Once the similarity matching measures are obtained, a pseudo-code description of the algorithm for constructing CD representations based on the similarity matching measures is as follows: suppose we have the following matrices: M_1, M_2, ..., M_m. Let the total number of nodes in CD be L. Let the number of nodes for c_i be L_i (ideally, $L_i = |F_{c_i}|$). [9] Define S_i to be L_i.

Let i=2
Repeat if i $\neq$ m
.For each set of entries (e.g. M_i (a, b) = S_{ab}, and M_i (b, a) = S_{ba}), allocate an appropriate number of nodes shared among those concepts in the entries, and subtract the same number of nodes from each of the node pools established for the related entries of the next lower numbered matrix
.i= i+1

Here *related entries* mean entries consisting of a set of concepts that is a subset of the original concept, for example, (a b c) for (a b c d). *An appropriate number of nodes* mean the number of nodes proportional to the similarity matching measure in question. For example, suppose $M_i(a, b) = S_{ab}$, and $M_i(b, a) = S_{ba}$, so the appropriate number of nodes will be $S_{ab} * L_b = S_{ba} * L_a$. According to the definition of similarities above, the equality always holds. It is because $S_{ab} * L_b = \frac{|F_a \cap F_b|}{|F_b|} * L_b = |F_a \cap F_b|$, and $S_{ba} * L_a = \frac{|F_a \cap F_b|}{|F_a|} * L_a = |F_a \cap F_b|$. However, in reality this equality may not hold: because human similarity judgments and measurements are always error-prone, inconsistency is inevitable. Besides, random noises alone are enough to upset the equality. When inconsistency is encountered, we can use the average of the two instead in the formula.

An issue is the subtraction of nodes from the node pools of the related entries in the lower order matrices. Because of the fact that a high-order similarity is part of some lower-order similarities, the (feature) nodes used in the high-order similarity are part of the node pools of the lower-order similarities. When we allocate nodes for a high-order entry, we must subtract the same number of nodes from each of the related lower-order entries. Since we establish node pools for similarities iteratively, from the lowest order up, each time we only need to subtract from the related entries of the closest lower-order matrix.

The question of which nodes to remove from a pool of nodes can be answered partially by considering constraints we have, that is, the fact that we have to preserve established similarities (i.e. node sharing situations). When the constraints we have are not enough to determine an uniquely correct way of removing nodes, we can make a tentative decision and backtrack later if necessary, that is, performing a search over the space of all possible ways of removing nodes, until a test shows that all similarity matching measures are implemented correctly.

[9] L_i is determined based on the principle of similarity-based representation: the more general a concept is, the fewer feature nodes there are for representing it.

A Determining Similarities and Microfeature Representations

STSIS, or *a Statistical Test-Score procedure for determining Intensional Similarity*, is a procedure for determining, by empirical means, the microfeature (intensional) similarities and for constructing microfeature representations. [7] This procedure can be applied without performing a thorough domain analysis to find out all relevant microfeatures (because not all areas have a set of microfeatures well analyzed as in geography).

Assume there is a set of concepts $c = \{c_1, c_2, c_3, ..., c_n\}$, and $\overline{c}$ is the vector composed of all these concepts. Matrix M_2 measures the pairwise similarities between elements of the vector $\overline{c}$, that is,

$$M_2 = S(\overline{c} \times \overline{c})$$

where $\times$ denotes outer-product, and S is the similarity matching measure: S([x]) = [S(x)]. (In other words, the similarity of a matrix is a matrix of the similarities of its elements).

For each matrix element,

$$S(a,b) = \frac{\sum_{i=1}^{n} S_i'(a,b)}{n}$$

where $S_i'(a,b)$ is an empirical measure of the similarity matching between a and b, ranging from 0 to 1. We can also calculate the mean squared error:

$$\delta(a,b) = \sqrt{\sum_{i=1}^{n}(S - S_i')^2}$$

Then for higher order similarities, we have

$$M_3 = S(\overline{c} \times \overline{c} \times \overline{c})$$

and

$$M_4 = S(\overline{c} \times \overline{c} \times \overline{c} \times \overline{c})$$

and so on. [8]

One problem with this approach is that there are too many entries to fill in each matrix, especially in higher order ones. One way to deal with this problem is determining which entry will be zero beforehand and thus avoiding computing that entry; for a large number of entries in these matrices will be zero, which can be determined by examining related entries in the lower order ones (if one of the related entries is zero, then the entry is zero). (*Related entries* are defined to be entries that contain a subset of concepts involved in the original entry).

Another problem is when we should stop, because obviously we do not want too many M matrices (we can produce matrices as many as the number of concepts). There is no theoretical result determining when to stop. However, we can set up some empirical criteria. For example, we can limit the number of matrices to be no more than half the number of the concepts involved.

[7] For related approaches, see, for example, Zadeh (1981).

[8] Ideally, we should have $S_{ab} = \frac{|F_a \cap F_b|}{|F_b|}$, $S_{abc} = \frac{|F_a \cap F_b \cap F_c|}{|F_c|}$, and $S_{abcd} = \frac{|F_a \cap F_b \cap F_c \cap F_d|}{|F_d|}$, etc.

[37] G. Towell, J. Shavlik, and M. Noordewier, Refinement of approximate domain theories by knowledge-based neural networks. Proc.AAAI-90, Morgan Kaufman, San Mateo, CA. New York, pp.861-866. 1990

[38] A. Tversky, Features of Similarity, *Psychological Review*, 84(4), pp.327-352, 1977

[39] S. Vosniadou & J. Ortony, (eds.) *Similarity and Analogical Reasoning*, Cambridge University Press, 1989

[40] D. Waltz and J. Pollack, Massively Parallel Parsing, *Cognitive Science*, 1985

[41] L. Zadeh, Fuzzy Sets, *Information and Control*, 8, pp.338-353, 1965

[42] L. Zadeh, Test-score semantics for natural languages and meaning-representation via PRUF, in: B. Rieger, (ed.) *Empirical Semantics*, Bochum: Brockmeyer, pp.281-349, 1981

[43] L. Zadeh, Fuzzy Logic, *Computer*, Vol.21, No.4, pp.83-93, April 1988

[44] M. Waterman, *An Introduction to Expert Systems*, Addison-Wesley, Reading, MA. 1985

[19] H. Leonard, *Principle of Reasoning*, Dover, New York, NY. 1967

[20] R. Michalski, Two-tiered concept meaning, inferential matching, and conceptual cohesiveness, in S. Vosniadou & J. Ortony, (eds.) Similarity and Analogical Reasoning, Cambridge University Press, New York, NY. 1989

[21] M. Minsky, *The Society of Mind*, Simon and Schuster, New York, NY. 1985

[21] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufman, San Mateo, CA. 1988

[22] M. Posner, (ed.) *Foundations of Cognitive Science*, MIT Press, Cambridge, MA. 1989

[23] J. Pustejovsky, *Generative Lexicon*, MIT Press, Cambridge, MA. 1992

[24] C. Riesbeck and R. Schank, *Inside Case-based Reasoning*, Lawrence Erlbaum Associate, Hillsdale, NJ. 1989

[25] J. McClelland, D. Rumelhart, and the PDP Research Group, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, MIT Press, Cambridge, MA. 1986

[26] G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, NJ. 1974

[27] R. Sun, A discrete neural network model for conceptual representation and reasoning, *Proc.11th Cognitive Science Society Conference*, pp.916-923, Erlbaum, Hillsdale, NJ. 1989 a

[28] R. Sun, Designing inference engines based on a discrete neural network model, *Proc.IEA/AIE*, ACM Press, New York, NY. p.1094, 1989 b

[29] R. Sun, Rules and Connectionism, *Proc.INNC-Paris*, p.545, Kluwer, Netherland. 1990 a

[30] R. Sun, The Discrete Neuronal Models, *Proc.INNC-Paris*, pp.902-907, Kluwer, Netherland. 1990 b

[31] R. Sun, The Discrete Neuronal Models and the Discrete Neuronal Models, in: B. Soucek ed. *Neural and Intelligent System Integration*, John Wiley & Sons, New York, NY. 1991 a

[32] R. Sun, Chunking and Connectionism, *Neural Network Review*, Vol.4, No.2, pp.76-78, 1991 b

[33] R. Sun and D. Waltz, Neurally Inspired Massively Parallel Model of Rule-Based Reasoning, in: B. Soucek ed. *Neural and Intelligent System Integration*, John Wiley & Sons, New York, NY. 1991

[34] R. Sun, A connectionist model of commonsense reasoning incorporating rules and similarities. *Knowledge Acquisition*, Vol.4, 293-321, 1992

[35] R. Sun, Beyond associative memories, logics and variables in connectionist networks. *Information Sciences*, Vol.70, 1993

[36] D. Touretzky, *The Mathematics of Inheritance*, San Mateo, CA: Morgan Kaufman, San Mateo, CA. 1986

REFERENCES

[1] J. Anderson and E. Rosenfeld (eds.) *Neurocomputing*, MIT Press, Cambridge, MA. 1988

[2] L. Bookman, A Connectionist Scheme For Modeling Context, In D. Touretzky et al. eds. *Proc.1988 Connectionist Summer School*, pp.281-290. San Mateo, CA: Morgan Kaufman, 1989

[3] C. Chang and R. C. Lee, *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, San Diago, CA. 1973

[4] A. Collins and R. Michalski, The Logic of Plausible Reasoning: A Core Theory, *Cognitive Science*, Vol 13, No 1, pp.1-49, 1989

[5] M. Derthick, Mundane reasoning by parallel constraint satisfaction, TR CMU-CS-88-182, Carnegie-Mellon University, 1988

[6] D. Dubois and H. Prade, An Introduction to Possibilistic and Fuzzy Logics, in: P. Smets et al (eds.) *Non-Standard Logics for Automated Reasoning*, Academic Press, San Diago, CA. 1988

[7] J. Feldman, Neural Representation of Conceptual Knowledge, Technical Report 189,Department of Computer Science, University of Rochester, 1986

[8] L. Fu, Recognition of Semantically Incorrect Rules, *Proc.IEA/AIE-90*, pp.??, 1990

[9] S. Gallant, Connectionist Expert Systems, *Communication of ACM*, V.31(2), pp.152-169, 1988

[10] J. Gelfand, D. Handelman and S. Lane, Integrating Knowledge-based Systems and Neural Networks for Robotic Skill Acquisition, *Proc.IJCAI*, pp.193-198, Morgan Kaufman, San Mateo, CA. 1989

[11] S. Grossberg, *The Adaptive Brain*, North-Holland, New York, NY. 1987

[12] P.J. Hayes, In defence of logic, *Proc.5th IJCAI*, pp.559-565, Morgan Kaufman, San Mateo, CA. 1977

[13] F. Hayes-Roth, D.A. Waterman and D.B. Lenat, eds. *Building Expert Systems*, Addison-Wesley, Reading, MA. 1983

[14] J. Hendler, Marker Passing and Microfeature, *Proc.10th IJCAI*, pp.151-154, Morgan Kaufman, San Mateo, CA. 1987

[15] J. Hendler, Integrating Marker Passing and Problem Solving, Lawrence Erlbaum Associates, Hillsdale, NJ. 1988.

