205

/A FUZZY SEMANTIC NETWORK/

by

RON RAY HIGHTOWER

B.S., Kansas State University, 1983

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering and Computer Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1986

Approved by:

[illegible]

Major Professor

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Knowledge representation is a key issue in artificial intelligence and related fields of study such as cognitive science. The representation used by a computer system governs which processing operations can be used effectively. This, in turn, is related to the power and capability of the system.

Three main questions should be asked in evaluating a particular knowledge representation. How easy is it to create and later modify information in this form of representation? How well does the representation support processing? Finally, how well does the representation portray the knowledge it represents?

Conceptual graphs are a notation for knowledge representation. The notation was created by John F. Sowa (1984) in his book Conceptual Structures: Information Processing in Mind and Machine. This representation is easy to create, at least in small examples. While it does not support von Neumann style processing very well, it may be easily implemented on special machines (Fahlman, 1979). This representation provides an open-ended, powerful mechanism for representing knowledge.

However, conceptual graphs suffer from being too rigid; this is partly due to the fact that they have been developed for database representations. Entities in a database can be rigidly defined, but objects in the real world do not lend themselves to this luxury. To augment the capabilities of the conceptual graph representation, the methods of fuzzy set theory have been used in this work to create a new version of conceptual graphs.

This thesis defines a representation of "fuzzy conceptual graphs." Whenever possible the original methods of Sowa are used; nevertheless, sometimes the desire for fuzziness creates a need for new operations. To support fuzzy conceptual graphs, it becomes necessary to define a method for comparing graphs and a method for defining fuzzy concepts. Sowa (1984) calls the methods that support conceptual graphs a semantic network. The term "fuzzy semantic network" is coined for the methods that support fuzzy conceptual graphs.

The thesis is divided into three sections. Section one defines fuzzy conceptual graphs. Section two defines the supporting mechanisms of graph comparison and concept definition. Section three illustrates an example application of this fuzzy semantic network.

I. FUZZY CONCEPTUAL GRAPHS: DEFINITION AND MOTIVATION

After a brief description of conceptual graphs and fuzzy concepts, this section explores the nature of a fuzzy conceptual graph. These three topics set the stage for the remainder of the thesis.

Conceptual Graphs

A conceptual graph is a description of a scene or a situation, written with a graph notation consisting of nodes and links. The nodes represent objects or concepts, and the links represent relations between the concept nodes. The physical world seems to divide itself into discernible objects as do other environments such as character recognition, game playing, and system design. Conceptual graphs are a natural notation for this kind of environment.

In Figure 1 is an example of how a conceptual graph can be used to describe an object such as the character T. As per Sowa's convention, the concept nodes are drawn as labeled rectangles and the relation links are drawn as arrows with label-circles affixed to them.

The letter T has two main components, the vertical and horizontal line segments. Each of these pieces are echoed in the conceptual graph as concept nodes. The conceptual graph, which can be thought of as a statement about the letter T, points out that the two lines are joined together. The conceptual graph can also be read as saying that the alignment (ALGN) of one line segment is horizontal (HORZ) and the alignment or the other line segment is vertical (VERT). (The ALGN relation belongs to a special family of relations which act as place holders for attribute values.)

The conceptual graph in Figure 1 does not completely describe the letter T in the diagram. Additional relation links and concept nodes are needed for a complete representation. The horizontal line is above the vertical line and this information should be shown with an ABOVE relation connected with the two LINE concept nodes. The two lines are also perpendicular, which should be stated explicitly in the graph.

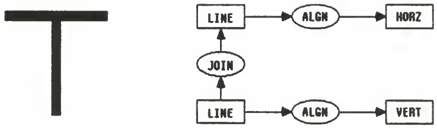

Figure 1. Character T and its conceptual graph

Another important fact is that the upper-end of the vertical line is connected to the middle of the horizontal line. Finally, it would be useful to roughly locate the two line segments within the framework of the letter by saying that the horizontal line is at the top and the vertical line is in the middle, between the left and right sides. All of these additional pieces of information would be necessary to provide a complete description. In general a conceptual graph can grow to be quite complex even when describing a simple scene.

Relation links usually connect only two concept nodes together but it is possible, and sometimes necessary, to join three or more nodes with a single relation link. However, a relation link must always connect at least two concepts, and can never be connected to just one concept node. In some ways the relation links used in the conceptual graph notation act like nodes of standard graph notation where only binary links are allowed. The constraint above insures against free-floating relation links and forces them to always exist between concept nodes as relations should.

The distinction between the concepts and relations seems fairly clear, but in some cases the two become interchangeable. The relation ALGN from the graph in Figure 1 relates a line segment with the value of the alignment attribute, but is not alignment a concept? Every relation is a concept, but it is not drawn as one until it is referenced in the graph description. An example of this is given in The Handbook of AI [see p. 182, Barr & Feigenbaum (1981)] concept node "BIRD" is connected with a NEST node via a relation of OWN. If the graph needed to describe when the bird owned the nest then the relation OWN became the concept node, OWN-1, and this node could be tied to a node with time and date values. In the process of referring to the OWN relation, it becomes a relationship and, therefore, a concept.

If it is possible to coerce an arbitrary relation into a concept, then it must be possible to find a common representation for both concepts and relations. Fahlman, in his design of a hardware-version of semantic networks, used a single element type to represent both concepts and relations [see p. 237 in Fahlman (1979)]. At some level of representation, concepts and relations are of very similar structures.

In the ensuing discussions, the emphasis will be on concept nodes and most examples will be of concepts. Anything said about the concepts should apply to the relations with little alteration. In the special situations where a distinction should be made, it will be discussed in some detail to insure completeness with respect to both concepts and relations.

Fuzzy Concepts and Relations

Conceptual graphs suffer from being too rigid in composition when used to describe scenes in the physical world. (Conceptual graphs were originally used to describe data base elements which can be rigidly described without fault.) Concepts and relations do not submit well to concise definition, as will be shown directly; this limits the usefulness of conceptual graphs in describing the physical world.

Suppose that we wish to generate a conceptual graph description of a line drawing, similar to Figure 1 but with many more lines. The first step would seem to be separating out the individual line segments, and curved lines, and labeling them ("concept label") in the graph as either LINE or ARC (for curved lines). It is necessary to have a definition for what will be accepted as a LINE and what will not be accepted. There should be some margin of error for how straight a line has to be in order to satisfy the definition of LINE, but how much margin? If an arbitrary cutoff point is used, then the definition will be too strict in some situations and too lenient in others. The answer to this problem is a fuzzy threshold which allows a gradation in the definition. As a line segment curves and becomes an arc, the change can be represented by a fuzzy measure which is a number from the interval zero to one. A perfect line would have the fuzzy measure of 1.0, and as the line becomes less-than-perfect the fuzzy measure approaches 0.0. Figure 2 shows a set of lines and curves with a possible set of fuzzy measures.

In graph form, the fuzzy concepts can be represented as a "concept label" and a "fuzzy measure." Because the graph is a description of a particular object, each component "concept label" refers to a particular element of that object. The fuzzy concept for that element describes 1) what name or label has been given to the element and 2) provides a fuzzy measure of how well the element matches with the label type.

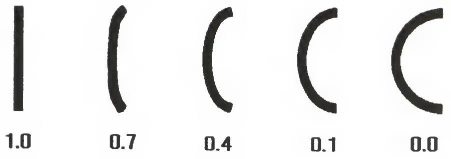

Figure 2. Examples of the "fuzzy concept" LINE.

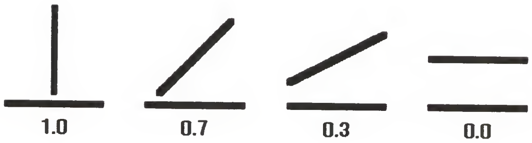

Figure 3. Examples of the "fuzzy relation" PERP (perpendicular).

In the normal conceptual graph only the "concept label" is used to describe an element and there is no measure of how well that label fits.

Relations can also be given a "fuzzy measure" which describes how well the "relational label" fits the particular scene being described. As an example of the fuzziness found in relations, we refer to the perpendicular (PERP) relation that might exist between two line segments.

In Figure 3, a set of lines is shown which represent the spectrum from perpendicular to parallel relationships. The perpendicular line segments are given the fuzzy measure of 1.0, while at the other extreme the fuzzy measure of 0.0 is given to the parallel lines segments.

