*Technical Report*

UCAM-CL-TR-25
ISSN 1476-2986

Number 25

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

# A clustering technique for semantic network processing

Hiyan Alshawi

May 1982

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

*http://www.cl.cam.ac.uk/*

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

*http://www.cl.cam.ac.uk/techreports/*

ISSN 1476-2986

# A CLUSTERING TECHNIQUE FOR SEMANTIC NETWORK PROCESSING

Hiyan Alshawi

Computer Laboratory, University of Cambridge

Corn Exchange Street, Cambridge CB2 3QG, England

May, 1982

Prepared for the 1982 European Conference on Artificial Intelligence, Orsay, France.

## Abstract

The paper describes techniques for performing serial processing on the type of semantic network exemplified by NETL. They make use of an indexing scheme that can be based on semantic clustering. The basic algorithm is aimed at performing fast intersection operations. It is claimed that the scheme is suitable for its current application in text processing. The semantic criteria for clustering that have been tried are briefly described. Extensions of the scheme are suggested for use with large networks.

## 1. Introduction

This paper describes techniques for performing marker propagation processing on a certain class of semantic networks. The processing and data structures to which these techniques are applicable are exemplified by Fahlman's NETL system [1].

Fahlman [2] proposes a hardware design for performing such processing efficiently. Woods [4] makes use of a hypothetical parallel machine structure when describing some algorithms for KL-ONE. The software scheme described here is aimed at making this type of processing tolerably efficient for small and medium sized networks on conventional serial machines. It is based on an indexing method that uses clusters of nodes. An efficient scheme is needed for experimentation with network representations and for processing usefully large bodies of knowledge.

The basic scheme has been implemented for the memory component of an experimental text processing system. Suggestions are also made for extending the algorithm for operating on large networks.

## 2. Processing model assumptions

It is assumed that the network being processed has a small number of link types. A 'marker propagation' is an operation that takes an initial set of nodes and traverses the network from these nodes following links of some specified type, and marking, with a specific marker, all the nodes on the paths that are followed. We need to be able to selectively address the set of nodes marked with a given combination of markers. Such a set may form the result of a sequence of marker propagations, or the initial set for a marker propagation. Addressing the set corresponds to finding the

intersection of sets of nodes which possess different markers.

The structure that will be imposed by clustering is not part of the network representational formalism and is only concerned with matters of efficiency; thus it should not be confused, for example, with partitioning as used by Hendrix [3]. However, the efficiency issues being addressed are relevant to different representational formalisms that use the type of processing model dependent on marking outlined above.

Processing the network with markers tends to lead to performing a large number of operations of the following two types: marking a node; and testing a node, i.e. comparing the markers on a node with a given combination of markers. These will be referred to as 'primitive operations'. It is assumed that a small, constant, effort is required for performing primitive operations.

## 3. Intersection algorithm using clustering

Suppose we have partitioned the nodes of the network into clusters of a chosen fixed size. The nodes in each cluster are then linked to a newly created cluster node with special clustering links. The cluster nodes are then clustered themselves, and this is repeated until there is only one node, the 'apex', to be clustered. The resulting tree structure is a 'pyramid' with the network nodes as its base and successive clusterings forming higher levels parallel to it.

When any network node is marked, all the cluster nodes above it are also marked with the same marker. The network nodes with a given combination of markers can now be found by starting at the apex and following the combination down through the cluster nodes to the base.

Thus in order to reach the intersection set corresponding to the required combination we will only examine the network nodes in clusters that have a node marked with each of the markers in the combination. This means, in particular, that the number of network nodes examined will be bounded above by the cluster size times the number of nodes in the smallest marked set. On the other hand if the nodes in the marked sets happen to be unions of clusters then only the nodes in the intersection will be examined. An upper bound on the number of primitive operations needed for finding the intersection of a number of marked sets is O(m log N) where N is the number of nodes in the network and m is the size of the smallest marked set. However, this is only approached when m is small compared to N and the set is badly correlated with the clustering. It is possible for the number of operations performed by the algorithm to be proportional to the size of the intersection set, which of course is the minimum. How closely this optimal growth rate is approached, on average, depends on how the clustering correlates with the marked sets.

## 4. Some comparisons

Fahlman characterizes his hardware scheme as being able to perform, quickly, certain types of intersections. The design allows direct hardware addressing of sets of nodes with particular combinations of markers. It is not possible for a serial machine to perform marker propagations and addressing operations as quickly as the NETL machine, but cluster-based processing is suitable for simulating such parallel hardware.

Fahlman points out that the standard serial algorithms for finding the intersection of two sets represented as lists require an effort of at least O(n') where n' is the length of the shorter list. As just noted

cluster-based intersection may imply less effort.

The gain with clustering comes from excluding areas of the network that need not be searched. The size of the gain depends very much on how clustering correlates with marked sets. Different criteria for clustering that have been tried are described in the following section. Note that if an intersection is constrained further by another marker, which does not mark the smallest set, then performing the intersection is faster for the clustering algorithm but slower for the standard one. This behaviour of the clustering algorithm seems natural and desirable. In addition, distinguishing clusters which have all their members marked with a marker allows us to make use of the pyramid when searching for nodes satisfying marking conditions that are more general than simple intersections, but this will not be discussed here.

