Generalization and Formalization of Precisiation Language with Applications to Human-Robot Interaction

Takehiko Nakama, Enrique Muñoz and Enrique Ruspini
European Center for Soft Computing, c/Gonzalo Gutiérrez Quirós, s/n, 33600 Mieres, Spain

Keywords: Precisiated Natural Language, Precisiation Language, Formal Logic, Propositional Logic, Predicate Logic, Quantificational Logic, Fuzzy Logic, Fuzzy Relation, Human-Robot Interaction.

Abstract: We generalize and formalize precisiation language by establishing a formal logic as a generalized precisiation language. Various syntactic structures in natural language are incorporated in the syntax of the formal logic so that it can serve as a middle ground between the natural-language-based mode of human communication and the low-level mode of machine communication. As regards the semantics, we establish the formal logic as a many-valued logic, and fuzzy relations are employed to determine the truth values of propositions efficiently. We discuss how the generalized precisiation language can facilitate human-robot interaction.

1 INTRODUCTION AND SUMMARY

In his computational theory of perceptions (e.g., (Zadeh, 2001), (Zadeh, 2002), (Zadeh, 2004)), Zadeh introduced the concept of precisiated natural language (PNL), which refers to a set of natural-language propositions that can be treated as objects of computation and deduction. The propositions in PNL are assumed to describe human perceptions, and they allow artificial intelligence to operate on and reason with perception-based information, which is intrinsically imprecise, uncertain, or vague.

Precisiation language is an integral part of this framework. Each proposition in PNL is translated into a precisiation language, which then expresses it as a set or a sequence of computational objects that can be effectively processed by machines (e.g., (Zadeh, 2001), (Zadeh, 2002), (Zadeh, 2004)). Zadeh proposed a precisiation language in which each proposition is a generalized constraint on a variable. This precisation language is called a generalized-constraint language.

Zadeh considered the primary function of natural language as describing human perceptions, and his PNL and precisiation language only deal with perceptual propositions ((Zadeh, 2001), (Zadeh, 2002), (Zadeh, 2004)). However, the importance of natural language is not limited to describing human perceptions. For instance, using a natural language, we describe not only perceptions but also actions. Therefore, it is important, both theoretically and practically, to extend PNL and precisiaton language to other types of proposition. Generalized constraints in Zadeh's precisiation language are suitable for precisiating perceptual propositions but not for precisiating action-related propositions (See Section 3).

One of the major fields that require the precisiation of action-related propositions in natural language is robotics. Recently, many studies (e.g., (Marble et al., 2004), (Dias et al., 2006), (Dias et al., 2008b), (Johnson et al., 2008), (Johnson and Intlekofer, 2008)) have been conducted to develop robotic systems in which humans and robots work as true team members, requiring peer-to-peer human-robot interaction. Such systems can be highly effective and efficient in performing a wide range of practical tasks—assistance to people with disabilities (e.g., (Lacey and Dawson-Howe, 1998), (Shim et al., 2004), (Feil-Seifer and Mataric, 2008),(Kulyukin et al., 2006)), search and rescue (e.g., (Kitano et al., 1999), (Casper and Murphy, 2003), (Dias et al., 2008a), (Norbakhsh et al., 2005)), and space exploration (e.g., (Wilcox and Nguyen, 1998), (Fong and Thorpe, 2001), (Fong et al., 2005), (Ferketic et al., 2006)), for instance. One of the major challenges of developing these robotic systems is the increased complexity of the human-robot interactions (e.g., (Goodrich and Schultz, 2007)). Although humans prefer natural language as a communication medium, it presents several major problems when used for

Nakama T., Muñoz E. and Ruspini E..
Generalization and Formalization of Precisiation Language with Applications to Human-Robot Interaction.
DOI: 10.5220/0004660703320343
In *Proceedings of the 5th International Joint Conference on Computational Intelligence* (SCA-2013), pages 332-343
ISBN: 978-989-8565-77-8

human-robot communications; natural-language expressions tend to be notoriously underspecified, diverse, vague, or ambiguous, so they often lead to errors that are hard to overcome (e.g., (Winograd and Flores, 1986), (Tomassi, 1999), (Shneiderman, 2000), (Forsberg, 2003), (Gieselmann and Stenneken, 2006)). Low-level sensory and motor signals and executable code are easy for machines to interpret, but they are cumbersome for humans and thus cannot, on their own, create an effective human-robot interface. Task descriptions or specifications for robotic systems typically involve action-related propositions, such as "*bring the box to the room*" and "*keep the robot in the building if it rains*." A generalized precisiation language that can precisiate not only perceptual propositions but also action-related propositions can effectively mediate human-robot interaction in robotic systems that employ a peer-to-peer communication mode.

Recently, Nakama et al. (Nakama et al., 2013) have taken a first step toward generalizing and formalizing precisiation language by establishing a formal logic as a generalized precisiation language. Their formal logic can generate infinitely many precisiated propositions, just as infinitely many propositions can be generated in natural language, while ensuring that every proposition in the formal logic is precisiated. In this paper, we further develop and elaborate on the framework proposed in (Nakama et al., 2013).

The remainder of this paper is organized as follows. In Section 2, we examine the properties of formal logic that are desirable for precisiating natural-language expressions. The syntax of our formal logic is explained in Section 3. In Section 4, we discuss the generality of our framework. In Section 5, we examine how to add a deductive apparatus to our formal logic so that we can infer and reason in it. In Section 6, we develop a hierarchy of propositions that enhances the expressive power and the interactivity of our formal logic. The semantics of the formal logic is explained in Section 7.

Since the precisiation of perceptual propositions has been examined (e.g., (Zadeh, 2001), (Zadeh, 2004)), we focus on examining how to precisiate action-related propositions in this paper. In (Muñoz et al., 2013), we provide details on how to extend our generalized precisiation language to perceptual propositions. To explain our formal scheme, we will first appeal to ordinary practices and then move on to formal considerations so that the reader can understand it intuitively. To generate examples of ordinary practice, we consider establishing task descriptions for human-robot interaction. As mentioned earlier, task descriptions inherently involve actions, so the precisiation of task descriptions is an important step toward generalizing precisiation language and PNL.

2 PROPERTIES OF FORMAL LOGIC SUITABLE FOR PRECISIATION OF NATURAL-LANGUAGE EXPRESSIONS

The application of formal logic to natural language is a paradigm of logical analysis (Tomassi, 1999); it provides genuine insight into the syntactic structures of natural-language sentences and the consequential characters of assertions expressed by them. This analysis is important for precisiating propositions in natural language.

In order for PNL to have high expressive power, it is desirable that precisiation language can generate infinitely many precisiated propositions while ensuring that every proposition in it is precisiated. Formal logic achieves these properties by a recursive definition of its syntax; it can generate infinitely many well-formed formulas while ensuring that every formula in it is well-formed.

