Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

ON THE EXPRESSIVENESS OF NETWORKS WITH HIDDEN VARIABLES

Rina Dechter

July 1991
CSD-910038

In Proceedings, AAAI-90, Boston, MA., 7/27-8/3/90, pp. 556-562.

Technical Report
R-147
February 1990

ON THE EXPRESSIVENESS OF NETWORKS WITH HIDDEN VARIABLES

Rina Dechter [(1)]

Computer Science Department
Technion -- Israel Institute of Technology
Haifa, Israel, 32000

ABSTRACT

This paper investigates design issues associated with representing relations in binary networks augmented with hidden variables. The trade-off between the number of variables required and the size of their domains is discussed. We show that if the number of values available to each variable is just two, then hidden variables cannot improve the expressional power of the network, regardless of their number. However, for $k \geq 3$, we can always find a layered network using k-valued hidden variables that represent an arbitrary relation. We then provide a scheme for decomposing an arbitrary relation, ρ, using $\frac{|\rho|-2}{k-2}$ hidden variables, each having k values ($k > 2$).

Topic: Knowledge representation.
Subtopic: connectionist architectures.

(1) This research was supported in part by NSF grant #IRI-8821444 and by an Air Force grant #AFOSR 88 0177 while the author was visiting the cognitive systems lab at UCLA.

1. Introduction

Hidden units play a central role in connectionist model, without which the model would not represent many useful functions and relations. In the early days of the Perceptrons [4] it was noted that even simple functions like the *XOR* were not expressible in a single layer perceptron; a realization that slowed research in the area until the notion of hidden units had emerged [6, 3]. Nevertheless, a formal treatment of the expressiveness gained by hidden units, and systematic schemes for designing systems with hidden units within the neural network paradigm are still not available.

Our intention is to investigate formally the role of hidden units and devise systematic schemes for designing systems incorporating hidden units. Specifically, we address the following task: given a relation on n variables, called visible, we wish to design a network having $n+h$ units whose stable patterns, (relative to the visible units) coincide with the original relation. This task is central to most applications of connectionist networks, in particular to its role as associative memory. The task will be investigated for a connectionist architecture which is different from classic connectionist networks in that it is based on **constraint networks.** The sequential constraint network model is defined next.

A **Network of binary constraints** involves a set of n variables $X_1,...,X_n$, each represented by its domain values, $D_1, \ldots, D_n$, and a set of constraints. A **binary constraint** R_{ij} between two variables X_i and X_j is a subset of the cartesian product $D_i \times D_j$ that specifies which values of the variables are compatible with each other. A solution is an assignment of values to all the variables which satisfy all the constraints, and the **constraint satisfaction problems (CSP)** associated with these networks is to find one or all solutions. A binary CSP can be associated with a **constraint-graph** in which nodes represent variables and arcs connect pairs of variables which are constrained explicitly. Figure 1a presents a constraint network where each node represents a variable having values $\{a, b, c\}$ and each link is associated with a strict lexicographic order (where $X_i < X_j$ iff $i < j$). (The domains and the constraints explicitly indicated on some of the links.)

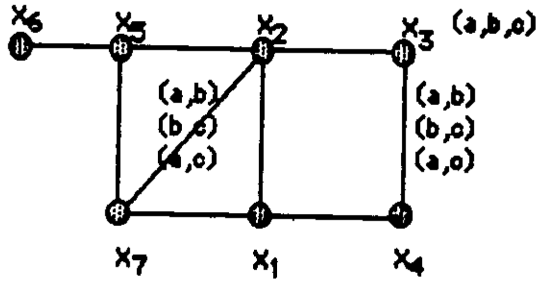


Figure 1: An example of a binary CN

Our constraint-based connectionist architecture assumes that each unit plays the role of a variable having k states, and that the links, representing the constraints, are quantified by compatibility relations between states of adjacent units. Each unit asynchronously updates its state (i.e., assigns itself one of its values) using a decision function based only on the states of its neighboring units and its compatibility relations with them. In a companion paper we provide a communication protocol for this model which is guaranteed to converge to a global consistent assignment of values [1]. Although this constraint-based architecture differs from classical connectionist architectures the resemblance is strong enough to shed interesting light on the latter architectures as well.