[16] K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators, Neural Networks, Vol.2 pp.359-366. 1989

[17] R. Lacher et al, Backpropagation learning in expert networks, IEEE Transactions on Neural Networks. 3, 62-72, 1992.

[18] T. Lange and M. Dyer, Frame selection in a connectionist model, *Proc.11th Cognitive Science Conference*, pp. 706-713, Lawrence Erlbaum Associates, 1989

5.2 Future Directions

Our future work will include the following:

- Using backward-chaining and other inference algorithms (except forward chaining, which is used here and analyzed fully in [35]), to enable a wider range of reasoning capabilities.
- Dealing with sequential events, for applying the model to planning and other areas.
- Developing variable binding capabilities in this two level model (cf. [30, 31]).

5.3 Summary

In short, we have proposed an idea for structuring knowledge that might have wide applicability in various domains. The idea follows from analyzing different kinds of flexibilities in reasoning in vague and continuous domains. Knowledge is divided into two kinds: (1) compound knowledge statements in the form of rules and (2) microfeatures associated with individual primitive knowledge statements (concepts). Thus, the two levels, the concept level and the microfeature level, follows naturally, which allow both efficient usage of knowledge statements explicitly represented in the system and wider coverage via plausible connections with these statements (i.e. rule application plus similarity matching). Several simple examples are presented that indicate possibilities of applying the structure to a variety of domains.

clusters of them) as in PDP models, we can obtain all similar cases at once, which facilitates comparisons, explanation generations, and other post-processing.

Comparing our approach with traditional rule-based approach (e.g. [44] and [13]) in constructing knowledge-based systems, the advantage is obvious: not only our model of two level structuring is (potentially) capable of doing almost all of the functionalities of traditional rule-based systems [35], but it can also deal with similarity-based reasoning in an efficient and massively parallel way. A pure rule-based system cannot deal with vague and continuous domains both completely and efficiently, for it needs greatly many more rules to cover all cases and all minor variations. Considering some other variants of rule-based reasoning, namely, probabilistic reasoning, such as [21] and [26], we believe that our approach is simpler than theirs, and takes into account cumulative evidentiality (the ability to accumulate evidence) with efficient computation. The weighted-sum computation used can be viewed as a simplification of probabilistic reasoning: suppose we have probabilistic events A_1, A_2, ..., A_n, and B. A rule such as $A_1 A_2 ....A_n \longrightarrow B$ can be readily interpreted as computation of p(B) from p(A_1), p(A_2), ..., p(A_n), i.e.

$$p(B) = p(B|A_1 A_2 ....A_n)p(A_1 A_2 ....A_n)$$

When A_i's are pair-wise independent and exhaustive, the above formula can be rewritten as

$$p(B) = p(B|A_1)p(A_1) + p(B|A_2)p(A_2) + ... + p(B|A_n)p(A_n)$$

The probabilistic law is similar in form to the calculation we perform according to the following rule: $A_1 A_2 .....A_n \longrightarrow B$ with weights $(w_1, w_2, ...w_n)$, that is,

$$ACT_B = \sum_i ACT_{A_i} * w_i$$

where $p(A_i)$ are activation values of the antecedents (i.e., ACT_{A_i}'s), $p(B)$ are the activation value for the consequent (i.e., ACT_B), and $p(B|A_i)$ is w_i. So we have an interpretation of the weights as conditional probabilities.

Comparing our approach with case-based reasoning [24], the similarity is evident: both approaches utilize similarities between the current situation and previously known situations to come up with a plausible conclusion. The differences on the other hand are the following:

- Rules (compound knowledge statements) are the basic coding mechanisms for both concrete and abstract knowledge in our model, which adopts a simple, uniform representation that encompasses both cases and rules.

- There are both rules and similarities in our approach, and they are intermixed. In other words, abstract knowledge and concrete analogous knowledge co-exist and they are integrated into a unified structure.

- Unlike most case-based systems, similarity matching (as well as rule application) is done here in a massively parallel fashion, and thus is very efficient.

1. dressed-like-sailor talks-like-sailor walks-like-sailor $\longrightarrow$ sailor
2. talk-about-TV-business $\longrightarrow$ interested-in-TV
3. talk-about-TV-business $\longrightarrow$ in-TV-business
4. in-TV-business $\not\longrightarrow$ sailor
5. dressed-as-sailor $\longrightarrow$ hobby-sailor
6. hobby-sailor $\longrightarrow$ rich-people
7. rich-people $\not\longrightarrow$ sailor

Figure 24: A List of Rules

does not have to be done by constraint satisfaction through minimizing some energy functions (as in [5]), but can be done by reasoning based on rules and similarities. This way the results are less susceptible to small changes in parameter settings.

The knowledge used for performing this type of mundane reasoning is encoded in rules as in Figure 24. In the bottom level, distributed representation is used so that similar concepts have shared nodes in that level. It is constructed according to STSIS, and each node is vaguely interpretable.

We provide a set of tests to verify the correctness of the system the same way as before. The system works as expected; for example, when given the input that "Ted is dressed up like a sailor" (in the form of activating nodes "dressed-like-sailor") the system will indicate that Ted is a sailor (in the form of activating nodes "sailor" strongly); when given the input that Ted is dressed up like a sailor but talks a lot about TV business (in the form of activating nodes "dressed-like-sailor" and "talk-about-TV-business"), the system will indicate hobby sailors (in the form of activating nodes "hobby-sailor" more strongly).

5 Concluding Remarks

5.1 Comparisons

Comparing our approach with PDP models [25], we notice both similarities and differences. In terms of similarities, both approaches utilize networks of simple processing elements, operate in a massively parallel fashion, and are capable of carrying out continuous functions for capturing flexible reasoning in vague domains. Unlike PDP models, our model does not rely exclusively on similarities – rules are implemented that can generate precision as well as flexibility [28, 29, 32]. Our model does not require extremely long time of training (as in case of backpropagation networks; cf. [25]), or extremely long settling time (as in case of Boltzmann machine; cf. [5]). Although the main means for knowledge acquisition in our system is extracting knowledge from external sources, some learning can be carried out for refining and rectificating knowledge bases (cf. [8], [37], etc.). In our system, instead of producing only the closest match among the (stored) training cases (or

1. Pot is cooking pot.
2. Pot could be marijuana.
3. Marijuana is illegal.
4. Cleaning kitchens implies cleaning cookware.
5. Using dishwashers implies cleaning cookware.
6. If one is having marijuana and the police are coming, then the police will see it.
7. Police seeing illegal substance results someone being arrested.
8. To avoid arrests, prevent the police seeing illegal substances.
9. To prevent somebody seeing something, hide it.
10. Putting something in a dishwasher is for washing it.
11. Putting something in a dishwasher could be for hiding it.
12. If there is a bankrobbery going on and the police are coming, then they are here to stop the bankrobbery.

Figure 23: A List of Rules

and similarity matching. [6]

4.3.2 Applications to Mundane Reasoning

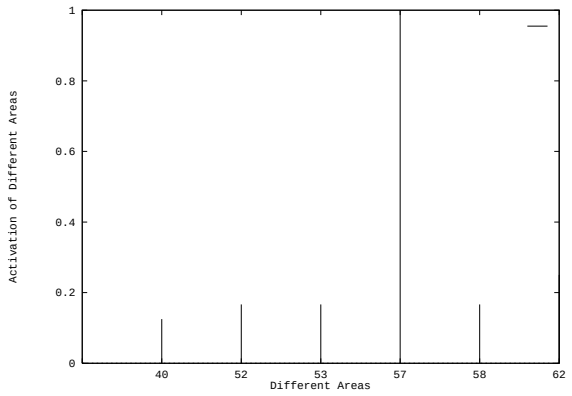
Mundane reasoning is another area in which commonsense knowledge is heavily involved. Mundane reasoning is used to refer to the type of reasoning that we do daily regarding mundane matters, for example, which chair to sit in, when to eat, etc. The goal of such mundane reasoning is to come up rapidly with an interpretation of a situation or to make a quick decision, given the current context. In applying the two level structuring, we have to identify all the concepts and rules involved in a particular task. We also have to identify all the microfeatures associated with the concepts.

We will look into the "Ted" example from [5]. Instead of using constraint satisfaction as in [5], we perform rule-based reasoning plus similarity matching. The problem can be stated as follows:

> Ted is seen walking along a pier, dressed like a sailor. Ted launched into an excited monolog on the influence of TV programming. It seems reasonable to conclude that Ted is a professional sailor, and that he is interested in television. But another possibility is that Ted is TV tycoon and a millionaire playboy and has a hobby of sailing.

The point is that normally Ted should be taken to be a professional sailor, but under certain circumstances, given some pertinent clues, Ted should be interpreted as a hobby sailor. This task

[6] Note that some subtle linguistic elements are not taken into account in the present implementation, for example, "*when* the police are coming" vs. "*because* the police are coming", etc. When finer distinctions are needed, more nodes will have to be added into the system, along with possibly other mechanisms, to take those elements into consideration.



40	"guiana-pgs"	0.125
52	"chile-s"	0.1666666666666667
53	"chile-c"	0.1666666666666667
57	"Argentina-andeanhighland"	1
58	"columbia-w"	0.1666666666666667
62	"ecuador-highlands"	1/4

Figure 22: Output From the System: Case 4

along with concepts, or use some statistical methods to determine similarities and, based on that, construct microfeature representation (see Appendix for details).

One simple example regarding lexical disambiguation (cf. [40], [2], and [23]) that lends itself to our discussion is regarding "Pot", taken from [18]:

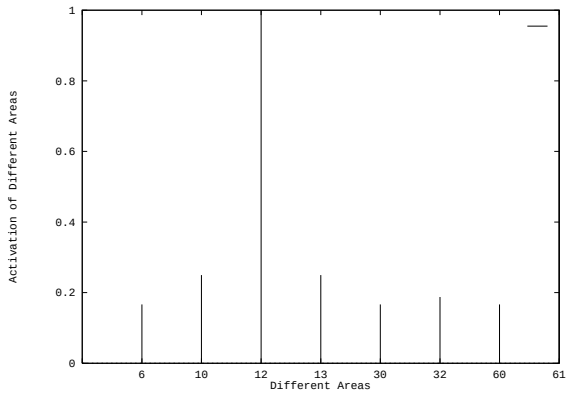
> John put the pot inside the dishwasher, because the police are coming.