As in other formal logics, we can reason logically in our formal logic by adding a deductive apparatus to it; the resulting analytical machinery allows us to determine when one sentence in the formal language follows logically from other sentences. Thus our formal logic precisiates the inference and the reasoning in which humans engage using a natural language. See Section 5.

Our scheme also reflects the theory of descriptions in formal logic, which was introduced by Russell (Russell, 1984). He claimed that the reality consists of logical atoms, which can be considered indecomposable, self-contained building blocks of all propositions in formal logic, and that logical analysis ends when we arrive at logical atoms. In our precisiation language, precisiation ends when we arrive at logical atoms, which will be represented by atomic propositions at the lowest level of a hierarchy of propositions. See Section 6.

3 SYNTAX OF THE FORMAL LOGIC

In this section, we describe how to form propositions in our formal language. To explain our scheme, we will first appeal to ordinary practices and then

move on to formal considerations so that the reader can understand it intuitively. To generate examples of ordinary practice, we consider establishing task descriptions for human-robot interaction, but keep in mind that our scheme is not limited to precisiating propositions that describe tasks. (Also notice that, since tasks typically involve actions, we will be extending PNL and precisiation language to action-related propositions.) We will discuss the generality of our formal logic in Section 4. In this formal logic, each proposition has a syntactic form observed in natural language. Our formal logic generalizes Zadeh's generalized-constraint language by incorporating multiple syntactic forms so that it can deal with not only perceptual propositions but also action-related propositions.

In Section 3.1, we describe the components of such propositions. In Section 3.2, we describe how to form an atomic proposition. In Section 3.3, we describe how to form a compound proposition. In Section 3.4, we provide a recursive definition of well-formed formulas that allows our formal logic to generate infinitely many well-formed formulas while ensuring that every formula in it is well-formed.

In these sections, we will provide examples of rather simple task descriptions, but our scheme can be easily applied to robotic systems that require more intricate task descriptions. See Section 4.

3.1 Component Sets

In our formal logic, we generate propositions using elements in component sets. To provide concrete examples, we consider the sets S, V, O, A and C shown in Table 1 as component sets.

In Section 3.2, we will explain how to generate atomic propositions using elements in S, V, O, and A. In this example, the elements in S are also in O because they can not only perform a task but also receive an action in V. The element labeled as "null," called the null element, is included in O and A. In Section 3.2, we will explain how the null element is used in forming atomic propositions. In Section 3.3, we will explain how to form compound propositions using the connectives in C.

Although the component sets in Table 1 are rather simple, they can be made as rich as necessary, and other types of component sets can be incorporated in our formal logic. See Section 4.

3.2 Atomic Propositions

In our formal logic, an atomic proposition is defined to be a tuple in the Cartesian product of component sets, and the Cartesian product specifies each admissible tuple structure. To develop formal propositions that can be easily identified with natural-language sentences, we employ tuple structures that reflect syntactic structures observed in natural languages such as English. For instance, using the component sets described in Section 3.1, we can define each atomic proposition in our formal logic to be an SVOA clause (The S, V, O, and A in SVOA stand for subject, verb, object, and adverbial phrase, respectively) by setting the admissible tuple structure to $S \times V \times O \times A$. The SVOA structure is observed in many languages, including English, Russian, and Mandarin. Using the null element in O and A, we can also generate SVO, SVA, and SV clauses. See the following examples of atomic propositions resulting from the component sets in Table 1:

Table 1: Examples of component sets.

set	elements
S	agents that can perform tasks e.g., $S = \{robot1, robot2, user\}$
V	verbs that characterize actions required by tasks e.g., $V = \{find, deliver, go, move, press\}$
O	objects that may receive an action in V or compose an adverbial phrase e.g., $O = \{box, button, table, room1, room2, robot1, robot2, user, null\}$
A	adverbial phrases that can be included in task descriptions e.g., $A = \{in\ \gamma, from\ \gamma, from\ \gamma_1\ to\ \gamma_2, to\ \gamma, null \mid \gamma, \gamma_1, \gamma_2 \in O\}$
C	connectives that can be used to combine multiple propositions in forming compound propositions e.g., $C = \{and, if, or, then\}$

- $\frac{robot1}{S}\ \frac{move}{V}$. (The actual form of this proposition is $\frac{robot1}{S}\ \frac{move}{V}\ \frac{null}{O}\ \frac{null}{A}$, but we will omit instances of the null element to simplify the resulting expressions.)
- $\frac{robot2}{S}\ \frac{move}{V}\ \frac{to\ room1}{A}\ \left(\frac{robot1}{S}\ \frac{move}{V}\ \frac{null}{O}\ \frac{to\ room1}{A}\right)$.
- $\frac{robot2}{S}\ \frac{find}{V}\ \frac{ball}{O}$.
- $\frac{robot1}{S}\ \frac{deliver}{V}\ \frac{box}{O}\ \frac{from\ room1\ to\ room2}{A}$.

For humans, these propositions (task descriptions)

are easy to specify and understand. Meanwhile, the structural and lexical constraints noticeably limit the diversity and flexibility of everyday language to ensure that robots can unambiguously interpret the resulting propositions (i.e., the specified tasks can be precisely interpreted and executed by robots).

Atomic propositions can be considered building blocks of all propositions. As will be explained in Section 6, we establish a hierarchy of propositions. At the lowest level of the hierarchy, each atomic proposition is directly associated with an indecomposable, self-contained executable code, and atomic propositions at each level compose propositions at higher levels.

There are several ways to deal with the undesirable or nonsensical atomic propositions that can be formed in $S \times V \times O \times A$. (Note that in formal logics, there can be well-formed formulas that are self-contradictory.) We can remove all such propositions from the cartesian product to ensure that each resulting atomic proposition is a precisiated proposition. (In this case, we abuse the notation and let $S \times V \times O \times A$ denote the "cleaned" cartesian product.) Also, we can consider them as always false so that they will never be executed in practice (see Section 7).

In our formal logic, the atomic propositions need not be expressed as generalized constraints on variables. By incorporating the SV, SVO, SVA, and SVOA structures in the syntax, we can precisiate action-related propositions rather naturally and effectively. Clearly, other syntactic structures can be incorporated in our formal logic; see Section 4.

3.3 Compound Propositions

In our formal logic, we generate each compound proposition by combining multiple atomic propositions using one or more connectives in the component set C. With the component sets in Table 1, we can form the following compound propositions:

$$\bullet\ \frac{\frac{robot2}{S}\ \frac{go}{V}\ \frac{to\ room1}{A}}{\text{atomic proposition}}\ \frac{if}{C}\ \frac{\frac{user}{S}\ \frac{press}{V}\ \frac{button}{O}}{\text{atomic proposition}}.$$

$$\bullet\ \frac{\frac{robot1}{S}\ \frac{go}{V}\ \frac{to\ room1}{A}}{\text{atomic proposition}}\ \frac{if}{C}\ \left(\frac{\frac{user}{S}\ \frac{call}{V}\ \frac{robot1}{O}}{\text{atomic proposition}}\ \frac{or}{C}\ \frac{\frac{user}{S}\ \frac{press}{V}\ \frac{button}{O}}{\text{atomic proposition}}\right). \quad (1)$$

In formal logic, parentheses are used to indicate the scope of each connective. In our examples, parentheses disambiguate the manner in which atomic tasks are performed.