The paper is organized as follows. Section 2 continues with definitions and preliminaries. In section 3 we show that bi-valued hidden variables add no expressional power, while, in section 4 we show that if the hidden variables have 3 values or more they can decompose any relation. Bounds on the trade-off between the number of hidden variables and the cardinality of their values are given as well. Section 5 extends the decomposition scheme to those having some initial inner decomposition, section 6 presents examples and section 7 provides concluding remarks. Due to space limitation most results are presented with sketchy or no proofs. For formal proofs see [2].

2. Definitions and preliminaries

Since communication links are pairwise our constraint-based architecture is restricted to expressing **binary** constraint networks only. A general relation on n variables is not necessarily expressible as a binary network of constraints on the same set of variables. The question we pose, therefore, is how to express any relation in a binary constraint network, with the aid of new **hidden** variables.

Let $rel(R)$ denotes the relation associated with network R. (i.e., $rel(R)$ is the set of all solutions to R). Let ρ be an n-ary relation over variables $X = \{X_1, \ldots, X_n\}$, each having k values. We now define the notion of decomposability with hidden variables.

Definition: Relation ρ is ***h*-network-decomposable** if there exist h additional variables, $Y = \{Y_1, \ldots, Y_h\}$, having k values or less, for which there is a binary network $R = R(X,Y)$, on $X \cup Y$, such that

$$\rho = \Pi_X rel(R(X,Y)). \tag{1}$$

$\Pi_U(\rho)$ denotes the projection[1] of relation ρ on subsets of variables U. When no hidden variables are required for network decomposability we say that the relation is **network decomposable**. Any relation, ρ, can be associated with a unique binary network that is generated by projecting the relation on each pair of its variables. This network is called **the minimal network** [5] and it is known to provide the best approximation to ρ. Namely, $\rho \subseteq rel(M)$ and if R is any other binary network on the original set of variables s.t. $\rho \subseteq rel(R)$ then $\rho \subseteq rel(M) \subseteq rel(R)$. It follows that the minimal network can determine whether a relation is network decomposable or not.

Theorem 1: A relation is network decomposable if and only if its minimal network represents it precisely. Namely, if $\rho = rel(M)$.

□

Every non-decomposable relation has a trivial **star-decomposition** using one hidden variable and an **unrestricted** number of values. In this decomposition the hidden variable, Y, needs t values, when t is the cardinality of the relation. Each value of Y is needed to "index" each tuple in the relation. This is achieved by constraining the hidden variable with each original variable as follows. The constraint between the hidden variable, Y, and an original variable X_j makes the i^{th} value of Y compatible with one and only one value of X_j, the value that appears in the i^{th} tuple of the relation. That way each value of Y is made consistent with exactly one tuple

(1) The projection of variables $X_{i_1}, \ldots, X_{i_l}$ on relation ρ is given by: $\Pi_{X_{i_1},\ldots,X_{i_l}}(\rho) = \{(x_{i_1}, \ldots, x_{i_l}) \mid \exists\, \bar{x} \in \rho \ s.t. \ \forall\, i_j \ \bar{x}_{i_j} = x_{i_j}\}$

(see Fig 2). The resulting constraint network, which has a star shape (hence its name), clearly represents the target relation (i.e. projecting it on all original variables yields the original relation).

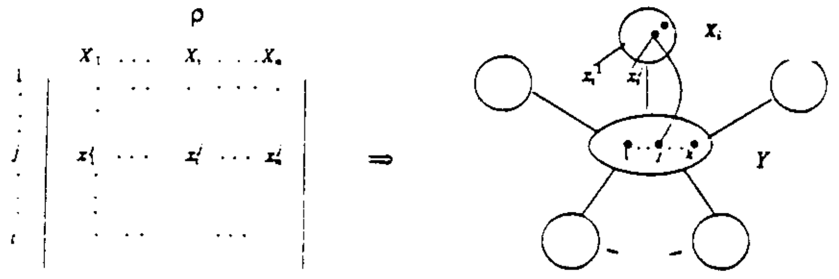

ρ
$X_1 \ldots X_i \ldots X_n$
1
j
t
$x_1^j \ldots x_i^j \ldots x_n^j$
$\Rightarrow$
X_i
x_i^1
x_i^j
Y

Figure 2: A star decomposition

Having the minimal network on one extreme, (a potential decomposition with no hidden variable) and the star network (requiring one hidden variable and unrestricted number of values) on the other, we are now interested in exploring middle ground cases. In particular, we wish to establish how many hidden variables are required, were we to restrict the size of their domains.

