and interpretability requirements [44].

2.1 Evolving Intelligent Systems

Approaches to extract meaningful information from data streams have recently been developed [45–63]. Methods and algorithms directed toward this end are known as Evolving Intelligent Systems. Evolving intelligent systems focus on nonstationary processes and embody online learning methods and one-pass incremental algorithms that evolve or gradually change individual models to guarantee life-long learning and self-organization of the system structure.

Evolving systems are a step toward a higher level of adaptability compared to conventional adaptive systems from control theory [64], classical identification systems [65], and traditional data mining systems [66, 67]. While the term 'intelligent' comes from the use of fuzzy and neuro-fuzzy (computational intelligence) techniques, the evolving aspect of these systems accounts for unbounded (infinite) amounts of data, changing concepts, and structural adaptation of models.

Formally stated, a system is said to be evolving if it: (i) learns continuously from data streams; (ii) does not store previous samples; (iii) does not depend upon prior structural knowledge; (iv) self-adapts its structure when needed; (v) is independent of statistical properties of data; and (vi) does not use 'prototype' initialization. Moreover, it is much desired that evolving systems assimilate knowledge fast using small memory requirements to support real-time applications. Evolving systems must account for the fact that the unknown is likely to matter.

In terms of implementation, evolving systems usually achieve their final purpose in software level, but they may be performed in physical embodiments including intelligent agents, embedded systems, and ubiquitous computing [43].

2.1.1 Historical Landmarks

In the beginning of this century, two mainstreams of research in evolving intelligent systems were introduced: evolving fuzzy systems [68] and evolving connectionist systems [69]. Their origins are independent of one another.

Evolving fuzzy systems (eFS) were proposed by Angelov [68], being evolving Takagi-Sugeno (eTS) fuzzy systems [70] a milestone in the field of structurally adaptive rule-based systems. The eTS is an eFS paradigm for function approximation and control that fulfils the requirements for flexible and adaptive approaches of a variety of modern applications such as automation processes, autonomous systems, intelligent sensors, and defense. eTS assumes that the antecedent and consequent parameters of functional fuzzy rules as well as the number of rules in a rule base can gradually change by learning from experience based

on data streams. This characteristic provides eTS approaches with the fundamental ability to pursue online modeling of time-varying nonstationary functions. Evolving fuzzy classifiers (eClass) [47] [71] are another approach derived from eFS when the consequent part of fuzzy rules is a class label. In eClass the number of classes needs not be known in advance and new classes can be incorporated at any time. eClass models were seminal to the field of evolving classifiers which possess the ability to capture both concept drift and shift [72].

Evolving connectionist systems (eCOS) were proposed by Kasabov [53] [69]. eCOS are artificial neural networks that operate continuously in time and adapt their structure and functionality through interaction with the environment and other systems. A paradigm of eCOS is called evolving fuzzy neural network (EFuNN) [73], which is the earliest and perhaps most influential model of eCOS. All neurons in EFuNN are created and updated during learning. They represent membership functions and rules. Information carried by a data stream is memorized on neurons and connections, and further used for predictions. The EFuNN structure evolves from hybrid (supervised and unsupervised) algorithms. Particularly, the fuzzy aspect of EFuNN permits the neural network to be interpreted as a fuzzy rule-base. Other noteworthy approaches supporting the context of eCOS are evolving self-organizing maps (eSOM) [74] and dynamic evolving neural-fuzzy inference systems (DENFIS) [75].

Common to both eFS and neurofuzzy eCOS are fuzzy sets, which are formed on a basis of numeric data through incremental clustering. Clusters give rise to fuzzy membership functions that considered together convey a global view of the available data. In evolving systems, fuzzy membership functions play a key role as the core of modeling approaches. They aim to represent similar data in a concise manner. After cluster identification, a recursive algorithm is usually used to refine local parameters and functions. In both platforms, eFS and neurofuzzy eCOS, expert knowledge can be incorporated.

From the granular computing point of view, eFS and great part of eCOS can be considered granular modeling frameworks. Fuzzy sets, used to represent numeric data, are instances of granules whereas computations in eFS and eCOS are based on the result of information granulation. However, in general, evolving intelligent systems cannot be regarded as evolving granular systems in the greatest sense of the term because they do not deal with input and output granular data and quite often do not produce granular estimation. In other words, evolving systems are granular systems internally, and singular systems externally.

Since the conception of evolving intelligent systems a diversity of studies suggesting extension of the original content has taken place. Approaches regarding primarily computational intelligence principles and ideas follow the essential notions of the original evolving intelligent systems. Conversely, there exist parallel research lines where structurally adaptive learning approaches from data streams are mostly based on data mining and statistics. Such approaches are often not referred to as 'evolving'; however, the central idea of capturing gradual and abrupt changes in nonstationary data streams is the same independently of the different terminologies. The next section reviews state-of-the-art works.

2.1.2 State of the Art

This section summarizes recent research related to learning methods capable of handling numeric data streams. We do not intend to give an exhaustive review of the literature, but to overview works closely related to the concepts and ideas of granular modeling.

The evolving participatory learning (ePL) approach [76, 77] combines the concept of participatory learning [78] with evolving Takagi-Sugeno fuzzy systems. The ePL approach is based on unsupervised clustering and therefore is a candidate to find rule base structures in adaptive fuzzy modeling. ePL uses participatory learning fuzzy clustering instead of scattering or information potential-based clustering used by eTS. At each time step, ePL updates the rule base structure using convex combinations of new data samples and the closest cluster center. The parameters of the consequent part of a rule are adapted using a recursive least squares algorithm.

