COMPUTATIONAL SITUATION THEORY

Erkan Tin and Varol Akman

BILKENT UNIVERSITY

Department of Computer Engineering
and
Information Science

Technical Report BU-CEIS-93-15

Computational Situation Theory

Erkan Tın and Varol Akman**

Abstract

Situation theory has been developed over the last decade and various versions of the theory have been applied to a number of linguistic issues. However, not much work has been done in regard to its computational aspects. In this paper, a medium (called BABY-SIT) which uses situation-theoretic constructs as its computational foundation is proposed. The features of this experimental environment are compared to those of the existing approaches towards 'computational situation theory.'

Keywords situation theory, situation semantics, PROSIT, ASTL, situation schemata

Contents

1 Introduction

Situation theory is an attempt to develop a mathematical theory of meaning which will clarify and resolve some tough problems in the study of language, information, logic, philosophy, and the mind [10]. It was first formulated in detail by Jon Barwise and John Perry in 1983 [11] and has matured over the last decade [24]. Various versions of the theory have been applied to a number of linguistic issues, resulting in what is commonly known as *situation semantics* [7, 8, 9, 23, 29, 31, 33, 48]. The latter aims at the construction of a unified and mathematically rigorous theory of meaning, and the application of such a theory to natural languages.

*Department of Computer Engineering and Information Science, Faculty of Engineering, Bilkent, 06533 Ankara, Turkey. E-mail: {tin, akman}@bilkent.edu.tr or {tin, akman}@trbilun.bitnet. Fax: +90–4–2664126.

Mathematical and logical issues that arise within situation theory and situation semantics have been explored in numerous works [8, 9, 11, 23, 24, 31]. In the past, the development of a *mathematical* situation theory has been held back by a lack of availability of appropriate technical tools. But by now, the theory has assembled its mathematical foundations based on intuitions basically coming from set theory and logic [1, 8, 23, 25]. With a remarkably original view of information (which is fully adapted by situation theory) [27, 28], a 'logic,' based not on truth but on information, is being developed [24]. This logic[1] will probably be an extension of first-order logic [5] rather than being an alternative to it.

Individuals, properties, relations, spatio-tempo-ral locations, and situations are basic constructs of situation theory. The world is viewed as a collection of objects, sets of objects, properties, and relations. *Infons* ('unit' facts) [25] are discrete items of information and *situations* are first-class objects which describe parts of the real world. Information flow is made possible by a network of abstract 'links' between high-order uniformities, viz. *situation types*. One of the distinguishing characteristics of situation theory vis-à-vis another influential semantic and logical tradition [26] is that information content is *context-dependent* (where a *context* is a situation).

All these features may be cast in a rich formalism for a computational framework based on situation theory. Yet, there have been few attempts to investigate this [16, 40, 43]. Questions of what it means to do computation with situations and what aspects of the theory makes this suitable as a novel programming paradigm have not been fully answered in the literature. This is what we hope to achieve here. Accordingly, in this paper, a computational approach to situation theory and its associated environment (called BABY-SIT) are proposed.[2] The proposed approach especially adopts the ontological features which were originally put forward in [11]. Existing approaches towards a computational account of situation theory unfortunately incorporated only some of these [14, 15, 16, 39, 40, 43]; the remaining features were omitted for the sake of achieving particular goals. This has caused conceptual and philosophical divergence from the ontology of the original theory—a dangerous and unwanted side effect. Our work will try to avoid this pitfall by simply sticking to the essentials of the theory.

The remaining parts of this paper are structured as follows. Situation theory and situation semantics are reviewed in Section 2. In Section 3, computational aspects of the theory is discussed. An approach to compute *via* situations is proposed and existing approaches are examined. Section 4 describes the proposed computational environment (BABY-SIT). Methods of computation and how computation proceeds within this medium are explained in this section. Section 5 emphasizes the role of situation theory in natural language semantics. Section 6 summarizes the current state of BABY-SIT and the remaining work.

2 Situation Theory and Situation Semantics

Situation theory is a mathematical theory of meaning [24]. According to the theory, individuals, properties, relations, spatio-temporal locations, and situations are the basic ingredients. The world is viewed as a collection of objects, sets of objects, properties, and relations.

Individuals are conceived as invariants; having properties and standing in relations, they persist in time and space. *Objects* are the parts of individuals. (Words are also objects, i.e., invariants across utterances.) All individuals, including spatio-temporal locations, have *properties* (like being *fragile* or *red*) and stand in *relations* to one another (like being *earlier*, being *under*).

A sequence such as $\langle r, x_1, \ldots, x_n \rangle$ where r is an n-ary relation over the individuals $x_1, \ldots, x_n$ is called a *constituent sequence*. Suppose Alice was eating ice cream yesterday at home and she is also eating ice cream now at home. Both of these situations share the same constituent $\langle$*eats, Alice, ice cream*$\rangle$. These two events, occurring at the same location but at different times, have the same *situation type* s (cf. [4] for the origin of this idea). Situation types are partial functions from relations and objects to the truth values 0 and 1 (a.k.a. *polarity*). The situation type s, in our example, assigns 1 to the constituent sequence $\langle$*eats,*

[1] According to *The Advanced Learner's Dictionary of Current English* (by A. S. Hornby, E. V. Gatenby, and H. Wakefield, London, U.K.: Oxford University Press, 1958), logic is the science or art of reasoning, proof, and clear thinking. Thus, the commonly accepted equation *logic = first-order logic* is highly suspect. (Cf. [6] for an extended argument on this.)

[2] The environment is dubbed BABY-SIT because we believe that presently it includes far too many provisional, make-shift design decisions. We trust, on the other hand, that as the 'baby' grows up, these haphazard dimensions will be trimmed and a natural, settled, and balanced kernel will endure as a truer representative of a situation-theoretic computational framework.

Alice, ice cream$\rangle$:

In s: eats, Alice, ice cream; 1.

Thus, $\langle$*eats, Alice, ice cream*; 1$\rangle \in s$.

Actually, situation types can be more general. For example, a situation type in which someone is eating something at home 'contains' the situation in which Alice is eating ice cream at home. Suppose Alice is not present in the room where this paper is being written. Then, "Alice is eating ice cream" is not part of our situation s and hence gets no truth value in s. Thus, situation theory allows *partiality* in a strong sense [31].

Situations in which a sequence is assigned both truth values are called *incoherent*. For instance, a situation s' is incoherent if $\langle$*has, Alice, A*$\heartsuit$; 0$\rangle \in s'$ and $\langle$*has, Alice, A*$\heartsuit$; 1$\rangle \in s'$. This is a situation in which Alice has the A$\heartsuit$ and she does not have the A$\heartsuit$ in a card game. There cannot be a real situation s' validating this. Nevertheless, the constituent sequence $\langle$*has, Alice, A*$\heartsuit\rangle$ may be assigned these truth values for spatio-temporally distant situation types (say, s' and s'').

Situation types are, however, independent of locations. A location and a situation type mold a *state of affairs* which in fact is a static situation. In order to keep track of change, *courses of events* are used. A course of events is a partial function from locations to situation types and may contain information about events at more than one location. The course of events e that Alice is eating ice cream at location l_1 (say 11:00 a.m., at home) and is sleeping at a temporally succeeding location l_2 (say 12:15 p.m., at home) is represented as follows:

In e, at l_1: eats, Alice, ice cream; 1,
at l_2: sleeps, Alice; 1,
$l_1 < l_2$ and l_1 @ l_2.

Spatio-temporal locations are allowed to stand in relation with each other in different ways: l_1 temporally precedes l_2 ($l_1 < l_2$), l_1 temporally overlaps l_2 ($l_1 \circ l_2$), l_1 spatially overlaps l_2 (l_1 @ l_2), l_1 is temporally included in l_2 ($l_1 \subseteq_t l_2$), l_1 is spatially included in l_2 ($l_1 \subseteq_s l_2$), and l_1 is spatio-temporally included in l_2 ($l_1 \subseteq l_2$).[3]

Permitting only coherent situations gives the advantage of distinguishing between logically equivalent statements. For example, the statements "Bob is angry" and "Bob is angry and Bob is shouting or Bob is not shouting" are logically equivalent in the classical sense [5]. In situation semantics, these two sentences will not have the same interpretation. A course of events e describing the situation in which Bob is only angry will not contain any sequence about Bob's shouting, i.e., e will be 'silent' on Bob's shouting whereas another courses of events e' describing Bob's being angry and his shouting or not shouting will be defined.