These compound propositions are still quite easy for humans to specify and understand, and the syntactic structures imposed on the clauses and the compositions ensure effective interpretation and execution by robots. Note that these compound propositions are precisiated propositions although they are not expressed as generalized constraints on variables. In fact, it can be quite difficult or ineffective to translate them into generalized constraints.

In (1), the clause "*user call robot1*" or "*user press button*" represents a condition that must be checked in determining whether to send the robot to room1, and the clause "*robot1 go to room1*" is an imperative. There are many task descriptions that can be effectively expressed as compound propositions consisting of conditions and imperatives. In our scheme, task descriptions consist of atomic propositions described in Section 3.2, and each atomic proposition is either a condition or an imperative. The type of each atomic proposition in a compound task description is unambiguously determined by the logical connective that connects it and by the location of the atomic proposition relative to the connective. An atomic proposition that forms a subordinate clause immediately following the connective "if" is considered a condition, whereas an atomic proposition that forms a clause immediately preceding the connective is considered an imperative. If the proposition is a condition, the robotic system monitors the described condition. If it is an imperative, the system executes the described action provided that all the required conditions are satisfied.

3.4 Recursive Definition of Well-Formed Formulas

In order for a precisiation language to have high expressive power observed in natural language, it should be able to generate infinitely many precisiated propositions while ensuring that every proposition in it is precisiated. As described in Section 2, we can attain these properties in formal logic by recursively defining its syntax; formally, our formal logic can generate infinitely many well-formed formulas while ensuring that every formula in it is well-formed.

The syntax of the formal logic described in Sections 3.1–3.3 can be recursively defined as follows:

1. Any $x \in S \times V \times O \times A$ is an atomic well-formed formula.

2. If α and β are well-formed formulas, then $\alpha\ c\ \beta$, where $c \in C$, is also a well-formed formula.
3. Nothing else is a well-formed formula.

This recursive definition allows our formal logic to generate infinitely many precisiated propositions while ensuring that every proposition in it is precisiated. As regards the examples described in Sections 3.1–3.3, with sufficiently rich component sets, we can describe any task that must be performed by the robotic system, and, using the syntax, we can ensure that the robotic system does not operate on any ill-formed task descriptions (i.e., task descriptions that are not interpretable).

4 GENERALITY OF THE FORMAL LOGIC

In Sections 3–3.4, we facilitated the exposition of our formal logic by explaining it with rather simple component sets and syntactic structures. Obviously, we can easily extend our scheme to more sophisticated component sets and syntactic structures so that our formal logic can deal with highly complex actions and perceptions. Each component set can be made as large as necessary, and other component sets or clause structures can be incorporated in the formal logic. For instance, in addition to the SV, SVO, SVA, and SVOA structures described and used in Sections 3–3.4 (and in Section 6), we can also incorporate other commonly observed clause structures (see, for instance, (Biber et al., 2002)), such as the SVC, SVOC, and SVOO structures, in the syntax of atomic propositions. Furthermore, we can extend the clause structures so that a phrase can be used as the subject or the object in an atomic proposition. Negation, a unary logical connective, can certainly be incorporated in the formal logic. We can also include Zadeh's generalized constraints, which are suitable for expressing perceptual propositions, in our formal logic; each generalized constraint can be considered an atomic proposition that has the SVC structure, and it can be combined with other propositions by connectives to form a compound proposition.

We can establish not only a propositional logic but also a quantificational logic, which fully incorporates quantifiers and predicates in well-formed formulas. Since propositions that describe perceptions often include quantifiers (see, for instance, (Zadeh, 2002), (Zadeh, 2004)), it is desirable to develop a quantificational logic as a precisiation language that covers both actions and perceptions.

5 INFERENCE AND REASONING IN THE FORMAL LOGIC

As in other formal logics, we can infer and reason in our formal logic by adding a deductive apparatus to it. Syntactically, a set of inference rules can be constructed, and axioms can also be established. (The hierarchy described in Section 6 represents non-logical, domain-dependent axioms.) Typical induction- and elimination-rules in formal logics, such as modus ponens and modus tollens, can be easily incorporated in our formal logic. Consequently, we can form a sequent, which consists of a finite (possibly empty) set of well-formed formulas (the premises) and a single well-formed formula (the conclusion), and we can examine its provability (derivability) using proof theory; we can determine if a conclusion follows logically from a set of premises by examining whether there is a proof of that conclusion from just those premises in the formal logic.

As will be described in Section 7, we can employ fuzzy relations to establish the semantics of our formal logic. This semantics allows us to investigate the truth conditions and the semantic validity of each proposition or sequent. As in other formal logics, comparative truth tables can be used to determine semantic validity.

6 HIERARCHY OF PROPOSITIONS

The importance of the hierarchy of propositions described in this section is threefold. First, it enhances the expressive power of the formal logic by building up its vocabulary while ensuring the precisiability of each resulting proposition. Second, it fortifies the high interactivity of our formal logic by allowing human-robot communications to take place at various levels of detail. Third, it strengthens the deductive apparatus of the formal logic by establishing domain-dependent axioms that can be used for inference and reasoning.

We will first explain the hierarchy intuitively using the task description scheme described in Section 3. A complex task can be described rather concisely, i.e., it can be expressed by an atomic proposition. In many cases, naive users will prefer specifying a complex task using an atomic proposition as opposed to a more lengthy compound proposition. On the other hand, in order for a robotic system to actually execute a complex task, the task must be broken down into simpler subtasks, and the manner in which the subtasks

are performed must be specified. Consequently, the atomic proposition describing a complex task can be reexpressed as a compound proposition that consists of atomic propositions describing the required subtasks. Some of the subtasks may have to be further decomposed in order to fully specify how to execute them. Expert users may want to establish and combine these subtasks carefully so that the robotic system can perform the task effectively and efficiently.

Thus, we can precisiate an atomic proposition by reexpressing it as a compound proposition consisting of atomic propositions that precisiate it. This process also leads to flexibility in the level of detail. In the task description example, the flexibility gives naive users an efficient, user-friendly interface with robots while giving expert users the power to customize tasks. As a result, the hierarchy allows human-robot interactions to take place at various levels of detail, and it also helps determine the appropriate level of detail for each human-robot interaction; the hierarchy determines whether, from the point of view of a given user, a given task is "atomic" or "compound."