The evolving multivariable Gaussian approach (eMG) [56] is an evolving functional fuzzy modeling approach which, differently from eTS, uses an evolving Gaussian clustering algorithm based on the concept of participatory learning. The clustering algorithm is one-pass and updates the eMG rule base continuously. Fuzzy sets in eMG are multivariable Gaussian membership functions which are adopted to preserve information between input variable interactions. The parameters of the membership functions, that is, cluster centers and dispersion matrices, are estimated by the clustering algorithm. A weighted recursive least squares algorithm updates the parameters of the rule consequents. The eMG clustering algorithm is particularly robust to noisy data and outliers through the use of a mechanism to smooth incompatible input data.

A data-driven incremental algorithm called flexible fuzzy inference system (FLEXFIS) [57] [79] was proposed to evolve Takagi-Sugeno fuzzy systems. A modified version of vector quantization was suggested for rules evolution. The FLEXFIS algorithm adapts linear functions of rules consequent and premise parameters (fuzzy membership functions) in online mode. Clusters of data are automatically generated based on the nature, distribution and quality of new data. Convergence toward the optimal parameter set in the least-squares sense has been achieved by the algorithm.

Self-organizing fuzzy modified least-square neural network (SOFMLS) [59] is a neurofuzzy network capable of adapting itself in real-time to a changing environment. In SOFMLS, parametric and structural model adaptation is performed simultaneously. The neural network generates a new rule if the smallest distance between a new numeric data vector and rule parameters is higher than a pre-specified radius. A density-based pruning procedure controls the network growth over time. SOFMLS does not require retraining of the whole model and has proved to be able to escape from local minima and be stable to concept changes.

The fuzzy min-max neural network (GFMM) [51] is a generalization of the fuzzy min-max clustering and classification neural networks [80]. It handles labeled and unlabeled data simultaneously in a single neural model. GFMM combines supervised and unsupervised learning to give hybrid clustering and classification. The learning process places and adjusts hyperboxes

(expansion-contraction paradigm) in the feature space in a few or one pass over data sets. GFMM is able to classify interval data and can be viewed as an incremental granular classifier.

Learn++.NSE [81] is an ensemble of classifiers-based approach for time-varying data distribution modeling. Learn++.NSE considers consecutive batches of data and makes no assumptions about the nature and rate of concept drift. The algorithm learns incrementally, similar to other algorithms of the Learn++ family [82]. Learn++.NSE trains one new classifier for each batch of data it receives and combines these classifiers using a dynamically weighted majority voting procedure. This procedure allows the algorithm to recognize and react to changes in the underlying data distributions. Since data batches are discarded after use, Learn++.NSE is suitable for online modeling of large volumes of data.

Very fast decision trees (VFDT) [83] is a method to discover knowledge in databases that builds decision trees using constant memory space and constant time to process a sample. VFDT operates on high-volume data streams and gradually creates branches and leaves if necessary. The approach uses Hoeffding bounds to guarantee that its output is asymptotically nearly identical to that of a conventional batch learner. VFDT is designed for classification purpose.

The ultra fast forest of trees (UFFT) [84] is a one-pass incremental algorithm able to detect concept drift. Trees are split according to new information appearing in a numeric data stream. In multi-class classification problems UFFT builds a binary tree for each possible pair of classes, leading to a forest of trees. Decision nodes and leaves contain naive Bayes classifiers to detect changes in class distribution and classify test examples. When changes in class distributions are detected, sub-trees rooted at representative nodes are pruned.

Differently from VFDT and UFFT, evolving fuzzy linear regression trees (eFRT) [55] convey a linear regression model in each leaf. Thus, eFRT can be used for function approximation and prediction. In general, the number of tree nodes and the number of inputs can be changed given a new sample. The tree starts with a single leaf and grows replacing leaves with sub-trees and adding more variables to the regression model. The eFRT topology is updated on the fly using a statistical model selection test that considers accuracy and number of parameters to provide accurate and parsimonious trees.

Massive Online Analysis (MOA) [85] is a software environment for learning from evolving data streams. MOA supports incremental classification and clustering approaches that do not scale with the volume of information. For classification, MOA considers boosting, bagging, and Hoeffding trees with and without naive Bayes classifiers at the leaves. For clustering, it implements the algorithms StreamKM++, CluStream, ClusTree, Den-Stream, D-Stream, and CobWeb. The aim of MOA is to provide analysis tools and insight about real-world data stream mining problems. MOA can interact with the software WEKA, the Waikato Environment for Knowledge Analysis [67].

2.2 Granular Data Streams

Physical systems change over time and usually produce considerable amounts of nonstationary data. Data streams in online environment can be granular from different perspectives. A more intuitive perspective concerns data that are granular by themselves. To elaborate, consider a simple example of predicting variable y from the last available observation x. This leads us to search for an approximand p to describe the process function f based on pairs (x, y). Here, instances x and y are singular (real numbers), and function f is single-valued. Singular data do not restrain models to be singular but rather a granular system may use granular models whose size and placement reflect the information carried by singular data. A hypothesis is that granular representation helps to assess the structure of detailed singular data and organizes the data into a more interpretable format.

Consider $x = [\underline{x}, \overline{x}]$ and $y = [\underline{y}, \overline{y}]$ as instances of a granular data stream, intervals in this case. To exemplify, $\underline{x}$ and $\overline{x}$ may denote the minimum and maximum price of an economical index during a day, and $\underline{y}$ and $\overline{y}$ the range of fluctuation of the price in the next day. In this example, data are originally granular, and models $[\underline{p}, \overline{p}]$ must be granular to support granular data. Figure 2 illustrates the granular modeling approach for function approximation.