Situation semantics uses statements to classify real situations by the claims these statements make. Claims are represented by coherent courses of events. These courses of events classify the real situations which *validate* them; a real situation r validates a course of events e iff if $\langle r, x_1, \ldots, x_n;\ 1\rangle \in e_l$ (or $\langle r, x_1, \ldots, x_n;\ 0\rangle \in e_l$), then in r, the objects $x_1, \ldots, x_n$ stand (or do not stand) in the relation r at l. A course of events e at a location l, e_l, is also called a situation type. For example, assume the existence of a real situation in which Bob is really angry at l. A coherent course of events e making the claim "Bob is angry" at l is validated by this real situation.

According to situation theory, meanings of expressions reside in systematic relations between different types of situations. They can be identified with relations on *discourse situations d*, *(speaker) connections c*, the utterance φ itself, and the described situation e. Some public facts about φ (such as its speaker and time of utterance) are determined by the discourse situations [44]. The ties of the mental states of the speaker and the hearer with the world constitute c [34].

A discourse situation involves the expression uttered, its speaker, the spatio-temporal location of the

[3]Some utterances are about different situation types 'meeting' in one. Consider the utterance "Alice did not eat ice cream because she was ill." The courses of events may be formulated as follows:

In e_2, at l_2: because, e_0, e_1; 1,
where in e_1, at l_1: is, Alice, ill; 1,
in e_0, at l_0: eats, Alice, ice cream; 0,
$l_0 \circ l_1$, $l_0 \subseteq l_2$, and $l_1 \subseteq l_2$.

utterance, and the addressee(s). Each of these defines a linguistic role (the role of the speaker, the role of the addressee, etc.) and we have a *discourse event*. For example, if the indeterminates a, b, α, and l denote the speaker, the addressee, the utterance, and the location of the utterance, respectively, then a discourse event D is given as:

$$\begin{aligned} D := \text{at } l:\ & \text{speaking}, a; 1, \\ & \text{addressing}, a, b; 1, \\ & \text{saying}, a, \alpha; 1. \end{aligned}$$

Using a name or a pronoun, the speaker refers to an individual.[4] A situation s in which the referring role is uniquely filled is called a *referring* (*anchoring*) *situation*. If in s the speaker uses a noun phrase ν to refer to a unique individual, this individual is called the *referent* of ν.[5]

Tense markers of tensed verb phrases can also refer to individuals, e.g., spatio-temporal locations. Therefore, an anchoring situation s can be seen as a partial function from the referring words ν_i to their referents $s(\nu_i)$. This function is the speaker's connections for a particular utterance [44].

The utterance of an expression φ 'constrains' the world in a certain way, depending on how the *roles* for discourse situations, connections, and described situation are occupied. For example, "I am crying" describes a three-place relation $[\![\text{I am crying}]\!]$ on the utterance situation (the discourse situation and the connections) u and the described situation e. This relation defines a meaning relation written in the following form:

$$d, c[\![\text{I am crying}]\!]e.$$

Given a discourse situation d, connections c, and a course of events e, this relation holds just in case there is a location l_d and a speaker a_d such that a_d is speaking at l_d, and in e, a_d is crying at l_d.

In interpreting the utterance of an expression φ in a context u, there is a flow of information, partly from the linguistic form encoded in φ and partly from contextual factors provided by the utterance situation u. These combine to form a set of constraints on the described situation e. e is not uniquely determined; given u and an utterance of φ in u, there will be several situations e that satisfy the constraints imposed. The meaning of an utterance of φ and hence its interpretation are influenced by other factors such as stress, modality, and intonation [31]. However, the situation in which φ is uttered and the situation e described by this utterance seem to play the most influential roles. For this reason, the meaning of an utterance is essentially taken to be a relation defined over φ, u (d, c), and e. This approach towards identifying linguistic meaning is essentially what Barwise and Perry call the *Relation Theory of Meaning* [11, 12].

The constituent expressions of φ do not describe a situation when uttered in isolation. Uttering a verb phrase in isolation, for example, does not describe a situation e. Other parts of the utterance (of which this verb phrase is a part) must *systematically* contribute to the description of e by providing elements such as an individual or a location. For example, the situational elements for the utterance of the tenseless verb phrase 'running' provide a spatio-temporal location for the act of running and the individual who is to run. For the tensed verb phrase 'is running,' an individual must be provided. The situational elements form the *setting* σ for an utterance. The elements provided by σ can be any individual, including spatio-temporal locations. The meaning of φ is a relation defined not only over d, c, and e, but also over σ.

3 Situations: A Computational Perspective

Intelligent agents generally make their way in the world by being able to pick up certain information from a situation, process it, and react accordingly [24, 27, 28, 36]. Being in a (mental) situation, such an agent has information about situations it sees, believes in, hears about, etc. Alice, for example, upon hearing an utterance of "A bear is running towards you," would have the information, by relying on the utterance situation, that her friend is the utterer and that he is addressing her by the word "you." Moreover, by relying on the situation the utterance described, she would know that there is a bear around and it is running towards her.

[4] A name directly refers to an individual, independent of whether the individual is imaginary or real. A pronoun can either refer to an individual deictically or else it may be used indirectly by co-referring with a noun phrase.

[5] Obviously, the speaker may not refer to anything at all. In this case, the role of the referent is left empty.

Situations can be of the same type. Among the invariants across situations are not just objects and relations, but also aggregates of such. Having heard the warning above, Alice would realize that she is faced with a type of situation in which there is a bear and it is running. She would form a 'thought' over the running bears—an abstract object which carries the property of both being a bear and running—and on seeing the bear around, would individuate it.

Realization of some type of situation causes the agent to acquire more information about that situation as well as other situation types, and to act accordingly. Alice, upon seeing the bear around, would run away, being in possession of the previously acquired information that bears might be hazardous. She can obtain this information from the situation by means of some constraint—a certain relationship between bears and their fame as life-threatening creatures. Attunement to, or awareness of, that constraint is what enables her to acquire and use that information.[6]

An important phenomenon in situation theory is that of *structured* (*nested*) information [28]. Assuming the possession of prior information and/or awareness of other constraints, the acquisition by an agent of an item of information can also provide the agent with an additional item of information. On seeing a square, for example, one gains the information that the figure is a rectangle, and that it is a parallelogram, and that its internal angles are 90 degrees, and so on.

Reaping information from a situation is not the only way an agent processes information. It can also act in accordance of the obtained information to change the environment. Creating new situations to arrive at new information and conveying information it already had to other agents are the primary functions of its activities. Having the information that there is a bear around, Alice would run away, being attuned to the constraint that the best way to avoid danger in such situations is to keep away from the bear. Or, having realized that she cannot move, she would yell for help, being aware of the constraint that calling people in such situations might work.

In short, an intelligent agent has the ability to acquire information about situations, obtain new information about them by being attuned to assorted constraints, and act accordingly to alter its environment. All these are ways of processing information about situations. An information processing environment for such an agent should have the following properties:

- Partitioning of information into situations.
- Parametrization of objects to give a proper treatment of abstraction over individuals, situations, etc.
- Structuring of situations in such a way that they allow nested information.
- Access to information partitioned in this way.
- Access to information in one situation from another situation connected to the former via some relation.
- Constraint satisfaction to control flow of information within and between situations.

These properties would naturally define the underlying mechanisms for a situation-theoretic computational environment. But what constructs are provided by situation theory to build such an environment?

In situation theory, *infons* are the basic units of information [25]. Abstraction can be captured in a primitive level by allowing parameters in infons. Parameters are abstractions or generalizations over classes of non-parametric objects (e.g., individuals, spatial locations). Parameters of a parametric object can be associated with objects which, if they were to replace the parameters, would yield one of the objects in the class that parametric object abstracts over. The parametric objects actually define types of objects in that class. Hence, letting parameters in infons results in what is called *parametric infons*. For example, $\langle see, X, Alice; 1\rangle$ and $\langle see, X, Y; 1\rangle$ are parametric infons where X and Y are parameters over individuals. These infons are said to be parametric on the first and first and second argument roles of the relation *see*,

[6]To rehearse another classical example due to Barwise [24], a tree stump in a forest conveys various types of information to say, a hunter. If he is aware of the relationship between the number of rings in a tree trunk and the age of the tree, the stump will provide him the age of the tree when it was felled. If the hunter is able to recognize various kinds of bark, the stump can provide the information as to what type of tree it was, its probable height, shape, etc. To someone else the same tree stump could yield information about the weather the night before, the kinds of insects that live in the vicinity, and so on.

respectively. Parametric infons can also be allowed to be indetermined with respect to relation and polarity, e.g., $\langle R, X, Y; I\rangle$ where R and I are parameters over relations and polarity, respectively. *Parameter-free infons* are the basic items of information about the world (i.e., 'facts') while parametric infons are the basic units that are utilized in a computational treatment of information flow.

