Processing Entailments and Accessing Facts in a Uniform Frame System*

Anthony S. Maida

Institute of Cognitive Studies
University of California, Berkeley
Berkeley, California 94720

Computer Science Dept.
Penn State University
University Park, PA 16802

Abstract

This paper: 1) describes the structure of a "uniform" frame system; 2) shows how entailments can be computed within the system; and, 3) shows how contingent facts that are related to a concept can become accessable as a function of how deeply the meaning of that concept is processed. The system is called **UniFrame** and differs from slot-filler frame systems primarily in its commitment to uniformly representing all concepts and to maintaining a representation which is semantically well knit.

I. INTRODUCTION

This paper describes the structure of a "uniform" frame system, showing how entailments can be computed within the system, and how contingent facts that are related to a concept can become accessable as a function of how deeply the meaning of that concept is processed.

The system is called **UniFrame** and differs from slot-filler frame systems primarily in that it does not use slots. Instead it places emphasis on four desireable characteristics in a knowledge representation. They are: 1) The **uniformity** of the formalism; 2) The **semantic coherence** of the conceptual structures represented in the formalism; 3) Whether new concepts are defined by **differentiating** them from existing concepts; and, 4) Whether the representation allows for a variable **depth of processing** of concepts depending on the contextual situation. We will describe each of these dimensions in turn.

Uniformity: This dimension embodies the intuition that all concepts are made of the same "stuff" (cf., [8]). For instance, humans invariably explain one concept in terms of other concepts and, although the concepts used in the explanation are different concepts, they are nonetheless the same kind of object. Below are two examples of common representational situations we wish to avoid because we wish to maintain a uniform representation. They can be viewed as "bugs" in hierarchical slot-filler frame systems [e.g., FRAIL [3]; NETL [5]; PEARL [4]].

Frames vs Slots. A frame for the concept PHYSICAL OBJECT is likely to have the slot COLOR-OF. If, in addition there is a frame for the concept COLOR, there is likely to be no indication that the COLOR-OF slot in the PHYSICAL OBJECT frame and the COLOR frame are semantically related. Somehow, the COLOR-OF slot should be in part derived from the concept COLOR (Wilensky, [11], [12])**.

Uniformity. In a frame system, the concept PERSON would be represented as a frame but a concept such as MURDERER would be the slot, MURDERER-OF, in the MURDER-EVENT frame. This violates the intuition of uniformity because MURDERER has as much reason to be represented as a full-fledged concept as PERSON does.

This lack of uniformity propogates itself in representing semantically similar assertions. Consider the sentences:

All persons belong in jail.
All murderers belong in jail.

Despite the semantic similarity of these sentences, their representations would be dissimilar. The former sentence references a frame whereas the latter references a slot.

Semantic Coherence: Concepts can be related to one another in various ways (cf., [1]). We would like concepts in a data base to be well-knit, while remaining semantically precise. We have identified three forms of relatedness.

Similarity. The most obvious form of relatedness has to do with the sharing of common components of meaning. This can be seen in families of verb senses which have similar meaning. For instance, the verbs: kill, murder, strangle, and suicide all share similar components.

Definitional Relationships. There are at least two kinds of definitional relationships, namely *derived concepts* and *component concepts*. For instance, the concept MURDERER is derived from the concept MURDER-EVENT. A MURDERER is one who murders somebody. However, the concept DIE bears a different relationship to the concept MURDER-EVENT. Namely, DIE is used as a component in the definition of MURDER-EVENT.

*-Acknowledgements: Many of these views emerged from discussions in a knowledge representation seminar given by Robert Wilensky. Other participants of that seminar were: M. Butler, D. Chin, C. Cox, B. D'Ambrosio, P. Jacobs, M. Luria, J. Martin, J. Mayfield, P. Norvig, L. Rau, J. Sokolov, and N. Ward. Another language which emerged from that seminar was Wilensky's KODIAK. Richard Alterman and Nigel Ward provided helpful comments on this paper. This research was supported by the A. P. Sloan foundation. Author's present address: Computer Science Dept., Whitmore Lab, Penn State University, University Park, PA 16802.

**-The idea of patching frame systems in this manner is due to

Contingent Relationships. There is a form of concept relatedness that stems from contingencies about how the world happens to be organized. The relatedness of the concepts MURDERER and JAIL can be explained by the highly salient empirical fact that murderers are supposed to be put in jail.