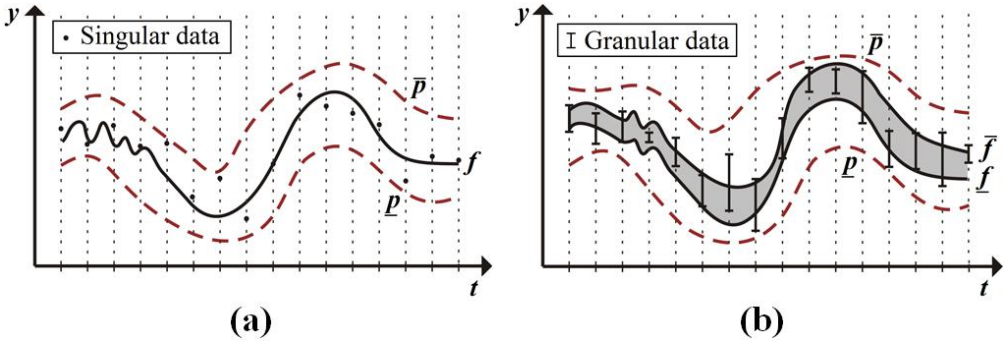

Figura 2: Granular models: (a) single-valued function, (b) granular function

Figures 2(a) and 2(b) show that granular models outer approximate single-valued and granular functions, respectively. Outer approximations of functions can always be obtained, e.g., at the top level, the coarsest possible granular approximation is the problem domain. Although merely enclosing a solution may sound at first shallower than finding the solution itself, we should reflect that the degree of satisfaction involved in embracing a solution depends strongly on the width of the enclosure obtained

[16]. Moreover, when processing stream data, we rarely have an idea about the error range and uncertainty associated with the data. On the contrary, if we can compute with granules containing a solution, then we can take for example the midpoint as a numeric approximation. Hence, we obtain both an approximate numeric solution and tolerance bounds on the approximation. The key task of approximating functions with granules is seeking for the tightest envelope for the approximand.

Another perspective for the materialization of granules in data streams is concerned with the uncertainty introduced during preprocessing steps. Incomplete data makes precise discrimination of examples difficult. Missing values are usually predicted through imputation methods [86] where the imputed data is uncertain by the very nature of the prediction and motivates granules. In privacy-preserving data mining, uncertainty may be added to the data in order to preserve the privacy of the results [87]. Additionally, noise and disturbances of bounded-error dynamic context also demand information granulation.

Granular data may arise when measurements are inaccurate or variables are hard to be quantified. For example, in sensor streams imprecision arises from inaccuracies in the underlying data acquisition equipment. Often, data are purely numerical, but the process which generated the data is uncertain. In these cases, uncertainty in data representation may be useful to improve the quality of the results. For example, an instance with greater uncertainty may not be as important as one with smaller uncertainty.

Sometimes, stream data are derived from expert knowledge. Granular computing provides a general framework to represent real-world perception in natural language [34] [38]. Various considerations can affect one's choice of data representation. Foremost among these is what Zadeh calls cointention [13], the ability of the representing object to convey the meaning of the concept it is being used to represent [88].

In a nutshell, stream data can be intervals, probability distributions, rough sets, and fuzzy intervals [38]. We define granular data streams as a sequence of samples that conveys granular information about a process. Evolving granular models are built from granular data streams. Interval and fuzzy granular data streams generalize numeric data streams by allowing interval representation and fuzziness.

2.3 Evolving Granular Modeling

Nonstationary granular system modeling encompasses adaptive and flexible learning procedures to deal with many types of data such as numbers, intervals, and fuzzy intervals. Granular computing provides a rich framework for modeling nonstationary systems using granular data streams.

Evolving granular modeling [53, 54] [70] [89–95] comes not only as an approach to capture the essence of stream data but also as a framework to extrapolate spatio-temporal correlations from lower-level raw data and provide a more abstract human-like representation of them. Research effort into granular computing toward online environment-related tasks is supported by a manifold of relevant applications such as financial, health care, video and image processing, GPS navigation, click stream analysis, online information security, process control, etc.

Our definition of evolving granular system is as follows: evolving granular systems are systems that are able to derive interpretable rule-based models and provide granular function approximation using an incremental learning algorithm and imprecise stream data (with imprecise data being numbers, intervals, fuzzy intervals, etc.) Association rules given in the form of If-Then statements can be extracted from an evolving granular construct at any time. The evolved rule base means, in essence, a granular description of a process.

In practice, evolving granular systems extend evolving intelligent systems in their capability to handle singular and granular input-output data, and give single-valued and granular approximations of original single-valued or granular functions. Granular approximation comes with a linguistic description in addition to a numeric, pointwise approximation typical of evolving intelligent systems.

Evolving granular systems rely fundamentally on the concepts of granular view, information granule, and granular mapping in the process of modeling stream data. Emphasis is on the tasks of data granulation and computing with granules [10] [96] [97]. The granularity of information explicitly embedded into granular systems offers valuable features in dynamic modeling such as transparency and flexibility. Naturally, we are concerned with a certain way of compressing granular data into more intelligible granular models.

Granular data streams are responsible for creation, expansion and shrinkage of granular models along one or more dimensions of the input and output spaces, guide parameter adaptation, and order the most appropriate granularities. Concept change, missing and noisy values, superfluous and outlier samples are common in online environments and require automatic intervention. Whenever a sample arrives, evolving algorithms should decide whether to discard it or to use it to update the current knowledge. Evolving granular learning algorithms designed to handle online granular data face odd challenges concerning the value of the current knowledge, which reduces as the concept changes; and the impossibility to neither store nor retrieve the data once read. Learning must be one-pass. Constructive (bottom-up) and decomposition-based (top-down) mechanisms predominate.