To construct a computational model of situation theory, it is convenient to have available abstract analogs of objects. As noted above, by using parameters we can have abstracts which are parametric objects, including parametric situations, parametric individuals, etc. This yields a rich set of data types. Abstract situations can be viewed as models of real situations. They are set-theoretic entities that have only some of the features of real situations, but are amenable to computation. We define abstract situations as structures consisting of a set of parametric infons. Information can be partitioned into situations by defining a hierarchy between situations. A situation can be larger, having other situations as its subparts. For example, an utterance situation for a sentence consists of the utterance situations for each word forming the sentence. Being in this larger situation gives the ability of having information about its subsituations. The *part-of* relation[7] of situation theory can be used to build such hierarchies among abstract situations and the notion of nested information can be accommodated.

Being in a situation, one can derive information about other situations connected to it in some way. For example, from an utterance situation it is possible to obtain information about the situation it describes. Accessing information both via a hierarchy of situations and explicit relationships among them require a computational mechanism. This mechanism will put information about situation types related in some way into the comfortable reach of the agent and can be made possible by a proper implementation of the *supports relation*, $\models$, of situation theory (cf. the 'extensionality principle' in [24, p. 72].) Given an infon σ and a situation s, this relation holds if σ is made true by s, i.e., $s \models \sigma$.

Constraints enable one situation to provide information about another and serve as links. (They actually link the types of situations.) Constraints can be treated as inference rules. When viewed as a backward-chaining rule, a constraint can provide a channel for information flow between types of situations, from the antecedent to the consequent. This means that such a constraint behaves as a 'definition' for its consequent part [47]. Another way of viewing a constraint is as a forward-chaining rule. This approach enables an agent to alter its environment.[8]

3.1 Approaches to 'Computational Situation Theory'

3.1.1 PROSIT

PROSIT (PROgramming in SItuation Theory) is the pioneering work in this direction. PROSIT is a situation-theoretic programming language developed by Nakashima *et al.* [40]. It has been implemented in Common Lisp [46].

PROSIT is tailored more for knowledge representation in general than for natural language processing. One can define situation structures and assert knowledge in particular situations. It is also possible to define relations between situations in the form of constraints. PROSIT's computational power is due to an ability to draw inferences via rules of inference which are actually constraints of some type. There is an inference engine similar to a Prolog interpreter [47]. PROSIT offers a treatment of partial objects, such as situations and parameters. It can deal with self-referential expressions [9], and their unification.

One can assert facts that a situation will support. For example, if the situation S1 supports the fact that Bob is a young person, this can be defined in the current situation S as:

S: ($\models$ S1 (young "Bob")).

Note that the syntax is similar to that of Lisp and the fact is in the form of a predicate. The supports relation, $\models$, is situated so that whether a situation supports a fact depends on within which situation the query is made. Queries can be posed about one situation from another, but the results will depend on

[7] The *part-of* relation is reflexive, anti-symmetric, and transitive. Hence, it provides a partial-ordering of the situations [24, p. 72].

[8] For instance, being aware of a man ringing the door bell, Alice utters the sentence "A man is at the door." This in turn results in Carol's (another agent's) opening the door. Or it introduces into the discourse a noun phrase for pronominalization in the subsequent discourse, e.g., Carol's question: "Is he the mailman?"

where the query is made.

There is no notion of situation type in PROSIT. For this reason, one cannot represent abstractions over situations and specify relations between them without having to create situations and assert facts to them.

PROSIT has a constraint mechanism. Constraints can be specified using either of the three relations $\Rightarrow$, $\Leftarrow$, and $\Leftrightarrow$. Constraints specified using $\Rightarrow$ (respectively, $\Leftarrow$) are forward (respectively, backward) chaining constraints; the ones using $\Leftrightarrow$ are both backward- and forward-chaining constraints. Backward chaining constraints are of the form ($\Leftarrow$ *head* $fact_1 \ldots fact_n$). If all the facts are supported by the situation, then the head fact is supported by the same situation. Forward chaining constraints are of the form ($\Rightarrow$ *fact* $tail_1 \ldots tail_n$). If *fact* is asserted to the situation, then all the tail facts are asserted to the same situation. Backward chaining constraints are activated at query-time while forward-chaining constraints are activated at assertion-time. By default, all the tail facts of an activated forward-chaining constraint are asserted to the situation, which may in turn activate other forward-chaining constraints recursively.

For a constraint to be applicable to a situation, the situation must be declared to 'respect' the constraint. This is done by using the special relation *respect*. For example, to state that every man is human, one would write:

S: (respect S1 ($\Leftarrow$ (human *X) (man *X))).

This assertion states that S1 respects the stated constraint. This assertion is made with respect to S. (*X denotes a variable.) Since assertions are situated, a situation will or will not respect a constraint depending on where the query is made. If we assert that:

S: ($\models$ S1 (man "Bob")),

then PROSIT will answer yes to the query:

S? ($\models$ S1 (human "Bob")).

The question mark indicates that the expression on its right is a query expression for the situation on its left.

Constraints in PROSIT are about local facts within a situation rather than about situation types. That is, the interpretation of constraints does not allow direct specification of constraints between situations, only between infons within situations. (Situation theory allows constraints between situation types.)

Situated constraints offer an elegant solution to the treatment of *conditional constraints* which only apply in situations that obey some condition. For example, when Alice throws a basketball, she knows it will come down—a constraint to which she is attuned, but which would fail if she tried to play basketball on the moon. This is actually achieved in PROSIT since information is specified in the constraint itself. Situating a constraint means that it may only apply to appropriate situations. This is a good strategy to achieve *background conditions*. However, it might be required that conditions set not only within the same situation, but also between various types of situations. Because constraints have to be situated in PROSIT, not all situations of the appropriate type will have a constraint to apply.

PROSIT does not provide an adequate mechanism for specifying *conventional constraints*, constraints which can be violated. An example of this sort of constraint is the relation between the ringing of the bell and the end of class. It is not necessary that the ringing of the bell should mean the end of class.

Parameters, variables, and constants are used for representing entities in PROSIT. Variables, rather than parameters, are used to identify the indeterminates in a constraint. Parameters might be used to refer to unknown objects in a constraint. Variables have a limited scope; they are local to the constraint in which they appear. Parameters, on the other hand, have a global scope throughout the whole description. Variables match any expression in the language and parameters be can equated to any constant or parameter. That is, the concept of *appropriateness conditions* is not exploited in PROSIT. Appropriateness conditions, in fact, specify restrictions on the types of arguments a relation can take, and any restrictions between these arguments [24, p. 115]. It is more useful to have parameters that range over various classes rather than to work with parameters ranging over all objects. Such particularized parameters are known as *restricted parameters* [24, p. 53].

Some treatment of parameters is given in PROSIT with respect to *anchoring*. Given a parameter of some type (individual, situation, etc.), an anchor is a function which assigns an object of the same type to the parameter [24, pp. 52–63]. Hence, parameters work by placing restrictions on anchors. There is no appropriate anchoring mechanism in PROSIT since parameters are not typed.

In PROSIT, set operations such as $\cup$ and $\cap$ are possible on sets of facts supported by a situation. As mentioned before, situations are closed under constraints and rules of inference. However, all these make PROSIT diverge from the ontology of situation theory.

PROSIT has been used to show how problems involving cooperation of multiple agents can be solved, especially by combining reasoning about situations. In [39], Nakashima *et al.* demonstrate how the *Conway paradox*[9] can be solved. The agents involved in this problem use the *common knowledge* accumulated in a shared situation. This situation functions as a communication channel containing all information known to be commonly accessible. One agent's internal model of the other is represented by situations. Individual knowledge situation together with the shared situation helps an agent to solve the problem.