3. Using bi-valued hidden variables

When the hidden variables have only two values we get a surprising negative result:

Theorem 2: Relations which are not network decomposable cannot be decomposed by adding any number of 2-valued hidden variables.

Sketch of proof: Lets ρ be a relation that is not network decomposable and let M be its minimal network. The minimal network, since not representing ρ, allows a tuple $\bar{x} = \bar{x}_1, \ldots, \bar{x}_n$ which is not part of ρ. The task of any hidden variable is to disallow such tuple while at the same time allow all tuples in ρ. Assume Y is such bi-valued hidden variable that when added to the network M it is inconsistent with $\bar{x}$ while consistent with any tuple in ρ. Y has to be consistent with each value of $\bar{x}$ (since $\bar{x}$'s values are generated from the minimal network). Namely, each value of $\bar{x}$ is consistent either with Y's "0" or with Y's "1". We further claim that **all** $\bar{x}$'s values are consistent either with the "0" or with the "1". Since if not there is a value, $\bar{x}_i$, not consistent with $Y=0$ and a value, $\bar{x}_j$, not consistent with $Y=1$ and the pair $(\bar{x}_i, \bar{x}_j)$ is not consistent with any value of Y. However, since this pair is allowed by the minimal network, excluding it must eliminate a legal tuple of ρ which yields a contradiction. The argument can be extended by induction to any number of hidden variables [2].

□

4. Multi-valued hidden variables

4.1 A conditional decomposition scheme

This section investigates decomposition schemes utilizing multi-valued (i.e., more then 2 values) hidden variables. In particular we wish to explore the trade-off between r, the number of hidden variables, and k, their domain sizes, required for decomposing an arbitrary relation.

We first restrict ourselves to $r = 1$. Clearly, a relation having t tuples is 1-decomposable by the star network. One may expect that when using also the minimal constraints between the original variables as part of the network decomposition the number of values needed by the centered hidden variable can be reduced. It can be shown, however, that for some relations, t, is also the smallest number of values required for decomposition (when using one hidden variable).

Let us define the **unit relation**, U_n, to be the "0-1" relation on n variables whose i^{th} tuple consists of a value "1" for variable X_i and a value "0" for all other variables (see Figure 3a).

$$U_5 = \left\{\begin{array}{ccccc} X_1 & X_2 & X_3 & X_4 & X_5 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array}\right\} \qquad U_{10,3} = \left\{\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 2 \end{array}\right\}$$

(a) (b)

Figure 3: (a) the unit relation U_5, (b) relation $U_{10,3}$

It can be shown that relation U_n cannot exploit the minimal constraints in order to reduce the value cardinality of the hidden variable:

Theorem 3: The smallest value, k, for which the unit relation U_n is 1-decomposable is $k = n$.

Sketch of proof: The minimal network of U_n allows the extra all "0" tuple. To exclude it any value of the hidden variable must be inconsistent with at least one $X_i = 0$ and therefore consistent with $X_i = 1$. As such it can "extend" only the i^{th} tuple of U_n hence we need n values.

$\square$

Let us define $H_\rho(h)$ to be the the minimum number of values (per variable) required for an h-decomposition of relation ρ. We can conclude:

Corollary 1: For every ρ, $H_\rho(1) \le |\rho|$ and for some ρ's, $H_\rho(1) = |\rho|$.