Consider the following task description:

$$\frac{\frac{robot1}{S}\ \frac{examine}{V}\ \frac{patient1}{O}\ \frac{in\ room1}{A}}{}. \tag{2}$$

This atomic proposition can be reexpressed as a compound proposition that consists of three atomic propositions representing subtasks that must be performed to accomplish the task:

$$\frac{\frac{robot1}{S}\ \frac{find}{V}\ \frac{patient1}{O}\ \frac{in\ room1}{A}}{\text{atomic proposition 1}}$$

$$\frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{check}{V}\ \frac{patient1}{O}}{\text{atomic proposition 2}}$$

$$\frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{send}{V}\ \frac{data}{O}}{\text{atomic proposition 3}}. \tag{3}$$

Atomic propositions 1 and 2 in (3) can also be reexpressed as compound propositions that clarify how they are performed; atomic proposition 1 in (3) can be defined as

$$\frac{\frac{robot1}{S}\ \frac{go}{V}\ \frac{to\ room1}{A}}{\text{atomic proposition}}\ \ \frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{search}{V}\ \frac{patient1}{O}}{\text{atomic proposition}}, \tag{4}$$

and atomic proposition 2 in (3) can be defined as

$$\frac{\frac{robot1}{S}\ \frac{go}{V}\ \frac{to\ patient1}{A}}{\text{atomic proposition}}\ \ \frac{then}{C}\ \ \left(\frac{\frac{robot1}{S}\ \frac{measure}{V}\ \frac{heart\ rate}{O}}{\text{atomic proposition}}\ \ \frac{and}{C}\ \ \frac{\frac{robot1}{S}\ \frac{measure}{V}\ \frac{blood\ pressure}{O}}{\text{atomic proposition}}\right). \tag{5}$$

Therefore, using (4)–(5) and atomic proposition 3 in (3), we can reexpress (2) as

$$\frac{\frac{robot1}{S}\ \frac{go}{V}\ \frac{to\ room1}{A}}{\text{atomic proposition}}$$

$$\frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{search}{V}\ \frac{patient1}{O}}{\text{atomic proposition}}$$

$$\frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{go}{V}\ \frac{to\ patient1}{A}}{\text{atomic proposition}}$$

$$\frac{then}{C}\ \ \left(\frac{\frac{robot1}{S}\ \frac{measure}{V}\ \frac{heart\ rate}{O}}{\text{atomic proposition}}\ \ \frac{and}{C}\ \ \frac{\frac{robot1}{S}\ \frac{measure}{V}\ \frac{blood\ pressure}{O}}{\text{atomic proposition}}\right)$$

$$\frac{then}{C}\ \ \frac{\frac{robot1}{S}\ \frac{send}{V}\ \frac{data}{O}}{\text{atomic proposition}}. \tag{6}$$

Figure 1 visualizes this hierarchy, which consists of three levels (levels 0, 1, and 2). For simplicity, each atomic proposition is represented by its verb; for instance, the atomic proposition at the highest level (level 2), "Robot1 *examine* patient1 in room1," is represented by "examine." The task expressed by the atomic proposition at level 2 is described in more detail at the intermediate level (level 1), where the atomic propositions that involve the verbs "find," "check," and "send" describe the subtasks that constitute the task. These subtasks are described in more detail at the lowest level (level 0), where they are expressed by the atomic propositions that involve the verbs "go," "search," "measure," and "send."

The hierarchy clearly shows how atomic proposition (2) at level 2 is precisiated. At level 0, we have atomic propositions that are not decomposable; each of them is directly associated with a self-contained executable code that is run to perform the corresponding task. Thus, atomic propositions at level 0 can be con-

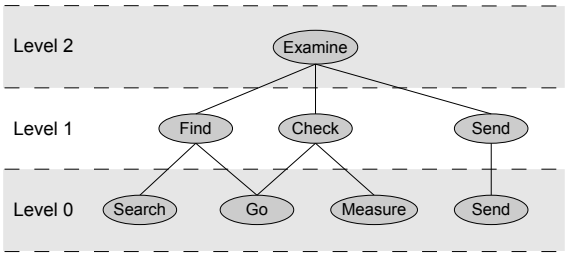

Figure 1: Hierarchical task description. At the highest level (level 2), the task is expressed as atomic proposition (2), represented by "examine." At the intermediate level (level 1), the task is expressed as compound proposition (3), which consists of atomic propositions represented by "find," "check," and "send." At the lowest level (level 0), the task is expressed as compound proposition (6), which consists of atomic propositions represented by "go," "search," "measure," and "send."

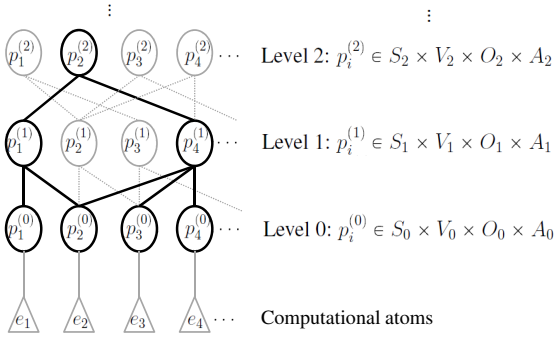

Figure 2: Hierarchy of propositions. At level i, the formal logic described in Sections 3–3.4 is established with component sets S_i, V_i, O_i, A_i, and C_i. Atomic propositions in $S_0 \times V_0 \times O_0 \times A_0$ are the logical atoms and the building blocks of all propositions, and each of them is directly associated with a computational atom. For each $i \geq 1$, every atomic proposition at level i is connected to atomic propositions at level $i-1$ that precisiate it.

sidered logical atoms described in Section 2, and they precisiate each proposition at higher levels.

The suitability of a given proposition depends on the level of granularity required for it. As regards the task description scheme, naive users will most likely prefer describing tasks at level 2, thus preferring (2). For expert users, there may be situations where they prefer specifying a given task step by step or reconfiguring its subtasks according to various circumstances; in such cases, interacting with robots at level 1 using (3) or at level 0 using (6) will be desirable. Thus, the hierarchy allows a variety of users to interact with robots at various levels of detail.