3.1.2 ASTL

Black's ASTL (A Situation Theoretic Language) is another programming language based on situation theory [16]. ASTL is aimed at natural language processing. One can define in ASTL constraints and rules of inference over the situations. An interpreter, a basic version of which is implemented in Common Lisp [46], passes over ASTL definitions to make inference and to answer queries about the set of constraints and basic situations.

ASTL allows of individuals, relations, situations, parameters, and variables. These definitions form the basic terms of the language. Complex terms are in the form of i-terms (to be defined shortly), situation types, and situations. Situations can contain facts which have those situations as arguments. Sentences in ASTL are constructed from terms in the language and can be constraints, grammar rules, or word entries.

The complex term *i-term* is simply an infon[10] $\langle rel,\ arg_1,\ \ldots,\ arg_n, pol\rangle$ where rel is a relation of arity n, arg_i is a term, and pol is either 0 or 1. A *situation type* is given in the form $[param|cond_1 \ldots cond_n]$ where $cond_i$ has the form *param* $\models$ *i-term*. If situation S1 supports the fact that Bob is a young person, this can be defined as:

S1: [S | S $\models$ $\langle$young, bob, 1$\rangle$].

The single colon indicates that S1 supports the situation type on its right-hand side. The supports relation in ASTL is global rather than situated. Consequently, query answering is independent of the situation in which the query is issued.

Constraints are actually backward-chaining constraints. Each constraint is of the form $sit_0 : type_0 \Leftarrow sit_1 : type_1, \ldots, sit_n : type_n$, where sit_i is a situation or a variable, and $type_i$ is a situation type. If each sit_i, $1 \leq i \leq n$, supports the corresponding situation type, $type_i$, then sit_0 supports $type_0$. For example, the constraint that every man is a human being can be written as follows:

*S: [S | S $\models$ $\langle$human, *X, 1$\rangle$] $\Leftarrow$
    *S: [S | S $\models$ $\langle$man, *X, 1$\rangle$].

*S, *X are variables and S is a parameter. An interesting property of ASTL is that constraints are global, i.e., have a non-explicitly stated scope. Thus, a new situation of the appropriate type need not have a constraint explicitly added to it. For example, assume that S1, supporting the fact that Bob is a man, is asserted:

[9] During a card game both Bob and Alice have an ace. Each of them knows that "Either Bob or Alice has an ace" is a fact. Now suppose Emily were to come along and ask them both whether they knew if the other one had an ace. They would answer "no," of course. And if Emily asked again (and again, ...), they would still answer "no." But now suppose Emily said to them, "Look, at least one of you has an ace. Now do you know whether the other has an ace?" They would again both answer "no." But now something happens. Upon hearing Bob answer "no" Alice would reason as follows: "If Bob does not know I have an ace, having heard that one of us does, then it can only be because he has an ace." Bob would reason in the same way. So they both figure out that the other has an ace. Somehow, Emily's statement must have added some information. But how can that be, since Emily told them something that each of them already knew? This is what is known as the Conway paradox.

[10] We use Black's notation almost verbatim rather than adapting it to the 'standard' notation of our paper.

S1: [S | S $\models$ ⟨man, bob, 1⟩].

This together with the constraint above would give:

S1: [S | S $\models$ ⟨human, bob, 1⟩].

Grammar rules are another form of constraints with similar semantics. An example grammar rule describing the utterance of a sentence consisting of a noun phrase and verb phrase can be defined as:

*S: [S | S $\models$ ⟨*cat*, S, sentence, 1⟩] $\rightarrow$
*NP: [S | S $\models$ ⟨*cat*, S, nounphrase, 1⟩],
*VP: [S | S $\models$ ⟨*cat*, S, verbphrase, 1⟩]

where *cat* denotes the category of the construct, and $\rightarrow$ indicates that this is a grammar rule. This rule can be read: "When there is a situation *NP of the given type and situation *VP of the given type, there is also a situation *S of the given type."

Although one can define constraints between situations in ASTL, the notion of a background condition for constraints is not available. Similar to PROSIT, ASTL cares little about coherence within situations. This is left to the user's control. Accordingly, there is no mechanism in ASTL to specify the constraints that can be violated.

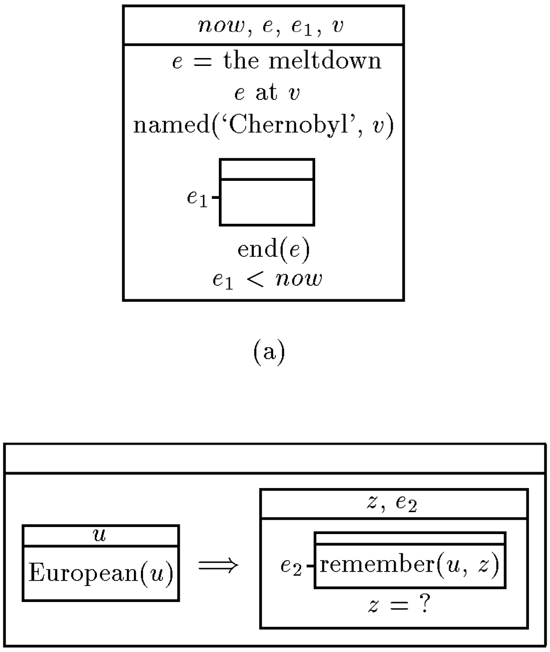
Declaring situations to be of some type allows abstraction over situations to some degree. But, the actual means of abstraction over objects in situation theory, viz. parameters, carry little significance in ASTL.

As in PROSIT, variables in ASTL have scope only within the constraint they appear. They match any expression in the language unless they are declared to be of some specific situation type in the constraint. Hence, it is not possible to declare variables as well as parameters to be of other types such as individuals, relations, etc. Consequently, anchoring on parameters cannot be achieved appropriately in ASTL. Moreover, ASTL does not allow definition of appropriateness conditions for arguments of relations. 'Speaking' relation, for example, might require its speaker role to be filled by a human. Such a restriction could be defined only by using constraints of ASTL. However, this requires writing the restriction each time a new constraint about 'speaking' is to be added. Having appropriateness conditions as a built-in feature would be better.

ASTL does not have a mechanism to relate two situations so that one will support all the facts that the other does. This might be achieved via constraints, but there is no built-in structure between situations (as opposed to the hierarchy of situations in PROSIT).

The primary motivation underlying ASTL is to figure out a framework in which semantic theories such as situation semantics [8] and *Discourse Representation Theory* (DRT) [37] can be described and possibly compared.[11] (Such an attempt can be found in [15].) In DRT, a *discourse representation structure* (DRS) is defined at each stage in a discourse describing the current state of the analysis. A DRS consists of two parts: a set of *domain markers* (*discourse referents*), which can be bound to objects introduced into the current discourse, and a set of conditions on these markers. DRSs are typically drawn as boxes with the referents on the top window and conditions below. Figure 1 shows the DRSs for the sentences "The meltdown at Chernobyl has ended" and "Every European will remember it," respectively [3]. Individual discourse referents are denoted by *now*, u, v, and z while event discourse referents are denoted by the letter e (with or without subscripts). e_1 represents the whole event (ending of the meltdown at Chernobyl) described by the first sentence. Conditions are defined using basic predicates and logical operators. The DRS in Figure 1(b) is true if for every European, he can remember z. z is a discourse referent identified by the anaphoric pronoun 'it' and the rules of DRS construction require that 'it' be matched with some previously introduced discourse referent. However, at the present stage there is no discourse referent with appropriate features. DRS construction can be completed by adding the discourse referents and the conditions introduced for the latter sentence to those declared for the former. Since the DRS for the first sentence contains a discourse referent with appropriate features, the second sentence can now be resolved. The 'unified' result is depicted in Figure 2.

[11] For this reason, ASTL has specific built-in features for natural language processing. It is claimed that these features can be justified from a situation-theoretic view [16, p. 21].



(b)

Figure 1: DRSs to model the discourse segment "The meltdown at Chernobyl has ended. Every European will remember it."

3.1.3 Situation Schemata

Situation schemata has been introduced by Fenstad *et al.* [31] as a theoretical tool for extracting and displaying information relevant for semantic interpretation from linguistic form. It is in fact an attribute-value system which has a choice of primary attributes matching the primitives of situation semantics. In this format, it is just another knowledge representation mechanism. The boundaries of situation schemata are however flexible and depending on the underlying theory of grammar, are susceptible to emendation. Hence, available linguistic insights can be freely exploited.