$\square$

A straightforward extension of the star decomposition to two hidden variables presents itself by expressing the $|\rho|$ values of one hidden variable by $|\rho|$ different pairs of values on the pair(1) of hidden variables as follows. All pairs of values are of the form $(i, 0)$ or $(0, i)$, where $\frac{|\rho|}{2} \ge i > 0$, and the value $(i, 0)$ is associated with the i^{th} tuple while the pair $(0, i)$ is associated with the $\frac{|\rho|}{2} + i$ tuple. Since $|\rho|$ different pairs can be expressed in this way by two $(\frac{|\rho|}{2})$-valued hidden variables, we can infer that $H_\rho(2) \le \frac{|\rho|}{2}$. It can be shown, however [2] that this bound is tight, namely:

Corollary 2: For every ρ, $H_\rho(2) \le \frac{|\rho|}{2}$ and for some ρ' (e.g., U_n) $H_\rho(2) = \frac{|\rho|}{2}$.

$\square$

The above property is extensible to any number of hidden variables and we can show that the unit relation U_n needs at least r, $(\frac{n}{r})$-valued hidden variables in order to be decomposed. We conclude that:

(1) For notational convenience we assume that all fractional expressions represent the ceiling integer of that fraction.

Corollary 3: For some ρ's $H_\rho(r) \geq \frac{|\rho|}{r}$.

□

Our approach for systematically decomposing a relation is to start from a star decomposition using a $|\rho|$-valued hidden variable, and then, if only k-valued variables are available, to simulate the star hidden variable by a relation that obeys the value restriction and that can be network decomposed. If the latter relation is not network decomposable we will apply the same principle to it and so on. This approach is detailed in the following paragraphs.

Let us extend the notion of unit relation into a k-valued relation as follows: the k-valued unit relation, $U_{n,k}$ has n tuples and $r = \frac{n}{k-1}$ variables such that the $(i(k-1)+j)^{th}$ tuple, $i \leq r$, $j \leq k-1$, has zero everywhere accept the i^{th} variable whose value is j (see Fig. 3b).

We focus, first, on the decomposition of U_n. The unit relation U_n can be **conditionally star-decomposed** via $\frac{n}{k-1}$ k-valued hidden variables. We first generate the concatenated relation $U_n U_{n,k}$, and then decompose it via a two layered network. The first layer consists of the original variables $X_1, \ldots, X_n$ and the second layer has the hidden variables $Y_1, \ldots, Y_{\frac{n}{k-1}}$. The only constraints in the decomposing network are those relating each hidden variable with an original variable (see Figure 4a). The constraints themselves are generated by projecting the concatenated relation $U_n U_{n,k}$ on the corresponding pairs of variables (Figure 4b). We say that $U_{n,k}$ "conditionally" decomposes U_n in the following sense: for any instantiation of the hidden variables to a legal tuple in $U_{n,k}$ the network allows only legal tuples among the original variables, namely, those participating in U_n.

Let ρ_1 and ρ_2 be two relations, having the same number of tuples, on disjoint sets of variables. Let us denote by $R(\rho_1, \rho_2)$ the two layered network in which the top layer contains ρ_1's variables, the bottom layer contains ρ_2's variables and there are constraints between any variable in ρ_1 and any variable in ρ_2. The constraints themselves are the projection of the concatenated relation $\rho_1 \rho_2$ on the corresponding pairs of variables. Using this notation we can say that $R(U_n, U_{n,k})$ is a conditional decomposition of U_n.

The problem now is that the new appended relation $U_{n,k}$, by itself, is not network decomposable. Namely, even if we add all the minimal constraints between the hidden variables it will not exclude the "0" tuple on the hidden variables which in turn will allow any combination of values on the original variables. As its name indicates it is just a conditional decomposition, i.e. conditioned on our ability to further decompose $U_{n,k}$.