The point is that normally the word "pot" should be interpreted as "cooking pot", but under certain circumstances, given some pertinent clues, it should be interpreted as marijuana.

We provide a set of situations to test our system in its ability, which is not in [18]: (1) John put the pot inside the dishwasher (the solution should be "cooking pot" in this case); (2) John put the pot inside the dishwasher, because the police are coming (the word "pot" means "marijuana" in this case); (3) John put the pot inside the dishwasher, because the police are coming and John wants to make the kitchen clean (the solution should be "cooking pot" in this case); (4) John put the pot inside the dishwasher, because John wants to make the kitchen clean (the solution should be "cooking pot" in this case); (5) John put the pot inside the dishwasher, when the police come for the bankrobbery across the street (the situation is pretty ambiguous, and the system could interpret it as "marijuana"); (6) John is cleaning the kitchen, putting the pot inside the dishwasher, when the police come for the bankrobbery across the street (the solution should be "cooking pot" in this case).

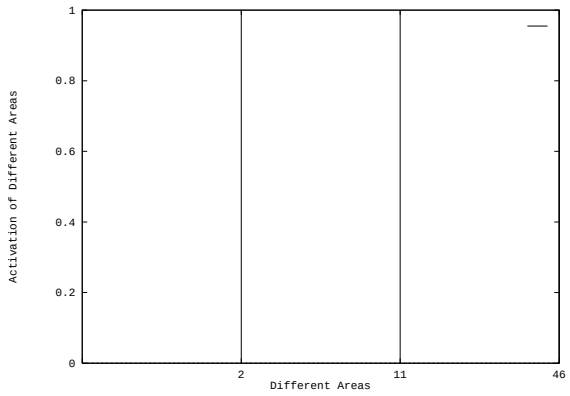
We constructed a system that can solve the original problem and pass the six tests as follows: Each of the concepts involved in the problem description is represented by one node in the top level. Knowledge in the form of rules is extracted from commonsense knowledge about the concepts involved in the story. Rules that are actually put into the systems are in Figure 23. On the other hand, the bottom level is basically a distributed version of the top level by sharing nodes among related concepts, so that continuity/similarity can be explored. This structure is constructed according to STSIS (see Appendix). Each node is vaguely interpretable. The rules are duplicated diffusely in the bottom level.

When the parameters are appropriately set the system performs the task correctly. It gives correct answers to all tests, distinguishing the often very subtle differences, through rule application



6	"Uruguay-coastal"	0.1666666666666667
10	"fruit-veg-growing-area"	0.2499997615814209
12	"producing-banana"	0.9999990463256836
13	"producing-tropical-fruits"	0.2499997615814209
30	"e-peru",	0.1666666666666667
32	"Bolivia-orient-rainforest",	0.1875
60	"columbia-basin"	0.1666666666666667
61	"ecuador-coast"	1

Figure 20: Output From the System: Case 2



2	"cattle-country"	0.9999990463256836
11	"sheep-country"	0.9999990463256836
46	"Brazil-s"	1

Figure 21: Output From the System: Case 3

for "cattle" in the results. The output is in Figure 21. The result indicates that the area does produce cattle and sheep. Nothing else in the network fires strongly or distinguishably in this case.

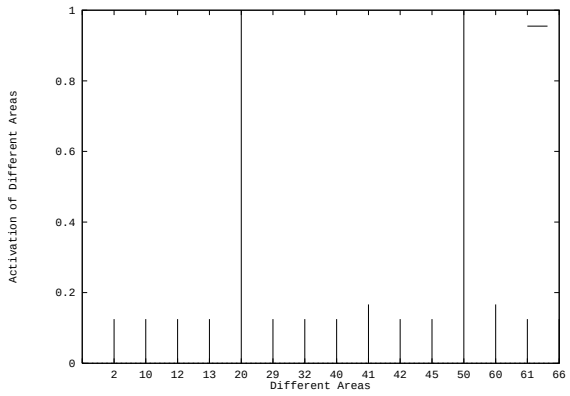
Yet another example is about "Argentina-Andean-Highland": Is it a cattle country? We activate the node representing the area, and the output is in Figure 22. This result shows clearly that it is not a cattle country.

4.3 Other Applications

Now the question is: Can this same method be applied to other domains where no such natural division of concepts and features seems to exist? We will show that the same approach does work for other domains. Due to the space limitation, only some brief hints as to how this structure can be applied to these other domains will be provided.

4.3.1 Applications to Natural Language Understanding

Natural language understanding is an area in which commonsense reasoning is crucial. For practical purposes, we can either perform a thorough domain analysis to identify useful microfeatures



2	"cattle-country"	0.1249998807907104
10	"fruit-veg-growing-area"	0.1249998807907104
12	"producing-banana"	0.1249998807907104
13	"producing-tropical-fruits"	0.1249998807907104
20	"rubber-producing-area"	0.9999990463256836
29	"c-peru"	0.125
32	"Bolivia-orient-rainforest"	0.125
40	"guiana-pgs"	0.125
41	"guiana-hilly-country-forest"	0.1666666666666667
42	"guiana-hilly-country-savanna"	0.125
45	"Brazil-cw"	0.125
50	"Brazil-n"	1
60	"columbia-basin"	0.1666666666666667
61	"ecuador-coast"	0.125
66	"suriname-plateau"	0.125

Figure 19: Output From the System: Case 1

```
( 2, "cattle-country", 0.1249998807907104 )
 ( 10, "fruit-veg-growing-area", 0.1249998807907104 )
 ( 12, "producing-banana", 0.1249998807907104 )
 ( 13, "producing-tropical-fruits", 0.1249998807907104 )
 ( 20, "rubber-producing-area", 0.9999990463256836 )
 ( 29, "c-peru", 0.125 )
 ( 32, "Bolivia-orient-rainforest", 0.125 )
 ( 40, "guiana-pgs", 0.125 )
 ( 41, "guiana-hilly-country-forest", 0.1666666666666667 )
 ( 42, "guiana-hilly-country-savanna", 0.125 )
 ( 45, "Brazil-cw", 0.125 )
 ( 50, "Brazil-n", 1 )
 ( 60, "columbia-basin", 0.1666666666666667 )
 ( 61, "ecuador-coast", 0.125 )
 ( 66, "suriname-plateau", 0.125 )
```

The result shows that it is a rubber-producing area for sure (with confidence value equal to 0.999999), and it is similar, to a small extent, to "Guiana hilly country" and "Bolivia orient rainforest area" etc. If we want to choose one answer out of many, we can simply use a winner-take-all network on top of this, but this is not an intrinsic part of the system. See Figure 19.

Another example is as follows: suppose we want to know about the Ecuador coastal area, we will give the system a query: What is the main agricultural product of "Ecuador-coast"? by activating the node representing "Ecuador-coast". To answer this question, we let the system run to perform its reasoning. The output is in Figure 20. The result indicates that the area is producing banana (with confidence value equal to 0.99999) and is very likely producing tropical fruits and other fruits/vegetables. It is similar, in some way, to "Uruguay-coastal", "eastern-Peru" and "Columbia-basin".

As yet another example, let us reason about "Brazil-south". We will start by giving a query: Does "Brazil-south" produce cattle? by activating the node representing "Brazil-south" and looking

temperate	arctic	woodland	plain	mediterrainian
plateau	Mts	coastal-land	lake	tropical
lowland	hill	river-valley-basin	swamp	rainforest
evergreen	deciduous	highland	upland	sparsely-populated
densely-populated	fertile	infertile	flood	prairie
dependable-rainfall	scrub	farming	rugged	subtropical
rainy	savanna	dry-arid	grassland	desert

Figure 18: Geographical Features Included in the System

In extracting information from source books, there are some subtleties that have to be taken into considerations. Each article regarding a particular region is written by a particular researcher familiar with that region, and varies in depth, presentation, amount of details and emphasis. This diversity inevitably has adverse effects on the accuracy of the specification. The problem is the lack of details on the one hand and too much detail on the other hand. When there are not enough details from one sourcebook, we can find another sourcebook and try to fill in what is needed. In case of too much detail, we have to be very careful in selecting the most important and relevant information out of the tangled web of irrelevant descriptions. As a rule of thumb, we usually disregard information associated with phases such as "plus", "in addition", "besides", "although", "a small portion of", "mostly..... but.....", etc. A problem is that few regions are geographically homogeneous. What we want is a description that is applicable to the largest portion of a region, expressing its *essential* characteristics, without having irrelevant information or descriptions that can only be applied to a small part of that region. There is certainly a tension between (1) capturing important characteristics of a region, and (2) excluding information applicable only to a small part of a region. The tradeoff between these two aspects helps to decide what primitive geographical regions are and what information is to be included for each such region.

Now we are ready to describe the working of the system. Once a name of a geographical area is given to the system, as imposing a query, the system will find out its agricultural characterization, such as "cattle-country", "rice-growing-area", or "rubber-producing-area", through rule application or similarity matching, or a combination of the two. For example, let us choose to reason about "Brazil-north", which is described as "tropical rainforest hilly plateau". We will start by giving a query: What is the main agricultural product of "Brazil-north"? That amounts to activating the node representing "Brazil-north". To answer this question, we let the system run to perform its reasoning. The output is as follows:

```
>(consyderr 0)

TITLE:  GEOGRAPHY
focusing on context AGRICULTURE : remove feature NIL
setup done
starting running
top down
cl propagating
cd propagating
bottom up

the average activation is 0.1213409896658248
```

"chile-n"
"chile-s"
"chile-c"
"Argentina-ne"
"Argentina-pampa"
"Argentina-patagonia"
"Argentina-andeanhighland"
"columbia-w"
"columbia-e"
"columbia-basin"
"ecuador-coast"
"ecuador-highlands"
"Venezuela-Llanos"
"Venezuela-coastalplain"
"suriname-coastalplain"
"suriname-plateau"

Figure 16: Geographical Areas Included in the System

"cotton-producing-area"
"coffee-growing-area"
"wine-producing-area"
"potato-growing-area"
"rubber-producing-area"
"goats-area"
"rice-growing-area"
"wheat-growing-area"
"soybean-growing-area"
"rubber-producing-area"
"sheep-country"
"producing-banana"
"producing-tropical-fruits"
"corn-growing-area"
"sugar-producing-area"
"fruit-veg-growing-area"

Figure 17: Regional Characterization Included in the System

"Chaco"
"Honduras"
"Uruguay-coastal"
"Uruguay-plateau-highland"
"Mendoza"
"Lanos"
"w-peru"
"c-peru"
"e-peru"
"Bolivia-orient-grassland"
"Bolivia-orient-rainforest"
"Bolivia-cordillera-occidental"
"e-paraguay"
"w-paraguay-forest"
"w-paraguay-savanna"
"Panama-lowland"
"w-Texas"
"guiana-pgs"
"guiana-hilly-country-forest"
"guiana-hilly-country-savanna"
"guiana-plain"
"Bolivia-SW-highlands"
"Brazil-cw"
"Brazil-s"
"Brazil-saopaulo"
"Brazil-e"
"Brazil-ne"
"Brazil-n"

Figure 15: Geographical Areas Included in the System

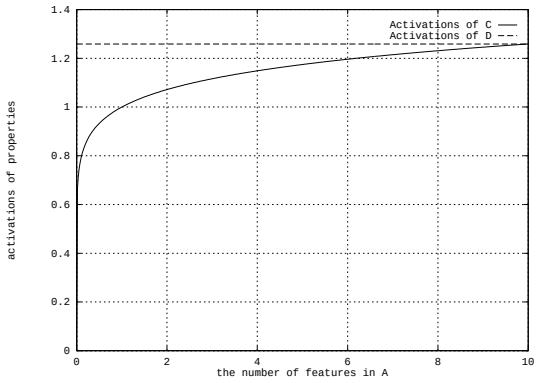


Figure 14: Inheritance

Activations of properties relative to the number of microfeatures in A, when B is fully activated, where $A \supset B$, $F_A \subset F_B$, $|F_B| = 10$, and $A \longrightarrow C$, $B \longrightarrow D$, with the weights equal to 1.

similar and the amount of overlapping is proportional to the degree of the similarity between them, as alluded to before. Thus, inferences are enabled through similarity matching, and a fuller coverage of the domain is ensured.