A simple sentence φ has the situation schemata shown in Figure 3(a). Here r can be anchored to a relation, and a and b to objects; $i \in \{0,1\}$ gives the polarity. LOC is a function which anchors the described fact relative to a discourse situation d, c. LOC will have the general format in Figure 3(b). IND.α is an indeterminate for a location, r denotes one of the basic structural relations on a relation set R, and loc_0 is another location indeterminate. The notation $[\ ]_\alpha$ indicates a repeated reference to the shared attribute value, IND.α. A partial function g anchors the location of SIT.φ, viz. SIT.φ.LOC, in the discourse situation d, c if

$g(loc_0) = loc_d$ and
$c(r)$, $g(\text{IND}.\alpha)$, loc_d; 1

where loc_d is the discourse location and $c(r)$ is the relation on R given by the speaker's connection c. The situation schema corresponding to "Alice saw the cat" is given in Figure 4.

Situation schemata can be adopted to various kinds of semantic interpretation. One could give some kind of operational interpretation in a suitable programming language, exploiting logical insights. But in its present form, situation schemata does not go further than being a complex attribute-value structure. It allows representation of situations within this structure, but does not use situation theory itself as a basis. Situations, locations, individuals, and relations constitute the basic domains of the structure. Constraints are declarative descriptions of the relationships holding between aspects of linguistic form and the semantic representation itself.

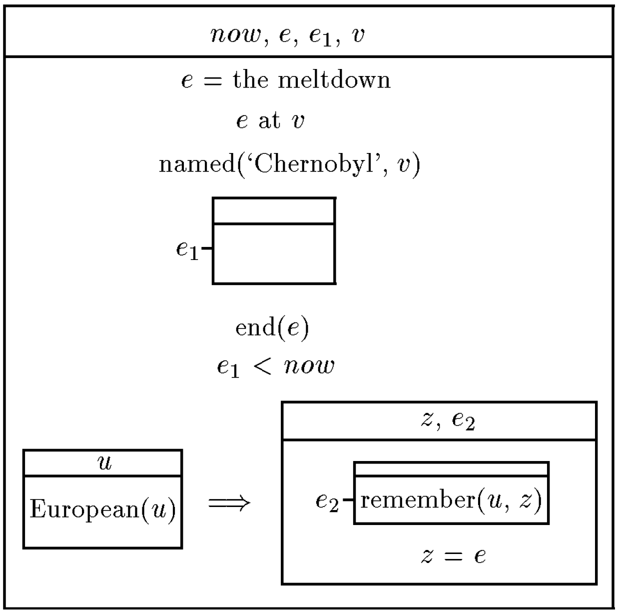


Figure 2: The unified DRS for Figure 1.

Theoretical issues in natural language semantics have been implemented on pilot systems employing situation schemata. The grammar described in [31], for example, has been fully implemented using a lexical-functional grammar system [32] and a fragment including prepositional phrases has been implemented using the DPATR format [19].

4 BABY-SIT: The Proposed Computational Model

The proposed computational model consists of seven primitive domains: *individuals* (D), *times* (T), *places* (L), *labels* (Q), *relations* (R), *polarities* (I), and *situations* (S). The structure of the proposed model, M, is then a tuple $<D, T, L, R, I, S>$. This structure is shared by three components of the system: *environment*, *background situation*, and *anchoring situation*. (These will be explained in Section 4.1.) Each primitive domain carries its own internal structure and is now briefly described:

- Individuals: Unique atomic entities in the model which correspond to real objects in the world.
- Times: Individuals of distinguished type, representing temporal locations.
- Places: Similar to times, places are individuals which represent spatial locations.
- Labels: Constants for representing objects in the model.
- Relations: Various relations hold or fail to hold between objects. A relation has argument roles which must be occupied by appropriate objects.
- Polarities: The 'truth values' 0 and 1.
- Situations: (Abstract) situations are set-theoretic constructs, e.g., a set of *parametric infons*, comprising relations, parameters, and polarities. A parametric infon is the basic computational unit of the form $\langle rel,\ arg_1,\ \ldots,\ arg_n;\ pol\rangle$, where *rel* is a relation, arg_i, $1 \leq i \leq n$, is a parameter of the appropriate type for that argument role, and *pol* is the polarity.[12] By defining a hierarchy between

[12]We will omit polarity when it is 1, i.e., $\langle rel, arg_1, \ldots, arg_n\rangle \equiv \langle rel, arg_1, \ldots, arg_n; 1\rangle$.

$$\text{SIT}.\varphi \begin{bmatrix} \text{REL} & r \\ \text{ARG.1} & a \\ \text{ARG.2} & b \\ \text{LOC} & - \\ \text{POL} & i \end{bmatrix} \qquad \begin{bmatrix} \text{IND} & \text{IND}.\alpha \\ \text{COND} & \begin{bmatrix} \text{REL} & r \\ \text{ARG.1} & [\,]_\alpha \\ \text{ARG.2} & loc_d \\ \text{POL} & i \end{bmatrix} \end{bmatrix}$$

(a) (b)

Figure 3: (a) A prototype situation schema, (b) the general format of LOC in (a).

them, situations can be embedded via the special relation *part-of*. A situation can be either (spatially and/or temporally) *located* or *unlocated*. Time and place for a situation can be declared by *time-of* and *place-of* relations, respectively.

Description of a model, D_M, consists of a definition of the structure M, a set of *constraints*, C, a set of *variables*, V, and a set of *parameters*, P. The definition of relations includes the *appropriateness conditions* for their argument roles. Appropriateness conditions define the domains to which arguments of a relation belong. Each argument can be declared to be from one or more of the seven primitive domains above. Parameters are ‘place holders’ for objects in the model; they are used to refer to arbitrary objects of a given type. They define abstractions or generalizations over classes of non-parametric objects. In addition to parameters, a set of variables is defined. Variables only appear in constraints and scope-wise are local to the constraint in which they appear. Variables as well as parameters must be declared to be from only one of the primitive domains so that variables can be unified and parameters can be anchored to objects of the appropriate type. (Cf. Figure 6 for an illustration of a possible set of declarations.)

The computational model is then defined as a tuple $<D_M, A, A', U>$ where A is an anchor for parameters, A' is an assignment for variables, and U is an interpretation for D_M. A is provided by the anchoring situation while A' is obtained through unification. U will be defined by the operational semantics of the computation.

4.1 Architectural Considerations

The proposed computational model for BABY-SIT is composed of seven major parts: *programmer/user interface*, *environment*, *background situation*, *anchoring situation*, *constraint set*, *inference engine*, and *interpreter* (Figure 5).

The interface allows interaction of the user with the system. One can develop and test his own program, and enter queries about situations.

The environment of the proposed model initially consists of static situation structures and their relationships. These structures can be dynamically changed and new relationships among situation types can be defined as the computation proceeds. Information conveyance among situations is basically made possible by defining a *part-of* relation among them. In this way, a situation s can have information about another situation s' which is part of s.

The background situation contains infons which are inherited by all situation structures in the environment. However, a situation can inherit an infon from the background situation only if it does not cause a contradiction in that situation.[13]

[13]The environment, for example, might be a model for situations in which a particular physical experiment is conducted. Assume that the background situation requires existence of air in all situations. However, there might be some situations which

$$
\text{SIT.1}\begin{bmatrix}
\text{REL} & \text{`see'} \\
\text{ARG.1} & \begin{bmatrix}\text{IND} & \text{`Alice'}\end{bmatrix} \\
\text{ARG.2} & \begin{bmatrix}
\text{IND} & \text{IND.2} \\
\text{SPEC} & \text{`the'} \\
\text{COND} & \begin{bmatrix}\text{REL} & \text{`cat'} \\ \text{ARG.1} & [\,]_2 \\ \text{POL} & 1\end{bmatrix}
\end{bmatrix} \\
\text{LOC} & \begin{bmatrix}
\text{IND} & \text{IND.3} \\
\text{COND} & \begin{bmatrix}\text{REL} & \text{`<'} \\ \text{ARG.1} & [\,]_3 \\ \text{ARG.2} & loc_d \\ \text{POL} & 1\end{bmatrix}
\end{bmatrix} \\
\text{POL} & 1
\end{bmatrix}
$$

Figure 4: Situation schemata for “Alice saw the cat.”