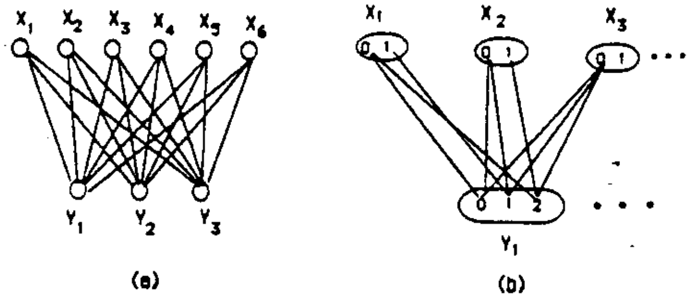


Figure 4: Decomposing U_6 by $U_{6,3}$

4.2. A general decomposition scheme

It seems as though we didn't solve anything! just transferred the decomposability problem from one unit relation (U_n) to another ($U_{n,k}$). Nevertheless, since the number of variables in $U_{n,k}$ is $\frac{n}{k-1} < n$, (for $k>2$) we can now decompose it with a new k-valued unit "relation", denoted $U^1{}_{n,k}$ having even a smaller number of variables. The unit relation $U^1{}_{n,k}$ is a pseudo-relation since it is a set of tuples which are not necessarily different. $U^1{}_{n,k}$ has $\frac{n}{(k-1)^2}$ variables and it is generated by taking the unit relation, $U_{\frac{n}{2},k}$, and duplicating each tuple in it (see Figure 5). The intention being that each tuple in $U^1{}_{n,k}$ will not distinguish between tuples in $U_{n,k}$ having non-zero values for the same variable. $U^1{}_{n,k}$ can conditionally decompose $U_{n,k}$ in the same manner that $U_{n,k}$ conditionally decompose U_n using the two layered network $R(U_{n,k}, U^1{}_{n,k})$.

This results in having a sequence of "pseudo" unit relations, each with a smaller number of variables and each disallowing a smaller tuple of "0"'s in the preceding relation. The resulting relation is a concatenation of "inflated" unit relations each having a $(k-1)$ fraction of the variables of the preceding relation. Let us denote the resulting relation by $U^*{}_n$ (see Figure 5). Clearly $U_n = \Pi_X U^*{}_n$.

$$\underset{U_{12}}{\begin{pmatrix} X_1 & X_2 & X_3 & X_4 & X_5 & X_6 & X_7 & X_8 & X_9 & X_{10} & X_{11} & X_{12} \\ 1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1 \end{pmatrix}} \underset{U_{12,3}}{\begin{pmatrix} Y_1 & Y_2 & Y_3 & Y_4 & Y_5 & Y_6 \\ 1&0&0&0&0&0 \\ 2&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&2&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&2&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&2&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&2&0 \\ 0&0&0&0&0&1 \\ 0&0&0&0&0&2 \end{pmatrix}} \underset{U^1{}_{12,3}}{\begin{pmatrix} Z_1 & Z_2 & Z_3 \\ 1&0&0 \\ 1&0&0 \\ 2&0&0 \\ 2&0&0 \\ 0&1&0 \\ 0&1&0 \\ 0&2&0 \\ 0&2&0 \\ 0&0&1 \\ 0&0&1 \\ 0&0&2 \\ 0&0&2 \end{pmatrix}} \underset{U^2{}_{12,3}}{\begin{pmatrix} T_1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}}$$

Figure 5: $U^*{}_{12}$; the relation generated from decomposing U_{12}

The network that decomposes $U^*{}_n$ is a layered network where each set of new hidden variables are connected t all variables in the preceding layer and the constraints are the projection of $U^*{}_n$ on the corresponding pairs of variables. The bottom layer consists of one or two variables whose allowed tuples can be controlled by a direct constraint. A schematic description of the network decomposition for U_{12} is given in Figure 6.

To summarize, U_n can be decomposed by intersecting sequences of two layered networks. The resulting network, $R^*{}_n$, is defined by: ("$\cdot$" denotes the intersection operation)

$$R^*{}_n = R(U_n, U_{n,k}) \cdot R(U_{n,k}, U^1{}_{n,k}) \cdot R(U^1{}_{n,k}, U^2{}_{n,k}), \cdot \ldots, \cdot R(U^i{}_{n,k}, U^{i+1}{}_{n,k}) \cdot, \ldots, \cdot R(U^{\log_k n - 1}{}_{n,k}, U^{\log_k n}{}_{n,k}). \quad (4)$$

We can now apply the same idea to an arbitrary relation. Namely, in order to decompose a given relation ρ using k-valued hidden variables we will generate the network