As Brachman, Fikes & Levesque [2] have pointed out, frame systems often confuse contingent and definitional relationships. For instance, one might see in a frame system the factual assertion that if person B is the victim of a murder event, then person B is dead. This assertion however does not represent a contingent fact about the world, but rather reflects an entailment.

Progressive Differentiation of Concepts: Hierarchical frame systems define new concepts by saying they are kinds of old concepts, but which have more slots. This is analogous to saying that an X is a Y that has such-and-such features. We shall opt for a different method of progressive differentiation. We say instead that an X is a Y which obeys such-and-such constraint, where the constraint relation can be more general than a feature. Instead of saying that a murder event has a murderer slot, a method slot, and a victim slot, we can define a murder event as the event of a volitional force causing a normally living thing to die by some action. This is our primary point of divergence from KRYPTON [2]. Advantages are:

a) The method offers increased expressive power over simply adding slots.

b) The constraint, which is used to differentiate a concept from its superordinate(s), can be used to compute semantic entailments.

c) Progressive differentiation is compatible with a highly plausible form of concept acquisition; namely, we acquire new concepts by differentiating concepts we already have [6].

Depth of Processing: As will be seen, UniFrame's constraint relation allows for a variable depth of processing of the meaning of a concept. As the meaning of a concept is processed more deeply, an increasing number of contingent facts related to that concept become accessable.

II. DETAILS OF UNIFRAME

The Concept Frame. In accordance with the principle of uniformity, UniFrame uses only one device, called the **concept frame**, to define concepts. A concept frame describes a concept by saying how it is a different concept from one of its superordinates. This differentiation process results in an ISA hierarchy. The concept frame depicted in **Table 1** represents the concept MURDER-EVENT.

Wilensky, although the virtue of uniformity in a conceptual hierarchy can be traced back at least to Quillian [9], and the idea of having only nodes (and not links) representing concepts in a semantic network is due to Shapiro [10].

```
[concept: MURDER-EVENT      isa: (KILL-EVENT
                                  INTENTIONAL-DO)
      with:        ($agent:         VOLITIONAL-FORCE
                    $instrument:      ACTION
                    $victim:          LIVING-THING)
constraint: (INTENTIONAL-DO $agent
                       (NAIVE-CAUSE $instrument
                                          (DIE $victim)))
derived concepts:        (MURDERER          $agent)
                         (MURDER-VICTIM $victim)
slot-functions: (MURDERER-OF          $agent) ]
```

Table 1

A **concept frame**, for the moment, consists of four parts: 1) a **concept name**, 2) an **isa-specification**, 3) a typed **variable list**, and, 4) an optional **constraint relation template** which is used to specify how the concept that is currently being defined is defined in terms of constraints on less differentiated concepts. The isa-specification indicates the category of entity that results when MURDER-EVENT is instantiated. The variable list declares the obligatory "slots" of the frame and indicates the category of object that can fill the slot. The constraint template, by making reference to the variable names in the variable list, serves to embed the slots in a relationship of previously defined concepts. The template in **Table 1** indicates that MURDER-EVENT is an intentional action in which a volitional force causes a living thing to die by some action.

The concept frame defines concepts by making references to other concepts. These concepts are referenced in three places: 1) The ISA specification; 2) The typings of the variable list; and, 3) All atomic expressions in the constraint template which are not variables. Thus, to define the concept MURDER-EVENT, we make reference to the following other concepts: KILL-EVENT, INTENTIONAL-DO, VOLITIONAL-FORCE, ACTION, LIVING-THING, NAIVE-CAUSE, and DIE.

The major effect of specifying the concept frame in **Table 1** is to define a new three-place relation to the system, MURDER-EVENT. For instance, after defining MURDER-EVENT, we can use expression (1) to assert that John murders Bill by suffocation (provided that SUFFOCATE has been defined as a concept).

```
(1) (MURDER-EVENT John
           (SUFFOCATE John Bill)  Bill)
```

The constraint template allows concepts to be explicated as constraints on relations between more general and simpler concepts. UniFrame can optionally explicate expression (1) into expression (2) below.

```
(2) (INTENTIONAL-DO John
           (NAIVE-CAUSE (SUFFOCATE John Bill)
                        (DIE  Bill)))
```

That is, MURDER-EVENT can be treated at an unanalyzed level, or it can be explicated as "intentional cause to die." The component concepts: INTENTIONAL-DO, SUFFOCATE and DIE, are themselves represented as concept frames which may have template fields that can in turn be explicated.