In the discussion about fuzzy concepts, the fuzzy measure was related to how well a particular concept "matched" the concept type. Another way of viewing the fuzzy measure is as a distance of some sort between the particular and the ideal concept type. This "conceptual distance" implies a linear scale where particulars from the concept type can be placed for comparison. Figures 2 and 3 are drawn to represent some linear scales for the concept of LINE and the relation of PERP. The fuzzy measures at the bottom of these figures are not distance values. As Figure 4 shows, there is a mapping from the values of distance to the values used as fuzzy measures.

There are two reasons for the mapping from distance to fuzzy measures. The first reason is that the fuzzy interval (0,1) provides a common scaling for distances. Curvature will be measured in one set of units while perpendicularity is measured in another. Fuzzy measures allow for a standardization of distance measures.

The second reason for mapping from conceptual distances to fuzzy measures is the fact that conceptual distance does not always have a linear scale for comparisons to be made against. For high-level concepts, it becomes difficult or else meaningless to calculate a number and call it conceptual distance. How far is a tree from a bush and what are the units of distance? It is always possible, however, to create some function which gives a fuzzy measure. Fuzzy measure functions always represent an approximation to the conceptual distance whether it can be calculated or not. A fuzzy measure function can be based on

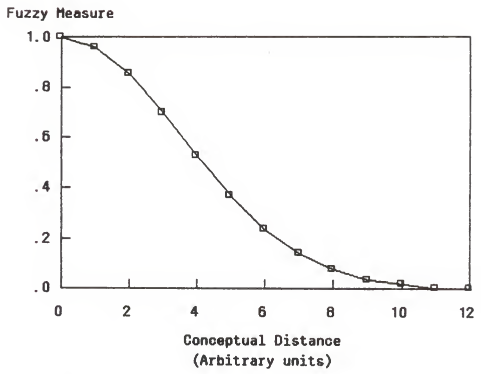

Figure 4. Mapping from the conceptual distance to the fuzzy measure.

psychological experiments rather than intuitive notions of what they should be. This makes it possible to use fuzzy measures as if they actually were a mapping from conceptual distance values.

What is the point of discussing conceptual distance, if all calculations and comparisons will be done in terms of the fuzzy measure? The idea of conceptual distance helps to provide a consistent dealing with fuzzy numbers during the design of comparison functions and combination functions. It is very easy to throw some numbers between zero and one around and call those numbers "fuzzy measures." By always relating the fuzzy measures back to the idea of conceptual distance at least an effort is made to give the fuzzy measures a consistent basis of meaning.

Fuzzy measure functions are usually created ad hoc, at least as far as research situations. Kurt Schmucker says, "It cannot be overemphasized that these definitions are biased decisions of the system designer. While it is hoped that they reflect the normal meanings given to the english terms they represent, there is much room here for disagreement." [see p. 28 of Schmucker (1984)]. This was in reference to the fuzzy set definitions of words rather than concepts, but the two are related. He later makes a similar remark about the operations that are performed on the fuzzy sets and how these operations may differ from those operations that people supposedly use (p. 38).

Although the fuzzy measures used in this paper are not perfectly correct in terms of experiment optimization, they suffice for demonstration purposes. The intuitive definition of a fuzzy concept, as presented above, can now be used to discuss what a "fuzzy conceptual graph" is and how it can be used.

Fuzzy Conceptual Graphs

Fuzzy conceptual graphs are simply conceptual graphs which use fuzzy concepts and fuzzy relations, described earlier, as the graphs' nodes and links. A fuzzy conceptual graph is still used to describe a scene or situation but the descriptive elements are now moderated with fuzzy measures. This will improve the act of graph matching, and graph matching is the basic operation performed on conceptual graphs.

One way of evaluating the graph matching ability of a system is to look at the set of entities from the data environment which can be described by a given graph. In Figure 5a the picture of a letter A has been described with a conceptual graph made by a graph generator. Figure 5b shows that many similar pictures from the character environment can be described with the same graph. In this case the graph is a normal conceptual graph with rigidly defined concepts and relations. This causes the set of acceptable characters to be a well-bounded group, and unfortunately some members of the set shouldn't belong while some non-members should belong.

Just as an aside, I point out that the test of a recognition system is ultimately whether we as human beings agree with what the system decides. It would be impossible to build a system which recognizes the exact same set of characters as a human being, so some errors are inevitable. There are two types of errors: accepting incorrect characters and not accepting correct characters. Trying to increase the number of recognized characters makes the system lenient and more incorrect characters are accepted. Trying to reduce the number of these incorrect acceptances will decrease the total number of characters recognized.

Fuzzy conceptual graphs make it possible to increase the number of recognized characters without necessarily accepting more incorrect characters. This is due to the ability to accept characters in a partial, fuzzy manner.

Figure 5c shows that fuzzy conceptual graphs describe a fuzzy set of characters from the character environment. Each character is a partial member of the set that can be described by a given graph. A fuzzy conceptual graph gives partial recognition over a larger set of characters. The advantage of this is that the final decision, of whether to accept or reject, can be made later when more information is available from other partial matchings. If the decision to accept or reject is forced early in the process of trying to recognize a character, then the effect of an error can propagate to other critical decision points, making recognition impossible. This is where the partial, fuzzy recognition of a character comes into the best use.

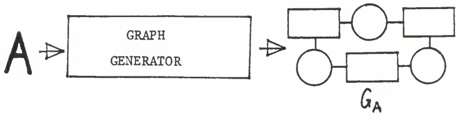

Figure 5a. Graph description of a character.

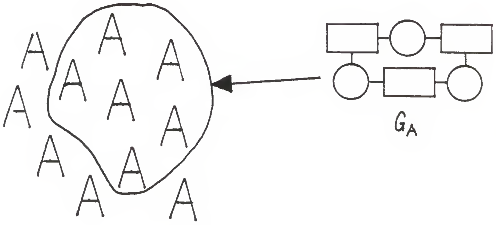

Figure 5b. Set of characters described by non-fuzzy conceptual graph.

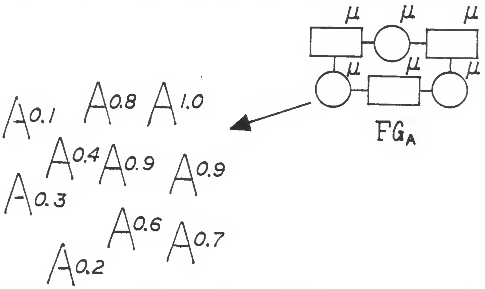

Figure 5c. Fuzzy set of characters described by fuzzy conceptual graph.

II. SUPPORTING MECHANISMS FOR THE FUZZY SEMANTIC NETWORK

The first part of this section explains the proposed method for comparing two graphs and calculating a fuzzy measure for the distance between them. This "graph matching algorithm" is then employed as the basis of a concept-definition method described in the second part.

Fuzzy Graph Matching

As previously mentioned, a conceptual graph is a description of an object or a situation. A basic operation on graphs is the comparison of one graph description with another. Pattern recognition, for example, is performed by matching graph descriptions of input data with graph descriptions of data patterns that are stored in memory. Graph matching provides a method for performing symbolic recognition, which can be used for performing functions, such as diagnosis and problem solving, in addition to vision and hearing [see pp. 52-53 of Fahlman (1979)].

The key problem in graph matching is that there will always be instances when the description of the input data does not exactly match the stored descriptions of known data patterns. What is needed is a graph matching algorithm which calculates a fuzzy measure of how well two graphs match, instead of just deciding that the graphs do or do not match.

Before suggesting a particular graph matching algorithm, it is worth noting that:

1) Graph components can have different degrees of importance.

2) Finding a correspondence between the components of two graphs is a difficult task in general.

3) The resultant fuzzy measure of comparing two graphs together should have a meaningful interpretation that is consistent with its later use.

Figures 6a through 6d show four characters and their respective descriptions in graph form. It can be seen by studying Figures 6b and 6d that some relations should be more important than others. The letter H and the broken letter H are very similar characters which can lead one to think that the JOIN relation, missing from Figure 6d, is not as important as other relations. Another unimportant relation is the relation PART which only acts as a place keeper for other concepts.

It will be necessary to weight the contributions of some concepts and relations to be less than others. The particular weighting values used will depend on the application area. In some situations the weighting values may change as a function of context or usage. The weighting of concepts and relations is a form of knowledge which is stored in the system by the system designer.

Graph matching is performed at a component-to-component level, and the results are then combined into one overall result. Typically one graph is used as a template and a second graph is matched against it.

The algorithm for graph comparison given here consists of three major steps:

1) A correspondence is found between the components of the template graph and those of the input graph.

2) The corresponding components of the two graphs are compared, and a fuzzy measure is calculated to represent how well each pair matches.

3) The collection of fuzzy measures for the pairs is combined into a single fuzzy measure, representing how well the input graph matches the template graph.

