CLUSTER: An Approach to Modeling Context*

Yigal Arens

Computer Science Department
University of Southern California

1. Introduction: CLUSTER and the Context of an Utterance

This paper outlines a theory of modeling the context of an utterance. It is part of the CLUSTER** theory of language understanding in context (Arens, 1986). It has been implemented in a version of UC (the UNIX Consultant) (Wilensky, Arens, & Chin, 1984).

CLUSTER builds on the following observations concerning the process of language understanding:

- Language understanding is not sentence-based.
- New input is understood in reference to the context, not necessarily the previous text.
- The context is updated and changed in the course of processing input. Insertion of a fact into the context may cause the insertion of a group of associated facts.
- Elements of the context have a degree of salience associated with them.
- This salience is a status that decays over time and may eventually be lost.
- Information in the context should be recorded periodically in the more permanent form.

CLUSTER theory posits several levels of memory structures. At the lowest level are *entries,* 'atomic' representations of elements of the context. Entries are grouped together in *clusters.* Such a grouping indicates that the entries tend to co-occur in the world. A collection of clusters is stored in an understanding system's *Long Term Memory.* The mention of an entry in an input sentence, say, will result in the recall of clusters containing it. The recalled clusters will then be placed in *the Context Model,* where they will be available to any processes that require knowledge of the context.

1.1. Entries

There are three general classes of entries. Each contains representations of different aspects of the world within which the understanding system operates.

Objects – Entries in this class stand for physical objects, events, and actions. An entry of this type must be present in the Context Model before a reference to such an object may be understood.

Assertions – Entries in this class describe states of the world, state-changes, or actions that have taken place. An entry of this type must be present before the effects of the thing it represents may be accessed.

Intentions – These entries describe intentions the understanding system has of performing certain actions. The actions the system can perform include output to the user (i. e., saying something), adding or deleting an existing entry in the Context Model, and more.

* Some of the work reported here was performed when the author was at the University of California at Berkeley, where it was sponsored by the Office of Naval Research under contract N00014-80-C-0732, and DARPA (DOD), ARPA order No. 3041, monitored by the Naval Electronic Systems Command under contract N00039-82-C-0235. The work performed at the University of Southern California was supported by the USC Faculty Research and Innovation Fund, and by TRW Defense Systems under Subcontract No. D71810AN5S.

** Contextual Language Understanding with Salience Tracking and Environment Recall.

The same real-world object or event may be represented by an entry of more than one type. For example, the system's intention to say something to the user may also be represented in an entry of type *object*. This would permit the system to understand references to this intention in the user's input, a situation that would not be possible otherwise.

1.2. Clusters

Entries do not, as a rule, occur in isolation. A collection of entries is arranged in a cluster if those entries tend to be found together in the real world. The choice of *associatedness* as the basis for cluster structure is supported by psychological evidence of its importance. There is considerable psychological evidence for the existence of an automatic process that makes information available on the basis of associative relatedness (McKoon & Ratcliff, 1979) (Neely, 1977) (Warren, 1977) (and see Anderson (1983) for a review). The co-occurrence of entries signified by their forming a cluster is usually due to the fact that they represent one of the following:

- the various parts of a complex object;
- the several steps of a complex action;
- the actors and instruments of a particular activity;
- intended or associated uses of a particular object;
- standard associations between certain stimuli and responses, causes and effects; or
- actions related by convention (e. g., an inquiry into whether one can pass a salt shaker and a request that the salt shaker actually be passed).

For example, a cluster from the UNIX domain relating the making of a file executable to the command that must be issued in order to do so will include the following entries*:

?F is a file
?U issues command 'chmod' with argument '755'
?F changes its 'executability' state to 'on'

The existence of this cluster in long term memory asserts, among other things, that the last two entries are related, i. e., issuing the command 'chmod 755 <filename>' is associated with making a file executable, and vice versa. Mentioning any one of these two facts in a conversation causes the second to become present in the Context Model, hence in context.

A more complex cluster, containing a variety of entries is displayed below:

?A is an action
?U has asked how to perform action ?A
?U wishes to know how to perform ?A
?U cannot perform action ?A
System intends to find plan for ?A
?U expects a response

This cluster reflects the association between the user asking how to perform some action, the user's wish to know how to perform it, and the system's intention of finding such a plan and informing the user of it.

* Entries are presented in a simplified notation. ?F and ?U are variables, allowing this cluster to be used regardless of who the particular user (?U) is, and which file (?F) is involved.