The hierarchy of propositions can be characterized more formally as follows. Let S_i, V_i, O_i, A_i, and C_i denote the component sets for level i of the hierarchy ($i \geq 0$). The elements in these sets reflect the degree of detail suitable for level i. Then at level i, we establish a formal logic with these component sets, as described in Section 3. Atomic propositions in $S_0 \times V_0 \times O_0 \times A_0$ are the logical atoms and the building blocks of all propositions; each of them is indecomposable and directly associated with a self-contained computational unit. We will call such a computational unit as a computational atom. For each $i \geq 1$, every atomic proposition in $S_i \times V_i \times O_i \times A_i$ can be decomposed into atomic propositions in $S_{i-1} \times V_{i-1} \times O_{i-1} \times A_{i-1}$. Figure 2 visualizes a typical form of this hierarchy. For each i and j, $p_j^{(i)}$ denotes an atomic proposition at level i, and e_j denotes a computational atom associated with $p_j^{(0)}$. In the figure, $p_j^{(0)}$ is connected to e_j for each j, and for each $i \geq 1$ and j, $p_j^{(i)}$ is connected to atomic propositions at level $i-1$ that precisiate $p_j^{(i)}$. The figure shows that, for instance, $p_2^{(2)}$ can be expressed as a compound proposition consisting of two atomic propositions ($p_1^{(1)}$ and $p_4^{(1)}$) at level 1 and also as a compound proposition consisting of four atomic propositions ($p_1^{(0)}$, $p_2^{(0)}$, $p_3^{(0)}$, and $p_4^{(0)}$) at level 0.

The hierarchy clearly shows the definition of each proposition by expressing it in terms of precisiated propositions at lower levels. Thus, non-logical, domain-dependent axioms result from the hierarchy, and they can be used for inference and reasoning in the formal logic.

Different levels of granularity may require different component sets, but the same syntactic structure is enforced at all levels. Using the hierarchy, we can ensure that all the resulting propositions remain precisiated at each level, and we can attain flexibility in the level of detail.

7 SEMANTICS OF THE FORMAL LOGIC

The semantics of formal logic specifies how to determine the truth value of each proposition. In two-valued logics, for instance, the truth value is either 1 (true) or 0 (false). As described by Zadeh (e.g., (Zadeh, 2001), (Zadeh, 2004)), this bivalence is not suitable for PNL, so we develop a many-valued semantics for our formal logic. The meaning of the truth value depends on the context. For the task description scheme described in Sections 3–6, for instance, one can evaluate each proposition and let its truth value reflect the feasibility of the corresponding task specification; 1 indicates that the task certainly can be carried out whereas 0 indicates that it certainly cannot be. In this case, it is more realistic and practical to let

the degree of feasibility take on not only the values 0 and 1 but also other values between 0 and 1. In real-world problems, it can be highly practical to evaluate the feasibility of a task description before any serious attempt is made to execute it. If robotic systems interact with a variety of users, including users who have no knowledge of these systems, some users may describe tasks that are virtually impossible to accomplish. It is more desirable to disregard such highly infeasible tasks immediately than to waste resources by attempting to realize them. Also, the user may want to be informed of the degree of feasibility of the task that he specifies before the system attempts to perform it. When multiple options are available for performing a specified task, the user may want to compare their degrees of feasibility before determining which option to take.

To determine the truth value of each proposition in our formal logic systematically and effectively, we use fuzzy relations. A fuzzy relation is a generalization of a classical ("crisp") relation (see, for instance, (Klir and Folger, 1988)). It is a mapping from a Cartesian product to the set of real numbers between 0 and 1. While a classical relation only expresses the presence or absence of some form of association between the elements of factors in a Cartesian product, a fuzzy relation can express various degrees or strengths of association between them. (Hence a classical relation can be considered a "crisp" case of a fuzzy relation.) In our formal logic, each proposition consists of pre-specified components, so a function that assigns a truth value to each proposition can be represented by a fuzzy relation on the Cartesian product of the components. Using some of the operations defined on fuzzy relations (they are described in Appendix), we can systematically and economically determine the truth value of each proposition.

We will explain the semantics of our formal logic using concrete examples of task descriptions for human-robot interaction so that the reader can understand it intuitively. To facilitate our exposition, we consider very simple task descriptions resulting from atomic propositions in $S \times V \times O$. Notice that even in this rather simple case, we need an efficient scheme for determining the truth value of each proposition. For instance, if each of the component sets S, V, and O contains ten elements, then there are 10^3 atomic propositions in $S \times V \times O$, and it may be impractical to determine the truth values of all the atomic propositions individually. Moreover, if the component set C consists of three connectives, then we can generate a total of $3 \cdot 10^6$ compound propositions that consist of two atomic propositions. In practice, it may be necessary to promptly evaluate and compare the truth values of a large number of task descriptions represented by such compound propositions in order to determine which option to execute, so even this simple case requires an efficient, systematic scheme for examining the truth conditions of propositions. (Also note that infinitely many propositions can be generated from these component sets.)

In Section 7.1, we explain how to determine the truth values of atomic propositions. In Section 7.2, we explain how to determine the truth values of compound propositions.

7.1 Truth Conditions of Atomic Propositions

For concreteness, we consider establishing a fuzzy relation on $S \times V \times O$, which is a mapping from the Cartesian product to a totally ordered set called a valuation set. In our formulation of many-valued logic, the valuation set is the unit interval [0, 1]. We consider the following component sets: $S = \{robot1, robot2\}$, $V = \{recognize, hold\}$, $O = \{ball, pen\}$. Table 2 shows the atomic propositions in $S \times V \times O$. Since the component sets considered here are small, we only have eight atomic propositions in the Cartesian product. However, as mentioned earlier, the total number of atomic propositions becomes quite large with large component sets, and we need an efficient, systematic scheme for determining the truth conditions of these propositions. We establish our scheme using three operations on fuzzy relations: projection, cylindric extension, and cylindric closure. These operations are explained in Appendix.

Table 2: Atomic propositions resulting from $S = \{robot1, robot2\}$, $V = \{recognize, hold\}$, and $O = \{ball, pen\}$.

$s \in S$	$v \in V$	$o \in O$
$robot1$	$recognize$	$ball$
$robot1$	$recognize$	pen
$robot1$	$hold$	$ball$
$robot1$	$hold$	pen
$robot2$	$recognize$	$ball$
$robot2$	$recognize$	pen
$robot2$	$hold$	$ball$
$robot2$	$hold$	pen

Suppose that the truth conditions (for concreteness, we assume that they represent degrees of feasibility) of these atomic propositions are determined for a robotic system under the following conditions:

(a) Robot1 is equipped with a high-resolution camera that enables it to recognize various objects, in-

cluding a ball and a pen. However, it does not have any arm, so it cannot hold any object.

(b) Robot2 has an arm that enables it to hold various objects, including a ball and a pen. However, it is not equipped with a high-resolution camera, so it is not fully capable of identifying objects.

(c) With the high-resolution camera, a ball is easier to recognize compared to a pen.

(d) With the arm, a pen is easier to hold compared to a ball.

Our strategy is to derive a fuzzy relation on $S \times V \times O$ from fuzzy relations on $S \times V$ and on $V \times O$. Hence, we first establish fuzzy relations on $S \times V$ and on $V \times O$. Let $R_{S\times V} : S \times V \to [0,1]$ denote a fuzzy relation on $S \times V$. Based on conditions (a) and (b), we set the values of $R_{S\times V}$ as shown in Table 3. Recall

Table 3: Fuzzy relation $R_{S\times V}$ on $S \times V$ based on conditions (a) and (b).