Finding a correspondence between graphs will be referred to as "conforming" in this thesis. The input graph will be conformed to the template graph. This will be represented as a mapping function from the components of the template graph to the components of the input graph. In general the template graph will be smaller than the input graph, and only a subgraph of the input graph will be matched.

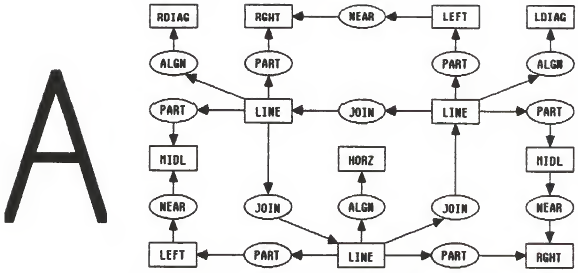

Figure 6a. Character A and its conceptual graph.

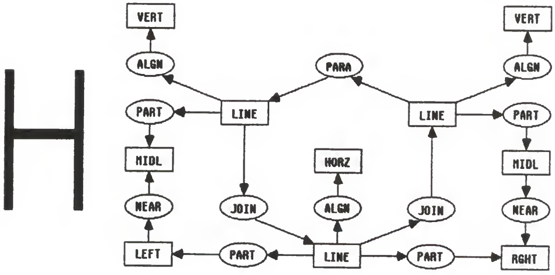

Figure 6b. Character H and its conceptual graph.

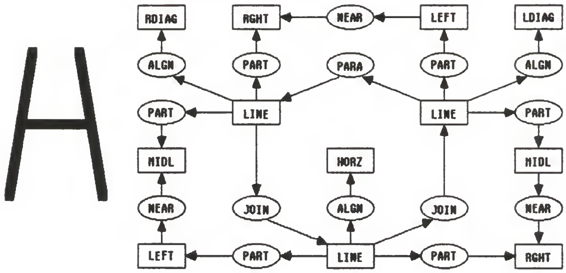

Figure 6c. Broken top A and its conceptual graph.

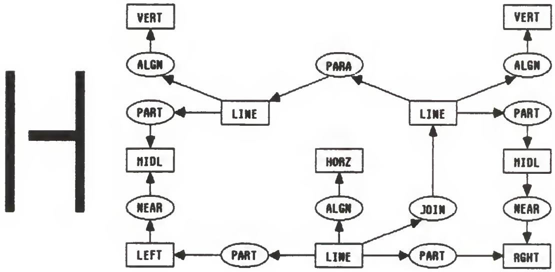

Figure 6d. Broken bar H and its conceptual graph.

The act of conforming two graphs is similar to the operation of "maximal joining" put forth by Sowa. A maximal join connects two graphs together so they can share information. That differs from this research, where the purpose is to recognize subgraphs. A maximal join is only effective if it joins the two largest subgraphs available within the graphs; and this is the same as the goal here. Sowa says, "For practical applications...the labels on the nodes reduce the combinations to a manageable number." [see p. 102 of Sowa (1984)]. Sowa then cites McGregor (1982) as a source of efficient algorithms for computing maximal joins. The issue of finding the mapping between graphs will not be discussed in this work. In addition to McGregor, Shapiro and Haralick (1981) are to be a source of algorithms for not only graph mappings but inexact graph mappings. (The inexact graph mapping is not made use of here, but in future work it might lead to a better fuzzy graph matching algorithm.)

A difference between the graphs used in the articles just mentioned and the graphs used here may prevent the application of those "efficient algorithms" in their current form. In fuzzy conceptual graphs, a node may have more than one valid concept label or concept type simultaneously. A particular node may be a LINE or an ARC at the same time, and the choice is dependent only on which makes a better fit. This problem should not increase the complexity of the algorithms to an unmanageable extent, but whatever algorithm is used it must be able to account for this interchange of concept labels.

The interchangeability of concept labels is not an entirely new situation. Sowa proposes a similar operation which is termed a "restriction" [see p. 94 of Sowa (1984)]. The restriction is a "vertical" change in type labels, meaning that the label moves from a higher, more abstract concept to a lower, more specific concept. The concept HUMAN could be restricted to the concept WOMAN in order to match a concept node of WOMAN in another graph. The interchange of labels used here will be of a "horizontal" nature, because the concepts will be on the same level of specification or abstraction. The change is different, but the effect is similar with respect to the graph conforming algorithm.

Sowa allows any number of restrictions prior to a join or maximal join operation. He has not pointed out that the need for a restriction would not be discovered until during the process of joining the two graphs.

Another problem, if the labels are so different, then how did the system know that the two graphs should be compared? The options of making hueristic guesses and comparing all graphs do not seem appropriate. Although the topic is not explored here, it would seem that an additional mechanism is needed to assist in choosing the correct concept label during the process of trying to match two graphs together.

Changing the label on a concept node is done to improve the match between two graphs. Changing the label of a node from the input graph to agree with a node label in the "template" graph (ideal concepts) also requires a change of the fuzzy measure for that node. For some sets of interchangeable concept types, it may be possible to calculate the new fuzzy measure in terms of the last fuzzy measure, for example, when the fuzzy functions are complements of each other. Usually it will be necessary to return to the original input data and calculate the new fuzzy measure. This may be costly to do for a high-level concept which, in turn, consists of many other low-level concepts, each requiring a fuzzy measure calculation.

After a mapping between the input graph and the template graph has been made, each pair of corresponding components are compared together. A fuzzy measure must be calculated for each pair; this indicates how well the component from the input graph matches the component from the template graph. More often than not, the fuzzy measures on the template graph will be 1.0 because the template graph will consist of ideal concepts. The fuzzy measure between the input and template nodes in this case simply will be the fuzzy measure on the input node, because the template node is the ideal concept. In cases where the template node has a fuzzy measure of less than 1.0, the fuzzy measure will be calculated by taking the absolute difference between input and template fuzzy measures and then subtracting it from 1.0. As an example, support the template has a note with the type of LINE with a fuzzy measure of 0.7 and the corresponding input node is 0.4 a LINE concept. The resulting fuzzy measure is given by:

$\mu(0.7,0.4) = 1.0 - abs(0.7 - 0.4) = 0.7$

This particular equation makes an assumption of monotonicity in the fuzzy measure, which will be true usually, and an assumption of linearity in the fuzzy measure will not be true usually. Fortunately the effects are worst at the smaller end of the fuzzy measure scale where the problem may not be serious.

Having found the component-to-component fuzzy measure for each pair of corresponding graph nodes it is now possible to combine them into a single, overall fuzzy measure. The overall combining function must satisfy a few constraints and desired behaviors. The output is a fuzzy number, for instance, and must exist on the fuzzy interval (0,1). Keeping in mind that certain concept types are more important than others, the following characteristics are desired from the combining function:

1) Predicates (i.e. concepts and relations) with low importance should not appreciably affect the overall comparison measure.

2) Predicates with high importance should affect the overall measure in proportion to their contribution to the graph description.

3) In combination the effect of low-importance predicates should be greater than the sum of these predicates had they occurred in isolation.

The method suggested here for combining the individual fuzzy measures is a weighted average. Every concept or relation type is given a weighting number reflecting its relative importance in the application area. This weighting of the concept and relation types is of a different nature from fuzzy numbers.

The suggested combining method does not satisfy the third desired characteristic listed above but still performs adequately. Before formally defining the Weighted Average Combining Function, however, it will be necessary to define some formal notation for fuzzy conceptual graphs. This notation is after Shapiro and Haralick (1981) and many of comments and observations in this section are echoed in that article.

The notation for Sowa's conceptual graphs can be mapped into their notation with little trouble and the result of combining the two notations is basically what follows.

```
     A fuzzy conceptual graph, Gx, is a pair
          Gx = (Cx,Rx)
where Cx is a set of concept nodes
          Cx = (Cx1,Cx2,...,CxM)
and RX is a set of relation links
          Rx = (Rx1,Rx2,...,RxK).

     Each concept node is a triplet,
          Cxm = (NCm,REFm,WCm)
where NCm is the name of the concept which labels the node,
REFm is the referent marker of the node,
and WCm is the fuzzy measure of how well the node's referent
satisfies the concept given by the concept name.

     Each relation link is a triplet,
          Rxk = (NRk,TUPLEk, WRk)
where NRk is the name of the relation type,
TUPLEk is an N-tuple of concepts
          (Cxi1,Cxi2,...,CxiN)
that the N-ary relation, Rxk, relates together,
and WRk is the fuzzy measure of how well the relationship named NRk is
satisfied by the concept nodes specified in the N-tuple TUPLEk.
```

In words then, a fuzzy graph is a set of concept nodes and a set of relation links where the relations connect the concepts.