We tabulate some of the data stored in the system: Figure 15 and Figure 16 list some of the geographical areas included in the system, most of which are in South America; Figure 17 lists concepts for characterizing a geographical area in terms of its agricultural products, such as rice-growing-area, cattle-country, etc.; Figure 18 lists microfeatures used.

It should be stressed that the process of knowledge acquisition for this system is straightforward and systematic, and nothing is tuned arbitrarily just for getting one outcome or the other. Specifically, the knowledge in the system is obtained from encyclopedias, such as *Encyclopedia Britannica* or *Encyclopedia Americana*, in the form of a basic geographical region (a region with relatively uniform characteristics), its products, and its geographical features. Since those types of information is well documented and rather extensive in source books, the task is straightforward, although it is extremely tedious and time-consuming.

For example, a typical entry is like this:

```
Peru--Montana:
lowland, tropical, woodland
```

This is extracted from the following descriptions:

> Peru–Montana:
>
> From west to east the major surface features of Peru are 1) the desert coast, 2) the Andes highlands, 3) the Montana.
>
> **Montana**: The Montana is the remote region containing the lower, densely forested slopes of the eastern Andes and the flat, tropical Amazon plains called Selva. The region experience high temperatures and abundant rains throughout the year. The natural vegetation cover is a tall tropical rainforest, with a leafy canopy shading a damp, open forest floor. Grasses grow in poorly drained areas, while coarse grasses and brush grow on sandy, porous soils and rocky slopes.

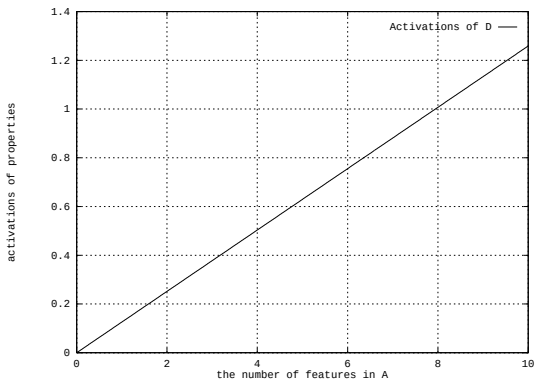


Figure 11: Inheritance

Activations of properties relative to the number of microfeatures in A, when A is fully activated, where $A \supset B$, $F_A \subset F_B$, $|F_B| = 10$, and $B \longrightarrow D$ with the weight equal to 1.

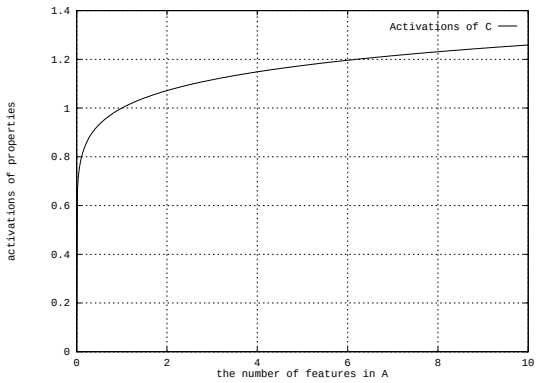


Figure 12: Inheritance

Activations of properties relative to the number of microfeatures in A, when B is fully activated, where $A \supset B$, $F_A \subset F_B$, $|F_B| = 10$, and $A \longrightarrow C$, with its weight equal to 1.

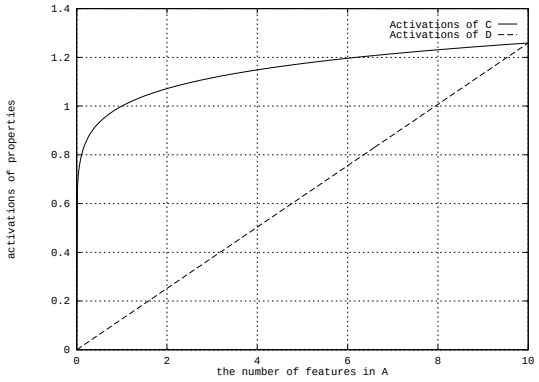


Figure 13: Inheritance

Activations of properties relative to the number of microfeatures in A, when A is fully activated, where $A \supset B$, $F_A \subset F_B$, $|F_B| = 10$, and $A \longrightarrow C$, $B \longrightarrow D$, with the weights equal to 1.

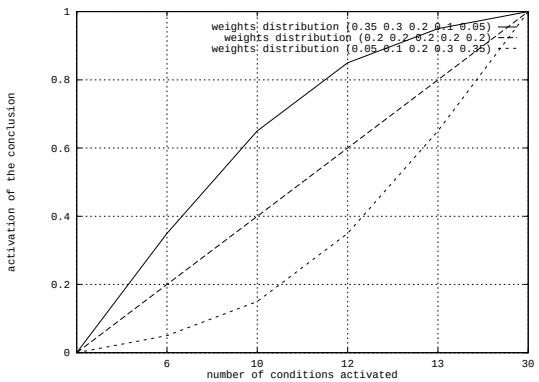


Figure 10: Rule Application

Activations of the consequent relative to the number of antecedents activated, with different weight distributions.

Let us look into the cases of rule application. Activations resulting from rule application are determined by

$$ACT_B = \sum W_i * ACT_{A_i}(t)$$

where W_i's are link weights (representing strengths or weights of rules). These formulas are implemented at the both levels. To test this, we collected activation data as in Figure 10, which shows the different activations of the consequent with different number of antecedents activated (and to different degrees), in a fixed order but with different weight distributions.

Let us look into the cases of inheritance, which is carried out with mixed similarity matching and rule application: that is, classes are represented as nodes and microfeature sets, properties are represented as separate nodes and microfeature sets, and links between them indicates which class has what properties. These are implemented with the two level structure. We can examine the following figures: Figure 11, Figure 12, Figure 13, and Figure 14. These figures are for bottom-up inheritance, top-down inheritance, bottom-up inheritance with cancellation, and top-down inheritance with cancellation, respectively. Those are the cases where each property value is represented by a distinct single node.

4.2 Reasoning with Geographical Knowledge

We will look into reasoning with geographical information. The knowledge representation in this system utilizes the two level idea by dividing the geographical knowledge represented in the system into two categories: concepts (primitive knowledge statements), which include basic geographical areas and regional characterizations (such as "cattle-country"), and microfeatures, which include basic geographical descriptions of areas, such as "highland", "mountainous", and "tropical", etc. Concepts are represented in the top level, and microfeatures are represented in the bottom level. Each area is connected to concepts describing its agricultural products by rules, implemented as links, if this knowledge is available to the system. Each geographical area represented in the top level is connected to its corresponding microfeatures in the bottom level, and because of the fact that microfeatures are shared by similar concepts, the microfeature representation is similarity-based, i.e., two concepts have overlapping microfeature representations if and only if the two are

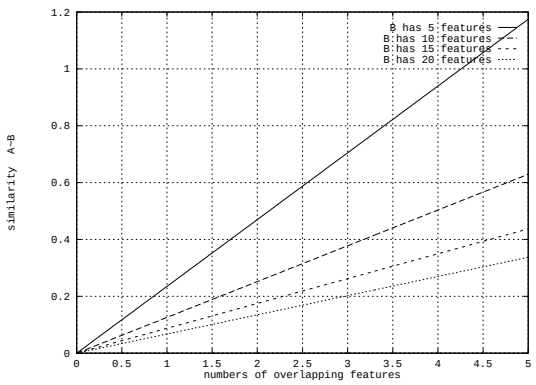


Figure 8: Similarity Matching in the Structure
when A has 20 microfeatures and B has small variable numbers of microfeatures, where $S_{AB} \approx \frac{|F_A \cap F_B|}{|F_B|}$

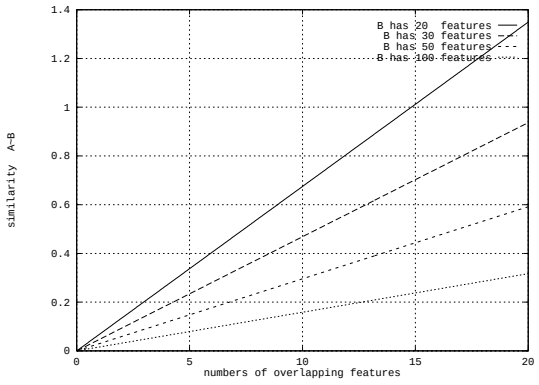


Figure 9: Similarity Matching
when A has 20 microfeatures and B has large variable numbers of microfeatures, where $S_{AB} \approx \frac{|F_A \cap F_B|}{|F_B|}$

(5) For A $\longrightarrow$ B, B $\longrightarrow$ C, C $\longrightarrow$ D, if A is activated, then

$$ACT_D \approx ACT_A * r_{AB} * r_{BC} * r_{CD}$$

(6) For A $\sim$ B, B $\longrightarrow$ C, C $\longrightarrow$ D, if A is activated, then

$$ACT_D \approx ACT_A * s_{AB} * r_{BC} * r_{CD}$$

(7) For A $\longrightarrow$ B, B $\sim$ C, C $\longrightarrow$ D, if A is activated, then

$$ACT_D \approx ACT_A * r_{AB} * s_{BC} * r_{CD}$$

(8) For A $\longrightarrow$ B, B $\longrightarrow$ C, C $\sim$ D,

$$ACT_D \approx ACT_A * r_{AB} * r_{BC} * s_{CD}$$

(9) For A $\sim$ B, B $\longrightarrow$ C, C $\sim$ D,

$$ACT_D \approx ACT_A * s_{AB} * r_{BC} * s_{CD}$$

4 Experiments

This section presents a number of experiments that demonstrate the capability of the two-level structure in accounting for some commonsense reasoning tasks, to supplement, verify and strengthen our theoretical discussions and developments.

