A Connectionist Model for Commonsense Reasoning Incorporating Rules and Similarities

Ron Sun
Honeywell SSDC
3660 Technology Dr.
Minneapolis, MN 55413
rsun@orion.ssdc.honeywell.com
(612) 782-7379

-

Running Title: Connectionist reasoning

bigskip bigskip

To appear in: Knowledge Acquisition. Academic Press, Cambridge, UK. 1992

SUMMARY

For the purpose of modeling commonsense reasoning, we investigate connectionist models of rule-based reasoning, and show that while such models can usually carry out reasoning in exactly the same way as symbolic systems, they have more to offer in terms of commonsense reasoning. A connectionist architecture, CONSYDERR, is proposed for capturing certain commonsense reasoning competence, which partially remedies the brittleness problem in traditional rule-based systems. The architecture employs a two-level, dual representational scheme, which utilizes both localist and distributed representations and explores the synergy resulting from the interaction between the two. CONSYDERR is therefore capable of accounting for many difficult patterns in commonsense reasoning with this simple combination of the two levels. This work shows that connectionist models of reasoning are not just "implementations" of their symbolic counterparts, but better computational models of commonsense reasoning.

1 Introduction

There has been quite a lot of research being done in understanding commonsense reasoning in artificial intelligence. There is little doubt that rule-based reasoning has been the most prominent and the most successful paradigm of traditional symbolic artificial intelligence, including work in understanding and modeling commonsense reasoning, up to now. Based on this paradigm, numerous AI systems for theoretical or practical purposes have been built and/or deployed. Despite various problems and objections leveled against them, they have been proven to be at least partially valid models. So when connectionism came to the scene, it was natural to ask the question of how to account for this type of reasoning in connectionist models. Whether connectionist models can be a viable alternative to symbolic AI depends, to a certain extent, on their ability to account for rule-following behaviors and rule-based reasoning (cf. Pinker & Prince 1988, Fodor & Pylyshyn 1988, and Touretzky & Hinton 1985), which are important aspects of commonsense reasoning.

To account for rules in connectionist models, some approaches have been explored, and several different systems have been implemented that can carry out rule-based reasoning rather completely. However, most of them are straight "implementations" of symbolic rule-based reasoning, that is, they carry out symbolic rule-based reasoning *faithfully* in a connectionist framework, serially or in parallel, without any fundamental difference from symbolic systems in terms of logical capabilities or reasoning capacities (especially from the standpoint of modeling commonsense reasoning).

In order to really justify connectionist models in terms of their ability of modeling various kinds of commonsense reasoning, we have to ask the following questions: Can connectionist models do *more and better* in terms of accounting for robust, flexible, and multifaceted human commonsense reasoning, especially in the types of reasoning that are most often dealt with by rule-based paradigms? Can those data (such as in Collins & Michalski 1989) that are difficult to account for by existing symbolic rule-based systems be explained by connectionist rule-based reasoning models? There is no way we can answer these questions abstractly from a purely theoretical standpoint. Instead we will have to answer these questions by building connectionist reasoning systems that *do* account better for commonsense reasoning. This work develops such a system and shows that connectionist models of reasoning are not just an “implementation” of their symbolic counterparts; rather, they are better computational models of commonsense reasoning, taking into consideration the approximate, evidential and flexible nature of rule-based reasoning, and similarity-based reasoning (a limited form of analogy), and in the same time accounting for the spontaneity and parallelism in reasoning processes.

In the rest of the paper, we will first look into various existing models of commonsense reasoning, including rule-based and connectionist ones. Then we will identify some problems that are difficult for these models to solve. We will move on to develop a new connectionist architecture that can better deal with these problems. Examples will be explained in detail to show the working of the system. This paper, however, is concerned more with general approaches, ideas and possibilities than with precise technical details (which can be found elsewhere, as will be pointed out along the way).

2 Previous Work

Let us look into some previous work in rule-based models of commonsense reasoning, especially connectionist rule-based reasoning models. Symbolic rule-based systems have a long history in artificial intelligence and cognitive science. The early successful work such as those described in Buchanan & Shortliffe (1984) and Hayes-Roth et al (1983) (e.g., MYCIN, PROSPECTOR, and DENDRAL) demonstrated the promise of this approach, which adopts a simple representation with modular units called rules composed of a small set of conditions and conclusions. Many elaborate cognitive theories of learning, problem solving and memory, etc. were proposed based on this paradigm. For example, Klahr et al (1989) contains a large variety of them, including the ACT theory and SOAR.

Roughly speaking, symbolic rule-based systems can be classified into two general categories: systems based on formal logics and production systems. Formal logics are simple, formally defined languages, capable of expressing rules in a rigorous way (Chang & Lee 1973). There are basically two types of formal logics: propositional logic and predicate logic. Propositional logic deals with declarative sentences (propositions) that can be either true or false. Propositions can be combined, using logical connectives ($\vee$, $\wedge$, $\neg$), to form compound propositions. Many different axiomatic systems are proposed and studied, which concerns the truth value of a proposition or the relations between truth values of different propositions. Predicate logic (first order logic) is made up of predicates, which contains arguments, and therefore there are also terms for arguments and quantifiers constraining arguments in this type of logic. This type of logic is more complex because of the introduction of arguments and consequently its bindings to terms, but it is thus more expressive: it can represent n-ary relations, among other things. In formal logics, generally rules are expressed as logical implications, and they have to follow the formal definitions and the associated axioms regarding logical implications. Usually rules are encoded using the Horn clause formalism, in which conditions are the conjunction of atomic propositions and conclusions are single atomic propositions, and a logical implication connects them; for example, $A \wedge B \wedge C \longrightarrow D$ is a Horn clause. There are also many extensions to traditional (propositional and predicate) logics, such non-monotonic logics, intensional logics (including modal logics), or fuzzy logic (cf. Shoham 1990, Ginsberg 1987, and Zadeh 1983 regarding these logics). However, formal logic is all too restrictive to account for human reasoning. It has to have all conditions precisely specified in order to perform one step of inference; it does not take into account the gradedness of concepts, propositions, and rules (except fuzzy logic); there is no built-in device for inheritance, reasoning with incomplete information (except default logic), or similarity-based inference; the paradigm is plagued by the *brittleness* problem, as will be discussed later.

Production systems are no different from logics in this respect (cf. Klahr et al 1989, Posner 1989, and Anderson 1983). Production systems usually consist of a production (rule) base and a working memory and use a control structure to decide which rules are eligible to fire, how rules are picked for actual firing, and in which order they are supposed to fire. Usually, conditions of rules are matched against data in the working memory; those rules whose conditions match the working memory data are eligible to fire. Conflicts are resolved by ordering rules and firing the rule that comes first in the order. Usually the most specific rule fires first. Other policies are also used in various systems, for example, the policy of refractoriness for avoiding a high-priority rule being executed repeatedly with the same data, or the policy of giving preference to rules that match the data most recently created, or the policy of giving preference to rules with stronger conditions, etc. In more recent systems, *goal symbols* are used to control the order and to focus on relevant rules. A goal symbol can be created by a rule, and thereafter only rules that have that goal symbol in their conditional parts are allowed to fire, until new goal symbols are created. This way attention is focused on only those rules that are most relevant to the current situations, and also an order in which rules are fired can be created dynamically (Posner 1989).

Because of the (partial) success of these rule-based systems in accounting for commonsense reasoning, one of the main challenges for connectionism was how to implement rule-based reasoning in a network, or massively parallel, fashion, without losing the basic characteristics of connectionist models. This challenge was met by a number of research efforts that build connectionist systems that can perform rule-based reasoning in some way. Touretzky & Hinton (1985) is the first work towards this end. They basically emulate the structure of a symbolic rule-based (production) system, with separate modules for working memory, rules, and facts; an elaborate pull-out network is designed to match working memory data against rules and to decide which matching rule is to fire. Competition and winner-take-all are used for that purpose. The result is the equivalent of a simple sequential symbolic rule-based system (dealing only with 3-tuples consisting of one predicate and two arguments).

Barnden (1988) represents another early attack on this problem. In his system, data reside in grid-like networks (called Configuration Matrices), coded with the help of adjacency relations and highlighting techniques. Hardwired rules are used to detect the presence of data that match particular rules, and an "Action Part" module can be used to add a new data structure representing the conclusion from the matched rule. Although there is some parallelism, it is mostly a sequential rule-based system at the knowledge level, carrying out symbolic processing.