Every concept node describes an individual object or concept and gives that individual an appropriate type label. The referent marker of the node can be thought of as a serial marker for every object in the universe. The idea of a referent (Sowa p. 85) makes it easier to distinguish between an individual and a concept label. If a certain man is the referent of a node he might be given the concept labels of MAN, HUMAN, FATHER, EMPLOYEE, etc. and yet he would always remain the same object. The act of labeling the universe is an imperfect process and in that act lies the roots of fuzziness. Thus it is natural for a concept to be a triplet of referent, label, and fuzzy measure.

The referent for a relation is the set of concept nodes that it joins together. For relations, identity is shown through the referents of the concepts specified in the TUPLE list. The relation link says that the referents of the specified concepts satisfy the relationship called NR, and the fuzzy measure specifies how well NR is satisfied.

The weighted average combining function can almost be presented now but some additional notation needs to be presented.

Recall that the first step in graph matching is finding a correspondence between components of the two graphs. For graphs Go and Gx, consisting of component predicates

$$Po = (Po1, Po2, \ldots, PoL)$$

and

$$Px = (Px1, Px2, \ldots PxL),$$

the correspondence is given by a mapping

$$h: Go \longrightarrow Gx$$

and so

$$h(Poi) = Pxi.$$

The predicates of a graph are all the concepts and relations used in that graph so the mapping includes both concepts and relations, but it is not allowed to map from concepts to relations and vice versa.

In general, the input graph, Gx, will have more components than the template graph, Go, and the inverse of mapping h will not exist for every component member of the input graph Gx.

In some cases a component from the template graph, Go, will not have a corresponding component in the input graph, Gx, and no candidate will exist for a type coercion. In these situations the mapping, h, will map the template node to a null value and then later during the fuzzy measure calculations, every mapping to a null value will automatically result in a zero fuzzy measure.

The final piece of required notation specifies the weighting functions for the concepts and relations.

Let WNC be the weighting function for concept types and let WNR be the weighting function for relation types. Both functions map from their respective set of type labels into the interval (0,1). The purpose of weighting concepts differently, again, is to reflect the relative importance some concepts and relations have over others.

The complete method of graph matching is given as follows. To compute the fuzzy measure of comparisons, u(Go,Gx), for the template graph Go and the input graph Gx, the procedure is:

1) Conform input graph Gx to template graph Go by making a mapping function.

   $h:Go \longrightarrow Gx$

   It may be necessary to do a vertical or horizontal restriction on some components to coerce the input graph to match the template graph. If no component can be found to correspond with a component of the template graph, then this component of the template will map to a null symbol. The act of restriction changes the fuzzy measure for the restricted node.

2) The fuzzy measure between each pair of the corresponding components must be calculated and the measure is given by:

   $\mu(Coi,Cxi) = 1.0 - abs(WCoi - WCxi)$

   where

   $h(Coi)=Cxi$,

   Similiarly for relations, we have:

   $\mu(Roj,Rxj) = 1.0 - abs(WRoj - WRxj)$

   where

   $h(Roj)=Rxj$.

   The special case of null mappings always evaluates to zero:

   $\mu(Poi,NULL) = 0.0$

   where Poi is Coi or Roj.

3) The fuzzy measures of the individual components are combined into a single, final result, using some combining function. The function proposed here is the weighted average combining function given by:

$$u(Go,Gx) = \frac{\sum_{Co} \mu(Coi,Cxi)*WNC(NCi) + \sum_{Ro} \mu(Roj,Rxj)*WNR(NRj)}{\sum_{Co} WNC(NCi) + \sum_{Ro} WNR(NRj)}$$

where

$Go=(Co,Ro),$
$Coi \xrightarrow{\subseteq} Co,$
$Roj \xrightarrow{\subseteq} Ro,$

and

$h:Go \longrightarrow Gx,$

then

$h(Coi)=Cxi,$
$h(Roj)=Rxj.$

(For clarity, note that the template graph has M concepts and K relations, or M+K predicates.

$Poi=Coi$ for $i=1,2,\dots,M$ and
$Poi=Roj$ for $i=M+j,\dots,M+K$ where $j=i-M$.)

An Example of Fuzzy Graph Matching

To demonstrate this function we compare the input graph for the "broken top" letter A to the template of the character H, shown in Figures 6c and 6b, respectively.

The graph of the template is:

```
Go = [
       Co = [
              Col  = ( LINE,      001, 1.0 )
              Co2  = ( LINE,      002, 1.0 )
              Co3  = ( LINE,      003, 1.0 )
              Co4  = ( MIDDLE,    014, 1.0 )
              Co5  = ( LEFTEND,   015, 1.0 )
              Co6  = ( MIDDLE,    016, 1.0 )
              Co7  = ( RIGHTEND,  017, 1.0 )
              Co8  = ( VERT,      028, 1.0 )
              Co9  = ( HORZ,      029, 1.0 )
              Co10 = ( VERT,      030, 1.0 )
            ]

       Ro = [
              Rol  = ( JOIN, (Col,Co2), 1.0 )
              Ro2  = ( JOIN, (Co2,Co3), 1.0 )
              Ro3  = ( ALGN, (Co1,Co8), 1.0 )
              Ro4  = ( ALGN, (Co2,Co9), 1.0 )
              Ro5  = ( ALGN, (Co3,Co10),1.0 )
              Ro6  = ( PART, (Co1,Co4), 1.0 )
              Ro7  = ( PART, (Co2,Co5), 1.0 )
              Ro8  = ( PART, (Co2,Co7), 1.0 )
              Ro9  = ( PART, (Co3,Co6), 1.0 )
              Ro10 = ( NEAR, (Co4,Co5), 1.0 )
              Ro11 = ( NEAR, (Co6,Co7), 1.0 )
              Ro12 = ( PARA, (Co1,Co3), 1.0 )
            ]
     ]
```

In comparing the graphs in Figures 6c and 6b we note the following differences. The input graph, Figure 6c, contains the extra concepts RGHT and LEFT and the extra relations PART, NEAR, and another PART. Since these are not part of the template graph, Figure 6b, they will not affect the comparison of the graph. The other differences between the graphs include mismatches between VERT and RDIAG, VERT and LDIAG, and a perfect PARA (parallel) relation against a 0.77 PARA relation.

Conforming the diagonal concepts into vertical concepts requires a change in the fuzzy measure. The functions used in calculating the fuzzy measures for base concepts and relations are given in Appendix B. In this case the lines are 8.5 degrees from the vertical or .149 radians from being vertical. The equation is

```
F = 1.0 - 2.594*A**2
```

This yields the fuzzy measure as 0.942.

In conforming the input graph, Gx, to the template graph, Go, in this example, the only non-identity mappings are:

$$\begin{aligned} h:\ & Co8(VERT,1.0) \longrightarrow Cx8(RDIAG,1.0) \\ & Co10(VERT,1.0) \longrightarrow Cx10(LDIAG,1.0) \\ & Ro12(PARA,1.0) \longrightarrow Rx12(PARA,0.77), \end{aligned}$$

After conforming the graphs these mappings become:

$$\begin{aligned} h:\ & Co8(VERT,1.0) \longrightarrow Cx8(VERT,0.94) \\ & Co10(VERT,1.0) \longrightarrow Cx10(VERT,0.94) \\ & Ro12(PARA,1.0) \longrightarrow Rx12(PARA,0.77). \end{aligned}$$

The following values are used for the weighting functions WNC and WNR:

Concepts	WNC	Relations	WNR
LINE	0.95	NEAR	0.95
LEFT	0.90	ALGN	0.90
RGHT	0.90	PERP	0.90
MIDL	0.90	JOIN	0.40
VERT	0.90	PARA	0.30
HORZ	0.90	PARA	0.30
RDIA	0.90		
LDIA	0.90		

The second step in graph matching in computing the fuzzy measure between each corresponding pair of predicates. This fuzzy measure is given by

$$1.0 - \mathrm{abs}(WCoi - WCxi)$$

as previously stated. In our example, however, the weights on all the concepts and relations are 1.0 and the equation above becomes

$$1.0 - \mathrm{abs}(1.0 - WCxi)$$

or

$$1.0 - (1.0 - WCxi)$$

and finally

$$WCxi.$$

Typically, the concepts in the template graph are representatives of ideal concepts.

Having calculated the individually fuzzy measures of the graphs, the third and last step is using the combining function to generate the overall, single fuzzy measure.