1.3. Activation

Entries in the Context Model are marked with a measure of their salience in the context, indicated by a level of activation. In CLUSTER, the level of activation is viewed as a point on a scale, thus distinguishing it both from Quillian's (1969) and Charniak's (1983) marker passing models. The processing of input is accompanied by the activation of entries representing concepts mentioned in it and a flow of activation from those entries to others in their cluster(s), and so forth. Through the accumulation of activation flow from several sources, the Context Modeler has the ability to identify entries that are salient in the current context, although not closely associated with any single entry.

Activation flow is *cluster-directed*. When an entry's activation level is increased, all clusters to which it belongs are identified. The entry's increase is divided by the number of clusters and that amount flows on to each cluster. Within each of the clusters, however, every entry receives the same increase, regardless of the size of the cluster. In this manner, CLUSTER distinguishes between entries associated with many different situations (e. g., a file and its numerous uses) and entries associated with only a few (e. g., the notion of execution). In the latter case we would like to infer that the current context is probably one of those few clusters/situations, regardless of the number of entries in each. Thus, when 'execution' is mentioned we can realize that there is some program being discussed regardless of the number of facts we know about the execution of programs.

The level of activation of an entry in the Context Model influences the choices made by processes that need to know about the context. As a rule, a process in search of a particular type of entry will choose the one of that type with the highest level of activation. See section 2. for examples.

Unless reinforced, as described above, the activation of all entries in the Context Model decays gradually with the passage of time. When the activation of an entry decreases below a certain threshold it is dropped from the Context Model.

Entries of the class *intention* are triggered when their activation grows high enough. Their intended action is then carried out.

1.4. Processing Outline

New entries are passed on to the Context Modeler by outside processes. In the current implementation, the only such process is the language analyzer. It occasionally identifies a fragment of the input language that is meaningful to it and passes its meaning to the Context Modeler. The latter forms an entry representing the meaning of the language fragment. The following process then takes place:

1. The Context Modeler attempts to unify the new entry with those already existing in the Context Model. Either a match is found or else a new entry is created in the Context Model.

   If an existing entry is found:

2a. All clusters of which the existing entry is a member are reinforced, with activation flow continuing on from those.

   Otherwise:

2b. All clusters associated with the new entry are retrieved from long term memory and their members are inserted recursively.

3. This process continues until stability is reached.

4. The Context Model is inspected and highly activated *intentions* are performed.

At this point the Context Modeler awaits new input from the language analyzer.

2. Usefulness of the Context Model to Language Analysis

In a conversational context the language analyzer cannot treat sentences as independent entities. Rather, it must relate their content and import to information gathered during the processing of previous utterances. In particular, objects, actions, and any other concepts mentioned in, or whose existence is implied by, an utterance must be disambiguated. Their interpretations must be identified from among those concepts present in the Context Model. For this to take place the language analyzer must be capable of identifying meaningful fragments in the input and passing them on to the Context Modeler as soon as possible.

A language analyzer with these characteristics is used in the implementation of CLUSTER. This is PHRAN, the PHRasal ANalyzer (Wilensky and Arens, 1980a, 1980b). Upon recognizing a phrasal construct in the input, PHRAN constructs its paired concept and passes it to the Context Modeler. The concept becomes part of the entry representing the meaning of the recognized language fragment.

2.1. Ambiguity Resolution

When PHRAN processes input it often encounters ambiguous language fragments. In such instances entries representing the meaning of *all* interpretations are inserted in the Context Model. In the absence of any selectional restrictions or other linguistic clues, or if such restrictions are insufficient to uniquely determine the meaning, PHRAN chooses the most highly activated entry of the appropriate type as representing the meaning of the ambiguous fragment. Naturally, the presence of related clusters in the Context Model will cause an entry's activation to be higher.

2.2. Definite References

By itself, a language analyzer is incapable of determining precisely which real world entities definite references refer to. This is because the understanding of most definite references inherently involves the use of information not present in the sentence being processed.

The Context Model provides a pool of candidates for the interpretation of referring expressions. As a rule, the language analyzer will choose as the referent the most highly activated entry consistent with the restrictions provided by the referring expression. When no such entry can be found, the Context Modeler realizes that a new entity is being discussed. It then creates an appropriate new entry and inserts it in the Context Model.

2.3. Pronouns and Demonstratives

As is the case with definite references, the language anlyzer searches the Context Model for a highly activated entry consistent with the restrictions derivable from the pronoun or demonstrative used. These will often include gender information.

3. Processing Example

This section contains an example of the operation of the UNIX Consultant (UC). I will sketch the processing involved in engaging in the dialogue that follows. I will pay special attention to the determination of the referent of the word 'it' in line [5].