4.1 Systematic Performance Evaluations

Although we derived the system parameters and functions above, we have not studied yet the whole spectrum of its behavior. So we want to design test cases that can cover rule applications, similarity matching functions, top-down weights, bottom-up weights, link weights, etc., to see how they work under different circumstances.

Let us look into the cases of similarity matching first:

$$s_{AB} = \frac{|F_A \cap F_B|}{|F_B|}$$

This formula is implemented with the two level structure. To test it, we collected the data in Figure 8 and Figure 9. The two figures demonstrate the difference in the final similarity outcome when one or more of the following three parameters change: $|F_A|$, i.e. the number of microfeatures for A, $|F_B|$, i.e. the number of microfeatures for B, and $|F_A \cap F_B|$, i.e. the number of overlapping microfeatures. The results show that each curve is almost straight while in fact a slower than linear function is employed. When the number for $|F_B|$ grows big, the addition of microfeatures in B has less effect on the similarity outcome. The number for $|F_A|$ has basically no effect on the similarity outcome.

weights td. Because of overlapping, some of the microfeature nodes of B are activated. Then at the bottom-up phase, the activation of the microfeature nodes of B goes up to the node B in the top level through weights bu ($= \frac{1}{g(|F_B|)}$). According to what we derived so far,

$$ACT_B = td * ACT_A * |F_A \cap F_B| * \frac{1}{g(|F_B|)}$$

To make B match what is obtained from a similarity measure, specifically,

$$s_{AB} = \frac{|F_A \cap F_B|}{|F_B|}$$

we have to choose td as

$$td_A = 1$$

for any node A in the top level, and we have to choose g as close to the identity function as possible, to make $\frac{ACT_B}{ACT_A} = s_{AB}$. Thus, we determine yet another parameter — the weights for top-down transmitting of activation.

Now we shall check to see if the parameters derived so far satisfy the requirements for correct rule application and mixed similarity matching and rule application (including the first two cases of inheritance). Let us verify the requirements identified before one by one:

(1) For A $\longrightarrow$ B, if A is activated, then $ACT_B = r_{AB} * ACT_A$ in the top level, where r_{AB} is the weight on the link between A and B (the same below), and the bottom-up activation is $ACT_A * |F_B| \frac{|F_A| \frac{r_{AB}}{f(F_A)}}{g(F_B)} \approx r_{AB} * ACT_A$, so the overall result is $ACT_B \approx r_{AB} * ACT_A$;

(2) For A $\sim$ B, B $\longrightarrow$ C, if A is activated, then

$$\begin{aligned} ACT_B &= \sum_{F_C} \frac{\sum_{F_B \cap F_A} ACT_A \frac{r_{BC}}{g(F_B)}}{g(F_C)} \\ &= \frac{|F_C|}{g(F_C)} ACT_A r_{BC} \frac{|F_B \cap F_A|}{g(F_B)} \\ &\approx ACT_A * s_{AB} * r_{BC} \end{aligned}$$

(3) For A $\longrightarrow$ B, B $\sim$ C, if A is activated, then

$$\begin{aligned} ACT_C &= \sum_{F_B \cap F_C} \frac{\sum_{F_A} ACT_A \frac{r_{AB}}{g(F_A)}}{g(F_C)} \\ &= \frac{|F_B \cap F_C|}{g(F_C)} ACT_A r_{AB} \frac{|F_A|}{g(F_A)} \\ &\approx ACT_A * r_{AB} * s_{BC} \end{aligned}$$

(4) For A $\longrightarrow$ B, B $\longrightarrow$ C, if A is activated, then

$$\begin{aligned} ACT_C &= \sum_{F_C} r_{BC} \frac{\sum_{F_A} ACT_A \frac{r_{AB}}{f(F_A)}}{g(F_C)} \\ &\approx ACT_A * r_{AB} * r_{BC} \end{aligned}$$

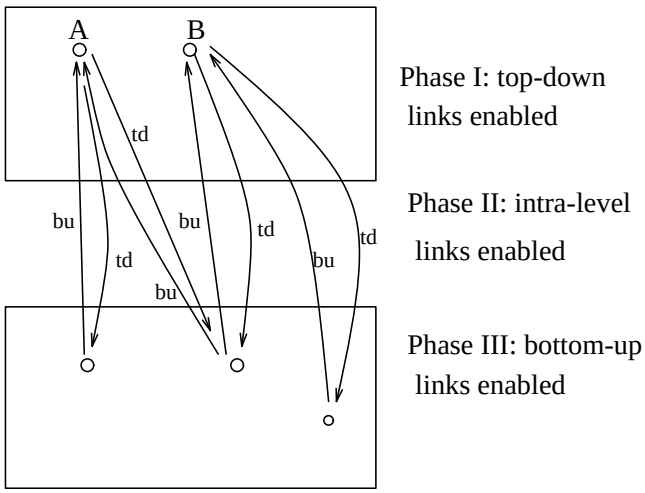


Figure 7: Similarity Matching

Let us look into similarity cases. Given the basic desiderata for similarity, we can think of many different measures (cf. [38], [22], and [11]), such as

$$s_{AB} = f1(|F_A \cap F_B|) - f2(|F_A - F_B|) - f3(|F_B - F_A|)$$

i.e. the contrast model of Tversky [38].

Or

$$s_{AB} = \frac{f1(|F_A \cap F_B|)}{f2(|F_A - F_B|) + f3(|F_B - F_A|)}$$

i.e. the ratio model of Tversky [38]. Yet others include

$$s_{AB} = \frac{f1(|F_A \cap F_B|)}{f2(|F_A|) + f3(|F_B|)}$$

$$s_{AB} = \frac{f1(|F_A \cap F_B|)}{f2(|F_A|) * f3(|F_B|)}$$

$$s_{AB} = \frac{f1(|F_A \cap F_B|)}{f3(|F_B|)}$$

Many more models can be constructed. However, when we measure them against our previous desiderata, only the last one is acceptable, because it does not involve F_A.

Looking at the matter from a different perspective, considering the implementational issues, we want as simple a formula as possible, not in terms of numbers of parameters or the time complexity of computation, but in terms of ease of implementing it in a connectionist fashion with a set of simple, autonomous, locally connected nodes. We want (1) all computation to be local, (2) only simple messages to be passed around, and (3) no extra nodes to be added (see Feldman [7] for similar points). With these three criteria in mind, again only the last model can be selected (details are omitted).

In Figure 7, suppose A is externally activated, then because of the microfeatures shared with A, B will later be activated. The activation of A will go top-down to its microfeature nodes through

$$
\begin{aligned}
&= |F_D| * \sigma * |F_B| * \lambda * \frac{r_{BD}}{f(|F_B|)} \\
&= \sigma * \lambda * r_{AC} * |F_B| * |F_D| * \frac{1}{f(|F_B|)} \\
&= \sigma * \lambda * r_{AC} * |F_D|
\end{aligned}
$$

provided f is the identity function and $r_{AC} = r_{BD}$, where λ is the activation of the microfeature nodes of A, which are all the same. Comparing the two formulas, clearly $D > C$, which is correct. But if we try to have a linear function or a function faster than linear, we will not get the correct result. For example, let $g(x) = x$, and then we have

$$bu_C = \frac{1}{|F_C|}$$

and

$$bu_D = \frac{1}{|F_D|}$$

We derive C and D the same way as before:

$$
\begin{aligned}
ACT_C &= \sum_{F_C} bu_C * \sum_{F_A} lw_{AC} * \lambda \\
&= r_{AC} * \lambda
\end{aligned}
$$

$$
\begin{aligned}
ACT_D &= \sum_{F_D} bu_D * \sum_{F_B} lw_{BD} * \lambda \\
&= r_{BD} * \lambda
\end{aligned}
$$

or C = D, which is wrong. So for this case we must have a function g that is slower than the identity function (or slower that linear functions in general, but not necessarily a constant). It is easy to verify that we will have $D > C$ with such a g.

Combining results from the above two cases, we conclude that g, as part of bu, has to be a function that is slower than linear, but faster than constants.

Although the above derivation assumes that $f(x) = x$, as we have shown before, $f(x)$ has to be slower than linear. To right the situation, we just have to make $f(x)$ as close to linear as possible, so that the non-linearity of $f(x)$ will not affect the obtained relation ($C > D$ or $D > C$), given the ranges of $|F_A|, |F_B|, |F_C|, |F_D|$. For example, we can choose $f(x) = x^{999/1000}$ and $g(x) = x^{9/10}$ for our applications.

For the first two cases of inheritance, they can be handled by a mixture of rule application and similarity matching. Case one can be described as $B \sim A, A \longrightarrow C$, and can be handled as mixed rule application and similarity matching. As will be analyzed later, if B is activated,

$$ACT_C = ACT_B * s_{BA} * r_{AC}$$

Case two can be described as $A \sim B, B \longrightarrow D$, and, as will be shown later, if A is activated,

$$ACT_D = ACT_A * s_{AB} * r_{BD}$$

$$= \sigma * \lambda * r_{AC} * |F_A|(|F_C| * \frac{1}{|f(F_A|)} + |F_D - F_C| * \frac{1}{f(|F_B|)})$$
$$= \sigma * \lambda * r_{AC}(|F_C| + |F_A| * |F_D - F_C| * \frac{1}{|F_B|})$$

provided that f is the identity function and $r_{AC} = r_{BD}$. Comparing the two formulas, clearly $D > C$, which is wrong. On the other hand, if we have bu inversely proportional to the size of the microfeature set of the CL node with which the bu is associated, that is,

$$bu_C = \frac{1}{|F_C|}$$

and

$$bu_D = \frac{1}{|F_D|}$$

then we will wind up having, obtained the same way as before,

$$\begin{aligned} ACT_C &= \sum_{F_C} bu_C * \sum_{F_A} lw_{AC} * \lambda \\ &= r_{AC} * \lambda \end{aligned}$$

and

$$\begin{aligned} ACT_D &= \sum_{F_C} bu_D * \sum_{F_A} lw_{AC} * \lambda + \sum_{F_D - F_C} bu_D * \sum_{F_A} lw_{BD} * \lambda \\ &= \lambda * r_{AC} * (\frac{|F_C|}{|F_D|} + \frac{|F_D - F_C|}{|F_D|}\frac{|F_A|}{|F_B|}) \\ &< r_{AC} * \lambda \end{aligned}$$

or C > D, which is correct. So what we can do now is simply to have a function, g, that is faster than a constant (with coefficients properly adjusted to guarantee that the asymptotic properties also hold for small values), and to let

$$bu_C = \frac{1}{g(|F_C|)}$$

for all C. Examples of such g include $\sqrt{x}$, $\log x$, or the identity function or other linear functions, etc, with coefficients equal to 1.