The two terms in the numerator are summations of products over the template graph. The first term is the summation of individual concept fuzzy measures times the weighting function for these concepts. The second term is the summation of individual relation fuzzy measures times the weighting function for those relations. For the twenty-two predicates in the template graph, Go, here are the fuzzy measures from Gx and the weighting function values:

Concept	Name	μ(Coi,Cxi)	Weight	Relation	Name	μ(Roj,Rxj)	Weight
Co1	LINE	1.0	0.95	Ro1	JOIN	1.0	0.40
Co2	LINE	1.0	0.95	Ro2	JOIN	1.0	0.40
Co3	LINE	1.0	0.95	Ro3	ALGN	1.0	0.90
Co4	MIDL	1.0	0.90	Ro4	ALGN	1.0	0.90
Co5	LEFT	1.0	0.90	Ro5	ALGN	1.0	0.90
Co6	MIDL	1.0	0.90	Ro6	PART	1.0	0.30
Co7	RGHT	1.0	0.90	Ro7	PART	1.0	0.30
Co8	VERT	0.94	0.90	Ro8	PART	1.0	0.30
Co9	HORZ	1.0	0.90	Ro9	PART	1.0	0.30
Co10	VERT	0.94	0.90	Ro10	NEAR	1.0	0.95
				Ro11	NEAR	1.0	0.95
				Ro12	PARA	0.77	0.90

The first term is found by taking the product of the two columns for concepts and then summing those together; the same procedure is used on the two columns for the relations. The numerator is given by

$$9.0465 + 7.293 = 16.339$$

The denominator is the sum of all the weights, both concepts and relations, without multiplying them by anything. The denominator is given by

$$9.15 + 7.5 = 16.65$$

The overall fuzzy measure for how well graph Gx matches graph Go is given by

$$16.339/16.65 = 0.981$$

This fuzzy measure becomes lower as the input character increasingly differs from the template character.

The graph in Figure 7 shows the results of two experiments. The letter A is slowly transformed into the letter H. The intermediate stages are put into graph notation and given to the graph matching algorithm. In the first experiment the graph for the letter A serves as the template, and in the second experiment the graph for the letter H is used as the template.

Along the bottom of the graph in Figure 7 are pictures of the letter A as it transforms into the letter H. The vertical axis shows the fuzzy measure calculated by the graph matching algorithm. The solid line represents comparisons with the template A and the broken line represents comparisons with the template H.

The intermediate stages all seems to make a good letter H in comparison with making a good letter A. However, the numbers in general seem too high. Should not the perfect letter H make a 0.0 comparison with the template for the letter A and vice versa? Not necessarily. The graph matching algorithm shows that from the entire universe of objects that can be described with conceptual graphs, the letters A and H are very close. In a character recognition system, when the whole universe is not of concern, it would be necessary to rescale the fuzzy measures from the graph matching algorithm.

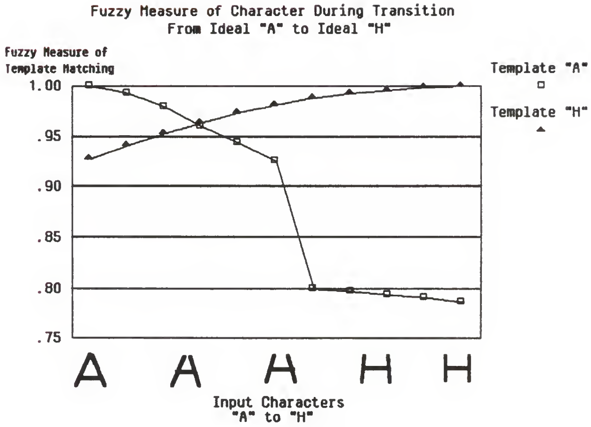

Figure 7. Graph matching experiment

One more comment should be made about this graph. The curve for the letter A template shows a discontinuity just to the right of the center. The reason for this discontinuity is that a large segment of the graph suddenly disappears as the transformation moves from left to right. Looking back at Figures 6a and 6b we note that five predicates at the top of the graph for A are not present in the graph for H. The center predicate, the relation NEAR, is the reason for the other four predicates to exist. During the transformation the relation NEAR goes from 1.0 to 0.0. When the graph generator can no longer find the relation NEAR, it ceases to include the other predicates in the graph. This causes a sharp drop in the fuzzy measure calculated by the graph matching algorithm.

The problem could be easily fixed in this case, but that fix would only apply to this application area. It may be impossible to have a graph matching algorithm which can be generalized to work in any application area. Special enhancements for the graph matching algorithm may be needed for every area of application.

Fuzzy Concept Definition

In the section on definition, a fuzzy concept is defined as a concept label and a fuzzy measure. The fuzzy measure indicated how well the object matched the concept's label. The graph matching algorithm calculates a fuzzy measure of how well one graph description fits another graph description. If a concept can be expressed as a conceptual graph then the graph matching algorithm can be used to calculate the fuzzy measure of this concept. This makes it possible to define concepts in terms of "graphs."

A single graph could be used to define an entire class of objects. This resembles a paradigmatic symbol [see p. 443 of Watanabe (1985)] which is a single example of a class of objects used to define the class to which it belongs. Watanabe calls pattern recognition "a process of seeing many-in-one", in the sense of giving a class label to a single individual.

As an example of concept definition, let's define the concept of the letter T. Figure 8 shows the ideal letter T and the graph description of it. The graph is, in fact, the definition of the concept T. To decide if an input graph represents the letter T, we simply use the fuzzy graph matching algorithm and calculate the fuzzy measure. The fuzzy measure tells us how well the input fits the concept definition. We can use it to make the decision. If the T occurs in isolation then an arbitrary threshold can be used to make the decision. If the T is part of a word, then the decision can be based on what the word might be and if it should contain the letter T or not. This is the advantage of fuzzy measures; decisions can be postponed until later when additional information is available.

A single graph can define a concept, but sometimes it may be desirable to use more than one graph in a definition. This is due to the that a concept may contain some major divisions within its boundaries. Each of the major divisions should have a template in order to improve on recognition ability. An example of this could be the concept of letter T which has upper and lower case Ts within the same group. A single template might be able to define both kinds of T, but neither type would match the template perfectly. By giving each T a separate template, both could be matched perfectly every time. Other examples exist, such as male HUMANs and female HUMANs, coniferous TREEs and deciduous TREEs, White HOUSE and shack HOUSE, etc., where a single concept has multiple incarnations.

The second reason for using multiple templates for concept definition is that an object may remain the same under certain transformations and multiple templates could be used to indicate this. Rotating a letter on the pages does not really change the letter's identity but the computer cannot know this unless it is told. One way of telling it is to allow certain operations on the graph prior to recognition, such as rotation. It may be easier, however, to simply store multiple orientations of the letter together as one definition. We would resort the fuzziness of the graphs to bridge the gaps between different views.

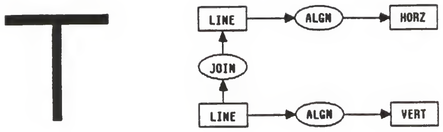

Figure 8. Character T and its conceptual graph.

When a concept is defined with more than one template, thereby forming a cluster of templates, the graph matching algorithm will be used to pick the best template. The template which has the best match to the input graph will be the template used. The fuzzy measure produced by the graph matching algorithm will be the fuzzy measure of the concept.

Sometimes a template used to define a concept will not be the ideal graph. To match perfectly with a half-ideal example is still only to be half-ideal. Because of this the templates in the cluster will be given weighting values. Half-ideal example templates will have a weighting value of 1.0 while the other less-than-half-ideal templates will have smaller weighting values. The weighting values will be used to limit the final resultant fuzzy measure when the input graph is a good match with a less-than-ideal template.

Formally, we state that a cluster G is a set of template graphs, i.e.

$$G = (G1, G2, \ldots, GN)$$

a weighting function over the graphs,

$$GW(Gi) \longrightarrow (0,1)$$

A concept definition function is used to calculate how well an input graph matches the concept defined by the cluster of templates. This function is given by:

$$\mu(G,Gx) = \max\left(\min_{i=1,2,\ldots,N} \left(\mu(Gi,Gx) , GW(Gi) \right) \right)$$

where

Gx = input graph,
G = the template cluster

and

Gi = ith member,
GW(Gi) = the template weighting function,
μ(Gi,Gx) = the graph matching algorithm applied to input Gx and the ith cluster template, Gi

Figure 9 shows three example characters to be used as templates in defining the concept T. The graphs describing these examples will not be shown but the cluster is a cluster of conceptual graphs. Underneath each character in Figure 9 is the value of the template weighting function for each character template. More specifically, GW(G1) = 1.0, GW(G2) = 1.0, and GW(G3) = 0.7 where G1, G2, and G3 are the graphs for the characters in Figure 9, left to right, respectively.