[1] User: How do I print the file fetch.l?
[2] UC: To print the file fetch.l type 'lpr fetch.l'.

. (intervening commands and questions)

[3] User: Has the file fetch.l been printed yet?
[4] UC: The file fetch.l is in the line printer queue.
[5] User: How can I cancel it?
[6] UC: To remove the file fetch.l from the line printer queue type 'lprm arens'.

The fragments recognized by PHRAN in the user's first question, [1], are, in the order in which they are identified:

I
the file fetch.l
How do I print the file fetch.l ?

The analysis of each fragment results in a structure representing its meaning. This structure is inserted in the Context Model. The third fragment recognized is a request concerning printing. Its processing results in a complete cluster being retrieved from long term memory and inserted in the Context Model. The cluster contains structures describing the command to print, its effect, the object printed and the fact that a printer is involved. This information is used to determine the answer provided by the system in [2].

After UC's reply, the Context Model contains representations of the following objects and actions:

(1) The user
(2) The file fetch.l
(3) The line printer
(4) The request to print fetch.l
(5) The command to print the file ('lpr fetch.l')
(6) A printing event
(7) The user has requested to print the file
(8) UC has told the user how to print the file

The contents of the Context Model are now preserved in long term memory in the form of a new cluster. This is done following an explicit request by the user, as this process is not currently automatic. The new cluster will be recalled when the representation of any of the entries (2), (4), (5), or (7), is encountered again.

The conversation now shifts to other topics and the structures previously in the Context Model are gradually deleted due to decay of their activation levels. By the time the user asks question [3], there is no trace left in the current context of the structures used in the earlier exchange. For the sake of this example, I will assume that when line [3] is input by the user the Context Model is empty.

When the user's second question is processed by PHRAN, the following fragments are recognized:

the file fetch.l
Has the file fetch.l been printed yet?

When an entry describing the first one is created in the Context Model, it serves as a key for retrieving the new cluster described above. The concept associated with the second fragment recognized causes the

retrieval of other clusters. These clusters enable the system to respond with [4], but are otherwise irrelevant to this example.

When question [5] is asked by the user, it is analyzed by PHRAN and the following input fragments are identified:

I
it
How can I cancel it?

The structure created to describe the first fragment is identified with the structure describing the user, which is already present in the Context Model.

When a structure is created to represent the meaning of the second fragment, *it*, no determination of its precise meaning by UC is possible. However, when the last fragment is recognized, PHRAN notes that the referent of *it* must be a command. The only command found in the Context Model is the one to print the file fetch.l. This was entry (5) in the cluster formed after processing question [1], which was stored in long term memory and retrieved during the processing of question [3] above. The identification of this entry as the referent enables UC to correctly interpret the user's question and eventually to provide the answer [6].

4. References

Anderson, J. R., 1983. A Spreading Activation Theory of Memory. *Journal of Verbal Learning and Verbal Behavior,* 22, pp. 261-295.

Arens, Y., 1986. CLUSTER: An Approach to Contextual Language Understanding. Ph. D. Thesis. Report No. UCB/CSD 86/293, Computer Science Division (EECS), UC Berkeley. April 1986.

Charniak, E., 1983. Passing Markers: A Theory of Contextual Influence in Language Comprehension. *Cognitive Science* 7, pp. 171-190.

McKoon, G., and Ratcliff, R., 1979. Priming in Episodic and Semantic Memory. *Journal of Verbal Learning and Verbal Behavior,* 18, pp. 463-480.

Neely, J. H., 1977. Semantic Priming and Retrieval from Lexical Memory: Roles of Inhibitionless Spreading Activation and Limited-capacity Attention. *Journal of Experimental Psychology: General,* 106, pp. 226-254.

Quillian, M. R., 1969. The Teachable Language Comprehender: A Simulation Program and Theory of Language. In *Communications of the ACM,* 12(8), August 1969, pp. 459-476.

Warren, R. E., 1977. Time and Spread of Activation in Memory. *Journal of Experimental Psychology: Learning and Memory,* 3, pp. 458-466.

Wilensky, R., and Arens, Y., 1980a. PHRAN: A Knowledge-Based Approach to Natural Language Analysis. Memo UCB/ERL M80/34, Electronic Research Laboratory, UC Berkeley, August 1980.

Wilensky, R., and Arens, Y., 1980b. PHRAN: A Knowledge-Based Natural Language Understander. In *Proceedings of the 18th Annual Meeting of the Association for Computational Linguistics,* Philadelphia, PA. June 1980.

Wilensky, R., Arens, Y., and Chin, D., 1984. Talking to UNIX in English: An overview of UC. In *Communications of the ACM.* June, 1984.