2.4 Time and Space Granulation

Data granulation may be performed in time and space domains. Approaches to building granules regard temporal granulation earlier than spatial granulation, as illustrated in Fig. 3. This order of granulation is maintained due to several reasons. Occasionally, samples are recorded at different time intervals, e.g., as in events stream. The need for synchronized analysis of manifold data streams and search for time-correlated structures give support to the possibility of considering temporal granulation first. Temporal granulation tends to slow down the data flow once several streaming instances can be wrapped by a granular object and further computations be based on granules. Time granules grant synchronism and smaller amount of granular data for subsequent

spatial analysis. Spatial correlation uniting heterogeneous data with multiple levels of granularity and different representations (intervals, fuzzy sets, rough sets, etc.) is captured during the process of spatial granulation. Structured representation of data is preserved over time as a synopsis of the data stream; it warrants structured problem solving at the practical level.

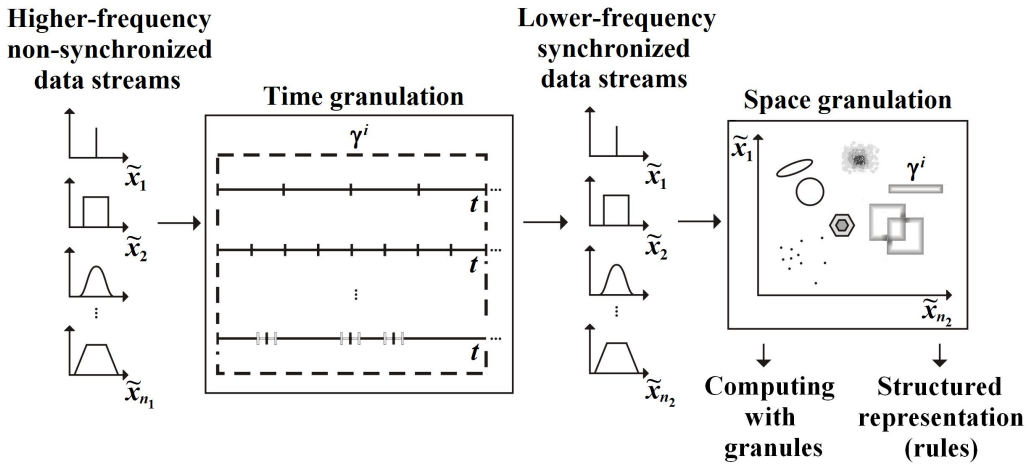

Figura 3: Time and space granulation

The flexibility of handling data streams using a granular computing framework enables us to describe granules in different application domains without deep knowledge about the problem. Tight time and memory constraints of online environment and interpretability requirements inspire granulated views of detailed data and computing at coarser granularities.

2.4.1 Time Domain Granulation

Time granulation aims at both reducing the sampling rate of fast data streams and synchronizing concurrent data streams that are input at random time intervals. A time granule describes the data for a certain time period.

Whenever the bounds of a time granule are aligned with significant shifts in the target function, the underlying granulation provides a good abstraction of the data. Conversely, if the alignment is poor, models may be inadequate [98]. Manifold granularities require temporal reasoning and respective formalizations. Time granules and time windows are distinguished as follows.

Time window [99] [100] stands for a pre-specified or adaptive duration interval within which data samples assemble a representation. Generally, a fixed number of samplings or an error value defines the size of the window. Windowing the time domain attempts to produce as few segments as possible to avoid data overfitting. Few time segments may hide information if the concept changes. Nonstationarity modifies "ideal" window lengths by its own dynamic. Approaches to testing window lengths are computationally costly and, hence, infeasible in environments with narrow time constraints. Essentially, there may exist several information granules in a time window. Data chunk analysis belongs to window-based approaches for information extraction and analysis.

A time granule groups data according to their indistinguishability in time. Since a time granule conveys similar data indexed in time, its bounds are naturally aligned with substantial changes in the function. The result of dynamic time granulation is a unique granule per segment. Time granules assume manifold levels of data abstraction and are aware of the pace of concept changes.

Event streams are examples of streams that usually come about at different time granularities. They require analysis of time-domain granules for commonalities extraction prior to space-domain analysis. Broadly stated, information evoked from time granules can be bounds of intervals, probability distributions or membership functions, and features such as frequency and correlation between events, patterns, prototypes. The internal structure of a granule and its associated variables provide full description and characterization of the granule.

Whenever manifold data streams mismatch each other at finer time granularities we resort to a granulated view of the time domain and a data mining and modeling approach. The resulting granulation should be at least as coarse as the coarsest individual stream to agree with the notion of outer approximation of functions and guaranteed solution.

2.4.2 Space Domain Granulation

Data granulation over the space domain is a process of organization for comprehension [1]. Granulation enables us to view different samples as being the same if low level details are neglected. Granulating the domain space is fundamental in methods of clustering and information integration [98] [101]. Resulting granules may compose antecedent and consequent parts of rules in rule-based systems [43] [102].

Whenever variables are recorded simultaneously and the sampling frequency is not so high that we have enough time to step recursive algorithms, the time granulation stage can be ignored and efforts fully concentrated on spatial granulation. In fact, time and space granulation are somewhat related. For instance, (*i*) with the minimal and maximal values occurring in a time granule we may form an interval granular object; (*ii*) taking a representative mean or median of instances resting into a time granule and a confidence interval around it we may form a statistical granular object; (*iii*) capturing the core and the uncertainty of instances falling in a same time granule may give rise to a fuzzy granular object. Granular objects of any precedence may be taken into consideration as input to the stage of spatial granulation.