A situation in the environment can only be realized if its parameters are anchored to objects in the real world. This is made possible by the anchoring situation which allows a parameter to be anchored to an object of appropriate type—an individual, a situation, a parameter, etc. But a parameter must be anchored to a unique object, i.e., it is anchored once. On the other hand, more than one parameter may be anchored to the same object. Restrictions on parameters are kept in the anchoring situation as well. These restrictions assure anchoring of one parameter to an object having the same qualifications as the parameter. The anchoring situation has a functionality similar to that of the connections, c, mentioned in Section 2.

In addition to the *part-of* relation among situations, constraints are potent means of information conveyance between and within situations. They link various types of situations. Constraints may be physical laws, linguistic rules, law-like correspondences, conventions, etc. In BABY-SIT, they are realized as forward-chaining constraints or backward-chaining constraints, or both.

Assertion of a new object into BABY-SIT activates the forward-chaining mechanism. Once their antecedent parts are satisfied, consequent parts of forward-chaining constraints are asserted into BABY-SIT unless this yields a contradiction. In case of a contradiction, the backward-chaining mechanism is activated to resolve it.

The interpreter forms the core of execution in BABY-SIT. Anchoring of parameters, evaluation of constraints, etc. are controlled by this part of the system.

are parts of the larger experiment situation, which is carried under vacuum conditions. Explicit assertion of the nonexistence of air in these situations prevents inheritance of the contradictory infon from the background situation.

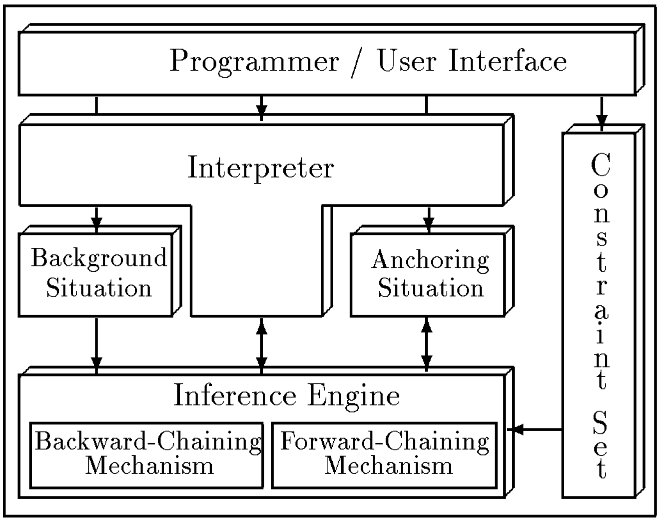


Figure 5: The architecture of the proposed model.

4.2 Methods of Computation

4.2.1 Unification

The basic data manipulation operation is *unification* which results in a substitution. Operationally, this substitution can be thought of as a simultaneous assignment of appropriate data structures to variables. As in logic programming [47], the *single-assignment property*, i.e., the ability to assign one variable to another and a structure containing variables to a variable, is preserved. The rules of unification are as follows:

- A variable is only unifiable with a structure of the same type.
- An individual cannot be unified with anything but itself.
- A label cannot be unified with anything but itself.
- A parameter is not unifiable with anything but itself. However, a parameter can be unified with another parameter of the same type if one of them is anchored to the other in the anchoring situation.
- Two infons can be unified only when they are componentwise unifiable.

Substitution of a parameter of some type with another parameter of the same type can be made possible through explicit assertion of the equality of parameters. A special relation, *equal*, is allowed to be used for this purpose within the model. Equality of two parameters requires the restrictions, in the form of infons, existing in the anchoring situation on both of the parameters to be unifiable.

Existing approaches [16, 40, 43] employ unification as the basic mechanism of computation as well. Although parameters in situation theory require special treatment, these approaches take them as if they were static variables throughout the model.

A situation, s, *supports* an infon if the infon is either explicitly asserted to hold in s, or it is supported by a situation s' which is part of s, or it can be proven to hold by application of backward-chaining constraints. Given an infon σ and a situation s, if s supports σ, then this is denoted by $s \models \sigma$. Otherwise, we say $s \not\models \sigma$.

In proving a 'supports' relation, proof by contradiction cannot be usefully employed. The reason is that for an infon σ and a situation s, it is possible that $s \models \sigma$, or $s \models \neg\sigma$, or neither $s \not\models \sigma$ nor $s \not\models \neg\sigma$. Therefore, refuting the case $s \models \neg\sigma$ does not entail $s \not\models \sigma$.

4.2.2 Constraints as Inference Rules

Barwise and Perry identify three forms of constraints [11]. *Necessary constraints* are those constraints by which one can define or name things, e.g., "Every dog is a mammal." *Nomic constraints* are like patterns that are usually called natural laws, e.g., "Blocks drop if not supported." *Conventional constraints* are those arising out of explicit or implicit conventions that hold within a community of living beings, e.g., "The first day of the month is the pay day." They are neither nomic nor necessary, i.e., they can be violated. All types of constraints can be *conditional* and *unconditional*. Conditional constraints can be applied to situations that meet some condition while unconditional constraints can be applied to all situations.

As stated earlier, constraints can be used as inference rules in a computational system. Some constraints can be defined as forward-chaining constraints, some as backward-chaining constraints, others as both forward- and backward-chaining constraints. In BABY-SIT, conditional constraints come with a set of *background conditions* which must be satisfied for the constraint to be applied. Each background condition is in the form of a $\models$ relation between a situation and an infon.

In BABY-SIT, a constraint becomes a candidate for activation when its background conditions, if any, are satisfied. A candidate forward-chaining constraint is activated whenever its antecedent part is satisfied. All the consequences are asserted if they do not yield a contradiction in the situation into which they are asserted. In this way, it is made possible to handle conventional constraints. New assertions may in turn activate other candidate forward-chaining constraints. If consequences cause contradictions within themselves, backward-chaining constraints are used to decide which one(s) will be successfully asserted. Candidate backward-chaining constraints are activated either when a query is entered explicitly or a query is issued by the forward-chaining mechanism.

In BABY-SIT, the following classes of constraints can be easily modeled [14]:

- Situation constraints: Constraints between situation types.
- Infon constraints: Constraints between infons (of a situation).
- Argument constraints: Constraints on argument roles (of an infon).

4.2.3 Querying

The basic computation upon posing a query is done by the application of backward-chaining constraints. The system's response when a query is entered may depend on its understanding of the intention of the user. There are several possible actions:

- Answering "true" or "false."
- Returning one solution if there is a possible anchor, either partial or full, provided by the anchoring situation for the parameters in the query. Infons anchoring any parameter in the query to an individual are displayed together with the solution.
- Returning all solutions.

5 Situation Semantics as Natural Language Semantics

Language is an integral part of our everyday experience. Some activities pertaining to language include talking, listening, reading, and writing. These activities are *situated*; they occur *in* situations and they are *about* situations [4]. What is common to these situated activities is that they convey information [24, 27, 28, 35]. When uttered at different times by different speakers, a statement can convey different information to a hearer and hence can have different meanings.[14]

[14]Consider the sentence "That really attracts me." Depending on the reference of the demonstrative, interpretation (and hence meaning) would change. For example, this sentence would be uttered by a boy referring to a cone of ice cream or by a cab driver referring to fast driving, meaning absolutely different things [34].

This *information-based* approach to the semantics of natural languages has resulted in what is known as *situation semantics*. Situation semantics makes simple assumptions about the way natural language works. Primary among them is the assumption that language is used to convey information about the world (the so-called *external significance* of language).[15] Even when two sentences have the same interpretation, i.e., they describe the same situation, they can carry different information.[16]

Classical approaches to semantics underestimate the role played by *context-dependence*; they ignore pragmatic factors such as intentions and circumstances of the individuals involved in the communicative process [4, 34, 35]. But, indexicals, demonstratives, tenses, and other linguistic devices rely heavily on context for their interpretation and are fundamental to the way language conveys information [2]. Context-dependence is an essential hypothesis of situation semantics; a given sentence can be used over and over again in different situations to say different things (the so-called *efficiency* of language). Its interpretation, i.e., the class of situations described by the sentence, is therefore subordinate on the situation in which the sentence is used. This context-providing situation, *discourse situation*, is the speech situation, including the speaker, the addressee, the time and place of the utterance, and the expression uttered. Since speakers are always in different situations, having different causal connections to the world and different information, the information conveyed by an utterance will be relative to its speaker and hearer (the so-called *perspectival relativity* of language).