Besides these early pieces of work, there are also other efforts, for example, Dolan & Smolensky (1989), Ajjanagadde & Shastri (1989), Lange & Dyer (1989), Sun (1989) (see also Sun & Waltz 1991), and Holldobler (1990). They basically have similar functionalities. Dolan & Smolensky (1989) presents a connectionist production system that uses tensor products for variable binding. Ajjanagadde & Shastri (1989) discusses a connectionist rule-based reasoning system, which represents each constant, predicate, and argument with an individual node and uses synchronized firing of nodes for binding constants (nodes) to arguments (nodes). Their system performs backward chaining for answering queries. Lange & Dyer (1989) and Sun (1989) both discuss forward chaining inferences in feedforward networks and variable binding as propagating a distinct constant value (or a *signature*, as called in Lange & Dyer) through argument nodes. Holldobler (1990) discusses how to implement Horn clauses in conenctionist networks.

From these above examples, it is quite clear now that connectionist models are capable of implementing rule-based reasoning in a variety of ways. So now the question is: Can connectionist models account *better* for commonsense rule-based reasoning? The evidential, robust, flexible, and multifaceted nature of commonsense reasoning is evident from various studies (such as in Collins 1979 and Collins & Michalski 1989; see detailed analyses below) yet they are all absent from these above models. What is really needed for a connectionist model of rule-based reasoning to be able to model commonsense reasoning adequately? We confront the above questions, by analyzing real protocol data and then actually building a new kind of connectionist models as a computational mechanism for commonsense reasoning.

3 Common Reasoning Patterns

Allan Collins collected a number of protocols of commonsense reasoning. He indicated the inadequacy of traditional logic in explaining those reasoning patterns, and argued for the use of different (logic-based) formalisms or frameworks in the study of *common reasoning patterns* found in various commonsense reasoning tasks. Collins & Michalski (1989) have done an impressive job in terms of analyzing the data and establishing a unifying framework for explaining them. What is needed is a computational mechanism from which various inference patterns contained in the data can emerge into existence. Such a mechanism, though benefited from existing analyses, does not have to conform to any existing frameworks, especially when existing frameworks are conceptually or computationally too complex. We believe that the mechanism ought to be analytically simple, structurally unified, and mechanistically sound. Connectionist models, as uniform networks of simple processing elements, in general fit these above descriptions very well; so they might provide such a mechanism.

Let us look at some examples from Collins & Michalski (1989). One protocol is as follows:

> Q: Do you think they might grow rice in Florida?
>
> R: Yeah. I guess they could, if there were an adequate fresh water supply, certainly a nice, big, warm, flat area.

In this example, the person answering the question deduced an uncertain conclusion based on partial information, with a piece of crucial information (the presence of fresh water) missing. This example also indicates the need for an additive procedure for accumulating evidence.

Another protocol is as follows:

> Q: Is the Chaco the cattle country?
>
> R: It is like western Texas, so in some sense I guess it's cattle country.

Here because there is no known knowledge, an uncertain conclusion is drawn based on similarity with known knowledge.

Yet another protocol is

> Q: Are there roses in England?
>
> R: There are a lot of flowers in England. So I guess there are roses.

Here the deduction is based on property inheritance (England HORTICULTURE flower; rose IS flower; so England HORTICULTURE rose), and the conclusion is partially certain and can be drawn only when there is no information to the contrary (i.e. no cancellation). A lot more protocol data similar to these are analyzed, and are presented in Appendix A.

Existing connectionist models, or any computational models for that matter, so far cannot deal very well with these above patterns in a single unified model. Although some connectionist or non-connectionist systems can deal with *some* of these problems, e.g., Fuzzy logic, Dempster-Shafer, and Bayesian reasoning, etc., no one can deal with them *all* very well in a unified framework.

4 The Brittleness Problem

Those examples analyzed above are actually manifestations of a general problem that has been plaguing symbolic AI for long, namely the brittleness problem, which is especially conspicuous in commonsense domains.

Brittleness implies being easily breakable — the slightest discrepancy between the knowledge base of a system and a input situation can lead to a breakdown of the system. The brittleness problem can be delineated, for the purpose of this research, as the inability of a system to deal with the following aspects of reasoning in a systematic way within a unified framework (Sun 1991): [1]

- partial information (for example, the first protocol),

[1] One of the goals of this research is to deal (to a certain extent) with these aspects in a unified framework; by no means is implied that this work solves all the problems.

- uncertain or fuzzy information,
- lack of matching rules (for example, the second protocol),
- rule interactions, i.e., lack of consistency and completeness in a fragmented rule base,
- generalization,
- top-down inheritance (for example, the third protocol),
- bottom-up inheritance,
- learning new rules and modifying existing rules.

Detailed analyses of these aspects (with a lot of examples) show that, while they look like a disparate set of problems, they can all be characterized as reasoning with rules supplemented with (feature) similarity-related inferences. We first define a measure of **conceptual similarity** (cf. Tversky 1977) as

$$(A \sim B) = \frac{|F_A \cap F_B|}{|F_A|} \in [0,1]$$

such that

$$if\ ACT_A = a,\ then\ ACT_B = a * (A \sim B)$$

where F_i is the feature representation of node "i", and ACT_i is the activation of node "i"; assume there is nothing else affecting ACT_B. We define a measure of **knowledge links** (i.e. rules) as

$$(A \longrightarrow B) = r \in [-1,1]$$

such that

$$if\ ACT_A = a,\ then\ ACT_B = a * (A \longrightarrow B)$$

where ACT_i is the activation of node "i", and r is the knowledge link (rule) strength between A and B. When there are multiple conditions in a rule, this measure becomes a vector, and the multiplication used here is generalized to inner-products of vectors. For the full details of rule encoding, see Sun & Waltz (1991) and Sun (1992a). Each of the above protocols can be analyzed and dealt with utilizing these two concepts; for example, the lack-of-matching-rule situation can be described as:

$$A \sim B$$

$$B \longrightarrow C$$

and A is activated ($ACT_A \neq 0$). So we have

$$ACT_C = ACT_B * (B \longrightarrow C)$$

$$= ACT_A * (A \sim B) * (B \longrightarrow C)$$

Other cases can be described similarly (Sun 1991), except learning new rules, which is a separate issue.

These two mechanisms will be embedded, as explained in the next section, in our new architecture: CONSYDERR (which stands for *a CONnectionist SYstem with Dual representation for Evidential Robust Reasoning*), so each of these aspects of brittleness in commonsense reasoning can be handled by our system.

5 A Sketch of the Model

The CONSYDERR architecture consists of two levels: CL and CD. CL is a connectionist network with localist representation, with one node for each concept, or roughly reasoning at the conceptual level (cf. Smolensky 1988). Rules (knowledge links) are represented in CL as links between two nodes representing the condition and the conclusion respectively, with computations as specified above. The scheme proposed, FEL or *Fuzzy Evidential Logic*, can handle a superset of (acyclic or hierarchical) Horn clause logic (in other words, there is no recursive structure) and Shoham's modal logic (or *Causal Theory*; Shoham 1987), so that it can fully accommodate traditional rule-based reasoning and capture commonsense causal knowledge (see Sun 1992a for proofs). Moreover, it is capable of approximate and cumulative evidential reasoning and works with partial and uncertain information. Unlike Horn clause logic, it can deal with negative as well as positive evidence. However, FEL handles only acyclic rule sets. Thus it forms feedforward networks, which in general has a simpler and more tractable dynamics. The basic operation of this scheme is simply weighted-sum computations; that is, suppose A_1, A_2, A_3, and B are propositions, and a rule is as follows:

$$A_1 \wedge A_2 \wedge A_3 \longrightarrow B$$

we encode the rule by connecting the nodes for A_i's to the node for B and computing in the node B the activation:

$$ACT_B = \sum_{i=1}^{3} W_i * ACT_{A_i}$$

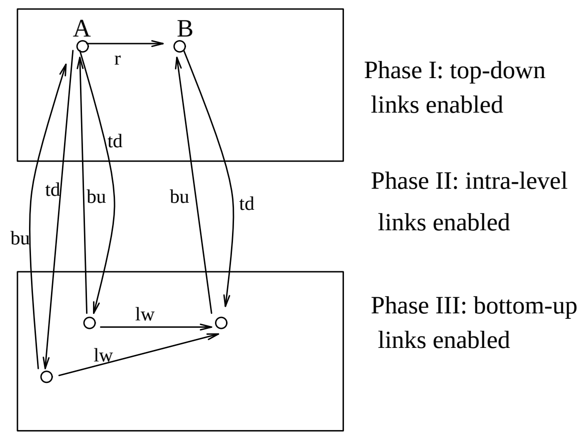


Figure 1: A Two-Level Architecture: CL is the top level and CD is the bottom level

when proper weights W_i's are given (Sun 1992a). Therefore, this scheme can be implemented, with ease, in a connectionist multilayer, feedforward network with weighted-sum node activation functions. In this paper, we will not discuss in detail the above points regarding rule representations; Sun (1992a) presents a complete treatment of these points. The scheme can handle variable binding by utilizing the DN (i.e., *the Discrete Neuron*) formalism, which establishes a mapping between input/state and output (Sun 1989 and Sun 1990b), and thus facilitates variable binding (see Appendix B).