Derived Concepts. The concept MURDERER is derived from the concept MURDER-EVENT. That is, a necessary and sufficient condition to be a murderer is to be the agent of a murder event. MURDERER is defined in **Table 2.**

```
[concept: MURDERER   isa: VOLITIONAL-FORCE

      with: ($murderer:   VOLITIONAL-FORCE)

constraint: ((lambda (x) (MURDER-EVENT x
                              (some ACTION)
                              (some LIVING-THING))
                  $murderer) ]
```

Table 2

A murderer is a volitional force with the property that it murders a normally living thing. This definition can be generated automatically by specifying the expression (MURDERER $agent) in the derived concepts field in **Table 1**.

How can this be done?*** Note that the variable, $agent, is typed as a volitional force in **Table 1**. Thus MURDERER is a volitional force. The constraint relation is constructed by doing lambda abstraction on $agent in **Table 1** and substituting typed existential quantifiers for any slot variables encountered.

What good is it? MURDERER is now a concept about which facts can be uniformly asserted (e.g., Murderers are dangerous. Murderers may be violent., etc).

There is one remaining field in **Table 1**. This is the slot-functions field. It enables us to make reference to the murderer of a particular murder-event rather than to the concept of the generic murderer.

III. ENTAILMENTS

Two types of inference operations are used to compute entailments. One is inheritance down the ISA hierarchy. For instance, if John strangles Bill, then it follows that John kills Bill by virtue of a STRANGLE-EVENT being a KILL-EVENT. The other operation concerns *explication* of the definition of a concept. Since inheritance is a well known tool, we will only discuss explication. Explication of a concept involves two things: 1) expansion of the relational template to make the *component concepts* explicit; and, 2) instantiation of the derived concepts to make the *derived concepts* explicit.

Template Expansion. Template expansion involves substituting a concept's constraint relation template, as was done in expression (1) to generate expression (2). Complete expansion would involve recursively expanding all of the concepts referenced in the template until primitives were reached. Expansion to two levels of the sentence "John murdered Bill" (ignoring tense and aspect) generates the following.

***-This has also been done by the use of the QUA link in Brachman's SI-NETS [7].

```
(3) (MURDER-EVENT John (some ACTION) Bill) -->

(4) (INTENTIONAL-DO John (NAIVE-CAUSE
                              (some ACTION) -->
                              (DIE Bill)))

(5) (INTENTIONAL-DO John
              (NAIVE-CAUSE
                   (some ACTION)
                   (STATE-CHANGE
                        (LIVING Bill)
                        (not (LIVING Bill)))))
```

Deciding how far to expand is a control problem dependent on the inference requirements of the task.

Instantiating Derived Concepts. Other entailments of expression (3) are expressions (6) and (7) below.

```
(6) (MURDERER John)
(7) (MURDER-VICTIM Bill)
```

These instantiations are obtained by applying the appropriate argument to the derived concepts of MURDER-EVENT, namely, MURDERER and MURDER-VICTIM.

Concept Explication. Explicating expression (3) one level involves: 1) instantiating the derived concepts, MURDERER and MURDER-VICTIM, with appropriate arguments; and, 2) expanding the template field of MURDER-EVENT to one level. If this is done according to **Table 1**, expressions (4), (6), and (7) result. The concepts directly mentioned in these expressions are considered to have been made explicit.

IV. ACCESSING FACTS: AN EXAMPLE

Contingent facts associated with a concept become accessable when that concept is made explicit, either by being used directly, or by the process of explication. Consider the task of referent identification in either of the situations below.

John murdered Bill. The funeral was held on Monday.
John murdered Bill. The trial was held on Monday.

"The funeral" refers to Bill's funeral but "the trial" refers to John's trial. Whatever processes are involved in identifying these referents, relevant facts must be accessed. In this case, the contingent fact that people who die have funerals (associated with DIE) determines that "the funeral" refers to Bill's funeral, and the information that murderers have trials (associated with MURDERER) determines that "the trial" refers to John's trial. When will these facts become available?