$s \in S$	$v \in V$	$R_{S\times V}(s,v)$
*robot*1	*recognize*	.9
*robot*1	*hold*	0
*robot*2	*recognize*	.2
*robot*2	*hold*	.8

that a fuzzy relation expresses various degrees or strengths of association between elements in component sets. The value assigned to $(robot1, recognize)$ is relatively large (.9) because *robot*1 is equipped with a high-resolution camera and is thus suitable for recognizing objects, whereas the value assigned to $(robot1, hold)$ is zero because *robot*1 is not equipped with an arm and is thus incapable of holding objects. Similarly, the value assigned to $(robot2, recognize)$ is relatively small (.2) because *robot*2 is not equipped with a high-resolution camera and is thus unsuitable for recognizing objects, whereas the value assigned to $(robot2, hold)$ is relatively large (.8) because *robot*2 is equipped with an arm and is thus suitable for holding objects. Technically, the fuzzy relation $R_{S\times V}$ is considered the underlying fuzzy relation on $S \times V \times O$ projected onto $S \times V$ (see Appendix for the operation of projection).

Analogously, based on conditions (c) and (d), we set the values of a fuzzy relation $R_{V\times O} : V \times O \to [0,1]$ as shown in Table 4. The value assigned to $(recognize, ball)$ is larger than that assigned to $(recognize, pen)$ because with a high-resolution camera, a ball is easier to recognize compared to a pen. Similarly, the value assigned to $(hold, pen)$ is larger than that assigned to $(hold, ball)$ because with an arm, a pen is easier to hold compared to a ball. Technically, the fuzzy relation $R_{V\times O}$ is considered the underlying fuzzy relation on $S \times V \times O$ projected onto $V \times O$ (see Appendix for the operation of projection).

Table 4: Fuzzy relation $R_{V\times O}$ on $V \times O$ based on conditions (c) and (d).

$v \in V$	$o \in O$	$R_{V\times O}(v,o)$
recognize	*ball*	.9
recognize	*pen*	.8
hold	*ball*	.7
hold	*pen*	.8

We establish a fuzzy relation $R_{S\times V\times O} : S \times V \times O \to [0,1]$ by combining the fuzzy relations $R_{S\times V}$ and $R_{V\times O}$. Formally, we obtain $R_{S\times V\times O}$ by first obtaining the cylindric extensions of $R_{S\times V}$ and $R_{V\times O}$ to $S \times V \times O$ and then computing the cylindric closure of the two cylindric extensions (see Appendix for cylindric extension and cylindric closure). First, we obtain the cylindric extensions of $R_{S\times V}$ and $R_{V\times O}$ to $S \times V \times O$. Table 5 shows the cylindric extension of $R_{S\times V}$ to $S \times V \times O$, which we denote by $R_{S\times V\uparrow S\times V\times O}$.

Table 5: Cylindric extension $R_{S\times V\uparrow S\times V\times O}$.

$s \in S$	$v \in V$	$o \in O$	$R_{S\times V\uparrow S\times V\times O}(s,v,o)$
*robot*1	*recognize*	*ball*	.9
*robot*1	*recognize*	*pen*	.9
*robot*1	*hold*	*ball*	0
*robot*1	*hold*	*pen*	0
*robot*2	*recognize*	*ball*	.2
*robot*2	*recognize*	*pen*	.2
*robot*2	*hold*	*ball*	.8
*robot*2	*hold*	*pen*	.8

This cylindric extension can be characterized as maximizing nonspecificity in deriving a fuzzy relation on $S \times V \times O$ from a fuzzy relation on $S \times V$ (see Appendix). Similarly, Table 6 shows the cylindric extension of $R_{V\times O}$ to $S \times V \times O$, which we denote by $R_{V\times O\uparrow S\times V\times O}$.

Table 6: Cylindric extension $R_{V\times O\uparrow S\times V\times O}$.

$s \in S$	$v \in V$	$o \in O$	$R_{V\times O\uparrow S\times V\times O}(s,v,o)$
*robot*1	*recognize*	*ball*	.9
*robot*1	*recognize*	*pen*	.8
*robot*1	*hold*	*ball*	.7
*robot*1	*hold*	*pen*	.8
*robot*2	*recognize*	*ball*	.9
*robot*2	*recognize*	*pen*	.8
*robot*2	*hold*	*ball*	.7
*robot*2	*hold*	*pen*	.8

This cylindric extension can be characterized as maximizing nonspecificity in deriving a fuzzy relation on $S \times V \times O$ from a fuzzy relation on $V \times O$ (see Appendix). Finally, we set $R_{S\times V\times O}$ to the cylindric closure of $R_{S\times V\uparrow S\times V\times O}$ and $R_{V\times O\uparrow S\times V\times O}$ on $S \times V \times O$

(see Appendix for the operation of cylindric closure), which is shown in Table 7.

Table 7: Cylindric closure $R_{S\times V\times O}$ of $R_{S\times V\uparrow S\times V\times O}$ and $R_{V\times O\uparrow S\times V\times O}$ on $S\times V\times O$.

$s\in S$	$v\in V$	$o\in O$	$R_{S\times V\times O}(s,v,o)$
*robot*1	*recognize*	*ball*	.9
*robot*1	*recognize*	*pen*	.8
*robot*1	*hold*	*ball*	0
*robot*1	*hold*	*pen*	0
*robot*2	*recognize*	*ball*	.2
*robot*2	*recognize*	*pen*	.2
*robot*2	*hold*	*ball*	.7
*robot*2	*hold*	*pen*	.8

Notice that the resulting truth values (degrees of feasibility) assigned to the eight atomic propositions reflect the conditions (a)–(d). For example, the truth values clearly indicate that robot1 is highly capable of recognizing objects (because it is equipped with a high-resolution camera) but is incapable of holding objects (because it does not have any arm). Similarly, the truth values clearly indicate that robot2 is highly capable of holding objects (because it is equipped with an arm) but is rather incapable of recognizing objects (because it is not equipped with a high-resolution camera).

It is efficient to derive a fuzzy relation on $S\times V\times O$ from fuzzy relations on $S\times V$ and on $V\times O$. Again, suppose that each of these component sets consists of ten elements. Then a total of 10^3 atomic propositions result from them, and it may be time-consuming to determine 10^3 truth values individually. With our scheme, we can derive the 10^3 truth values by determining $2\cdot 10^2$ values of the fuzzy relations $R_{S\times V}$ and $R_{V\times O}$. This efficiency of the scheme becomes more notable as the size of each component set or the number of component sets increases.