The location and size of a granule play a role in the process of granulation. Original stream data are compressed to a few granules whose location and granularity reflect the structure of the data. There are many granulated views of the same problem. When evolving granular structures, granules are created as instances of the current knowledge. Next, granules may expand and occupy the space wherever new instances arrive. Operations on granules combine granules to form a coarser granule or decompose a granule into finer granules. Operations on granules should be consistent with the size of the granules and relations between granules; they provide the basic ingredients for the granular computing.

While concept drift and shift are terms related to the joint time-space domain [72], the descriptions of data density, data clouds and information specificity [103] [104–106] concern the space domain and are options to guide spatial granulation. Bargiela and Pedrycz [1] state that granules should encompass as many data as possible while maintaining certain specificity in what they called principle of the maximization of the information density. The principle of the balanced information granularity [1] gives preference to the design of granules balanced along all dimensions rather than granules with unbalanced geometry. In particular, hyperbox-based spatial granulation provides descriptions fully compatible with the descriptions of intervals and fuzzy sets. With intervals and fuzzy sets, the pursuit of a balanced granularity and refining and coarsening of granules are reduced to operations on bounds of intervals and parameters of fuzzy membership functions.

3. SUMMARY

This article has addressed principles and definitions of granular computing that are useful for the development of interval and fuzzy rule-based models. We argued that information granulation plays a primary role both in handling data of uncertain nature and in representing concepts described in natural language. We emphasized interval and fuzzy granular computing frameworks - with intervals and fuzzy sets being instances of information granules. When processing granular data we are in fact handling a significant number of similar individual elements at the same time and therefore ignoring details. Aggregation operators which are pertinent for information fusion within granular computing environment were also covered.

Evolving granular systems combine granular computing and evolving intelligent systems concepts into a single framework. We argued that it is sometimes unnecessary or inefficient to discriminate numeric data precisely. Moreover, we discussed that systems are better supported by a granular framework to suit uncertain, granular stream data. Numeric data is a particular case in which a granule degenerates into a singleton. The necessity of building models in finer granularities, close to the singularity, is justified only when there are clear benefits on doing so. This article presented the historical landmarks and the state of the art of the research in evolving granular systems and discussed adaptive rule-based modeling from granular data streams.

REFERENCES

[1] Bargiela, A.; Pedrycz, W. Granular Computing: An Introduction. Kluwer Academic Publishers - Boston, 1st edition, 452p. 2002.

[2] Lin, T. Y. “Neural networks, qualitative fuzzy logic and granular adaptive systems.”World Congress of Computational Intelligence, pp: 566-571, 2002.

[3] Pedrycz, W. “Granular computing - the emerging paradigm.”Journal of Uncertain Systems, Vol. 1, pp: 38-61, 2007.

[4] Yao, J. T. “A ten-year review of granular computing.”IEEE International Conference on Granular Computing, pp: 734-739, 2007.

[5] Yao, Y. Y. “Granular computing: past, present and future.”IEEE International Conference on Granular Computing, pp: 80-85, 2008.

[6] Zadeh, L. A. “Fuzzy sets and information granularity.”In: Gupta, M. M.; Ragade, R. K.; Yager, R. R. (Eds.) Advances in Fuzzy Set Theory and Applications, North Holland - Amsterdam, pp: 3-18, 1979.

[7] Bargiela, A.; Pedrycz, W. “Granular mappings.”IEEE Transactions on System, Man, and Cybernetics - Part A, Vol. 35, Issue 2, pp: 292-297, 2005.

[8] Bargiela, A.; Pedrycz, W. “Toward a theory of granular computing for human-centered information processing.”IEEE Transactions on Fuzzy Systems, Vol. 16, Issue 2, pp: 320-330, 2008.

[9] Pedrycz, W.; Skowron, A.; Kreinovich, V. (Eds.) Handbook of Granular Computing. Wiley - Chichester, England, 1116p. 2008.

[10] Yao, Y. Y. "Human-inspired granular computing."In: Yao, J. T. (Ed.) Novel Developments in Granular Computing: Applications for Advanced Human Reasoning and Soft Computing, 2010.

[11] Pedrycz, W.; Gomide, F. Fuzzy Systems Engineering: Toward Human-Centric Computing. Wiley - Hoboken, NJ, USA, 526p. 2007.

[12] Russell, S.; Norvig, P. Artificial Intelligence: A Modern Approach. Series in Artificial Intelligence, 3rd edition, 2009.

[13] Zadeh, L. A. "Generalized theory of uncertainty (GTU) - principal concepts and ideas."Computational Statistics & Data Analysis, Vol. 51, pp: 15-46, 2006.

[14] Zadeh, L. A. "Toward a generalized theory of uncertainty (GTU) - an outline."Information Sciences, Vol. 172, pp: 1-40, 2005.

[15] Hansen, E. R.; Walster, G. W. Global Optimization using Interval Analysis. 2nd edition, Marcel Dekker, New York - Basel, 489p. 2004.

[16] Jaulin, L.; Keiffer, M.; Didrit, O.; Walter, E. Applied Interval Analysis. Springer-Verlag - London, 379p. 2001.

[17] Kearfott, R. B.; Kreinovich, V. Applications of Interval Computations. Kluwer Academic Publishers, 425p. 1996.

[18] Moore, R. E. Interval Analysis. Prentice Hall - Englewood Cliffs, NJ, 145p. 1966.

[19] Moore, R. E. Methods and Applications of Interval Analysis. SIAM - Philadelphia, 190p. 1979.