Figure 10 shows the input character from which a input graph is generated. This input graph is matched against all three of the template graphs using the graph matching algorithm. The results are given by:

$$\mu(g1,Gx) = 0.8$$
$$\mu(G2,Gx) = 0.92$$
$$\mu(G3,Gx) = 0.97$$

Using these values and the template weighting values, the concept definition is evaluated as:

$$\begin{aligned}\mu(G,Gx) &= \mathrm{Max}(\ \min(0.8,1.0)\ ,\ \min(0.92,1.0)\ ,\ \min(0.97,0.7\ )\ )\\ &= \max(\quad 0.8 \quad , \quad 0.92 \quad , \quad 0.7 \quad)\\ &= \ 0.92\end{aligned}$$

Note how the input graph, Gx, matches very well with the last template; nevertheless because G3 is not a perfect example of the letter T, it has a ceiling on how well it can be matched. The template weighting values act as limits for those templates which are less-than-ideal examples.

As mentioned earlier, there are two reasons to include more than one template in a concept definition. The last example shows the situation where one concept, the concept of the letter T, really has two different forms which cannot be correctly described with a single template. The second reason is to maintain the capability of recognition throughout the process of transformations on a graph. Certain transformations (e.g., rotation) change the graph-description of an object. This makes the object unrecognizable to the computer system unless it is told how to handle the transformation. One way to store the information on transformation is to make a template graph for intermediate stages of the transformation. In the case of rotation a different template could be stored for an objective every 45 degrees around a full, 360-degree rotation. The fact that each template defines a fuzzy set makes it possible to bridge the gaps between the separate templates.

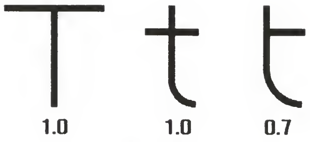

Figure 9. Three templates for the concept T.

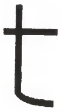

Figure 10. Input letter T

Representing this kind of information with a cluster may be inappropriate in certain situations. If the template is a complicated structure, processing several such structures will greatly slow down the system during matching. If a great number of templates are required to represent the transformation, e.g., as rotations in three-dimensional space, then the cluster will again be too large for reasonable processing. In some applications, it will be necessary to build graph transformations into the system for the sake of performance.

In Section I, the fuzzy conceptual graph has been defined. Fuzzy conceptual graphs are a straightforward combination of fuzzy set ideas and the graph notation of conceptual graphs. Fuzzy conceptual graphs should overcome the limitations of regular conceptual graphs which are well suited for rigid, database applications.

Section II introduces the graph matching algorithms and the concept definition cluster. These two collectively support the processing of fuzzy conceptual graphs. According to Sowa, "A conceptual graph has no meaning in isolation. Only through the semantic network are its concepts and relations linked to context...and perception." [see p. 77 of Sowa (1984)]. Sowa uses the term "semantic network" to mean the underlying relations and procedures that give conceptual graphs their meaning.

In this thesis, the relationships provided by fuzzy concept definitions, and the procedure of graph matching, give meaning to fuzzy conceptual graphs. Therefore, fuzzy concept definitions and the procedure of fuzzy graph matching by definition is a "fuzzy semantic network."

III. AN EXAMPLE APPLICATION OF A FUZZY SEMANTIC NETWORK

An example of character recognition is provided to show that fuzzy conceptual graphs improve recognition tasks and especially recognition-in-context tasks. The main operation is recognizing portions of the input graph and substituting a concept node in place.

Perception as Graph Matching

Perception can be defined as "the mental capture of objects through the senses." This definition can be altered when defining computer perception by substituting "input data" for "senses" and "internal representation" for "mental capture." This "internal representation of objects from input data" will be accomplished using conceptual graphs.

In some data environments objects are decomposable into recognizable parts called features. Recognizing or perceiving an object can be done by recognizing the pieces and then recognizing the group. It may be possible to find features which are decomposable into smaller, lower level features. Eventually decomposition of features must stop at the lowest level of base features.

In these terms, recognition begins with the lowest level features, attempts to find groupings, representing higher level features, and on up until the entire set of input data is recognized. This model is not psychologically complete. That is, it is not sufficient to explain human perception. This model, or a facsimile thereof, is discussed in great detail in Human Information Processing [see p. 274 of Lindsay and Norman (1977)]. The authors say, "...the type of feature analysis that we have just been performing is absolutely essential to the perceptual process, but by itself is not sufficient."

The fuzzy semantic network can be used to execute this model of perception. The briefest description of the perception model is given in four steps:

1) The camera or the eye transforms the world into input data.
2) Base-feature detectors convert (most of) the input data into concept nodes, waiting to be pieced together.
3) The concepts nodes are assembles together after the relations between them have been found. The result is a lower level graph description.
4) Portions of the graph are recognized and replaced with higher level concepts. This step is repeated until the graph is completely recognizable as a high level concept.

Some of the details of the sketchy description will be filled in while other matters will simply be discussed in terms of possible solutions.

Input Data and Base-Level Features

The object environment for this example is the domain of english alphabetic characters. The characters have been constrained to lines, portions of circles, and combinations thereof. All lines are of the same width and no serifs are allowed. The input data takes on the form of a bit image with reasonable resolution (20x20 to 40x40 pixels per character).

To compensate for this over-simplified situation, the only feature that is detectable is a long segment. Line segments are constrained to vertical, horizontal, and diagonal orientations as far as being recognizable. That is to say, except for very long lines, only the four orientations will be recognized. Anything else will be noted for future reference but not recognized as a line. Lines must also exist within a certain constraint of curvature to length ratio. The main point of these constraints is to show that with limited base-level support, the system can still recognize features and characters when required. The object and data environments have been constrained to provide a good showcase for the fuzzy semantic network!

Base-Level Relations and the Assembler

After the base-level features have been detected, they become concept nodes, ready to be put into a graph. At this state they retain some primitive information in numerical form, such as the (x,y) coordinates of the endpoints and middle of the lines. This information is used by the assembler to decide if any of the base-level relations are present.

For example, the base-relation of NEAR is given by the equation:

$$\mu_{\text{NEAR}}(\text{line1},\text{line2}) = \begin{cases} 1.0 & r < 0.1 \\ 1.0 - 5*(r-0.1) & 0.1 < r < 0.3 \\ 0.0 & r > 0.3 \end{cases}$$

where r is the ratio defined by

$$r = \frac{\text{the distance between lines 1 and 2}}{\text{length of line 1}}$$

The distance between lines 1 and 2 really means the distance between a point on line 1, such as the endpoint or midpoint, and a point on line 2. The distances are calculated in terms of the (x,y) information which has been retained for this computation. This function and the similar functions of JOIN, PARA, PERP, and ALGN have all been created ad hoc for this constrained enviroment. The functions could be redone on the basis of psychological experiments to provide them with a basis in reality, if that is a desired characteristic.

The search for relations must be a guided procedure; simply too many relations exist to be found by brute force. The endpoint of a line cannot be compared against all other endpoints to decide if it is NEAR or not. Similiarly, not every pair of lines can be compared for being parallel (PARA) or perpendicular (PERP). The search for these relations must be data driven and selective. Not every relation is relevant, even if it is true. If certain relations are ignored at this stage of processing, a complete system would allow for a return to this level of data in case a missing relation is needed at a higher level. This backtracking will not be shown here but the necessity is pointed out as a direction of future research.

The assembler gives rise to base-level graphs [see p. 34 of Sowa (1984)]. These graphs are basically a line-by-line description of the picture/input data. The computer can now see the input data, but it has not seen anything in the data yet. Perception begins here.

Subgraph Matching and Replacement

The next stage in the process involves recognition. As noted previously, recognition can be characterized as matching input data descriptions with stored data descriptions. The base-level graph is the input description and the stored descriptions will take the form of the concept definition clusters.

Instead of describing the process in a wordy manner, let us work an example during each matching and replacement. The input data is displayed in Figure 11.

The difficult part of interpreting this input data is the replacement of rounded characters with block characters (the Q and the U). Eventually the system should be able to match the lines that look like a Q with the Q template and the lines that look like a U with the U template, etc. Figures 12a and 12b show a major part of these templates.

The main difficulty resides in the concepts of CIRCLE and HALF-CIRCLE graphs which do not exist in the input graph. At this point the input graph consists of nothing but nineteen line concepts tied together with over one hundred other predicates. This graph will not appear in this thesis except for relevant subsections.