Another important strength of our scheme lies in updating the truth values of the atomic propositions. In practice, the values shown in Tables 3–4 will be determined dynamically based on the conditions of the robots. For instance, if the high-resolution camera of robot1 becomes dysfunctional, then we will use the fuzzy relation $R'_{S\times V}$ shown in Table 8 instead of the fuzzy relation $R_{S\times V}$ in Table 3 in computing the truth values of the atomic propositions.

Notice that the value of $R'(robot1, recognize)$ is .2, reflecting the fact that robot1 can no longer use its high-resolution camera to recognize objects. It is easy to verify that Table 9 shows the cylindric closure $R'_{S\times V\times O}$ of the cylindric extensions $R'_{S\times V\uparrow S\times V\times O}$ and $R_{V\times O\uparrow S\times V\times O}$.

Comparing Tables 7 and 9, we can see that the updated truth values (shown in Table 9) reflect the condition that the high-resolution camera of robot1 has become dysfunctional. Notice that we have efficiently updated the fuzzy relation on $S\times V\times O$ by just updating the fuzzy relation on $S\times V$. With our scheme, it is possible to keep the truth values of a large number of atomic propositions updated continually.

Table 8: Fuzzy relation $R'_{S\times V}$ on $S\times V$ (reflecting a damage to robot1's high-resolution camera).

$s\in S$	$v\in V$	$R'_{S\times V}(s,v)$
*robot*1	*recognize*	.2
*robot*1	*hold*	0
*robot*2	*recognize*	.2
*robot*2	*hold*	.8

Table 9: Cylindric closure $R'_{S\times V\times O}$ of $R'_{S\times V\uparrow S\times V\times O}$ and $R_{V\times O\uparrow S\times V\times O}$ on $S\times V\times O$ (reflecting a damage to robot1's high-resolution camera).

$s\in S$	$v\in V$	$o\in O$	$R'_{S\times V\times O}(s,v,o)$
*robot*1	*recognize*	*ball*	.2
*robot*1	*recognize*	*pen*	.2
*robot*1	*hold*	*ball*	0
*robot*1	*hold*	*pen*	0
*robot*2	*recognize*	*ball*	.2
*robot*2	*recognize*	*pen*	.2
*robot*2	*hold*	*ball*	.7
*robot*2	*hold*	*pen*	.8

7.2 Truth Conditions of Compound Propositions

Since we establish our formal logic as a many-valued logic, we treat the connectives in C as logic primitives of many-valued logic or fuzzy logic. Here we examine thee typical logical primitives: conjunction (represented by "and" in C), disjunction (represented by "or" in C), and implication (also called conditional, represented by "if" in C).

In evaluating the truth value of a compound proposition, conjunction is often implemented as a t-norm, whereas disjunction is often implemented as a t-conorm (e.g., (Klir and Folger, 1988), (Hájek, 1998)). Various forms of t-norm and t-conorm have been proposed. Some of the frequently used t-norms are the minimum t-norm, the product t-norm, and the Łukasiewicz t-norm, and some of the frequently used t-conorms are the maximum t-conorm, the probabilistic sum, and the Łukasiewicz t-conorm. In practice, the suitability of each of these t-norms or t-conorms depends on what the truth value represents. Also, there are various ways to implement implication in evaluating the truth value of a compound proposition

(e.g., (Trillas and Alsina, 2012)). Some of the main forms of implication are the material implication, the conjunctive conditional, the residuated conditional Sasaki hook, the Dishkant hook, and the Mamdani-Larsen conditional. Again, in practice, the suitability of each conditional depends on what the truth value represents.

8 CONCLUSIONS

We have taken a first step toward establishing a formal logic as a generalized precisiation language, which is essential for generalizing PNL. Various syntactic structures in natural language can be incorporated in our formal logic so that it precisiates not only perceptual propositions but also action-related propositions. The syntax of the formal logic allows us to create infinitely many precisiated propositions while ensuring that every proposition in it is precisiated. As in other formal logics, we can infer and reason in our formal logic. The resulting generalized precisiation language serves as a middle ground between the natural-language-based mode of human communication and the low-level mode of machine communication and thus significantly facilitates human-machine interaction.

ACKNOWLEDGEMENTS

This research is supported by the Spanish Ministry of Economy and Competitiveness through the project TIN2011-29824-C02-02 (ABSYNTHE).

REFERENCES

Biber, D., Conrad, S., and Leech, G. (2002). *A student grammar of spoken and written English.* Pearson ESL.

Casper, J. and Murphy, R. R. (2003). Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 33:367–385.

Dias, M. B., Harris, T. K., Browning, B., Jones, E. G., Argall, B., Veloso, M. M., Stentz, A., and Rudnicky, A. I. (2006). Dynamically formed human-robot teams performing coordinated tasks. In *AAAI Spring Symposium: To Boldly Go Where No Human-Robot Team Has Gone Before*, pages 30–38.

Dias, M. B., Kannan, B., Browning, B., Jones, E. G., Argall, B., Dias, M. F., Zinck, M., Veloso, M. M., and Stentz, A. J. (2008a). Evaluation of human-robot interaction awareness in search and rescue. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2327–2332.

Dias, M. B., Kannan, B., Browning, B., Jones, E. G., Argall, B., Dias, M. F., Zinck, M., Veloso, M. M., and Stentz, A. J. (2008b). Sliding autonomy for peer-to-peer human-robot teams. In *Proceedings of the 10th International Conference on Intelligent Autonomous Systems*.

Feil-Seifer, D. and Mataric, M. J. (2008). Defining socially assistive robotics. In *Proceedings of the International Conference on Rehabilitation Robotics*, pages 465–468.

Ferketic, J., Goldblatt, L., Hodgson, E., Murray, S., Wichowski, R., Bradley, A., Chun, W., Evans, J., Fong, T., Goodrich, M., Steinfeld, A., and Stiles, R. (2006). Toward human-robot interface standards: Use of standardization and intelligent subsystems for advancing human-robotic competency in space exploration. In *Proceedings of the SAE 36th International Conference on Environmental Systems*.

Fong, T., Nourbakhsh, I., Kunz, C., Flückiger, L., Schreiner, J., Ambrose, R., Burridge, R., Simmons, R., Hiatt, L., and Schultz, A. (2005). The peer-to-peer human-robot interaction project. *Space*, 6750.

Fong, T. and Thorpe, C. (2001). Vehicle teleoperation interfaces,. *Autonomous Robots*, 11:9–18.

Forsberg, M. (2003). Why is speech recognition difficult. *Chalmers University of Technology*.

Gieselmann, P. and Stenneken, P. (2006). How to talk to robots: Evidence from user studies on human-robot communication. *How People Talk to Computers, Robots, and Other Artificial Communication Partners*, page 68.

Goodrich, M. A. and Schultz, A. C. (2007). Human-robot interaction: a survey. *Found. Trends Hum.-Comput. Interact.*, 1:203–275.

Hájek, P. (1998). *Metamathematics of Fuzzy Logic*, volume 4. Kluwer Academic.