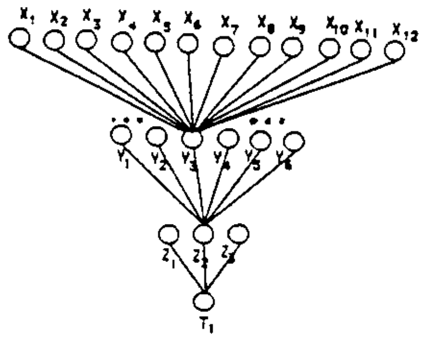


Figure 6: A layered decomposition of U^{*}_{12}

$$R^{*}(\rho) = R(\rho, U_{|\rho|,k}) \cdot R(U_{|\rho|,k}, U^{1}_{|\rho|,k}) \cdot, \ldots, \cdot R(U^{\log_k|\rho|-1}_{|\rho|,k}, U^{\log_k|\rho|}_{|\rho|,k}). \tag{5}$$

Let $H^{-1}{}_{\rho}(k)$ denotes the minimum number of k-valued hidden variables needed for decomposing ρ. We get the following theorem:

Theorem 4: Any relation is decomposable by $\frac{|\rho|-2}{k-2}$ k-valued hidden variables, when $k > 2$.

Proof: Let ρ be an arbitrary non-decomposable relation. It can be shown by simple algebraic manipulations that the decomposition presented in (5) requires $\frac{|\rho|-2}{k-2}$ variables.

□

From corollary 3 and from Theorem 4 it follows that the unit relation's decomposition cannot be substantially improved. Namely,

$$\frac{n}{k} \le H^{-1}{}_{U_n}(k) \le \frac{n-2}{k-2}. \tag{6}$$

To summarize, the decomposition scheme presented by (5) is optimal in the sense that for some relations (the U_n's) a better decomposition does not exist. Nevertheless, we still wish to find the minimum number of values needed for the star-decomposition, since it will provide a better bound for any general decomposition, namely:

$$H^{-1}{}_{\rho}(k) \le \frac{H_{\rho}(1)-2}{k-2}. \tag{7}$$

5. Decompositions of partially decomposable relations

One way of improving our scheme can be hinted by investigating the level of *inner* decomposition of the relation using only its original variables. We assumed that the relation is not binary network decomposable, however it may be losslessly decomposed to relations having arity greater then 2 yet smaller then n. Let a relation scheme $R = R_1, \ldots, R_l$ be a set of subsets of attributes of the original relation. We say that the scheme R is a **lossless decomposition** of ρ if:

$$\rho = \rho_1 \bowtie^{(1)} \rho_2 \bowtie, \ldots, \bowtie \rho_l. \tag{8}$$

We claim that if a lossless decomposition of the relation is available and if we use hidden vari-

(1) $\bowtie$ is the relational database *join* operator.

ables to decompose each component separately without introducing any conflicts between the components, the combined network is a decomposition of the target relation. The general scheme is as follows: given a lossless decomposition $R = R_1, R_2, \ldots, R_l$ of ρ which is defined over variables $X = X_1, \ldots, X_n$, and given a binary network decomposition, R'_i, for each subrelation ρ_i, utilizing a set of hidden variables Y_i, (i.e., $\rho_i = \Pi_X rel(R'_i)$), and denoting by $\rho' = rel(R'_1) \times, \ldots, \times rel(R'_l)$, it is always true that

$$\Pi_X \rho' \subseteq (\Pi_X rel(R'_1)) \bowtie, \ldots, \bowtie (\Pi_X rel(R'_l)) = \rho \tag{9}$$

If we take special care to ensure that the hidden variables used in different components will not interfere with each other (by utilizing disjoint subsets, for instance) and will not eliminate a legal tuple of ρ we will have an equality in the left hand side of (9). In that case the combined network $\bigcap_i R'_i$, is a network decomposition of ρ utilizing a set of hidden variables $\bigcup_i Y_i$.