In Case Six (Figure 5), we want the opposite: D > C. We can perform a similar analysis. In this case, we would rather have little or no influence from the sizes of the microfeature sets. It is easy to see why: If we assume that $bu = \sigma$ (i.e. g(x)=$\frac{1}{\sigma}$), then, derived the same way as before,

$$\begin{aligned} ACT_C &= \sum_{F_C} bu_C \sum_{F_A} \lambda * lw_{AC} \\ &= |F_C| * \sigma * |F_A| * \lambda * \frac{r_{AC}}{f(|F_A|)} \\ &= |F_C| * \sigma * \lambda * r_{AC} \end{aligned}$$

and

$$ACT_D = \sum_{F_D} bu_D \sum_{F_B} \lambda * lw_{BD}$$

$$= \sigma|F_A|\frac{r}{f(|F_B|)}\lambda$$

so C is activated more strongly than D. In the other case, if B is activated (with activation value λ; the same for its microfeature nodes),

$$\begin{aligned} ACT_D &= \sigma\sum_{F_B} lw_{BD}\lambda \\ &= \sigma|F_B|\frac{r}{f(|F_B|)}\lambda \end{aligned}$$

$$\begin{aligned} ACT_C &= \sigma\sum_{F_A} lw_{AC}\lambda \\ &= \sigma|F_A|\frac{r}{f(|F_A|)}\lambda \end{aligned}$$

so D is activated more strongly than C.

We are now ready to examine the cases five and six, which are more complicated: because F_C is embedded in F_D (because C is a superclass of D, as explained before), it is imperative that we pick the right *bu* function that takes into account all effects, desirable or undesirable, of sizes of microfeature sets. Look at Case Five (Figure 4). In order to have C activated more strongly than D, [5] we have to take into account the sizes of the microfeature sets of C and D (i.e., F_C and F_D) in determining *bu*. And the *bu* should be inversely related to the size of the microfeature sets of the node in the top level with which the particular *bu* is associated. Assume all microfeatures of C and D are activated to the same degree, and F_C is embedded in F_D, i.e., C has fewer microfeatures than D; thus if we have a uniform *bu* (equal to some σ), we will have an incorrect result (D being more strongly activated than C), that is,

$$\begin{aligned} ACT_C &= \sum_{F_C} bu_C * (activation\ \ of\ \ each\ \ node\ \ in\ \ F_C) \\ &= \sum_{F_C} bu_C \sum_{F_A} \lambda * lw_{AC} \\ &= |F_C| * \sigma * |F_A| * \lambda * \frac{r_{AC}}{f(|F_A|)} \\ &= |F_C| * \sigma * \lambda * r_{AC} \end{aligned}$$

provided f is the identity function, where λ is the activation of the microfeature nodes of A, which are all the same;

$$\begin{aligned} ACT_D &= \sum_{F_D - F_C} bu_D(activation\ \ of\ \ each\ \ node\ \ in\ \ F_D - F_C) + \\ &\quad \sum_{F_C} bu_D(activation\ \ of\ \ each\ \ node\ \ in\ \ F_C) \\ &= \sum_{F_D - F_C} bu_D \sum_{F_A} \lambda * lw_{BD} + \sum_{F_C} bu_D \sum_{F_A} \lambda * lw_{AC} \\ &= |F_D - F_C| * \sigma * |F_A| * \lambda * \frac{r_{BD}}{f(|F_B|)} + |F_C| * \sigma * |F_A| * \lambda * \frac{r_A C}{f(|F_A|)} \end{aligned}$$

[5] Let us assume that there is nothing going on in the top level, and all activations come bottom-up.

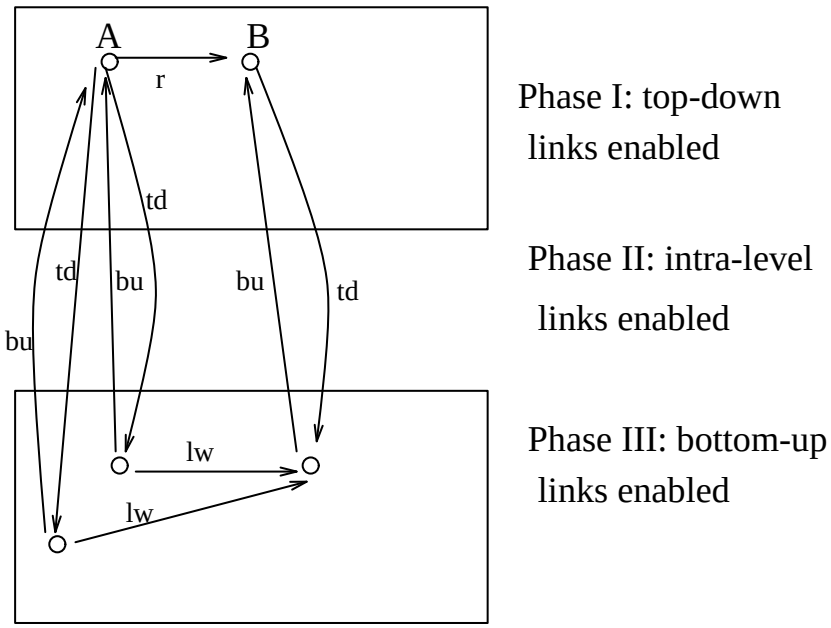


Figure 6: A Generic Model

F_C be its microfeature set.

$$lw_{AC} = \frac{r}{f(|F_A|)}$$

for any A and C, where f is a monotonic increasing function, linear or otherwise. Similarly, when B is activated (but not A), we want D to be activated more strongly than C in the bottom level. Because the microfeature set of A is a subset of that of B (as explained before), the total activation transmitted to C or D should be related to the sizes of the respective microfeature sets. Otherwise C and D will receive the same amount of activation, and therefore it will become impossible to differentiate the two. Since the total activation is equal to the size of the microfeature set of the originating concept times the activation transmitted along each individual link, to make sure that D is activated more strongly than C in this case, we must make f slower than linear, so that the total activation transmitted will be related to the size of the microfeature set of the originating concept.

It is easy to confirm that no matter what bu and td are used, with this lw function, C and D in the bottom level (F_C and F_D) will have the right activation in both cases. The details follow: suppose A is activated, and λ is the activation of the microfeature nodes of A due to top-down activation, and σ is the bottom-up weight (the same for both C and D, because they both have only one microfeature node):

$$\begin{aligned} ACT_C &= \sigma \sum_{F_A} lw_{AC}\lambda \\ &= \sigma |F_A| \frac{r}{f(|F_A|)}\lambda \end{aligned}$$

$$ACT_D = \sigma \sum_{F_A} lw_{BD}\lambda$$

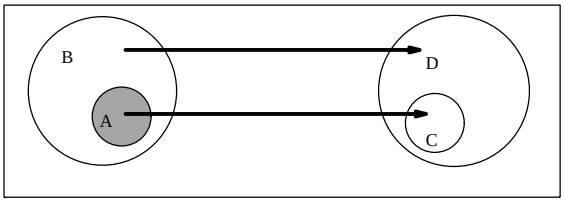


Figure 4: Inheritance Case V

$A \supset B$. $F_A \subset F_B$. $C \supset D$. $F_C \subset F_D$. Only the feature sets of the respective concepts are shown here. One arrow is drawn in place of pairwise connections.

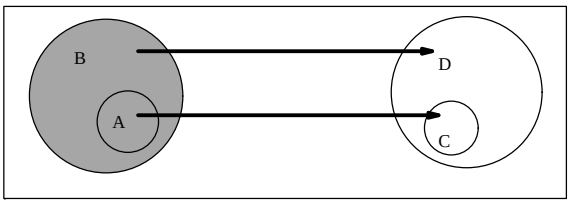


Figure 5: Inheritance Case VI

$A \supset B$. $F_A \subset F_B$. $C \supset D$. $F_C \subset F_D$. Only the feature sets of the respective concepts are shown here. One arrow is drawn in place of pairwise connections.

application problem, and the inheritance problem, with the same set of (appropriately set) parameters. We will call this structure CONSYDERR, which stands for *a CONnectionist SYstem with Dual-representation for Evidential Robust Reasoning.*

We will use continuous activation values (for representing confidence values), approximately between -1 and 1, in which 1 represents full confidence, 0 represents unknown, and -1 represents full negative confidence. We will not use thresholds for the simple reason that we adopted continuous activation values which represents continuous confidence values.

3.2 Parameters Derivation

Below we will analyze the requirements and derive the parameter. Assume in the following discussion the original rule strength in the top level connecting a concept and its property values are all the same, that is, the maximal value 1, for the sake of simplifying the discussion. We first direct our attention to inheritance/cancellation. Consider the cases three and four. When A is activated (but not B), we want C to be activated more strongly than D in the bottom level. Then during the bottom-up process, these activation values will be transmitted to the corresponding concept nodes. [4] To make sure that C is activated more strongly than D, lw (weights on the links that diffusely replicate the link in the top level) should be somehow inversely related to the size of the microfeature set of the originating concept, Assume the original link weight is r, and the weights for links in the bottom level that replicate (diffusely) the original link are (uniformly) lw. Let A be the source node, and F_A be its microfeature set. Similarly, let C be the destination node, and

[4] As a general assumption, we take the MAX of the corresponding top level activation values and the bottom-up values, which represents the combination of the results of the concept level and the microfeature level.

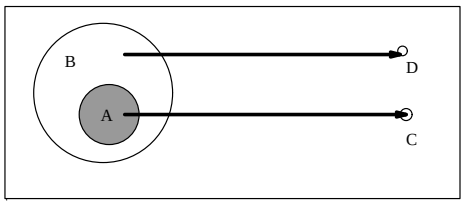


Figure 2: Inheritance Case III

$A \supset B$. $F_A \subset F_B$. Only the feature sets of the respective concepts are shown here. One arrow is drawn in place of pairwise connections.

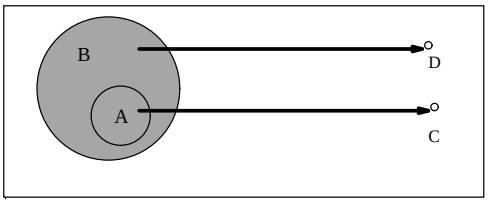


Figure 3: Inheritance Case IV

$A \supset B$. $F_A \subset F_B$. Only the feature sets of the respective concepts are shown here. One arrow is drawn in place of pairwise connections.

may bear relationships to each other as superclass/subclass. Therefore it is necessary to consider their mutual interaction, and to organize knowledge around an inheritance hierarchy for economical storage and organization. Let A $\supset$ B, and therefore $F_A \subset F_B$; that is, the larger the extension the smaller the intension; if A is a superset of B, the intension of A is the subset of the intension of B (see [19]). The following cases should be checked (see [36] for details):

(1) A has a property value C, and B has no specified property value. If B is activated, then C should be activated.