[20] Neumaier, A. Interval Methods for Systems of Equations. Cambridge University Press, Cambridge, 272p. 1990.

[21] Leite, D.; Costa P.; Gomide, F. "Interval-based evolving modeling."IEEE Symposium Series on Computational Intelligence: Workshop on Evolving Systems, Nashville - US, 8p. 2009.

[22] Leite, D.; Costa, P.; Gomide, F. "Granular approach for evolving systems modeling."In: Hullermeier, E.; Kruse, R.; Hoffmann, F. (Eds.) Lecture Notes in Artificial Intelligence (LNAI/IPMU), Vol. 6178, pp: 340-349, Springer - Verlag Berlin Heidelberg, 2010.

[23] Leite, D.; Costa, P.; Gomide, F. "Interval approach for evolving granular system modeling."In: Mouchaweh, M.; Lughofer, E. (Eds.) Learning in Non-stationary Environments: Methods and Applications, Springer - New York, pp: 271-301, 2012.

[24] Hickey, T.; Ju, Q.; van Emden, M. H. "Interval arithmetic: from principles to implementation."Journal of the ACM, Vol. 48, Issue 5, pp: 1038-1068, 2001.

[25] Moore, R. E.; Kearfott, R. B.; Cloud, M. J. Introduction to Interval Analysis. SIAM - Philadelphia, 223p. 2009.

[26] Hahn, G. J.; Meeker, W. Q. Statistical Intervals: A Guide for Practitioners. Wiley, USA, 387p. 1991.

[27] Moore, R. E.; Lodwick, W. "Interval analysis and fuzzy set theory."Fuzzy Sets and Systems, Vol. 135, Issue 1, pp: 5-9, 2003.

[28] Petkovic, M. S.; Petkovic, L, D. Complex Interval Arithmetic and Its Applications. Wiley - VCH, Germany, 280p. 1998.

[29] Zadeh, L. "Fuzzy sets."Information Control, Vol. 8, pp: 338-353, 1965.

[30] Lodwick, W.; Jamison, K. D. "Special issue: interfaces between fuzzy set theory and interval analysis."Fuzzy Sets and Systems, Vol. 135, pp: 1-3, 2003.

[31] Kaufmann, A.; Gupta, M. M. Introduction to Fuzzy Arithmetic: Theory and Applications. Van Nostrand Reinhold Company Inc., New York, 350p. 1985.

[32] Klir, G. K.; Yuan, B. Fuzzy Sets and Fuzzy Logic: Theory and Applications. Prentice Hall, 1st edition, 592p. 1995.

[33] Strother, W. "Continuous multi-valued functions."The Bulletin of Sao Paulo Mathematical Society, Vol. 10, pp: 87-120, 1958.

[34] Yager, R. R. "Learning from imprecise granular data using trapezoidal fuzzy set representations."In: Prade, H.; Subrahmanian, V. S. (Eds.) Lecture Notes in Computer Science, Springer - Berlim, Heidelberg, Vol. 4772, pp: 244-254, 2007.

[35] Dubois, D.; Kerre, E.; Mesiar, R.; Prade, H. "Fuzzy interval analysis."In: The Handbook of Fuzzy Sets, Vol. 1 - Fundamentals of Fuzzy Sets, Kluwer Academic - Bordrecht, pp: 483-581, 2000.

[36] Mendel, J. M. "Type-2 fuzzy sets and systems: an overview."IEEE Computational Intelligence Magazine, Vol. 2, Issue 2, pp: 20-29, 2007.

[37] Zadeh, L. "The concept of a linguistic variable and its application to approximate reasoning."Information Science, Vol. 8, pp: 199-249, 1975.

[38] Dubois, D.; Prade, H. "On the use of aggregation operations in information fusion processes."Fuzzy Sets and Systems, Vol. 142, Issue 1, pp: 143-161, 2004.

[39] Hamming, R. W. "Error detecting and error correcting codes."Bell System Technical Journal, Vol. 29, Issue 2, pp: 147-160, 1950.

[40] Cross, V. V.; Sudkamp, T. A. Similarity and compatibility in fuzzy set theory: assessment and applications. Physica-Verlag Heidelberg (Studies in Fuzziness and Soft Computing), 209p. 2002.

[41] Beliakov, G.; Pradera, A.; Calvo, T. Aggregation Functions: A Guide for Practitioners. Springer-Verlag, Berlin, Heidelberg, 1st edition (Studies in Fuzziness and Soft Computing), 361p. 2007.

[42] Darwin, C. R. The origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. John Murray - London, 6th edition, 1872.

[43] Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Intelligent Systems: Methodology and Applications. Wiley-IEEE Press Series on Computational Intelligence, 444p. 2010.

[44] Alonso, J. M.; Magdalena, L. "Special issue on interpretable fuzzy systems."Information Sciences, Vol. 181, pp: 4331-4339, 2011.

[45] Aggarwal, C. C.; Han, J.; Wang, J.; Yu, P. S. "A framework for on-demand classification of evolving data streams."IEEE Transactions on Knowledge and Data Engineering, Vol. 18, Issue 5, pp: 577-589, 2006.

[46] Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Fuzzy Systems - Preface to the Special Section. IEEE Transactions on Fuzzy Systems, Vol. 6, Issue 6, pp: 1390-1392, 2008.

[47] Angelov, P.; Zhou, X. "Evolving fuzzy-rule-based classifiers from data streams."IEEE Transactions on Fuzzy Systems, Vol. 16, Issue 6, pp: 1462-1475, 2008.