CD is a connectionist network with distributed representation, roughly corresponding to reasoning at the subconceptual level. Concepts and rules are diffusely represented by sets of feature nodes overlapping each other. The amount of overlapping of two sets of nodes representing two different concepts is proportional to the degree of similarity between these two concepts. We call this a *similarity-based representation*, in which units can be features, perceptual primitives, internal goals or affect states. Concepts are "defined" in terms of their similarity to other concepts in these primitive representations. We will utilize these primitives only as a substratum for similarity-based representations of higher level concepts.

Now we can link the localist network (CL) with this distributed network (CD), by linking each node in CL representing one concept to all the nodes in CD representing the same concept, and assign them appropriate weights (see Figure 1). Those cross-level links (*td* and *bu*) are moderated by a latch mechanism. The rule links in CL (labeled as r) are duplicated (diffusely) in CD (labeled as lw). The interactions of the two components are in fixed cycles: first the latch opens to allow the activation of CL nodes to flow into corresponding CD nodes, and then the two parts start settling down on their own simultaneously, and finally the latch opens to allow the activation of nodes in CD to flow back into CL to be combined with the activation of corresponding CL nodes.

From the above description it is clear that the system is a combination of rule-based and similarity-based components, interwoven together. It implements naturally the functions defined above for knowledge links and conceptual similarity. The synergy of the two types of representation and reasoning helps to deal with the brittleness problem discussed earlier and, therefore, to account for the aforementioned common reasoning patterns.

For example, in order to solve the no-matching-rule situation, we can explore the similarity between the current situation and the rule conditions as represented in the CD part of the system. Consider the following case:

> Cars are for traveling on ground. Airplanes are for traveling in air. Are buses for traveling on ground or in the air?

In the context of deciding modes of transportation, based on the similarity of relevant features (such as Having-wings, Having-tails, Wheels-on-both-sides, Aerodynamic-shapes, Landing-gears, etc.) as represented with the amount of overlapping in corresponding sets of units in CD, buses are closer to cars than to airplanes. So "traveling on ground" is activated more strongly. One concludes that buses are for traveling on ground.

Another example concerns the problem of inconsistent/incoherent rule bases (rule interaction), we can utilize the rule interactions in CD:

> If carrying cargo, buy utility vehicles. If carrying passengers, buy passenger vehicles. If carrying both cargo and passengers, what shall one buy?

Different types of vehicles are represented as features in CD. When the above two rules are both activated (in response to the question), all features corresponding to both utility and passenger vehicles will be activated in CD. All this information will go up to CL, and the things corresponding to the intersection of utility and passenger vehicles will be activated strongly (because they have all the features). So something like "van" will win.

Other aspects of the brittleness problem can be solved in a similar fashion, including the common patterns identified by Collins & Michalski (1989) [2]. This solution is quite different from Collins & Michalski (see Appendix A for a comparison). Our contention is that this model is conceptually simpler and computationally more efficient (by combining and eliminating many parameters).

[2] We are certainly not implying that we solved the brittleness problem completely. Rather, we are aiming for a simple and elegant model that can deal with some important and predominant aspects of the problem very effectively and efficiently.

Formal mathematical analyses were performed for each of these aspects, and we came up with a set of requirements and constraints for each aspect regarding the parameters (including td, bu, and lw; see Figure 1) of a system that can deal with that particular aspect. After analyzing how these requirements and constraints imposed by each of these aspects interact with one another, a synthesis was achieved, so that a unified system was formed with a unique set of parameter settings satisfying all requirements (Sun 1991). Based on that, a large-scale system (named GIRO) consisting of about two hundred nodes was built to test, in a realistic setting, how these fragments combine. The system utilizes geographical knowledge extracted from encyclopedias and performs commonsense reasoning based on that knowledge (see the next section). The results showed that the system performs very reasonably under various circumstances.

6 Detailed Examples

6.1 The Problem

Look at the "Chaco" example (Collins & Michalski 1989):

> Q: Is the Chaco the cattle country?
>
> R: It is like Western Texas, so in some sense I guess it's cattle country.

We can put it another way to straighten out the reasoning:

> Western Texas is cattle country.
> Chaco is similar to western Texas (in some relevant aspects).
> So Chaco is cattle country.

6.2 An analysis

In this example, because there is no known knowledge (or no applicable rules), an uncertain conclusion is drawn based on similarity with known knowledge (rules). Using the formalism we developed, it can be described as:

$$Chaco \sim WesternTexas$$

$$WesternTexas \longrightarrow cattlecountry$$

Given "Chaco" with $ACT_{Chaco} = 1$, "cattlecountry" is concluded with $ACT_{cattlecountry}$ calculated as follows:

$$ACT_{cattlecountry} = (WesternTexas \longrightarrow cattlecountry)$$

$$*ACT_{Chaco} * (Chaco \sim WesternTexas)$$

where the similarity measure is chosen to facilitate later implementations in the CONSYDERR architecture:

$$Chaco \sim WesternTexas = p * \frac{|F_{Chaco} \cap F_{WesternTexas}|}{|F_{Chaco}|}$$

$p \in [0, 1]$ is a parameter used for adjusting the system's behavior, from absolute rigidity to accommodating similarity-based associative thinking.

These equations can be readily translated into the CONSYDERR architecture: Links between nodes in both CL and CD represent rule strength measures (the link weights are defined to be the corresponding rule strengths), and similarity measures are implemented with CD representations (we use a set of nodes to represent all features in CD and the amount of overlapping between representations of two concepts expresses the conceptual similarity of these two concepts). See Figure 2.

6.3 The Working of the System

After starting to receive input data, the CONSYDERR system operates in fixed cycles: (1) Top-down phase, (2) Settling phase, and (3) Bottom-up phase. This cycle can be repeated to continuously track inputs.

In Top-down phase, the computation is as follows:

$$x_i(t+1) = \max_a ACT_a(t)$$

where a is any node in CL that has $x_i \in CD_a$.

In Settling phase, the computation is as follows:

$$\Delta ACT_a(t+1) = \alpha \sum W_i I_i(t) - \beta ACT_a(t)$$

and

$$\Delta x_i(t+1) = \mu \sum w_i i_i(t) - \nu x_i(t)$$

where W_i's and w_i's are rule strengths (weights) measures, and I_i's and i_i's are the activations of related concepts or features (premises or logical predecessors); α, β, μ, ν are parameters controlling the network dynamics: α and μ are rates of changes, and β and ν are decay rates. In this paper, we set $\alpha = \beta = \mu = \nu = 1$, so that the equilibrium state is, followed from the above two equations,

$$ACT_a = \sum W_i * I_i$$

$$x_i = \sum w_i * i_i$$

where I_i's and i_i's are final converged inputs to the two nodes, respectively. These inputs are from other nodes that are linked to a and x_i; they eventually converge to some constant values, when extenal inputs are clamped to a few nodes in the network and the feedforward network is given enough time to settle.

In Bottom-up phase, the computation is as follows:

$$ACT_b(t+1) = \max(ACT_b(t), \sum_{x_i \in CD_b} \frac{x_i(t)}{|CD_b|})$$

where b is any node in CL.

Applying this cycle to the example: Top-down phase will activate the CD representation of "Chaco" and activate partially the CD representation of "WesternTexas" based on their similarity; then in Settling phase, rules (links) take effect and this amounts to applying in CD the rule: *WesternTexas is cattle-country*, so the CD representation of "cattle country" is partially activated; finally in Bottom-up phase, the partially activated CD representation of "cattle country" will percolate up to activate the "cattle country" node in CL. The result can be read off from CL. See Figure 2.