The features of CIRCLE and HALF-CIRCLE will have to be recognized as groups of lines. The graph for a concept of a circle is shown in Figure 13. Since this is really a graph for an octagon description, as shown at the bottom of the figure, it has a template weighting only 0.8 in the circle cluster. It is also the only member of that cluster.

Only half of the boxes in the conceptual graph (Figure 13) were labeled. Note that the ring has been evenly divided into eight equal sections, alternating labeled and unlabeled sections. The reason for the distinction will be explained in terms of the pictures in Figure 14.

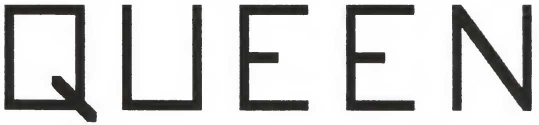

Figure 11. Picture of the input data "QUEEN".

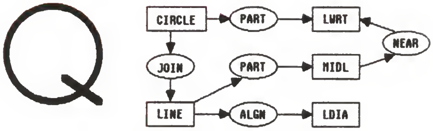

Figure 12a. Template for the Q concept.

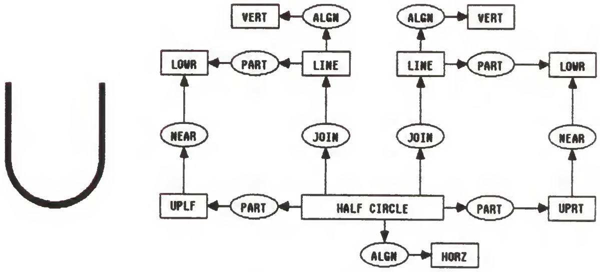

Figure 12b. Template for the U concept.

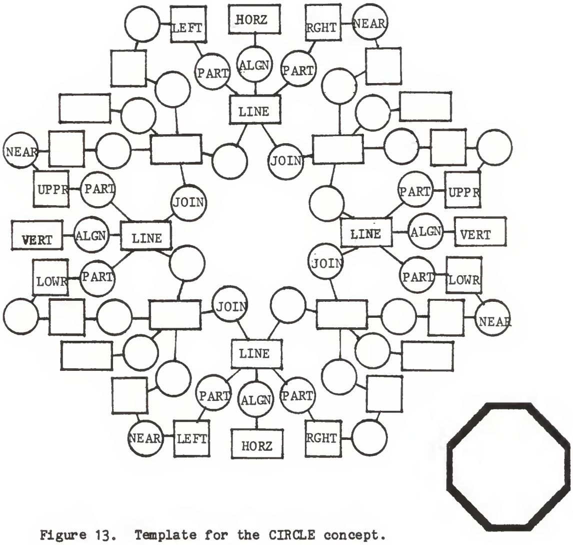

Figure 13. Template for the CIRCLE concept.

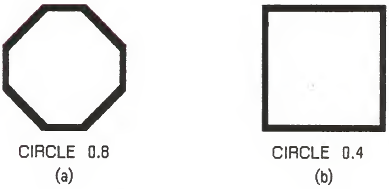

Figure 14. Two partial matches to the CIRCLE concept.

The labeled parts of the graph in Figure 13 correspond to the vertical and horizontal lines of the octagon in Figure 14a. These are also the parts of the template that can be mapped into the graph description of Figure 14b. The square in Figure 14b matches only one half of the template of Figure 14a. The parts that match together are the labeled portion of the graph in Figure 13.

The square in Figure 14b represents a portion of the input graph. The square is part of the bad Q of Figure 11. We wish to match this square against the circle definition in order to calculate how well it matches the circle concept. This can then be used to calculate how well the badly formed Q matches the concept Q. This will lead, eventually, to calculating the match of the entire input graph against the QUEEN concept.

Using the graph matching algorithm from the previous section, the fuzzy measure of fit between square and circle template is calculated. In this case, there are no partial matches and the fuzzy measure can be found by dividing the sum of the predicate weights in the square by the sum of the predicate weights in the template, which is

$$20.6/52.0 = 0.3962.$$

The square approximates a circle to a degree of 0.4.

The idea is to match portions of the input graph to feature templates and then substitute the appropriate concept node into the graph. Three major obstacles prevent this from being an easy operation. How does the system know to match the square against the circle definition? How does the conforming part of the graph matching algorithm know not to include extraneous pieces during matching, such as the line on the character Q? How does the replacement algorithm know where certain arcs are to be connected or disconnected? In this example, all these things can be handled with a bit of hand waving, but for general purpose use, these problems will have to be studied in greater detail for an answer.

Question one: "How did the system know to compare the square from the input graph with the circle concept definition?" Focusing the search requires 1) localizing or narrowing the portion of the input graph that is being searched, and 2) narrowing the number of concept definitions to be compared in the "knowledge base."

In this application the input graph is fairly easy to segment into a character by character recognition problem. In other applications, this will become a major problem of control for the perception algorithm.

Narrowing the search in the knowledge base can be accomplished by grouping concept definition together in various ways to help direct the search. Characters could be separated from the feature concepts. Words could be separated from the characters. Another approach would be to group objects according to similarities, such as all the letters that have two vertical lines (ADHMNOQUVW). The groups could be given a graph that represents the similar features within the group. This small graph would be a prototype of sorts and could be used to reduce the search time by a great deal.

Question two: "How does the conforming part of the graph matching algorithm know not to include extraneous parts of the input graph during matching?" In the example, the matching algorithm tries to make a correspondence between the diagonal bar of the Q and the diagonal lines in the octagon. In this case the diagonal directions of the two lines are exactly opposite and this prevents a match. This occurred only because the graph matching algorithm used knew that the LDIAG concept and the LINE concept were intimately tied together. This little bit of knowledge came from the application area and represented a special heuristic for the matching algorithm. This is another case where the graph matching algorithm seems to require special augmentation from the particular application area.

Question three: "How does the replacement algorithm know where to connect or disconnect concepts and relations?" This is a difficult problem but there are two possible solutions. The first involves a complicated set of rules for combining LINE concepts together, replacing concepts such as LOWR and RGHT with LOWER-RIGHT, eliminating redundant relations, etc. This would require an expert system of sorts just to substitute subgraphs back into complicated positions of the input graph.

The second method involves a return to the original input data. At this level everything can be marked with the name of the concept to which it belongs. This returns the system back to that part of the perception algorithm just prior to the assembler being the first graph. Instead of only lines, though, the assembler is also given circles and half-circles and other features to work with. The assembler builds the base-level graph from scratch which solves the problem of where to connect which link. Neither solution is without fault; this is another area where future work could be useful.

Continuing with the example, we assume that the square has been matched with the circle to a fuzzy degree of 0.396. Replacing the appropriate portion of the graph with the concept circle makes the input graph, or at least the Q part of it, look like the Q template in Figure 12a. The only difference between the input graph and the Q template is the 0.396 fuzzy measure on the concept node for CIRCLE.

Using the graph matching algorithm, it is possible to calculate how well the input graph matches the Q template. The ratio of weighted fuzzy measures is

$$6.876/7/45 = 0.923.$$

The system knows that the input graph contained the letter Q and the Q concept node replaces the entire subgraph.

A similar process is used to recognize the other letters in the input graph. The two Es and the N matched perfectly, 1.0, but the letter U matched only 0.967.

Figure 15 shows the template graph for the word "QUEEN." The concept and relation weightings are all 0.9. The final fuzzy measure, for the entire input graph, is given by:

$$\frac{(0.923)(0.9) + (0.967)(0.9) + (0.9)\ (7)}{0.9\ (9)} = 0.987$$

In other words, the final fuzzy measure is 0.987 for recognizing the word "QUEEN."

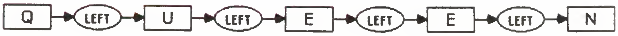

Figure 15. Template for the "QUEEN".

As a counter-example, the "system" was given a similar word but with a completely different first letter, such as "XUEEN." Assuming that the first letter completely mismatched the Q template and had a fuzzy measure of 0.0 still allows the final fuzzy measure to have a value of 0.889.

A test word of complete different letters, such as "XXXXI" has a fuzzy measure for the concept "QUEEN" of 0.444.

These numbers all shows that the fuzzy measure is higher than might be expected for a character recognition system. It should be pointed out, again, that these measures reflect differences between words on a scale that includes the entire universe of objects. A second point is that it does not matter how large the numbers are as long as the largest one belongs to the correct answer.

CONCLUSION

In this thesis the conceptual graph notation of Sowa has been modififed to make use of fuzzy measures in concept labeling. A graph matching algorithm is outlined for comparing graphs in the new notation and the basic method for defining fuzzy concepts is presented. The collection of structures and operations I have termed a "fuzzy semantic network."