Besides discourse situations, the interpretation of an utterance depends on the speaker's connections with objects, properties, times and places, and on the speaker's ability to exploit information about one situation to obtain information about another. Therefore, context supports not only facts about speakers, addressees, etc. but also facts about the relations of discourse-participants to other contextually relevant situations such as *resource situations*. Resource situations are contextually available and provide entities for reference and quantification. Their use has been demonstrated in the theory of definite descriptions of Barwise and Perry [11].[17]

Another key assumption of situation semantics is the so-called *productivity* of language: we can use and understand expressions never before uttered [18]. Hence, given a finite vocabulary, we can form a potentially infinite list of meaningful expressions. The underlying mechanism for such an ability seems to be *compositionality*.[18]

Situation semantics closes another gap of traditional semantic approaches: the neglect of *subject matter* and *partiality of information*. In traditional semantics, statements which are true in the same models convey the same information [13]. Situation semantics takes the view that logically equivalent sentences need not have the same subject matter, they need not describe situations involving the same object and properties. The notion of partial situations (partial models) leads to a more fine-grained notion of information content and a stronger notion of logical consequence that does not lose track of the subject matter (relevance).

The *ambiguity* of language is taken as another aspect of the efficiency of language. Natural language expressions may have more than one meaning. There are factors such as intonation, gesture, the place of

[15] For example, "Bob smashed his car yesterday" conveys the information that there is an individual named Bob, that he has a car, that he crashed it, that this event occurred in the past, and that he was the driver of the car at the spatio-temporal location of this unfortunate event. Thus, sentences describe situations in the world. These situations and the objects in them have properties and stand in relations to each other at spatio-temporal locations.

[16] For example, "Bob went to the theater" and "The father of Carol went to the theater" both describe the same situation in which Bob (an individual) went to the theater, assuming that Bob is Carol's father. However, while the first sentence says that this individual is Bob, the second sentence conveys the information that Carol (another individual) has a father who went to the theater.

[17] Imagine, for example, that there are two card games going on, one across town from the other: Max is playing cards with Emily and Claire is playing cards with Dana. Suppose Bob watching the former game mistakes Emily for Claire, and utters the sentence "Claire has the three of clubs." According to the classical (Russelian) theories [30], if Claire indeed has 3♣, this claim would be true since the definite noun phrases "Claire" and "the three of clubs" are used to pick out, among all the things in the world, the unique objects satisfying the properties of being an individual named Claire and being a 3♣, respectively; the sentence would be considered to contain no explicit contextually sensitive elements [9]. In contrast, situation semantics identifies these objects with respect to some limited situation—the resource situation exploited by Bob. The claim would then be wrong even if Claire had 3♣ across town. Thus, context is, in general, taken not to be a single situation, but a 'constellation' of related situations.

[18] The assumption that meaning of a larger linguistic unit is a function of the meanings of its individual parts is called the *principle of compositionality* [10]. It can be considered as a reflection of the similar principle in logic [5, 17] to the semantics of natural languages. (In logic, the truth-value of a well-formed formula is determined by the truth-value of its constituent parts.)

an utterance, etc. which play a role in interpreting an utterance [31]. Instead of throwing away ambiguity and contextual elements, situation semantics tries to build up a full theory of linguistic meaning by initially isolating some of the relevant phenomena in a formal way and by exploring how the rest helps in achieving the goal [11].

According to situation semantics, we use meaningful expressions to convey information not only about the external world but also about our minds (the so-called *mental significance* of language).[19] Situation semantics differs from other approaches in that we do not, in attitude reports, describe our mind directly (by referring to states of mind, ideas, senses, thoughts, etc.) but indirectly (by referring to situations that are external).

With these underlying assumptions and features, situation semantics provides a fundamental and appropriate framework for a realistic model-theoretic semantics of natural language [10]. Various versions of this theory have been applied to a number of linguistic issues (mainly) in English [7, 8, 9, 20, 22, 23, 29, 33, 41]. The ideas emerging from research in situation semantics have also been coalesced with well-developed linguistic theories such as *lexical-functional grammar* [45] and led to rigorous formalisms [31]. On the other hand, situation semantics has been compared to other influential mathematical approaches to the theory of meaning, viz. Montague Grammar [21, 26, 42] and DRT [37].

6 Concluding Remarks

Serious thinking about the computational aspects of the situation theory is just starting. There have been only a few proposals [16, 40, 43] which mainly offer a Prolog- or Lisp-like programming environment with varying degrees of divergence from the ontology of situation theory.

In this paper, we proposed a medium (called BABY-SIT) based on situation-theoretic constructs. BABY-SIT accommodates the following basic features of situation theory:

- Objects: The world is viewed as a collection of objects. The basic objects include individuals, times, places, labels, situations, relations, and parameters.
- Situations: Situations are first-class 'citizens' which represent limited portions of the world.
- Partiality: Infons can be made true or false, or may be left unmentioned by some situation.
- Coherence: A situation cannot support an infon and its dual at the same time.
- Circularity: A situation can contain infons which have the former as arguments.
- Constraints: Information flow is made possible via coercions that link various types of objects.

Compared to the existing approaches [16, 40, 43], BABY-SIT enhances the features listed above in the following ways:

- Situations are viewed at an abstract level. This means that situations are sets of parametric infons, but they may be non-well-founded (circularity) [1, 9].
- Parameters are place holders and can be anchored to unique individuals in the anchoring situation. The anchoring situation is required to cohere.
- A situation can be realized if its parameters are anchored, either partially or fully, by the anchoring situation. That is, only anchoring the parameters of an infon contributes a piece of information about the situation.

[19] Consider the sentence "A bear is coming this way" uttered by Bob. It can give us information about two different situations. The first one is the situation which we are located in. The second one is the situation which Bob believes. If we know that Bob is hallucinating, then we might learn the second situation, but not the first. Focusing on the second situation, if we could not see any bear around, we would normally focus on Bob's belief situation.

- Each relation has 'appropriateness conditions' which determine the type of its arguments. The basic computation regime is unification.
- Situations (and hence infons they support) have spatio-temporal dimensions.
- A hierarchy of situations can be defined both statically and dynamically. A situation can have information about another which is part of the former.
- Situations can be grouped to form a whole which provides a computational context. Such a whole has its own set of constraints which can be globally applied to the situations collected under it.
- Partial nature of situations facilitates computation with incomplete information.
- Constraints can be violated. This aspect is built directly into the computational mechanism: a constraint can be applied to a situation only if it does not introduce an incoherence.

BABY-SIT allows the use contextual information which plays a critical role in all forms of behavior and communication. Constraints enable one situation to provide information about another and serve as links between representations and the information they represent. Computation over situations occurs via constraints and is context-sensitive. In the existing approaches [14, 15, 16, 39, 40, 43], the notion of context is either poorly handled or left out completely. Furthermore, these approaches do not provide an apparatus for forming the background information which will assure the applicability of constraints. In BABY-SIT, the abstract nature of situations make it possible to form abstractions without asserting facts into them. The salient contributions of BABY-SIT are listed below:

- An interactive environment helps one to develop and test his program, observe its behavior vis-à-vis extra (or missing) information, make inference over the model, and issue queries.
- Objects in the environment and the attainment of information flow are compatible with the ontology of situation theory.
- Computation is highly context-sensitive and type-theoretic.
- The mode of computation is built upon conveyance and inheritance of information, consistency of the anchoring situation, and constraint satisfaction.
- Various types of constraints (i.e., necessary, nomic, conventional, conditional, and unconditional) [11] can be effectively utilized.
- Inheritance of information (from the background situation) is supported by BABY-SIT. This, together with the information conveyance among situations, enables one to use contextual information whenever necessary.
- Building models of reasoning from various 'points of view' is possible, especially via the anchoring situation associated with the environment.

The syntax of the declarations and the operational semantics of BABY-SIT are currently being defined. We are planning to implement BABY-SIT using the KEE^{TM} knowledge engineering environment [38]. KEE has a multi-world mechanism. Using KEE's built-in inference mechanism, one can define her own methods of inference. Combining forward- and backward-chaining mechanisms is also possible. Moreover, KEE's truth maintenance system enables one to handle inconsistency.

References

[1] P. Aczel. *Non-Well-Founded Sets*, CSLI Lecture Notes Number 14, Center for the Study of Language and Information, Stanford, Calif., 1988.

[2] V. Akman and E. Tın. "What is in a Context?" in E. Arıkan, editor, *Proceedings of the 1990 Bilkent International Conference on New Trends in Communication, Control, and Signal Processing, Volume II*, Amsterdam, Holland: Elsevier, 1990, pp. 1670–1676.