6.4 Another Example

Another example is as follows:

Q: Are there roses in England?

R: There are a lot of flowers in England. So I guess there are roses.

Here the deduction is based on property inheritance (England HORTICULTURE flower; rose IS flower; so England HORTICULTURE rose), and the conclusion is partially certain and can be drawn only when there is no information to the contrary (i.e. no cancellation).

This inheritance scenario can be expressed as follows:

$$England \longrightarrow flower$$

$$flower \sim rose$$

Although not shown explicitly, in this case we actually have $flower \supset rose$ and $F_{flower} \subset F_{rose}$. Given $ACT_{England} = 1$, the activation of "rose" can be calculated with:

$$ACT_{rose} = ACT_{England} * (England \longrightarrow flower)$$

$$*(flower \sim rose)$$

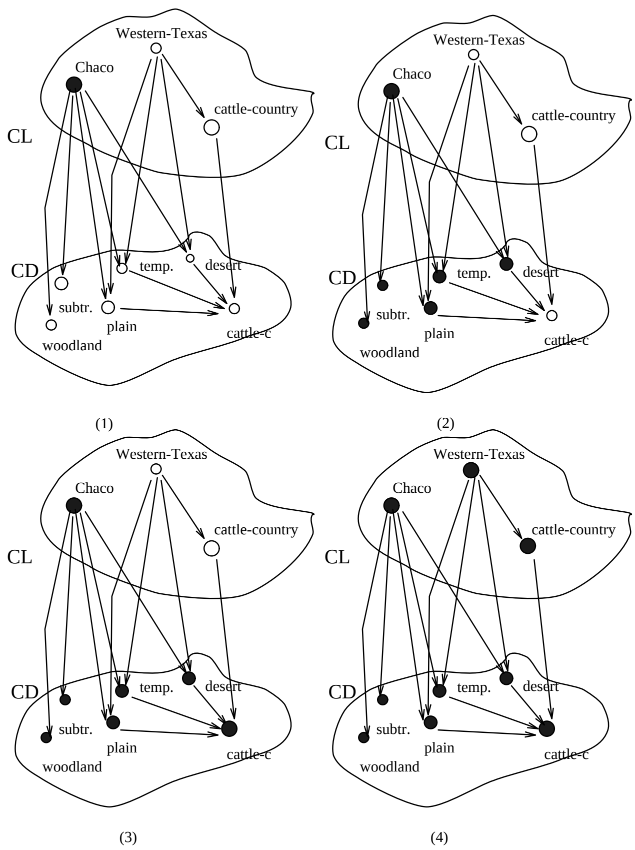


Figure 2: The Reasoning Process for the Chaco Protocol: (1) Receiving inputs, (2) Top-down, (3) Settling (rule application), and (4) Bottom-up.

(To save space, unrelated nodes are not shown here).

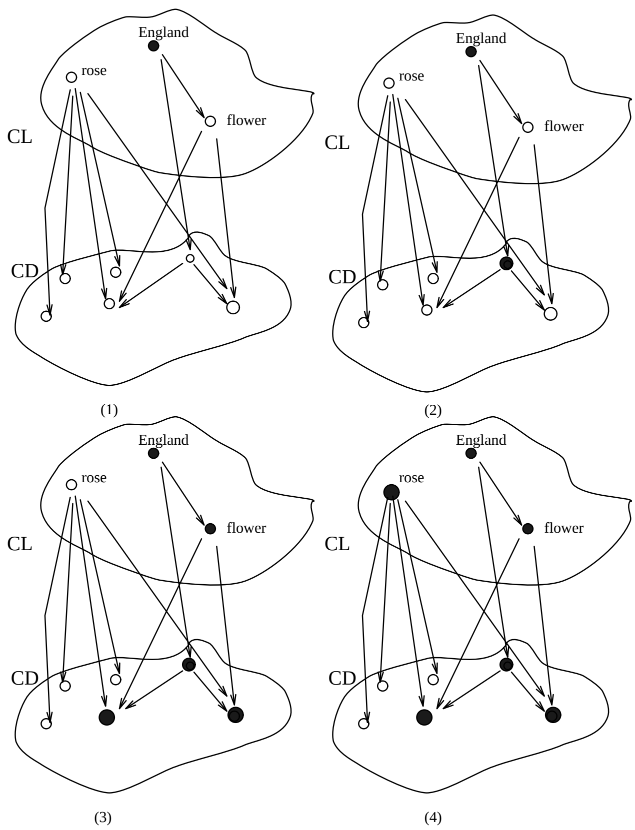


Figure 3: The Reasoning Process for the England Protocol: (1) Receiving inputs, (2) Top-down, (3) Settling (rule application), and (4) Bottom-up.

Therefore, the same way as before, this can be implemented in CONSYDERR with the two level dual representation and their interaction. (Here we use *rose* to represent *horticulture-rose* and *flower* to represent *horticulture-flower*, i.e., a property-value pair. See Sun 1991.)

We apply the same set of equations as before to this example, utilizing the three phases: the top-down phase will activate the CD representation of "England"; then in the settling phase, rules links take effect and amount to applying in CD the rule: *England* $\longrightarrow$ *flower*, so the CD representation of "rose" is partially activated due to similarity with "flower"; finally in the bottom-up phase, the partially activated CD representation of "rose" will percolate up to activate the "rose" node in CL. The result can be read off from CL. See Figure 3.

6.5 GIRO: Reasoning with Geographical Knowledge

Examining our previous examples, the question of sizes and scalability arises naturally: What if there are a large number of nodes in each level? What if there are a lot of similar concepts, rules and features? To show that systems for commonsense reasoning (in some particular domains at least) can be constructed systematically based on the CONSYDERR architecture and that it will work reasonably despite the existence of interference, distractions, and a large number of related concepts, we construct and study the system GIRO (standing for *a system for Geographical Information Reasoning and Organization*), which stores large amount of knowledge extracted from encyclopedias and reasons about agricultural characterizations of regions.

6.5.1 The Structure of GIRO

The knowledge representation in GIRO utilizes the two level idea in the CONSYDERR architecture by dividing the geographical knowledge represented in the system into two categories: concepts, which include basic geographic areas and regional characterizations (such as "cattle-country"), and features, which include primitive geographical descriptions of areas (such as "highland", "mountainous", and "tropical", etc.). Concepts are represented in CL, and features are represented in CD. Each geographic area represented in CL is connected to its corresponding features in the CD level, and because of the fact that features are shared by similar concepts, the CD representation is similarity-based, i.e., two concepts have overlapping CD representations if and only if the two are similar and the amount of overlapping is proportional to the degree of the similarity between them, as alluded to before. Each area is also connected to concepts describing its agricultural products by links, if this knowledge is available to the system. Figure 4 lists concepts for characterizing a geographical area in terms of its agricultural products, such as rice-growing-area, cattle-country, etc. Figure 5 lists features used in CD.

6.5.2 The Working of GIRO

GIRO operates this way: once a name of a geographical area is given to GIRO, as imposing a query, GIRO 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 GIRO a query: What is the main agricultural product of "Brazil-north"? That amounts to activating the node representing "Brazil-north" and

———————————————————

"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 4: Regional Characterization Included in the GIRO System

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 5: Geographical Features Included in the GIRO System

looking for agricultural products in the output. To answer this question, we let GIRO run to perform its reasoning. The actual output from GIRO is as follows:

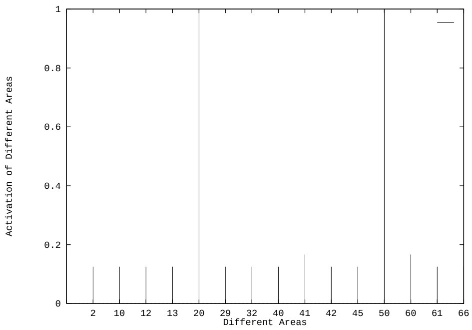
```
>(consyderr 0)

 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
 ( 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 GIRO and is not needed in this case.) See Figure 6.

Another test is as follows: suppose we want to know about Ecuador coastal area, we will give GIRO a query: What is the main agricultural product of Ecuador coast? by activating the node



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

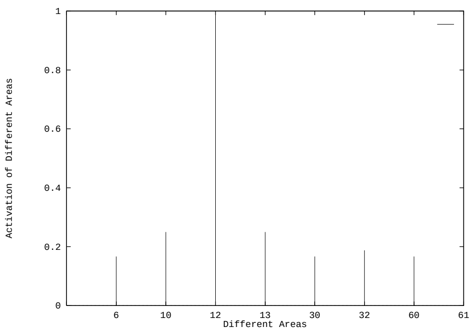
Figure 6: Output From GIRO: Case 1

representing "Ecuador coast". To answer this question, we let GIRO run to perform its reasoning. The actual output from GIRO 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 0.1433035089856102
 ( 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 )
```



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 7: Output From GIRO: Case 2

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". See Figure 7.