[48] Bouchachia, A.; Gabrys, B.; Sahel, Z. "Overview of some incremental learning algorithms."IEEE International Conference on Fuzzy Systems, pp: 1-6, 2007.

[49] Bouchachia, A. "An evolving classification cascade with self-learning."Evolving Systems, Vol. 1, Issue 3, pp: 143-160, 2010.

[50] Chen, S.; He, H. "Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach."Evolving Systems, Vol. 2, Issue 1, pp: 35-50, 2011.

[51] Gabrys, B.; Bargiela, A. "General fuzzy min-max neural network for clustering and classification."IEEE Transactions on Neural Networks, Vol. 11, Issue 3, pp: 769-783, 2000.

[52] Iglesias, J. A.; Angelov, P.; Ledezma, A.; Sanchis, A. "Evolving classification of agents?behaviors: a general approach."Evolving Systems, Vol. 1, Issue 3, pp: 161-171, 2010.

[53] Kasabov, N. Evolving Connectionist Systems: The Knowledge Engineering Approach. Springer-Verlag - London, 2nd edition, 451p. 2007.

[54] Leite, D.; Gomide, F.; Ballini, R.; Costa, P. "Fuzzy granular evolving modeling for time series prediction."IEEE International Conference on Fuzzy Systems, pp: 2794-2801, 2011.

[55] Lemos, A.; Caminhas, W.; Gomide, F. "Fuzzy evolving linear regression trees."Evolving Systems, Vol. 2, Issue 1, pp: 1-14, 2011.

[56] Lemos, A.; Caminhas, W.; Gomide, F. "Multivariable Gaussian evolving fuzzy modeling system."IEEE Transactions on Fuzzy Systems, Vol. 19, Issue 1, pp: 91-104, 2011.

[57] Lughofer, E. "FLEXFIS: A robust incremental learning approach for evolving Takagi-Sugeno fuzzy models."IEEE Transactions on Fuzzy Systems, Vol. 16 , Issue 6, pp: 1393-1410, 2008.

[58] Pouzols, F. M.; Lendasse, A. "Evolving fuzzy optimally pruned extreme learning machine for regression problems."Evolving Systems, Vol. 1, Issue 1, pp: 43-58, 2010.

[59] Rubio, J. J. "SOFMLS: Online self-organizing fuzzy modified least-squares network."IEEE Transactions on Fuzzy Systems, Vol. 17, Issue 6, pp: 1296 - 1309, 2009.

[60] Rubio, J. J. "Stability analysis for an online evolving neuro-fuzzy recurrent network."In: Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Intelligent Systems: Methodology and Applications, Wiley - IEEE Press, pp: 173-199, 2010.

[61] Vachkov, G. "Spatial-temporal knowledge base for modeling and analysis of evolving systems."Evolving Systems, Vol. 2, Issue 2, pp: 131-143, 2011.

[62] Pratama, M.; Anavatti, S. G.; Joo, M.; Lughofer, E. D. "pClass: An effective classifier for streaming examples."IEEE Transactions on Fuzzy Systems, Vol. 23, Issue 2, pp: 369-386, 2015.

[63] Beringer, J.; Hullermeier, E. "Online clustering of parallel data streams."Data & Knowledge Engineering, Vol. 58, Issue 2, pp: 180-204, 2006.

[64] Astrom, K. J.; Wittenmark, B. Adaptive Control. Prentice-Hall, Addison-Wesley, Boston, 2nd edition, 580p. 1994.

[65] Ljung, L. System Identification - Theory for the User. Prentice-Hall, Englewood Cliffs, NJ, 519p. 1988.

[66] Hastie, T.; Tibshirani, R.; Friedman, J. The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer-Verlag, 2nd edition, 768p. 2009.

[67] Witten, I. H.; Frank, E.; Hall, M. A. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, 3rd edition, 664p. 2011.

[68] Angelov, P. Evolving Rule-Based Models: A Tool for Design of Flexible Adaptive Systems. Springer-Verlag, Heidelberg, New York (Studies in Fuzziness and Soft Computing), 227p. 2002.

[69] Kasabov, N. Evolving Connectionist Systems: Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines. Springer-Verlag - London, 1st edition, 320p. 2003.

[70] Angelov, P.; Filev, D. "An approach to online identification of Takagi-Sugeno fuzzy models."IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 34, Issue 1, pp: 484-498, 2004.

[71] Angelov, P.; Zhou, X.; Filev, D.; Lughofer, E. "Architectures for evolving fuzzy rule-based classifiers."IEEE International Conference on Systems, Man and Cybernetics, pp: 2050-2055, 2007.

[72] Lughofer, E.; Angelov, P. "Handling drifts and shifts in on-line data streams with evolving fuzzy systems."Applied Soft Computing, Vol. 11, Issue 2, pp: 2057-2068, 2011.

[73] Kasabov, N. "Evolving fuzzy neural networks for supervised / unsupervised online knowledge-based learning."IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 31, Issue 6, pp: 902-918, 2001.

[74] Da Deng; Kasabov, N. "ESOM: An algorithm to evolve self-organizing maps from online data streams."IEEE International Joint Conference on Neural Networks, Vol. 6, pp: 3-8, 2000.

[75] Kasabov, N.; Song, Q. "DENFIS: Dynamic evolving neural-fuzzy inference system and its application."IEEE Transactions on Fuzzy Systems, Vol. 10, Issue 2, pp: 144-154, 2002.

[76] Lima, E.; Gomide, F.; Ballini, R. "Participatory evolving fuzzy modeling."International Symposium on Evolving Fuzzy Systems, pp: 36-41, 2006.