Explicating expression (3) one level makes both of these facts accessable. Since contingent facts become accessable as the concepts they are associated with become explicit then, expression (4) allows facts stored with DIE to become accessable, and expression (6) allows facts stored with MURDERER to become accessable.

As more templates composing the meaning of a concept are expanded, more facts related to that concept become available. However, search for contingent facts is highly constrained and can proceed only as the concept's definition is explicated. Assuming that search terminates when the relevant facts are found, we have a situation where the meanings of concepts are processed to a variable depth and this is controlled by the inference requirements of the task domain.

V. SUMMING UP

How do UniFrame's features match up to the desired characteristics of a representation mentioned in the introduction? We discuss each in turn.

Uniformity: Concepts are defined only by concept frames and concept frames themselves make reference only to other concepts. Concepts which are typically slots in other systems are full-fledged concepts in this system. Assertions about murderers can be made in the same way as assertions about persons.

Coherence of Conceptual Structures: In terms of similarity of related verbs, UniFrame is like most hierarchical frame systems. The concepts underlying these verbs would appear at proximal places in the concept hierarchy.

With respect to the relatedness that derives from definitional relations, UniFrame has better facilities than most frame systems. UniFrame can represent derived concepts, such as MURDERER. It also captures the way DIE is a component concept of MURDER-EVENT. It also distinguishes between the slot-function MURDERER-OF in the MURDER-EVENT frame and the concept MURDERER.

With respect to contingent relationships, factual knowledge can be added to UniFrame in the same way that it can be added to any frame system. However, UniFrame discriminates between the two kinds of knowledge. Death of a murder victim follows from the definition of the murder event, rather than as an asserted fact about the murder event.

Progressive Differentiation of Concepts: UniFrame progressively differentiates concepts by the use of a hierarchy, just as most frame systems do, but instead of adding slots in order to specialize concepts, the constraint relation uses other concepts to impose relationships between slots.

Depth of Processing: UniFrame makes it easy to process the meaning of a concept to a varying amount of depth, simply by deciding whether to expand a template, or instantiate derived concepts. As the meaning is processed more deeply (explicated), more contingent facts become accessable.

REFERENCES

[1] Alterman, R. "Event concept coherence in narrative text" In *Proc. Fifth Annual Conference of the Cognitive Science Society,* Rochester, New York, May, 1983.

[2] Brachman, R., Fikes, R., and Levesque, H. "KRYPTON: Integrating terminology and assertion" In *Proc. AAAI-83,* Washington, D.C., August, 1983, 31-35.

[3] Charniak, E. "A common representation for problem solving and language comprehension information." *Artificial Intelligence,* 16, 1981, 225-255.

[4] Deering, M., Faletti, J., and Wilenksy, R. "PEARL: An efficient language for artificial intelligence programming" In *Proc. IJCAI-81,* Vancouver, Canada, August, 1981.

[5] Fahlman, S.E. *NETL: A system for representing and using real-world knowledge.* Cambridge, Mass.: MIT Press, 1979.

[6] Kolodner, J. "Maintaining organization in a dynamic long-term memory." *Cognitive Science,* 7:4, 1983, 243-280.

[7] Leitner, H.H. & Freeman, M.W. "Structured inheritance networks and natural language understanding" In *Proc. IJCAI-79,* Tokyo, Japan, August, 1979, 525-530.

[8] Maida, A.S. & Shapiro, S.C. "Intensional concepts in propositional semantic networks." *Cognitive Science,* 6:4, 1982, 291-330.

[9] Quillian, M.R. "Semantic Memory." In M. Minsky (Ed.) *Semantic Information Processing,* Cambridge, Mass.: MIT Press, 1968.

[10] Shapiro, S.C. "A net structure for semantic information storage, deduction and retrieval" In *Proc. IJCAI-71,* vol. 2, 512-523, 1971.

[11] Wilensky, R. "Knowledge Representation - A Critique and a Proposal" In *Proc. First Annual Workshop on Theoretical Issues in Conceptual Information Processing,* Atlanta, Georgia, March, 1984.

[12] Wilensky, R. "KODIAK: A Knowledge Representation Language" In *Proc. 6th Annual Conference of the Cognitive Science Society,* Boulder, Colorado, June, 1984.