6.6 Initial Setup

The question of how we can gather data and set up a large system can be divided into two questions: how do we obtain rule weights and how do we obtain similarity measures?

In GIRO, rules are obtained by going through geography sourcebooks, picking out the relevant information and integrating it into the network with the CFRDN procedure (*Compilation of Fel Rules into DN networks*; Sun & Waltz 1991). The rules being put into the system include *WesternTexas is cattlecountry*, etc. In general, weights representing rules can be obtained by reading textbooks, instructions, or by using learning algorithms through interactions with the environment. There is no universally applicable way to do this, or in other words, it is domain-specific. One particular rule learning algorithms we developed is the IARL (*Incremental Associative Rule Learning*) algorithm, which is discussed in detail elsewhere (Sun & Waltz 1991).

Similarity measures are obtained by an indirect means: we first obtain all the relevant features needed for representing the concepts involved, and then naturally the amount of feature overlap determines the similarity between concepts involved. In order to come up with detailed feature representations for concepts, we pre-establish a set of feature nodes, and we then go through sourcebooks, establishing links (cross-level links) between a concept in CL and its features in CD, based on what we read in the sourcebooks. The features include: altitude, rainfall, vegetation, population, temperature, terrain, etc. with various ranges. Another possible way of obtaining similarity measures is to conduct a test, asking a group of subjects to rate the similarity of concepts concerned, and then construct CD representations based on the collected test scores with the STSIS procedure (Sun 1991).

One thing that needs to be stressed is that everything here is done with a rather mechanical process, and the architecture is modular: new nodes can be added and the existing ones can be deleted at either layers and the effect is localized, without affecting the global structure. Another thing is that similarity is context-sensitive, and mechanisms are needed to focus on relevant features and ignore or discount somehow the irrelevant ones, given the context (or the query, in the above-mentioned cases). This is done by the attention focusing module external to the system, in which a set of "context rules" are used to pick out all relevant features and suppress others (Sun 1991).

6.7 Comparisons

CONSYDERR utilizes parallelism inherent in the data to the maximum extent, especially when compared with Touretzky & Hinton (1985) or Dolan & Smolensky (1989) (which are basically sequential at the knowledge level). While most connectionist rule-based systems (Lange & Dyer 1989, Ajjanagadde & Shastri 1989, etc.) are functionally comparable to the CL part of CONSYDERR, the CD part is unique in that it provides an efficient way for similarity matching to supplement rule-based reasoning; the CL/CD dual representation scheme constitutes a principled way of accounting for the dichotomy of the conceptual level and the subconceptual (intuitive) level reasoning (Smolensky 1988, Sun 1991). More specifically, in CONSYDERR, once a CD representation of a concept is activated, many other concepts that share features with the concept will be activated to various degrees subsequently. In order to achieve this in purely rule-based systems, we have to have rule links from every concept to all of the other concepts that share some features with it; this can make systems too complex, and moreover it will be difficult in such systems to distinguish directly applicable rules from mere similarities and associations.

More recently Barnden & Srinivas (1990) describes an attempt at utilizing similar *cases* in reasoning in connectionist networks for the purpose of combating the rigidity that is undermining both symbolic reasoning systems and purely rule-based connectionist systems, which is the same idea as what motivates the CONSYDERR architecture. Their system is adopted from their original rule-based connectionist system; it has multiple configuration matrices (CMs), each of which contains a case for short term processing; a relative small set of gateway CMs provides the interface between short term processing and long term memory, which is not CM but instead encodes cases in connection weights; cases in non-gateway CMs compete to have their contents copied into gateway CMs, where it can cause cases similar to it to be retrieved from long term memory, and some symbol substitutions take place after the retrieval to adapt the cases to the current situation. Comparing CONSYDERR with Barnden & Srinivas (1990), we notice the following differences:

- CONSYDERR is more massively parallel: there are no gateways and complex copying and substitution processes that may hamper parallelism.
- Matching in CONSYDERR is more automatic (spontaneous): there is no special matching mechanism.

The two different systems represents two different attacks on the same problem, and therefore the two approaches might converge somehow.

The combination of rules and similarity in CONSYDERR is analogous in a way to a relatively new paradigm of AI (to which Barnden and Srinivas 1990 belong) — case-based reasoning (or CBR), which involves retrieval of relevant cases, matching, and adaptation of the matched cases (Riesbeck & Schank 1989). In CONSYDERR, the retrieval is accomplished automatically — we consider all existing knowledge at the same time. Similarity matching for finding the best case is done in a truly massively parallel fashion — all cases are matched against the current case at the same time, distributively with simple local computations. Differences between our approach and that of CBR include, for example, that we integrate rules with cases, we use uniform representation for cases and rules alike, and we utilize massive parallelism. It seems that although case-based inference (or the use of analogous knowledge in general) is very important for commonsense reasoning, it is supplementary to abstract knowledge encoded in rules. In other words, rules are primary, and concrete analogous knowledge are secondary, and they should be integrated into a unified framework, such as done in CONSYDERR.

One dire possibility of similarity matching is that we risk infering from one thing everything else. However, this situation will not happen, as demonstrated by the above examples, because of the following two facts: (1) the similarity relation is not transitive, that is, similarity will not propagate; only those concepts that actually match the concepts fired by rules will be activated; (2) most concepts activated from similarity have only weak activations, and are hence suppressed; we can use, for example, the average activation of a net as a dynamic threshold to filter out relatively weak activations.

7 Adding Variables to CONSYDERR

The system described above has only limited expressive power — because of a lack of variables, only a single object can be considered at a time. Adding variables to CONSYDERR is helpful in improving both expressive and reasoning power in performing rule-based reasoning. (For other reasons why variable binding is important, see Pinker & Prince 1988 and Fodor & Pylyshyn 1988.)

To add variables to CONSYDERR, first of all is the question of how to represent variables. Each variable in each predicate can be handled by one node. The variable can get bound by an application of a relevant rule that has in its conclusion the predicate that contains the variable. The binding is done by a numerical value, which represents a particular constant (a domain object, a binding), being passed along the link (which represents the rule in question) to the node representing the variables, from a node representing a variable having the same name in a predicate in the condition part of the rule applied. For example, suppose we have a rule: "if an area grows something, it is an agricultural area." It can be expressed as follows (where A represents "grow", and B represents "agricultural area"):

$$A(x, y) \longrightarrow B(x)$$

and the input is A(a, b) (meaning *area a* grows *b*), where a and b are constants, e.g., *a=Florida and b=orange*. The binding a is passed to the node representing x, and b to the one representing y in A(x,y). Then by applying the rule, a is further passed from the node representing x in A(x, y) to the node representing x in B(x).

Let us look at one level, say CL, to see how this can be accommodated in the CONSYDERR architecture. To add variables into CL, the same structure of CL can be kept just as before, but in place of a node now we have an *assembly*, or a collection of nodes interconnected in some way. To serve our purpose of incorporating variables and the variable binding mechanism into the model, an assembly should consist of a certain number of nodes, one of which is for computing and storing confidence (activation) values (we will call it the C node), exactly the same as before, and the rest are for variables (we will call them X nodes). We have to place an upper limit on the number of variables a predicate may have, because we can have only a fixed number of nodes in an assembly (or a network in general). Suppose we are allowed to have k variables, an assembly then contains k+1 nodes, each of which except one is for storing and passing along bindings of the variable which

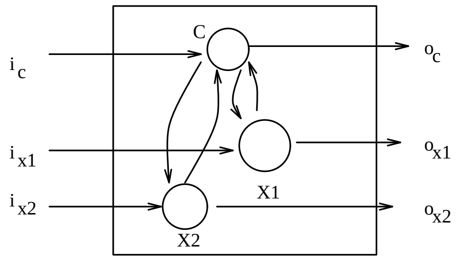


Figure 8: An Assembly

it represents. Among all the nodes, the C node of an assembly receives inputs from C nodes in other assemblies which are conditions of a rule that use this assembly for its conclusion predicate; each variable (X) node receives inputs from variable (X) nodes in the same other assemblies as its corresponding C node, picking up its own binding and passing it on. See Figure 8. Besides the above basic connections, each of the variable nodes in an assembly should also communicate with the C node to send its bindings, so that the C node can check consistency (and other things), and should also receive from the C node "instructions", which are to tell a variable node which binding to take in case there are multiple inputs for a variable node.