## 5. Clustering criteria

We would like marked sets to correlate with clustering as well as possible. The results of marking and intersection operations are not affected by the clustering methods used. Hence we need not worry here about the formal properties of a particular clustering but only the efficiency to be gained from it. In the implementation of the experimental memory component two 'semantic' clustering criteria were tried, and compared with clustering in a random manner.

In one semantic method the order in which the nodes are chosen for clustering is the same as the order in which the nodes are created. The majority of nodes in the network will be created as a result of processing input files written by the person who constructs the knowledge base. The

order of statements in these files tends to clump semantically 'close' information together, in a manner presumably related to the constructor's model of the knowledge being represented.

The other semantic method depends on so-called specialization hierarchies determined by links of certain types. The cluster order for nodes is the depth first search order starting from the top of the specialization hierarchy, of the subnetwork formed by these links. Specialization hierarchies are of course part of the representational formalism, but they have been used as the basis for clustering because they are heavily exploited during processing, in a way similar to the use of VC links for implementing virtual copies in NETL [1].

When a new node is added to the network it is assigned to a cluster. The choice of cluster is determined by a function that is consistent with the clustering method being applied. Clusters are allowed to grow to a maximum size, new clusters being created when this size is exceeded. It is not necessary to perform any special actions when a node is deleted. The whole network can be reclustered after a large number of updates has taken place.

The three clustering methods, 'creation', 'specialization', and 'random', have been tried with a small network database having about 200 nodes. The number of primitive operations that were executed when using the network was counted. This confirmed the expected result that random clustering was the least efficient, whereas specialization clustering led to the execution of the least number of primitive operations. The small size of the network means that the results of these test runs cannot support any strong conclusions. Note, however, that an appropriate semantic clustering

becomes more advantageous than random clustering as the size of the network grows. It proved to be fairly easy to determine a good initial cluster size experimentally. This was 8 for the test network.

## 6. Types of propagations

The way in which the network is used for text processing leads to a distinction between 'temporary' and 'context' markers. These can be characterized here as follows. Temporary markers are used transiently within a sequence of propagations, while context markers can participate in more than one such sequence. Thus a set of context markers can be intersected with the sets of nodes output by each of several sequences of propagations.

Unfortunately particular propagations using temporary markers may have to be repeated. In order to reduce the effort that this implies, the types of propagations that start at a single node and are expected to encounter a large number of branches are recorded at these nodes. These propagations are therefore performed only once.

During the processing of paragraphs of text we can expect to make frequent use of context markers, and usefully take advantage of recorded marker propagations. This means that the efficiency of performing network operations depends largely on how fast intersection sets can be reached. The algorithm based on clustering would therefore seem to be suitable for this application.

## 7. Cluster propagations for large networks

The scheme described applies to network processing where the dominant operation is intersection. However it can also be adapted to alleviate the cost, for very large networks, of marking large areas with temporary markers. Thus we could perform 'approximate' marker propagations on a high level of the cluster pyramid and then use the information gained in this way to perform restricted propagations, where necessary, on the base network.

This would require the introduction of 'cluster links' between the cluster nodes. Thus, if one or more pairs of nodes in different clusters are linked by some link type, then a cluster link of the same type would be created between the respective cluster nodes. Propagations that use these cluster links are approximate in the sense that a cluster link does not imply the existence of a network link between any two particular network nodes.

Then if temporary marking propagations are performed at the higher level, a desired combination of markers on a cluster node means that the nodes in that cluster might be in the desired set. The paths leading to the cluster nodes with this combination can then be traced back to the sources of the marker propagations. The network nodes corresponding to these paths would then be marked with special 'pass' markers. The propagations could then be performed on the base network only going through nodes that have pass markers. This would lead to marking all the nodes in the desired intersection, and should restrict marker spreading in the base network considerably.

## 8. Conclusion

The basic technique and its proposed extension for large networks are aimed at performing efficient marker propagation processing serially for NETL-like systems. The efficiency of the intersection algorithm depends on the clustering criteria applied. The techniques are suitable for simulating parallel hardware and for some AI applications.

## Acknowledgment

I would like to thank Karen Sparck Jones and John Tait for their helpful criticisms of earlier versions of this paper.

References

[1] Fahlman, S.E.

'NETL: A System for Representing and Using Real-World Knowledge'; MIT Press, Cambridge, Mass., 1979.

[2] Fahlman, S.E.

'Preliminary Design for a Million-Element NETL Machine'; Proceedings of the AAAI conference, 1980.

[3] Hendrix, G.G.

'Expanding the Utility of Semantic Networks Through Partitioning'; Proceedings of the Fourth International Joint Conference on Artificial Intelligence, 1975.

[4] Woods, W.A.

'Research in Natural Language Understanding'; Report No. 3797, Bolt Beranek and Newman Inc., April 1978.