Granular Computing—Computing with Uncertain, Imprecise and Partially True Data

Lotfi A. Zadeh

Department of EECS,
University of California
Berkeley
USA

As we move further into the age of machine intelligence and automated decision-making, a pressing need arises for methods of computation which can deal effectively with information which is imprecise, uncertain, incomplete and partially true. Granular computing is intended to serve this need.

In granular computing, the objects of computation are not the values of variables but information about the values of variables. As an illustration, suppose that a computation calls for the value of Vera's age. I do not know what it is, but what I know is that Vera has a son who is about 25, and a daughter who is about 35. Furthermore, I know that the child bearing age varies from about 16 to about 42. Given this information, how can I estimate Vera's age?

Existing methods of computation are based for the most part on bivalent logic and bivalent-logic-based probability theory. The problem is that bivalent logic is intolerant of imprecision and partial truth. Granular computing is based on fuzzy logic. In fuzzy logic, everything is or is allowed to be graduated, that is, be a matter of degree or, equivalently, fuzzy. Furthermore, in fuzzy logic everything is or is allowed to be granulated, with a granule being a clump of values which are drawn together by indistinguishability, similarity, proximity, or functionality. For example, an interval is a granule; so is a fuzzy interval; and so is a gaussian distribution. In granular computing, a granule is interpreted as a representation of the state of knowledge of the value of a variable. A granular value is a label of a granule. For example, the granular values of Age may be young, middle-aged and old. A granular variable is a variable which takes granules as values. A linguistic variable is a granular variable whose granular values are labels drawn from a natural language.

A concept which serves to precisiate the concept of a granule is that of a generalized constraint. The concept of a generalized constraint is the centerpiece of granular computing.

A generalized constraint is an expression of the form X isr R, where X is the constrained variable, R is the constraining relation, and r is an indexical variable which serves to identify the modality of the constraint. The principal modalities are: possibilistic (r=blank); veristic (r=v); probabilistic (r=p); usuality (r=u); random set (r=rs); fuzzy graph (r=fg); bimodal (r=bm); and group (r=g). The primary constraints are possibilistic, veristic and probabilistic. The standard constraints are bivalent possibilistic, bivalent veristic and probabilistic. Standard constraints have a position of centrality in existing scientific theories.

A generalized constraint, GC(X), is open if X is a free variable, and is closed (grounded) if X is instantiated. A proposition is a closed generalized constraint. For example, "Lily is young," is a closed possibilistic constraint in which X=Age(Lily); r=blank; and R=young is a fuzzy set. Unless indicated to the contrary, a generalized constraint is assumed to be closed.

A generalized constraint may be generated by combining, projecting, qualifying, propagating and counterpropagating other generalized constraints. The set of all generalized constraints together with the rules governing combination, projection, qualification, propagation and counterpropagation constitute the Generalized Constraint Language (GCL).

In granular computing, computation or equivalently deduction, is viewed as a sequence of operations involving combination, projection, qualification, propagation and counterpropagation of generalized constraints. An instance of projection is deduction of GC(X) from GC(X, Y); an instance of propagation is deduction of GC(f(X)) from GC(X), where f is a function or a functional; an instance of counterpropagation is deduction of GC(X) from GC(f(X)); an instance of combination is deduction of GC(f(X,Y)) from GC(X) and GC(Y); and an instance of qualification is computation of X isr R when X is a generalized constraint. An example of probability qualification is (X is small) is likely. An example of veristic (truth) qualification is (X is small) is not very true.

The principal deduction rule in granular computing is the possibilistic extension principle: f(X) is A $\longrightarrow$ g(X) is B, where A and B are fuzzy sets, and B is given by $\mu_B(v)=\sup_u(\mu_A(f(u)))$, subject to v=g(u). μ_A and μ_B are the membership functions of A and B, respectively.

A key idea in granular computing may be expressed as the fundamental thesis: information is expressible as a generalized constraint. The traditional view that information is statistical in nature may be viewed as a special, albeit important, case of the fundamental thesis.

A proposition is a carrier of information. As a consequence of the fundamental thesis, the meaning of a proposition is expressible as a generalized constraint. This meaning postulate serves as a bridge between granular computing and NL-Computation, that is, computation with information described in a natural language.

The point of departure in NL-Computation is (a) an input dataset which consists of a collection of propositions described in a natural language; and (b) a query, q, described in a natural language. To compute an answer to the query, the given propositions are precisiated through translation into the Generalized Constraint Language (GCL). The translates which express the meanings of given propositions are generalized constraints. Once the input dataset is expressed as a system of generalized constraints, granular computing is employed to compute the answer to the query.

As a simple illustration assume that the input dataset consists of the proposition "Most Swedes are tall," and the query is "What is the average height of Swedes?" Let h be the height density function, meaning that h(u)du is the fraction of Swedes whose height lies in the interval [u, u + du]. The given proposition "Most Swedes are tall," translates into a generalized constraint on h, and so does the translate of the query "What is the average height of Swedes?" Employing the extension principle, the generalized constraint on h propagates to a generalized constraint on the answer to q. Computation of the answer to q reduces to solution of a variational problem. A concomitant of the close relationship between granular computing and NL-Computation is a close relationship between granular computing and the computational theory of perceptions. More specifically, a natural language may be viewed as a system for describing perceptions. This observation suggests a way of computing with perceptions by reducing the problem of computation with perceptions to that of computation with their natural language descriptions, that is, to NL-Computation. In turn, NL-Computation is reduced to granular computing through translation/precisiation into the Generalized Constraint Language (GCL).

An interesting application of the relationship between granular computing and the computational theory of perceptions involves what may be called perception-based arithmetic. In this arithmetic, the objects of arithmetic operations are perceptions of numbers rather than numbers themselves. More specifically, a perception of a number, a, is expressed as usually (*a), where *a denotes "approximately a." For concreteness, *a is defined as a fuzzy interval centering on a, and usually is defined as a fuzzy probability. In this setting, a basic question is: What is the sum of usually (*a) and usually (*b)? Granular computing and, more particularly, granular arithmetic, provide a machinery for dealing with questions of this type.

Imprecision, uncertainty and partiality of truth are pervasive characteristics of the real world. As we move further into the age of machine intelligence and automated reasoning, the need for an enhancement of our ability to deal with imprecision, uncertainty and partiality of truth is certain to grow in visibility and importance. It is this need that motivated the genesis of granular computing and is driving its progress. In coming years, granular computing and NL-Computation are likely to become a part of the mainstream of computation and machine intelligence.