Let us look again at the aforementioned example:

$$A(x, y) \longrightarrow B(x)$$

and the input is A(a, b), where a and b are constants. We will have in CL two assemblies: one for A(x,y) and the other for B(x). The first assembly has three nodes: C, X, and Y. The second assembly has two nodes: C and X. The C node in the first assembly is linked to the C node in the second assembly. The X node in the first assembly is linked also to the X node in the second assembly. (There should also be some intra-assembly links, which are omitted for simplicity.) See Figure 9. First the C node in the first assembly is activated, and the binding a is passed to the X node in A(x,y), and b to the Y node. Then by applying the rule (i.e., propagating activation along the link), the C node in the second assembly is activated, and a is passed from the X node in A(x, y) to the X node in B(x). Thus a conclusion is reached: B(a).

The above description regarding CL is readily applicable to CD. From a computational standpoint, CD is exactly the same as CL. The difference is in the representational primitives: instead of concepts and propositions, we have features as basic elements in the CD representation. So whatever works for CL works for CD.

It should be pointed out that variable nodes do not necessarily contain a particular numerical

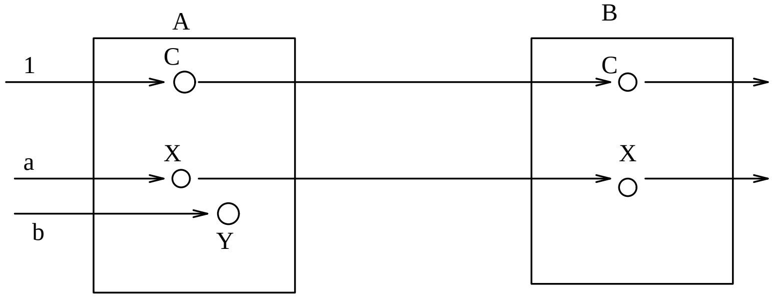


Figure 9: The Network for Representing the Rule: if A(X,Y) then B(X)

value or symbol that represents a particular domain object, or in other words, they can be free variables. For example, if we have a rule: $P(x) \longrightarrow Q(x)$, given P(z), we can derive Q(z), by representing z with a special symbol in variable nodes of the assemblies for P(x) and Q(x).

One issue we have to look deeper into is the inter-level connections between CL and CD. The main question is how bindings are passed along when top-down activation, or bottom-up activation, is in effect. A simplistic answer is that we can pass appropriate bindings along with activations the same way as in the intra-level case. One problem that arises from this solution is how we deal with the situation, at the bottom-up phase, where different features of the same CL node have different bindings associated with them (see Figure 10). One way to deal with this is to use the majority rule principle, choosing the binding that is associated with more features than any others. A related, but complementary, problem is, at the top-down phase, how to decide a proper binding when there are more than one binding received at a CD assembly representing a feature, from different CL assemblies (see Figure 11). This can occur when two assemblies representing two different predicates with different arguments and bindings but sharing some features are activated at the same time. This problem can be solved by choosing the binding associated with the assembly that has stronger activation (and hence stronger confidence). If they unfortunately have the same activation, a random choice has to be made. The solutions to the above two problems give the system a flavor of competition and winner-take-all. [3]

With variable binding, the structure of a CL network is quite complicated. We need a way of setting up a network, given a set of rules in FEL (extracted from experts or learned from data, etc.). Although the process of setting up such a network is complex, it can be done mechanically by following some fixed procedures (not with some "art of network building"). As mentioned before,

[3] Obviously, this overall scheme may not work for all cases. Nevertheless, it is adequate for dealing with most of the interesting cases.

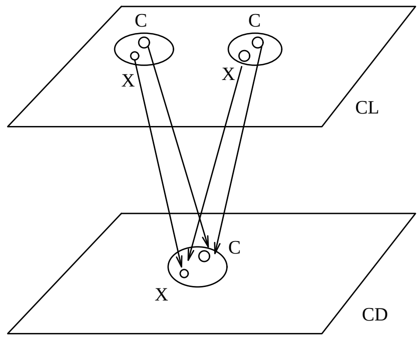


Figure 10: Multiple Bindings Received During the Top-Down Phase

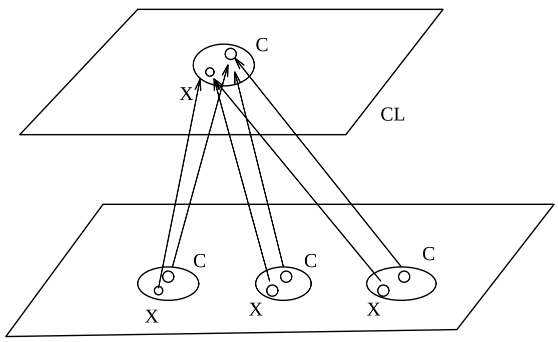


Figure 11: Multiple Bindings Received During the Bottom-up Phase

CFRDN is such a set of procedures for automatically transforming first order FEL rules into a forward chaining reasoning network. Sun (1992b) is concerned specifically with the analysis of some more intricate issues in variable binding and some comparisons with other proposed variable binding mechanisms (Ajjanagadde & Shastri 1989, Lange & Dyer 1989, and others).

8 Conclusion

In this paper, a connectionist architecture CONSYDERR is proposed to account for some common patterns found in commonsense reasoning (and a set of protocols), and to remedy to a certain extent the brittleness problem found in typical symbolic systems. In the paper, different from other existing connectionist systems, a dual representational scheme is devised, which has extensional objects (localist representation) as well as intensional objects (distributed representation with features). By using feature-based distributed representation in addition to the localist representation, we are able to explore the synergy resulting from the interaction between these two types of representations and between rule-based reasoning and similarity-based reasoning. This synergy helps to deal with problems such as partial information, no exact matching, property inheritance, rule interaction, with only very simple representation as shown above (some of these issues are not treated in depth here; readers are refered to the papers dealing specifically with them, such as Sun 1989, 1992a, 1992b, and Sun & Waltz 1991). Therefore, the CONSYDERR architecture is capable of accounting for many difficult reasoning patterns in one unified system. This work demonstrates that the simple combination of the two simple techniques (rules and feature similarity) can go a long way in solving commonsense reasoning problems. This architecture also demonstrates that connectionist models equipped with symbolic capabilities are powerful tools for reasoning and they are not mere implementations of their symbolic counterparts.

Future work will include applying CONSYDERR to a wider variety of problems, theoretical and practical, to demonstrate and further develop its capabilities; we also want to explore other ways of combining rule-based reasoning with connectionist models, to better account for commonsense reasoning; finally, FEL and logical capabilities of CONSYDERR shall be further studied.

A The Collins Protocols

This appendix will present and briefly examine all the protocols considered, most of which are from Collins (1979) and Collins & Michalski (1989) (some of them were presented before, but are still listed here for completeness).

(1) The first protocol shows uncertain, evidential reasoning:

> Q: Do you think they might grow rice in Florida?
>
> R: Yeah. I guess they could, if there were an adequate fresh water supply, certainly a nice, big, warm, flat area.

In this example, the person answering the question deduced an uncertain conclusion based on partial knowledge, with a piece of crucial information (i.e. the presence of fresh water) absent.

(2) the second protocol is as follows:

> Q: Is the Chaco the cattle country?
>
> R: It is like western Texas, so in some sense I guess it's cattle country.

Here because there is no known knowledge (in other words, no applicable rules), an uncertain conclusion is drawn based on similarity with known knowledge (or rules).

(3) the third protocol is:

> Q: Are there roses in England?
>
> R: There are a lot of flowers in England. So I guess there are roses.