[3] N. Asher. "Verbal Information, Interpretation, and Attitudes" in P. P. Hanson, editor, *Information, Language, and Cognition*, Vancouver, Canada: The University of British Columbia Press, 1990, pp. 29–56.

[4] J. L. Austin. "Truth," in J. O. Urmson and G. J. Warnock, editors, *Philosophical Papers of J. L. Austin*, Oxford, U.K.: Oxford University Press, 1961, pp. 117–133.

[5] J. Barwise. "An Introduction to First-Order Logic," in J. Barwise, editor, *Handbook of Mathematical Logic*, Amsterdam, Holland: North-Holland, 1977, pp. 5–46.

[6] J. Barwise. "Model-Theoretic Logics: Background and Aims," in J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, Berlin, Germany: Springer-Verlag, 1985, pp. 3–23.

[7] J. Barwise. "Noun Phrases, Generalized Quantifiers, and Anaphora," in P. Gärdenfors, editor, *Generalized Quantifiers*, Dordrecht, Holland: Reidel, 1987, pp. 1–29.

[8] J. Barwise. *The Situation in Logic*, CSLI Lecture Notes Number 17, Center for the Study of Language and Information, Stanford, Calif., 1989.

[9] J. Barwise and J. Etchemendy. *The Liar: An Essay on Truth and Circularity*, New York, N.Y.: Oxford University Press, 1987.

[10] J. Barwise and J. Etchemendy. "Model-Theoretic Semantics," in M. I. Posner, editor, *Foundations of Cognitive Science*, Cambridge, Mass.: MIT Press, 1989, pp. 207–243.

[11] J. Barwise and J. Perry. *Situations and Attitudes*, Cambridge, Mass.: MIT Press, 1983.

[12] J. Barwise and J. Perry. "Semantic Innocence and Uncompromising Situations," in A. P. Martinich, editor, *The Philosophy of Language*, New York, N.Y.: Oxford University Press, 1985, pp. 401–413.

[13] J. van Benthem. *Essays in Logical Semantics*, Dordrecht, Holland: Reidel, 1986.

[14] A. W. Black. "Constraints in Computational Situation Semantics," Lecture Notes circulated during the *Logic, Language, and Information Summer School*, Saarbrücken, Germany, 1991.

[15] A. W. Black. "Embedding DRT in a Situation-Theoretic Framework," in *Proceedings of the Fifteenth International Conference on Computational Linguistics*, Nantes, France, 1992, pp. 1116–1120.

[16] A. W. Black. "ASTL—A Language for Computational Situation Semantics," Manuscript, Department of Artificial Intelligence, University of Edinburgh, Edinburgh, U.K., 1992.

[17] C. C. Chang and H. J. Keisler. *Model Theory*, Third Edition, Amsterdam, Holland: North-Holland, 1990.

[18] N. Chomsky. *Lectures on Government and Binding*, Dordrecht, Holland: Foris, 1981.

[19] E. Colban. "Prepositional Phrases in Situation Schemata," Appendix A in Fenstad *et al.*, 1987.

[20] R. Cooper. "Tense and Discourse Location in Situation Semantics," *Linguistics and Philosophy*, **9**: 17–36, 1986.

[21] R. Cooper. "Meaning Representation in Montague Grammar and Situation Semantics," in B. G. T. Lowden, editor, *Proceedings of the Alvey Sponsored Workshop on Formal Semantics in Natural Language Processing*, 1987.

[22] R. Cooper. "Three Lectures on Situation Theoretic Grammar," in M. Filgueiras, L. Damas, N. Moreira, and A. P. Tomás, editors, *Natural Language Processing*, Lecture Notes in Artificial Intelligence, Vol. 476, Berlin, Germany: Springer-Verlag, 1991, pp. 102–140.

[23] R. Cooper, K. Mukai, and J. Perry, editors. *Situation Theory and Its Applications, Volume 1*, CSLI Lecture Notes Number 22, Center for the Study of Language and Information, Stanford, Calif., 1990.

[24] K. Devlin. *Logic and Information*, Cambridge, U.K.: Cambridge University Press, 1991.

[25] K. Devlin. "Infons as Mathematical Objects," *Minds and Machines*, **2**: 185–201, 1992.

[26] D. Dowty, R. Wall, and S. Peters. *Introduction to Montague Semantics*, Dordrecht, Holland: Reidel, 1981.

[27] F. Dretske. *Seeing and Knowing*, Chicago, Illinois: University of Chicago Press, 1969.

[28] F. Dretske. *Knowledge and the Flow of Information*, Cambridge, Mass.: MIT Press, 1981.

[29] D. A. Evans. "A Situation Semantics Approach to the Analysis of Speech Acts," in *Proceedings of the Nineteenth Annual Meeting of the Association for Computational Linguistics*, 1981.

[30] G. Evans. *The Varieties of Reference*, New York, N.Y.: Oxford University Press, 1991.

[31] J. E. Fenstad, P.-K. Halvorsen, T. Langholm, and J. van Benthem. *Situations, Language, and Logic*, Dordrecht, Holland: Reidel, 1987.

[32] J. E. Fenstad. "Natural Language Systems," in R. T. Nossum, editor, *Advanced Topics in Artificial Intelligence: 2nd Advanced Course*, Lecture Notes in Artificial Intelligence, Vol. 345, Berlin, Germany: Springer-Verlag, 1987, pp. 189–233.

[33] J. M. Gawron and S. Peters. *Anaphora and Quantification in Situation Semantics*, CSLI Lecture Notes Number 19, Center for the Study of Language and Information, Stanford, Calif., 1990.

[34] H. P. Grice. "Utterer's Meaning, Sentence-Meaning, and Word-Meaning," *Foundations of Language*, **4**: 1–18, 1968.

[35] Z. S. Harris. *A Theory of Language and Information: A Mathematical Approach*, New York, N.Y.: Oxford University Press, 1991.

[36] D. Israel and J. Perry. "What is Information?" in P. P. Hanson, editor, *Information, Language, and Cognition*, Vancouver, Canada: The University of British Columbia Press, 1990, pp. 1–28.

[37] H. Kamp. "A Theory of Truth and Semantic Representation," in J. Groenendijk, T. Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language*, Amsterdam, Holland: Mathematical Center, 1981, pp. 277–322.

[38] *KEE^{TM} (Knowledge Engineering Environment) Software Development System*, Version 3.01, IntelliCorp, Inc., Mountain View, Calif., 1988.

[39] H. Nakashima, S. Peters, and H. Schütze. "Communication and Inference through Situations," in *Proceedings of the Third Conference on Artificial Intelligence Applications*, Washington, D.C.: IEEE Computer Society Press, 1987, pp. 76–81.

[40] H. Nakashima, H. Suzuki, P.-K. Halvorsen, and S. Peters. "Towards a Computational Interpretation of Situation Theory," in *Proceedings of the International Conference on Fifth Generation Computer Systems*, Institute for New Generation Computer Technology, Tokyo, Japan, 1988, pp. 489–498.

[41] J. Perry. "Contradictory Situations," in F. Landman and F. Veltman, editors, *Varieties of Formal Semantics*, Dordrecht, Holland: Foris, 1984, pp. 313–324.

[42] M. Rooth. *Noun Phrase Interpretation in Montague Grammar, File Change Semantics, and Situation Semantics*, Report No. CSLI–86–51, Center for the Study of Language and Information, Stanford, Calif., 1986.

[43] H. Schütze. "The PROSIT Language v0.4," Manuscript, Center for the Study of Language and Information, Stanford University, Stanford, Calif., 1991.

[44] J. R. Searle. *Speech Acts: An Essay in the Philosophy of Language*, Cambridge, U.K.: Cambridge University Press, 1969.

[45] P. Sells. *Lectures on Contemporary Syntactic Theories*, CSLI Lecture Notes Number 3, Center for the Study of Language and Information, Stanford, Calif., 1985.

[46] G. L. Steele, Jr. *Common Lisp: The Language*, Second Edition, Bedford, Mass.: Digital Press, 1990.

[47] L. Sterling and E. Shapiro. *The Art of Prolog: Advanced Programming Techniques*, Cambridge, Mass.: MIT Press, 1986.

[48] S. Stucky. "The Situated Processing of Situated Language," *Linguistics and Philosophy*, **12**: 347–357, 1989.