(2) B has a property value D, and A has no specified property value, if A is activated, D should be activated too.

(3) A has a property value C and B has a property value D $\neq C$. if A is activated, C should win over D.

(4) A has a property value C and B has a property value D $\neq C$. if B is activated, D should win over C.

(5) A has a property value C which has a feature set F_C, and B has a property value D with $F_D \subset F_C$. If A is activated, C should win over D.

(6) A has a property value C which has a feature set F_C, and B has a property value D with $F_D \subset F_C$. If B is activated, D should win over C.

With all these constraints established, we can proceed to derive the exact specifications of the parameters, including top-down weights (denoted as td), bottom-up weights (denoted as bu), and weights (denoted as lw) for links between two microfeature nodes which diffusely replicate the rule links r (see Figure 6). **This structure can solve the similarity matching problem, the rule**

an incomplete knowledge base. A similarity measure has the following requirements (see [34] and references cited therein for detailed justifications; they are too long to replicate here):

$s_{AB} \propto |F_A \cap F_B|$, that is, the similarity between two concepts is proportional to the amount of their microfeature overlapping.

$s_{AB} \propto \frac{1}{|F_B|}$, that is, the similarity is inversely proportional to the number of microfeatures B has, when everything else is equal.

$s_{AB} \not\propto \frac{1}{|F_A|}$, that is, the similarity is not (inversely or not) proportional to the number of microfeatures A has, when everything else is equal.

Rules. Rules are needed for expressing precisely the knowledge a system does possess. Coupling such precise knowledge statements with similarities, a lot of plausible inferences can be made. The following cases of rules and mixed rules/similarities have respective requirements: [3]

(1) A $\longrightarrow$ B: if A is activated, then $ACT_B = r_{AB} * ACT_A$, where ACT_x is the activation value of x and r_{AB} is the strength of the rule between A and B (the same below);

(2) A $\sim$ B, B $\longrightarrow$ C: if A is activated, then $ACT_B = s_{AB} * ACT_A$, and $ACT_C = r_{BC} * ACT_B$, where s_{AB} is the similarity between A and B (the same below);

(3) A $\longrightarrow$ B, B $\sim$ C: if A is activated, then $ACT_B = r_{AB} * ACT_A$, and $ACT_C = s_{BC} * ACT_B$;

(4) A $\longrightarrow$ B, B $\longrightarrow$ C: if A is activated, then $ACT_B = r_{AB} * ACT_A$, and $ACT_C = r_{BC} * ACT_B$;

(5) A $\longrightarrow$ B, B $\longrightarrow$ C, C $\longrightarrow$ D: if A is activated, then $ACT_B = r_{AB} * ACT_A$, $ACT_C = r_{BC} * ACT_B$, and $ACT_D = r_{CD} * ACT_C$;

(6) A $\sim$ B, B $\longrightarrow$ C, C $\longrightarrow$ D: if A is activated, then $ACT_B = s_{AB} * ACT_A$, $ACT_C = r_{BC} * ACT_B$, and $ACT_D = r_{CD} * ACT_C$;

(7) A $\longrightarrow$ B, B $\sim$ C, C $\longrightarrow$ D: if A is activated, then $ACT_B = r_{AB} * ACT_A$, $ACT_C = s_{BC} * ACT_B$, and $ACT_D = r_{CD} * ACT_C$;

(8) A $\longrightarrow$ B, B $\longrightarrow$ C, C $\sim$ D: if A is activated, then $ACT_B = r_{AB} * ACT_A$, $ACT_C = r_{BC} * ACT_B$, and $ACT_D = s_{CD} * ACT_C$;

(9) A $\sim$ B, B $\longrightarrow$ C, C $\sim$ D: if A is activated, then $ACT_B = s_{AB} * ACT_A$, $ACT_C = r_{BC} * ACT_B$, and $ACT_D = s_{CD} * ACT_C$.

Inheritance. Inheritance is the inference regarding a concept (or a primitive knowledge statement) that is based on knowledge (statements) associated with a superclass or a subclass of the concept. This problem is important, because different concepts (primitive knowledge statements)

[3] We only deal with rules with only a single premise here; rules with multiple premises are just extensions of these cases (see [35] for a detailed treatment). Note also that, unlike symbolic systems, here various sorts of chaining have to be dealt with on a one-by-one basis.

because they share some common microfeatures due to similarity; so some of the microfeature representations of concepts will be partially activated if a concept *similar* to them is activated (in the bottom level). Finally in the bottom-up phase, fully or partially activated microfeature representations will go back up to activate the corresponding nodes in the top level. The result can be read off from the top level. [1]

Notice the massive parallelism in the above specified structure: activations are propagated, in a massively parallel fashion, from all pre-link nodes to all post-link nodes; each node receives inputs as soon as it can, and therefore fires as soon as it can, ensuring a maximum degree of parallelism in terms of rule application. In terms of similarity matching, all similar concepts are activated (in their microfeature representations) immediately once an original concept is activated, and simultaneously matched with the original one (through top-down and bottom-up flows); thus the structure is extremely efficient by employing the two level structure. The parallelism in this structure accounts well for the similar parallelism and spontaneity in human reasoning processes as identified in, for example, Collins & Michalski [4] and Sun & Waltz [33]. For other connectionist models handling rule-based reasoning, see Gallant [9], Lange & Dyer [18], Sun [27], and Sun & Waltz [33], etc. For similar two-level approaches, see Grossberg [11], Hendler [14, 15], and Gelfand et al [10].

One problem that we are not addressing here is the variable binding problem, that is, how we can associate arguments with predicates in inferences. This is especially difficult for connectionist networks (which we use as implementational means) to handle variable binding, because of the simplicity and homogeneity of such networks. Nevertheless, such networks are able to handle variable binding to a certain extent (see [35] for details).

3 Fine-Tuning the Structure

3.1 Basic Desiderata

To show that the two level structure proposed above is versatile enough to accommodate special requirements that are often associated with various kinds of reasoning tasks, we will see how a set of desiderata for dealing with the geographical commonsense reasoning tasks (see [34]) can be used to determine the parameters of this structure.

All of the desiderata and the requirements for some commonsense reasoning in geographical domains are determined in [34] and they can be summarized together as follows (divided into three categories: rule application, similarity matching, and inheritance):

Similarity. A similarity measure s_{AB} is for measuring the similarity between A (the target) and B (the source), namely, "$A \sim B$". [2] It is needed because in vague domains we can and must reason based on similarities of knowledge statements, in order to reach plausible conclusions with

[1] Each node in the system has one or more sites (cf. [7]), each of which computes the weighted-sum (or any other similar functions whenever needed) of the inputs. The maximum of the values computed by all the sites is taken to be the activation value of the node.

[2] Here similarity from A to B means that, when there is no direct knowledge about the concept A available, the concept B, which is similar to A, can be utilized to find plausible answers. This situation, which can also be termed similarity-based induction, is different from the generic, context-free notion of similarity.

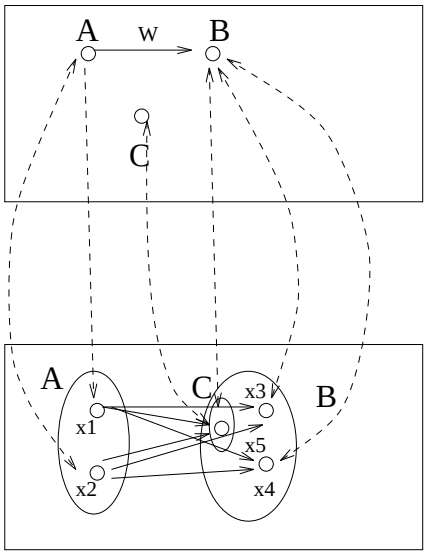


Figure 1: The Two Level Structure

For the settling phase, in the top level

$$ACT_C(t+1) = \sum_i r_i * ACT_{A_i}(t)$$

and in the bottom level

$$ACT_y(t+1) = \sum_j lw_j * ACT_{x_j}(t)$$

where r's and lw's are link weights (representing rule strength), and lw's can be determined from corresponding r's (as will be shown later); A_i's and x_i's are the activations of nodes that are related respectively to C and y's by links (rules). That is, in this case, each node receives activations from other nodes at the same level (which are related to it by rules) and does a weighted-sum for computing its own activation.

For the bottom-up phase,

$$ACT_C(t+1) = \max(ACT_C(t), \sum_{y_i \in F_C} bu_C * ACT_{y_i}(t))$$

where C is any top level node, y_i's are its corresponding microfeature nodes; bu is a weight (to be determined later). That is, a concept node receives activation from its corresponding microfeature node, and chooses the value as its activation if it is greater than its original activation.

When applying this cycle, first some nodes in the top level get activated by external inputs (and clamped). Then the top-down phase will activate (and clamp) the microfeature nodes corresponding to the active concept nodes. In the settling phase, links representing rules related to those activated nodes take effect in both levels. Concepts may have overlapping microfeature representations

reasoning capability in dealing with vague domains. An example from the commonsense geographical domain is as follows: "Columbia basin" is described as tropical river-basin and coastland, and "Ecuador coast" is described as a tropical, coastal lowland with rainforest cover. In order to deduce possible agricultural products of "Columbia basin", we notice its similarity to "Ecuador coast" in terms of their geographical features. Since the latter produces (among other things) bananas, we can conclude that the former may produce bananas too. For another example, "Northern Brazil" is described as a tropical, hilly plateau with rainforest, and "Bolivia oriente rainforest area" is described as tropical plain and lowland with rainforest. The former is a rubber producing area. Because of the feature similarity, we might conclude that the latter is likely to be one too.

To sum up, some precisely specified rules are needed, and in addition to such rules, we also need to deal with vagueness in a domain by utilizing continuous numerical evidential combination and similarity-based inferences. Similarities can be explored based on features (or microfeatures, as in [25]) to provide a fuller coverage of all the situations in a domain.