Here the deduction is based on *property inheritance*. Formally, England HORTICULTURE flower; rose IS flower; so England HORTICULTURE rose, to use the jargon of the inheritance theory. The conclusion is only partially certain and is drawn because there is no information to the contrary (i.e. no *cancellation* of properties).

(4) The fourth protocol is:

> Q: Is that [Llanos] where they grow coffee up there?
>
> A: I don't think the savanna is used for growing coffee. The trouble is the savanna has a rainy season and you can't count on rain in general [for growing coffee].

This protocol shows a chain of reasoning: Llanos is a savanna, savanna has a rainy season, and rainy seasons do not permit coffee growing.

(5) The fifth protocol is:

> Q: Is Uruguay in the Andes Mountains?
>
> A: It's a good guess to say that it's in the Andes Mountains because a lot of the countries [of South America] are.

Here there is no rule stating whether Uruguay is in the Andes or not. However, since most South American countries are in the Andes, the *default* is therefore *in the Andes*. Uruguay just "inherits" this default value (although incorrectly).

(6) The sixth protocol is:

> Q: Can a goose quack?
>
> A: No. A goose – well, it's like a duck, but it's not a duck. It can honk, but to say it can quack. No. I think its vocal cords are built differently. They have a beak and everything. But no, it can't quack.

Two patterns are present here. One is based on similarity between geese and ducks, independent of knowledge regarding geese, yielding the conclusion that geese may be able to quack. Another pattern is a rule: since geese do not have vocal cords built for quacking, they cannot quack.

(7) The seventh protocol is:

> Q: Is Florida moist?
>
> A: The temperature is high there, so the water holding capacity of the air is high too. I think Florida is moist.

In this example the concepts involved are not all-or-nothing, but somehow graded, so the conclusion must be graded, or lower in confidence, corresponding with the confidence values of known facts and rules.

(8) The eighth protocol is:

> Q: Will high interest rates cause high inflation rates?
>
> A: No. High interest rates will cause low money supply growth, which in turn causes low inflation rates.

This example shows a chaining of rules: High interest rates will cause low money supply growth, and low money supply growth will cause low inflation rates, so high interest rates will cause low inflation rates.

(9) The ninth protocol is:

> Q: What kind of vehicles are you going to buy?
>
> A: For carrying cargo, I have to buy a utility vehicle, but for carrying passengers, I have to buy a passenger vehicle. So I will buy a vehicle that is both a utility and a passenger vehicle. For example, a van.

This example shows the additive interaction of two rules: if carrying cargo, buy a utility vehicle, and if carrying passengers, buy a passenger vehicle. The result is the combination of the two rules: something that is both a utility vehicle and a passenger vehicle.

(10) The tenth protocol is:

> Q: Do women living in that [tropical] region have short life expectancy?
>
> A: Men living in tropical regions have short life expectancy, so probably women living in tropical regions have short life expectancy too.

This is another case of using similarity because of the lack of direct knowledge.

(11) The eleventh protocol is:

> Q: Are all South American countries in the tropical region?
>
> A: I think South American countries are in the tropical region, because Brazil is in the tropical region, Guyana is in the tropical region, Venezuela is in the tropical region, so on and so forth.

Although the conclusion is incorrect, this example illustrates *bottom-up inheritance* (a form of generalization). Since there is no knowledge directly associated with the superclass "South American countries" as to whether they are in the tropical or not, subclasses are looked at, and a conclusion is drawn based on the knowledge of the subclasses.

In Collins & Michalski (1989), these patterns are divided into the following categories:

- Derivation from mutual implication, in which particular values of different entities are related.
- Derivation from mutual dependence, in which functional relationships between two entities are exploited.
- GEN-based transformation, in which what is known about a particular class is generalized to its superclass.

- SPEC-based transformation, in which what is known about a class is specialized to a subclass.
- SIM-based transformation, in which what is known about a class is mapped to another class based on similarity.
- DIS-based transformation, in which what is known about a class is excluded from another class based on dissimilarity.

On the other hand, according to our analysis, there are much fewer categories: the first two categories have little difference and both are dealt with by rule applications; the rest categories are similarity related, that is, generalization and specialization are special cases of similarity, and therefore they can be dealt with by similarity matching. Dis-similarity based inference is extremely un-reliable: it is generally not the case that, just because the things are different in some particular aspects, they are different in some other aspects. Therefore, this kind of inference is not considered here. Overall, all the cases are described and dealt with by rules plus similarity in our analysis.

In the analysis by Collins & Michalski, the confidence of the conclusions reached depends on a number of parameters:

- Conditional likelihood
- Degree of certainty
- Degree of typicality of a subset within a set
- Degree of similarity of one set to another
- Frequency of the referent in the domain of the descriptor
- Dominance of a subset in a set
- Multiplicity of the referent
- Multiplicity of the argument

According to our analysis, however, a smaller set of parameters can be identified: rule weights and similarity measures. These two parameters can subsume the above parameters used by Collins & Michalski: the first two, conditional likelihood and degrees of certainty, can be easily captured by rule weights; the rest can be accounted for by similarity measures or a combination of rule weights and similarity measures.

B A Formal Treatment: the DN Nodes and Assemblies

A formal treatment of the basic elements in CONSYDERR, namely nodes, assemblies and networks, will be presented. We will use a very general formalism for connectionist models, the *Discrete Neuronal Model*, or the DN model. This formalism is first proposed in (Sun 1989) and later further developed in (Sun 1990b). It is a generalization of conventional connectionist models, aiming at resolving some difficulties inherent in these models (such as variable binding) by removing some unnecessary restrictions. The generalization in this formalism goes along several dimensions: internal states, differentiated outputs, and temporal responses (see Sun 1989 for auguments in favor of these). The resulting model can deal with variable binding, rule matching, and confidence value propagation all at the same time. Thus the model forms a reasonable basis for developing a complex reasoning system incorporating variables.

An automaton-theoretic description of a node in a DN model is a 7-tuple:

N=$< S, A, B, I, O, U, V >$

S= the set of all the possible states of a node,

O= the set of all outputs,

A= the set of all the actions (output symbols),

I= the set of inputs,

B= the set of all input symbols,

U= the state transition function,

V= the action function.

We think of a node as having a set of input and a set of output:

$$I = (i_1, i_2, ..., i_n) \in \Pi B$$

$$O = (o_1, o_2, ..., o_m) \in \Pi A$$

And the state transition and action functions are

$$V : S \times \Pi B \longrightarrow \Pi A$$

$$U : S \times \Pi B \longrightarrow S$$

A network of DN nodes is a graph with (multiple) directed edges between pairs of nodes. An added constraint is if there is a directed link from node i to node j, then $\Pi A_i = \Pi B_j$. Although the formalism is complex (it is aimed for generality beyond this work; Sun 1989, 1990b), the *actual* mechanism for variable binding is very simple, as will be seen below.

In this model a set of mapping between inputs and outputs is explicitly specified instead of a continuous activation function. This allows more flexible relationship between inputs and outputs, especially useful for performing variable binding. An internal state is introduced for a node to have a kind of memory. Distinct multiple simultaneous outputs are allowed in this model, which is very important for dealing with variable bindings. Because of the generality, this model can emulate other more conventional connectionist models (Sun 1989). Conversely, DN nodes can also be implemented with the conventional weighted-sum threshold model (Sun 1990b). These two facts together further ensure the compatibility of this formalism with more conventional models.

Let us see how the DN formalism can be used to construct one level of CONSYDERR, given some first order FEL rules with variables (we will use CL as an example for illustration). We can map from the first-order FEL to a CL network with variable binding as follows: A predicate is represented by an assembly of DN nodes in CL, and rules are encoded in a network by the connections an assembly has to the other assemblies. For example, Figure 12 shows how assemblies representing predicates are wired into a network. So CL is made up of assemblies which in turn are made up of nodes representing predicates and arguments, that is,

$$CL \cong \{AS, L\}$$

where AS is a set of assemblies

$$AS \cong \{(C, X_1, X_2, ..., X_k) | \text{k is an arbitrary number}\}$$

and L is the connectivity between assemblies

$$L \cong \{(as_1, as_2, i, j) | as_1, as_2 \in AS, \;\; i, j \in [0, k] \text{ are labels denoting nodes within an assembly}\}$$

The connections among nodes are specified in L by labels which indicate which node in one assembly is connected to which node in another assembly. (There may also be connections from each X to C and from C to each X, as explained before.)