Johnson, M., Feltovich, P. J., Bradshaw, J. M., and Bunch, L. (2008). Human-robot coordination through dynamic regulation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2159–2164.

Johnson, M. and Intlekofer, K. (2008). Coordinated operations in mixed teams of humans and robots. In *Proceedings of the IEEE International Conference on Distributed human-Machine Systems*.

Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahsahi, T., Shinjou, A., and Shimada, S. (1999). RoboCup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 739–743.

Klir, G. J. and Folger, T. A. (1988). *Fuzzy sets, uncertainty, and information*. Prentice Hall.

Kulyukin, V., Gharpure, C., Nicholson, J., and Osborne, G. (2006). Robot-assisted wayfinding for the visually impaired in structured indoor environments. *Autonomous Robots*, 21:29–41.

Lacey, G. and Dawson-Howe, K. M. (1998). The application of robotics to a mobility aid for the elderly blind. *Robotics and Autonomous Systems*, 23:245–252.

Marble, J., Bruemmer, D., Few, D., and Dudenhoeffer, D. (2004). Evaluation of supervisory vs. peer-peer interaction with human-robot teams. In *Proceedings of the Hawaii International Conference on System Sciences*.

Muñoz, E., Nakama, T., and Ruspini, E. (2013). Hierarchical qualitative descriptions of perceptions for robotic environments. *To appear in the Proceedings of the Fifth International Conference on Fuzzy Computation Theory and Application (FCTA)*.

Nakama, T., Muñoz, E., and Ruspini, E. (2013). Generalizing precisiated natural language: A formal logic as a precisiation language. *To appear in the Proceedings of the Eighth Conference of European Society for Fuzzy Logic and Technology (EUSFLAT)*.

Norbakhsh, I. R., Sycara, K., Koes, M., Yong, M., Lewis, M., and Burion, S. (2005). Human-robot teaming for search and rescue. *Pervasive Computing*, pages 72–79.

Russell, B. (1984). Lectures on the philosophy of logical atomism. In Marsh, R. C., editor, *Logic and Knowledge Essays 1901–1950*. George Allen & Unwin, London.

Shim, I., Yoon, J., and Yoh, M. (2004). A human robot interactive system "RoJi". *International Journal of Control, Automation, and Systems*, 2:398–405.

Shneiderman, B. (2000). The limits of speech recognition. *Communications of the ACM*, 43(9):63–65.

Tomassi, P. (1999). *Logic*. Routledge.

Trillas, E. and Alsina, C. (2012). From Leibnizs shinning theorem to the synthesis of rules through Mamdani-Larsen conditionals. In *Combining Experimentation and Theory*, pages 247–258. Springer.

Wilcox, B. and Nguyen, T. (1998). Sojourner on mars and lessons learned for future planetary rovers. In *Proceedings of the SAE International Conference on Environmental Systems*.

Winograd, T. and Flores, F. (1986). *Understanding computers and cognition: A new foundation for design*. Ablex Pub.

Zadeh, L. A. (2001). A new direction in ai: Toward a computational theory of perceptions. *AI magazine*, 22(1):73.

Zadeh, L. A. (2002). Some reflections on information granulation and its centrality in granular computing, computing with words, the computational theory of perceptions and precisiated natural language. *Studies in Fuzziness And Soft Computing*, 95:3–22.

Zadeh, L. A. (2004). Precisiated natural language (PNL). *AI Magazine*, 25(3):74–92.

APPENDIX

We describe three operations on fuzzy relations that are used in determining the truth conditions of atomic propositions in our formal logic: projection, cylindric extension, and cylindric closure. First, we establish notation. Let $X_1, X_2, \ldots, X_n$ be sets, and let $X_1 \times X_2 \times \cdots \times X_n$ denote their Cartesian product. We will also denote the Cartesian product by $\times_{i \in \mathbb{N}_n} X_i$, where $\mathbb{N}_n$ denotes the set of integers 1 through n. A fuzzy relation on $\times_{i \in \mathbb{N}_n} X_i$ is a function from the Cartesian product to a totally ordered set, which is called a valuation set. In our formulation, the unit interval [0, 1] is used as a valuation set. Each n-tuple $(x_1, x_2, \ldots, x_n)$ in $X_1 \times X_2 \times \cdots \times X_n$ (thus $x_i \in X_i$ for each $i \in \mathbb{N}_n$) will also be denoted by $(x_i \mid i \in \mathbb{N}_n)$. Let $I \subset \mathbb{N}_n$. A tuple $y := (y_i \mid i \in I)$ in $Y := \times_{i \in I} X_i$ is said to be a sub-tuple of $x := (x_i \mid i \in \mathbb{N}_n)$ in $\times_{i \in \mathbb{N}_n} X_i$ if $y_i = x_i$ for each $i \in I$, and we write $y \prec x$ to indicate that y is a sub-tuple of x.

Let $X := \times_{i \in \mathbb{N}_n} X_i$ and $Y := \times_{i \in I} X_i$ for some $I \subset \mathbb{N}_n$. Suppose that $R : X \to [0,1]$ is a fuzzy relation on X. Then a fuzzy relation $R' : Y \to [0,1]$ is called the projection of R on Y if for each $y \in Y$, we have

$$R'(y) = \max_{x \in X \,:\, y \prec x} R(x).$$

We let $R_{\downarrow Y}$ denote the projection of R on Y.

We continue with $X := \times_{i \in \mathbb{N}_n} X_i$ and $Y := \times_{i \in I} X_i$ ($I \subset \mathbb{N}_n$). Let $F : Y \to [0,1]$ be a fuzzy relation on Y. A fuzzy relation $F' : X \to [0,1]$ is said to be the cylindric extension of F to X if for all $x \in X$, we have

$$F'(x) = F(y),$$

where y is the tuple in Y such that $y \prec x$. We let $F_{\uparrow X}$ denote the cylindric extension of F to X. The cylindric extension $F_{\uparrow X}$ of a fuzzy relation $F : Y \to [0,1]$ is the "largest" fuzzy relation on X such that its projection on Y equals F; if we let $\mathscr{R}$ denote the set of all fuzzy relations $R' : X \to [0,1]$ such that $R'_{\downarrow Y} = F$, then for all $x \in X$, we have

$$F_{\uparrow X}(x) = \max\{R'(x) \mid R' \in \mathscr{R}\}.$$

For each j, let $Y_j := \times_{i \in I_j} X_i$, where $I_j \subset \mathbb{N}_n$. Let $R^{(j)} : Y_j \to [0,1]$ denote a fuzzy relation on Y_j. Then a fuzzy relation $F : X \to [0,1]$ is called the cylindric closure of $R^{(1)}, R^{(2)}, \ldots, R^{(m)}$ on X if for each $x \in X$,

$$F(x) = \min_{1 \le j \le m} R^{(j)}_{\uparrow X}(x).$$