[77] Maciel, L.; Gomide, F.; Ballini, R. "Enhanced evolving participatory learning fuzzy modeling: an application for asset returns volatility forecasting."Evolving Systems, Vol. 5, Issue 2, pp: 75-88, 2014.

[78] Yager, R. R. "A model of participatory learning."IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, Issue 5, pp: 1229-1234, 1990.

[79] Lughofer, E.; Cernuda, C.; Pratama, M. "Generalized FLEXible Fuzzy Inference Systems."12th International Conference on Machine Learning and Applications, 7p. 2013.

[80] Simpson, P. K. "Fuzzy min-max neural networks. Part I: classification."IEEE Transactions on Neural Networks, Vol. 3, Issue 5, pp: 776-786, 1992.

[81] Elwell, R.; Polikar, R. "Incremental learning of concept drift in nonstationary environments."IEEE Transactions on Neural Networks, Vol. 22, Issue 10, pp: 1517-1531, 2011.

[82] Muhlbaier, M.; Topalis, A.; Polikar, R. "Learn++.NC: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes."IEEE Transactions on Neural Networks, Vol. 20, Issue 1, pp: 152-168, 2009.

[83] Domingos, P.; Hulten, G. "Mining high-speed data streams."International Conference on Knowledge Discovery and Data Mining, pp: 71-80, 2000.

[84] Gama, J.; Medas, P. "Learning decision trees from dynamic data streams."Journal of Universal Computer Science, Vol. 11, Issue 8, pp: 1353-1366, 2005.

[85] Bifet, A.; Holmes, G.; Pfahringer, B.; Kranen, P.; Kremer, H.; Jansen, T.; Seidl, T. "MOA: Massive online analysis, a framework for stream classification and clustering."Journal of Machine Learning Research, Vol. 11, pp: 44-50, 2010.

[86] Little, R. J. A.; Rubin, D. B. Statistical Analysis with Missing Data. Wiley-Interscience, 2nd edition, 381p. 2002.

[87] Aggarwal, C. C.; Yu, P. S. (Eds.) Privacy-Preserving Data Mining: Models and Algorithms. Springer-Verlag (Series: Advances in Database Systems), Vol. 34, 513p. 2008.

[88] Yager, R. R. "Participatory learning with granular observations."IEEE Transactions on Fuzzy Systems, Vol. 17, Issue 1, pp: 1-13, 2009.

[89] Bargiela A.; Pedrycz, W. "Granulation of temporal data: a global view on time series."International Conference of the North American Fuzzy Information Processing Society, pp: 191-196, 2003.

[90] Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural network for semi-supervised data stream classification."World Congress on Computational Intelligence - International Joint Conference on Neural Networks, pp: 1877-1884, 2010.

[91] Leite, D.; Palhares, R.; Campos, V.; Gomide, F. "Evolving granular fuzzy model-based control of nonlinear dynamic systems."IEEE Transactions on Fuzzy Systems, Vol. 23, Issue 4, pp: 923-938, 2015.

[92] Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural networks from fuzzy data streams."Neural Networks, Vol. 38, pp: 1-16, 2013.

[93] Leite, D.; Gomide, F. "Evolving linguistic fuzzy models from data streams."In: Trillas, E.; Bonissone, P.; Magdalena, L.; Kacprycz, J. (Eds.) Combining Experimentation and Theory: A Hommage to Abe Mamdani (Studies in Fuzziness and Soft Computing), pp: 209-223, 2011.

[94] Lughofer, E. Evolving Fuzzy Systems - Methodologies, Advanced Concepts and Applications. Springer-Verlag, Berlin Heidelberg, 460p. 2011.

[95] Pedrycz, W. "Evolvable fuzzy systems: some insights and challenges."Evolving Systems, Vol. 1, Issue 2, pp: 73-82, 2010.

[96] Yao, Y. Y. "Interpreting concept learning in cognitive informatics and granular computing."IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 39, Issue 4, pp: 855-866, 2009.

[97] Zadeh, L. A. "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic."Fuzzy Sets and Systems, Vol. 90, Issue 2, pp: 111-127, 1997.

[98] Bargiela, A.; Pedrycz, W. "Recursive information granulation: aggregation and interpretation issues."IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 33, Issue 1, pp: 96-112, 2003.

[99] Last, M. "Online classification of nonstationary data streams."Intelligent Data Analysis, Vol. 6, Issue 2, pp: 129-147, 2002.

[100] Ozawa, S.; Pang, S.; Kasabov, N. "Incremental learning of chunk data for online pattern classification systems."IEEE Transactions on Neural Networks, Vol. 19, Issue 6, pp: 1061-1074, 2008.

[101] Aggarwal, C. C.; Yu, P. S. "A framework for clustering uncertain data streams."IEEE International Conference on Data Engineering, pp: 150-159, 2008.

[102] Kuncheva, L. I. Fuzzy Classifier Design. Springer-Verlag, Heidelberg, 321p. 2000.

[103] Yager, R. R. "Measures of specificity over continuous spaces under similarity relations."Fuzzy Sets and Systems, Vol. 159, Issue 17, pp: 2193-2210, 2008.

[104] Angelov, P. "Anomaly detection based on eccentricity analysis."IEEE Symposium on Evolving and Autonomous Learning Systems (EALS), 2014.

[105] Bezerra, C. G.; Costa, B. S.; Guedes, L. A.; Angelov, P. P. "A new evolving clustering algorithm for online data streams."IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), 2016.

[106] Kangin, D.; Angelov, P.; Iglesias, J. A. "Autonomously evolving classifier TEDAClass."Information Sciences, Vol. 366, Issue 20, pp: 1-11, Oct. 2016.