2.2 A Two level Model

We are now ready to present a two level approach for structuring knowledge to take care of the two types of vagueness. This structure was proposed in [34], and we will summarize important points below. One level of this structure is the *concept level*, which contains primitive knowledge statements, or concepts. This level consists of a collection of *nodes*, or processing elements, for representing these concepts in the domain. For expressing compound knowledge statements, or rules, these nodes are connected via links from antecedents of a rule to consequents of the rule. This level is the top level in Figure 1. The other level (the bottom level) is the *microfeature level*, which contains nodes each of which represents a fine-grained element (a microfeature) in the meanings of the concepts represented in the top level. Each node in the top level is connected to all the relevant microfeature nodes in the bottom level; once a concept node is activated, the related microfeature nodes will be activated subsequently from inter-level connection, and vice versa. Links in the top level are replicated diffusely in the microfeature level by multiple links between two sets of microfeature nodes representing an antecedent and a consequent of a rule respectively. The first type of vagueness is handled by utilizing weighted sum computation in each nodes, which is a continuous mapping, accumulating evidence from different sources (see [33, 35]), in computing its output activation. It is proven that such a function can actually implement Horn clause logic as a special case [35]. The second type of vagueness is handled by a *similarity matching process* based on microfeatures. In this structure, similarity matching is accomplished through the interaction between the two levels, by a top-down/settling/bottom-up cycle (see Figure 1).

Equations for the computation of the three phases is specified as follows:

For the top-down phase,

$$ACT_{x_i}(t+1) = \max_A(td_A * ACT_A(t))$$

where ACT is the activation value of a node and A is any node in the top level that has $x_i \in F_A$; td is a weight (to be determined later). That is, a microfeature node receives activation from the corresponding concept nodes, and chooses the largest value.

and explanations needed for user interaction. We simply choose *Horn clauses* as the form of such knowledge statements (see [3]). In its simplest case, a Horn clause just states a proposition or a concept, without any pre-conditions or possible consequences. In more complex cases, a Horn clause is a rule consisting of a number of antecedents, which are simple propositions/concepts, and a consequent, which is also a simple proposition/concept. We choose Horn clause because of the simplicity of the formalism, its popularity in knowledge-based systems, its expressive power, and its inferential efficiency [3]. Though Horn clause logic will be the basic form of our knowledge base, we will add various additional mechanisms for dealing with vague domains.

Second, we need to have a better grasp of the various kinds of vagueness in our domains. For one thing, there should be a degree of *confidence* or *certainty* associated with each primitive knowledge statement (or its instantiation), judging how close, and/or how likely an instance of that statement is to conform to the ideal of the concept involved (see [41]). For example, "warm areas are suitable for rice growing" can be expressed in the rule

$$warm \longrightarrow \textit{rice-growing}$$

Here "warm" is not a crisp concept (or knowledge statement); various degrees of "warm" exist. We should allow a degree of confidence to be associated with the knowledge statement. We can accept as input to a system an instance of the knowledge statement of a place being "warm" with an associated confidence degree. We then match it with an existing rule (or compound knowledge statement), to form an instantiation of the rule. And we derive the conclusion knowledge statement of a place being "rice-growing" with a corresponding degree of confidence determined by, among other things, the degree of confidence of being warm.

We also need a way of associating a confidence value with a rule itself, determining how likely it will hold. Moreover, various vague evidence leading to the same conclusion should be summed up somehow; that is, the degrees of confidence of different pieces of evidence in a rule have to be *accumulated* in some way, for example, by a weighted-sum. An example is as follows,

$$\textit{subtropical rainy flat evergreen} \neg \textit{flood} \longrightarrow \textit{rubber-producing-area}$$

That is, an area with subtropical and rainy weather, flat terrain, and evergreen vegetation cover but no frequent flood tends to be an rubber-producing area. Let us consider some possible scenarios. If one only knows that an area is subtropical and rainy but without flood, one may still conclude that it could be a rubber-producing area; if one knows that an area is subtropical, rainy, with flat terrain, and without flood, one may conclude that it is more likely to be a rubber-producing area; if one knows all of the above conditions, one can conclude that the likelihood is very high. A cumulative evidentiality is in working here, which should be taken into consideration in our system.

There is yet another type of vagueness, which can be further pinned down. Basic concepts (or primitive knowledge statements) are *similar* to each other to varying degrees. Even though something is best described by one particular knowledge statement, other statements may also apply. Especially when there is no inference that can be made with the best matched case, turning to other related cases will definitely be of help. This is in a way similar to analogical reasoning (or case-based reasoning), in that some seemingly different knowledge is brought together from a partial match of two chunks of knowledge. In some sense, we need a rudimentary form of analogical

In our discussion here, we generally assume that knowledge in a domain is composed of knowledge *statements*, such as Horn clauses, which could be atomic propositions or rules consisting of a number of atomic propositions expressing antecedents and consequents respectively. This is an acceptable assumption, because almost all knowledge-based systems are in such forms (or can be transformed into such forms, for example, frames, scripts, etc.; see [12]). One exception is neural networks, which encode knowledge in a set of numerical weights, which are difficult to interpret in terms of rules (but not impossible to interpret; cf. [8], [37], and [17]). (One of the goals of this paper is to show that there is an alternative to the black-box style neural network models, and our model not only serves as an alternative but also has some important advantages.) From here on, when we discuss a domain, we will think of it as composed of a *space* of *primitive statements* (i.e., a set of all possible primitive statements), and the main operation in that space is match. When two statements are deemed matching each other, inferences are enabled by associating knowledge of one statement with the other. For example, suppose we have the following space of primitive statements: $\{a, b, c, d, e, f\}$, and the following associated compound knowledge statements: $c \longrightarrow d$ and $e \longrightarrow f$. When given a, we search for matching statements. Suppose e matches a (exactly or approximately), then the knowledge statement $e \longrightarrow f$ can then be applied, and we can deduce f (exactly or approximately).

We will be mainly concerned with the *vague* domains. By vague domains, we mean the type of domains that are composed of inexact statements (fuzzy, probabilistic, etc.), and the match between two statements in a domain is not all-or-nothing: a continuum can be formed that ranges from perfect (exact) match to no match (irrelevance) to the exact opposite (the negative perfect match). The key features of such domains are (1) inexactness, allowing partial match situations, and (2) continuity, with varying degrees of partial match.

Given the above, it is impossible to have knowledge statements that cover all possible situations in such domains: the number of possible situations can be infinite, because a continuum can exist going from one statement (any particular statement) to the statement that is the exact opposite. Given this infinite space, we need to structure our knowledge statements economically, in ways that cover a domain as accurate as possible within the constraint of resources. We also have to consider how to structure knowledge in ways that are most helpful in guiding and facilitating reasoning in a domain.

In the rest of this paper, we first provide some discussion of vagueness, and then we propose a two level structuring of knowledge for vague domains. We consider a particular set of requirements for plausible inferences from a knowledge base, and we derive the parameter values of a two-level system satisfying the set of requirements. A set of experiments are presented to further illustrate the structure. Finally, brief comparisons and conclusions complete the paper.

2 Developing A Two Level Structure

2.1 Some Analyses

Let's consider how we can better structure our knowledge, given the aforementioned considerations. First of all, we need to have explicit knowledge statements in our system. This is because explicitly stored knowledge statements can provide clarity, modularity, human comprehensibility,

1 Introduction

One of the most important problems in building intelligent systems is how to represent knowledge, that is, how to structure knowledge in a way that facilitate common types of inferences to be made. For research and practice in knowledge-based systems, some types of logic or rule-based frameworks are usually adopted ([44] and [13]). However, they are far away from matching the capacity and flexibility of human reasoning (see [4] and [39]). What is the problem?

Knowledge is hard to grasp, as discovered by many leading researchers (cf. Minsky [21] and Hayes-Roth et al [13]), notwithstanding the fact that small chunks of it for a narrowly defined domain can be extracted and structured into rule-based, frame-based, or other similar systems. Psychological experiments reveal that in human cognition various kinds of knowledge exist and they are used in different ways [22]. Many different types of inferences can be performed in commonsense reasoning, sometimes based on the same basic set of knowledge [20]. Besides, most parts of knowledge are uncertain, fuzzy, or probabilistic (cf. [21], [41], and [6]).

In view of these problems, in order to build intelligent systems that are more capable in producing flexible, plausible and useful inferences, we do need to sharpen up our means for representing knowledge; that is, we need to structure knowledge in ways that maximize the inferential capability while minimizing the inferential complexity (we do have to consider the *performance* issue). This is a tough goal to achieve, since the two subgoals are apparently mutually contradictory, but this goal nevertheless can serve as an ideal to strive for. (For some domains which we will look into here, some nice balance between the two subgoals is indeed achievable.)

Some attempts at providing more flexibility to intelligent systems result in some partially successful models for representing knowledge and/or performing reasoning. One such model is fuzzy logic [43], which allows vague concepts and deals with vagueness in reasoning. It envisages a concept as a set of objects each of which belongs to the set to a certain degree, as measured by a *grade of membership*. Thus, vague concepts are accommodated and objects may belong to a particular concept *partially*, without the need of forcing a dichotomatical true/false decision. We will adopt this view of concepts as partially true descriptions of objects, but will not adopt the logical operations defined on top of that (i.e., MIN/MAX, as in [43]).

Beside fuzzy logic, there are a number of other rule-based approaches, for example, probabilistic reasoning [21], Dempster-Shafer evidential reasoning [26], and so on, that are good at capturing some aspects of the vagueness of real world knowledge. However, they are not *meant* to deal with other aspects of vague knowledge.

Another popular approach is that of PDP models (see [25] and [1]). Using networks of simple processing elements, typically with a global multilayer feedforward structure and with a continuous sigmoidal activation function in each processing element, these models are capable of virtually any continuous mapping [16], including dealing with flexible reasoning based on similarity as mappings between two sets of concepts. This purely similarity-based, generic framework can be applied to a large variety of knowledge intensive tasks, by encoding knowledge distributively in link weights acquired from applying learning algorithms. However, this type of models does lack certain basic characteristics for flexible yet precise commonsense reasoning, namely, explanation, symbolic processing, interaction with human experts in acquiring and modifying knowledge, interpretation of internal knowledge, and handling of discontinuous cases (see [33]; more on this later).

Structuring Knowledge In Vague Domains

Ron Sun
Department of Computer Science
College of Engineering
The University of Alabama
Tuscaloosa, AL 35487

rsun@cs.ua.edu

January 10, 1993

INDEX TERMS: artificial intelligence, knowledge-based systems, neural networks, reasoning, knowledge representation, vagueness

Abstract

In this paper, we propose a model for structuring knowledge in vague and continuous domains where similarity plays a role in coming up with plausible inferences. The model consists of two levels, one of which is an inference network with nodes representing concepts and links representing rules connecting concepts, and the other is a microfeature based replica of the first level. Based on the interaction between the concept nodes and microfeature nodes in the model, inferences are facilitated and knowledge not explicitly encoded in a system can be deduced via mixed similarity matching and rule application. The model is able to take account of many important desiderata of plausible reasoning, and produces sensible conclusions accordingly. Examples will be presented to illustrate the utility of the model in structuring knowledge to enable useful inferences to be carried out in several domains.