In each assembly, there are k+1 nodes of the DN type: C,$X_1, X_2, ..., X_k$. The C node contains the activation value of the predicate (which represents the confidence, plausibility or certainty of the predicate) and connects to other C nodes in other assemblies for representing different predicates that are related to it by rules. The X nodes take care of variable bindings (i.e., for arguments to

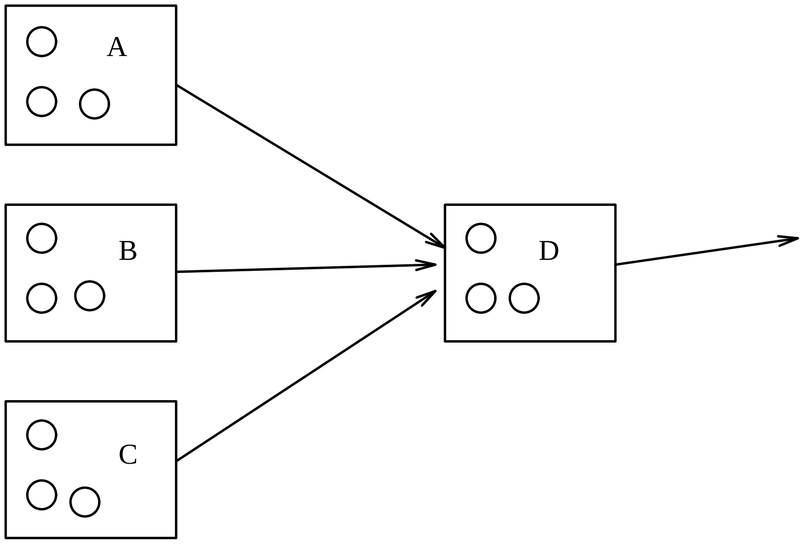


Figure 12: The Wiring of a Network for Representing the Rule: if A(X) B(Y) C(Z), then D(W) where X, Y, Z and W are vectors of arguments.

the predicate) for a maximum of k variables. The set of states and the set of input/output symbols in the variable nodes are numerical values representing all possible bindings in a domain. The set is fixed a priori. Each element in an assembly, C or X's, is a node with a specific action function (a state transition function is not need for this application), as specified below:

For C:

$$V_0 = f(I_1, I_2, ...I_k) \in A_C$$

NOTE. Action function f here is a weighted-sum function. $I_i's$ are outputs of other C nodes in other assemblies impinging on this node. A_C is the set of output symbols in the node C. (Node C can also receive inputs from X nodes in the same assembly, useful in certain situations where consistency checking is needed in variable binding; see Sun 1992b.)

And for X_i: i=1,2,....k,

$$V_i = f_i(I_1, I_2, ...I_n) \in A_{Xi}$$

NOTE. Action function f_i here is a mapping specifying the output of X_i based on I_i's, inputs from X_i nodes in other assemblies. This function always pass on the binding it receives (some other complications regarding consistency checking, unification, selections, and inputs from C nodes are explained in Sun 1992b). A_{Xi} is the set of output symbols in the node X_i.

CONSYDERR can be defined as follows, with variables and the variable binding mechanism included:

> **CONSYDERR** is an architecture composed of a hierarchical (acyclic) CL and a hierarchical (acyclic) CD, both built with the DN nodes and assemblies (as defined above), with separate top-down and bottom-up inter-level connectivity (or path) and phasic interaction inter-level dynamics (i.e., a three phase cycle).

Acknowledgments I wish to thank Dave Waltz, James Pustejovsky, and Tim Hickey for many helpful discussions. My thanks also go to the editor and the reviewers for their helpful suggestions.

References

V. Ajjanagadde and L. Shastri, 1989. Efficient Inference with Multi-place Predicates and Variables in a Connectionist System. In Proc.11th Cognitive Science Society Conference, 396-403. Hillsdale, NJ: Erlbaum.

J. R. Anderson, 1983. *The Architecture of Cognition*, Cambridge, MA: Harvard University Press.

J. Barnden, 1988. The right of free association: relative-position encoding for connectionist data structures. In Proc.10th Conference of Cognitive Science Society, 503-509. Hillsdale, NJ: Erlbaum.

J. Barnden and K. Srinivas, 1990. Overcoming Rule-Based Rigidity and Connectionist Limitations Through Massively Parallel Case-based Reasoning. Technical Report MCCS-90-187, Computing Research Laboratory, New Mexico State University, Las Cruces, NM.

B. Buchanan and E. Shortliffe, 1984. Rule-Based Expert Systems. Reading, MA: Addison-Wesley.

C. Chang and R. C. Lee, 1973. *Symbolic Logic and Mechanical Theorem Proving*, Academic Press.

A. Collins, 1979. Fragments of a theory of human plausible reasoning. Proc.TINLAP.

A. Collins and R. Michalski, 1989. The logic of plausible reasoning. Cognitive Science, 13(1), 1-49.

C. Dolan and P. Smolensky, 1988. Implementing a connectionist production system using tensor products. In D. Touretzky et al. eds. Proc.1988 Connectionist Summer School, 265-272. San Mateo, CA: Morgan Kaufman, 1989.

J. Fodor and Z. Pylyshyn, 1988. Connectionism and Cognitive Architecture: A Critical Analysis. in Pinker and Mehler (eds.) *Connections and Symbols.* Cambridge, MA: MIT Press.

M. Ginsberg (ed.) 1987. *Readings in Non-monotonic Reasoning*, pp.53-55, San Mateo, CA: Morgan Kaufman.

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

S. Holldobler, 1990. On High-level inferencing and the variable binding problem. in Dorffner, G. (ed.), OEGAI-90. pp.180-185. Berlin, FRG: Springer-Verlag.

J. Klahr et al, eds. 1989. Production System Models of Learning and Development. Cambridge, MA: MIT Press.

T. Lange and M. Dyer, 1989. Frame selection in a connectionist model. In Proc.11th Cognitive Science Conference, 706-713. Hillsdale, NJ: Erlbaum.

A. Pinker and A. Prince, 1988. On Language and Connectionism. in S. Pinker and J. Mehler (eds.), *Connections and Symbols.* Cambridge, MA: MIT Press.

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

C. Riesbeck and R. Schank, 1989. *Inside Case-based Reasoning.* Hillsdale, NJ: Lawrence Erlbaum Associates.

L. Shastri, 1988. A Connectionist Approach to Knowledge Representation and Limited Inference. Cognitive Science, 12, 331-392.

Y. Shoham, 1987. Reasoning about Change: Time and Causation from AI Standpoint. Ph.D Dissertation, Yale University, New Haven, Connecticut.

Y. Shoham, 1990. Non-monotonic Reasoning and Causation. Cognitive Science, 14, 213-252.

P. Smolensky, 1988. On the proper treatment of connectionism, Behavioral and Brain Sciences, 11, 1-43.

R. Sun, 1989. A Discrete Neural Network Model for Conceptual Representation and Reasoning. In Proc.11th Cognitive Science Society Conference, 916-923. Hillsdale, NJ: Erlbaum.

R. Sun, 1990a. Rules and Connectionism. In Proc.INNC-Paris, p.545. Netherlands: Kluwer.

R. Sun, 1990b. The Discrete Neuronal Models. In Proc.INNC-Paris, 902-907. Netherlands: Kluwer.

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

R. Sun, 1991. Integrating Rules and Connectionism for Robust Reasoning: An Architecture with Dual Representation. Ph.D Thesis, Computer Science Department, Brandeis University, Waltham, MA.

R. Sun, 1992a. Beyond associative memories, logics and variables in connectionist networks. *Information Sciences*, special issue on AI and neural networks.

R. Sun, 1992b. On variable binding in connectionist networks. *Connection Science*, submitted.

D. Touretzky and G. Hinton, 1985. Symbol Among Neurons. In Proc.9th International Joint Conference on Artificial Intelligence, pp.238-243. San Mateo, CA: Morgan Kaufman.

A. Tversky, 1977. Features of Similarity. Psychological Review, 84(4), 327-352.

D. Waltz and J. Pollack, 1985. Massive parallel parsing. Cognitive Science, 9, 51-74.

L. Zadeh, 1983. Commonsense knowledge representation based on fuzzy logic. *Computer*, Vol.16, No.10, 61-66.