The fuzzy conceptual graphs, based on the new notation, should improve the outcome of any symbolic recognition task. Symbolic recognition based on high-level or multiple-level encodings of the input data instead of using it directly. By encoding the data into higher-level representations, the system can do recognition in terms of only the relevant data features. Fuzzy conceptual graphs can represent the input data as high-level features and the fuzzy measures make it possible to record how well a high-level feature fits the data. Regular conceptual graphs can encode higher-level features also but there is no measure of how correct this encoding is. Fuzzy conceptual graphs combine high-level symbolism with moderating effect of fuzzy measures.

The graph matching algorithm of Section II. makes the fuzzy graph nottion into a useable notation. Matching graphs is a fundamental operation in recognition, graph generation, and concept definition. There is also potential uses for graph matching in learning models, problem solving and behavior modeling. Eventually other operations on fuzzy conceptual graphs could be needed; especially application area oriented transformations. For this semantic network, however, the graph matching algorithm was sufficient.

Fuzzy concepts were defined in this thesis using graph descriptions. The definition graphs (templates) consisted of feature concepts which could be decomposed into lower-level feature concepts. Eventually everything was defined in terms of the lowest feature, the line segment. Because every concept was based in the sensory domain of the system, the symbols used in the conceptual graphs were connected to reality.

I see this as a step toward the "paradigmatic symbol" that humans have, as opposed to the "empty computer symbols" that are devoid of any meaning [see p. 447 of Watanabe (1985)].

This "fuzzy semantic network", then, is meant to capture some of the fuzzy concept processing that provides power to human pattern recognition abilities.

BIBLIOGRAPHY

Barr, Avron and Feigenbaum, Edward A. The Handbook of Artificial Intelligence, Volume I, William Kaufmann, Inc., Los Altos, California. 1984

Cohen, Paul R. and Feigenbaum, Edward A. The Handbook of Artificial Intelligence, Volume III, William Kaufmann, Inc., Los Altos California. 1982

Fahlmar, Scott E. NETL: A System for Representing and Using Real-World Knowledge. MIT Press, Cambridge, Massachusetts. 1979

Lindsay, Peter H. and Norman, Donald A. Human Information Processing: An Introduction to Psychology. Academic Press, New York. 1977

Shapiro, L. G. and Haralick, R. M. "Structural Descriptions and Inexact Matching", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-3, Number 5, September 1981

Schumucker, Kurt J. Fuzzy Sets, Natural Language, Computations, and Risk Analysis. Computer Science Press, Rockville, Maryland. 1984

Sowa, John F. Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley Publishing, Company, Reading, Massachusetts. 1984

Watanabe, Satosi. Pattern Recognition: Human and Mechanical. John Wiley and Sons. 1985

APPENDICES

APPENDIX A

Glossary of Predicate Abbreviations

ALGN:	alignment. The angular alignment of a line segment.
CIRC or CIRCLE:	A closed loop resembling a perfect circle.
HALF-CIRCLE:	
HORZ:	horizontal. A possible "value" for alignment.
JOIN:	joined. Attached or very close line segments.
LDIA or LDIAG:	left-diagonal. Upper left to lower right.
LEFT (concept):	left end of a line segment.
LEFT (relation):	left of. The Q is left of U in QUEEN.
LINE:	A line segment.
LOWR:	lower end of a line segment.
LWLF:	lower left quadrant of a character cell.
LWRT:	lower right quadrant.
NEAR:	relatively close line segments or picture elements.
MIDL:	middle point of a line segment.
PARA:	parallel. Parallel line segments.
PART:	part of. A definable sub-piece of a picture element
PERP:	perpendicular. Perpendicular line segments.
RGHT:	right end of a line segment.
RDIA or RDIAG:	right diagonal. Lower left to upper right.
UPPR:	upper end of a line segment
UPLF:	upper left quadrant of a character cell.
UPRT:	upper right quadrant of a character cell.
VERT:	vertical. A vertical line segment.

APPENDIX B

During the earliest stages of character recognition special mechanisms are needed to calculate the fuzzy measure for base-level relations. Appendix B briefly describes the mechanisms used for experiments in this thesis.

NEAR

$$\text{NEAR} = \begin{cases} 1.0 & r < 0.1 \\ 1.0 - 5*(r-0.1) & 0.1 < r < 0.3 \\ 0.0 & 0.3 < r \end{cases}$$

$$r = \frac{\text{the distance between the line and the given point}}{\text{the length of the given line segment}}$$

The NEAR function calculates the fuzzy measure for how close a given point is to a given line segment. The length of the line segment, in the numerator, acts as a scaling factor in this equation. This is based on the assumption that a large line segment has a larger neighborhood in terms of the NEAR relation than a smaller line segment.

JOIN

$$\text{JOIN} = \exp(-d*d)$$

$$d = \frac{\text{distance between line segments}}{\text{width of line segments}}$$

The JOIN function calculates a fuzzy measure representing how well two line segments are joined. The fuzzy measure is calculated as a function of the distance between the two line segments. The scaling factor for this function is the width of the line segment, whereas for the NEAR relation the scaling factor was the length of the line segment.

PARA

$$\text{PARA} = \begin{cases} 1.0 - 2.594*(a**2) & 0 < a < pi/8 \\ 0.2 + 2.594*(pi/4-a)**2 & pi/8 < a < pi/4 \end{cases}$$

a = interior angle between line segments

The PARA function generates a fuzzy number as a measure for how well two lines satisfy the parallel relation. The angle between the two line segments is measured in radians.

This same function is used with slight modifications for other angular fuzzy relations such as PERP, HORZ, RDIAG, LDIAG, and VERT.

PERP

$$\text{PERP} = \begin{cases} 1.0 - 2.594*(b**2) & 0 < a < pi/8 \\ 0.2 + 2.594*(pi/4-b)**2 & pi/8 < a < pi/4 \end{cases}$$

b = interior angle between line segments - 90 degrees

The PERP function is exactly the same as the PARA function but a different parameter is used as input. The "best" angle for being perpendicular is ninety degrees while the "best" angle for being parallel is zero degrees. The PERP function simply subtracts the ninety degrees (1.57 radians) and then uses the same equation used in the PARA function.

HORT, VERT, RDIAG, and LDIAG

The same technique as described above for the PERP function can be used for any angular predicate measure. The angle between a line segment and an imaginary, horizontal reference line is measured. This radian measure is compared against the desired angle, such as 45 degrees for RDIAG lines, and the difference is used as the input parameter to the equation for the PARA function.

Example: an input line has an angular measure of 52 degrees or 0.908 radians. For RDIAG the desired angle is 0.785 radians. Subtracting these values gives:

$$a = 0.908 - 0.785 = 0.122$$

In the equation for the PARA function this produces a final result of 0.985.

LEFT END, MIDDLE-POINT and RIGHT-END

The NEAR relation is only used between specified points on line segments (in this thesis). The right end of one line segment may be near the middle point of another line segment. The points on a line segment are fuzzy concepts and the following fuzzy measure function is used to generate them:

$$\text{POINT} = \begin{cases} 1.0 - 8.0*(d**2) & 0 < d < 0.25 \\ 8.0*(d-.5)**2 & 0.25 < d < 0.5 \end{cases}$$

$$d = \frac{\text{the distance from a point on the line segment}}{\text{the length of the line segment}}$$

The length of the line segment acts as the scaling factor in this function. The distance from the right end or the middle point determines the fuzzy measure depending on whether the RGHT predicate or the MIDL predicate is desired.

FUZZY SEMANTIC NETWORKS

by

RON RAY HIGHTOWER

B.S., Kansas State University, 1983

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering and Computer Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

ABSTRACT

This thesis presents a knowledge representation structures termed a fuzzy conceptual graph. This structure is similar to the conceptual graphs of John Sowa but the concepts and relations have been replaced with fuzzy concepts and fuzzy relations, respectively. It is rationalized that the performance of the fuzzy conceptual graphs is superior to the non-fuzzy counterparts in some pattern recognition tasks.

A method for comparing two fuzzy conceptual graphs is given; it calculates a fuzzy measure for how well the two graphs compare. This new fuzzy measure function is employed as the basis for fuzzy concept definitions. The two methods of fuzzy comparisons and fuzzy definitions are demonstrated in the content of a character recognition application.

A semantic network is a set of relations and operations underlying the processing of conceptual graphs. In this thesis, the fuzzy versions of the mechanisms of conceptual graphs, graph comparing, and concept definition are proposed, thereby rendering the semantic network to become the fuzzy semantic network.