We can associate a star decomposition with each component separately. Namely, if l hidden variables are available, each devoted to a star decomposition of one subrelation, then the hidden variable Y_i of subrelation ρ_i will need $|\rho_i|$ values. We will get therefore that

$$H^{-1}{}_{\rho}(\max_i |\rho_i|) \leq l \tag{10}$$

If only k-valued hidden variables are available we can decompose each component subrelation using disjoint subsets of k-valued hidden variables. This way the non-interfering property is maintained. When applying the bound of Theorem 4 to each component separately and summing over all components we get:

$$H^{-1}{}_{\rho}(k) \leq \sum_{i=1}^{l} \frac{|\rho_i| - 2}{k-2} = \frac{\sum_{i=1}^{l} |\rho_i| - 2l}{k-2} \tag{11}$$

We see, therefore, that the "level" of inner decomposition can affect ρ's decomposability. We further conjecture that if ρ cannot be losslessly decomposed at all then $H_{\rho}(1) = |\rho|$. Examples conforming with this conjecture is the unit relation and the parity relation [2].

6. Examples

The following two examples, taken from chapter 8 of [7], demonstrate the use of inner decompositions once they are available.

Example 1: addition

Consider the problem of adding two-digit numbers, where the digits are 0 or 1. Denote the first number by X_1Y_1, the second number by X_2Y_2, let T stand for the carry and let $Z_1Z_2Z_3$ stand for the resulting sum. The *add* relation is given in figure 7a.

A star decomposition of *add* with one hidden variable requires 16 values, namely $H_{add}(1) \leq 16$. Using 3-valued variables our scheme requires 14 variables (EQ. (7)). Consider now the lossless decomposition of the *add* relation given by $R = R_1, R_2$ where $R_1 = T, Y_1, Y_2, Z_3$ and $R_2 = T, X_1, X_2, Z_1, Z_2$ (see figure 7b.) By decomposing each component separately we get that one 8-valued variable is needed for $rel(R_2)$ and a 4-valued variable is needed for $rel(R_1)$, each of the two can star-decompose its corresponding subrelation. In that case nothing is gained by the inner decomposition since a decomposition of the relation with 8-valued variables can be

$$add = \left\{ \begin{array}{cccccccc} X_1 & Y_1 & X_2 & Y_2 & T & Z_1 & Z_2 & Z_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{array} \right\}$$

(a)

$$R_2 = \left\{ \begin{array}{ccccc} T & X_1 & X_2 & Z_1 & Z_2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{array} \right\} \quad R_1 = \left\{ \begin{array}{cccc} T & Y_1 & Y_2 & Z_3 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{array} \right\}$$

(b)

Figure 7: (a) The *add* relation, (b) A lossless decomposition of *add*

directly applied to the overall relation using just two variables. However, if only 3-valued variables are permitted available, $rel(R_1)$ would require two such hidden variables while $rel(R_2)$ will require 6 variables, resulting in a total of 8, 3-valued variables as in Figure 8.

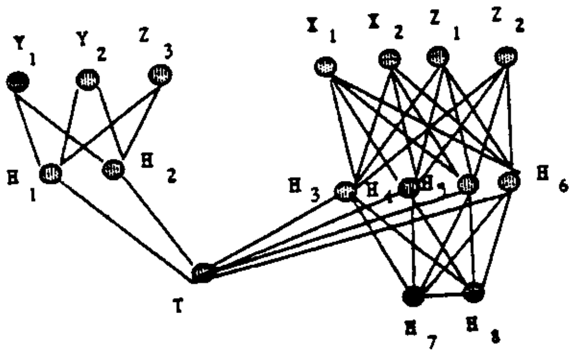


Figure 8: A 3-valued decomposition of *add*.

Example 2: the negation problem:

Consider a situation where the input to a system consists of a pattern of $n+1$ bits and n output bits. One of the input bit is called "the negation bit". When it is "0" the rest of the n input bits should be mapped directly to the output patterns. If it is "1" then each bit should be mapped to its negation. Figure 9a describes the relation defined over the negation variable N, the input variables X_1, X_2, X_3 and the output variables Y_1, Y_2, Y_3.

A direct decomposition of this relation requires 2^n values for a star decomposition and when 3-valued variables are available $2^n - 2$ hidden variables are required. Consider now the following lossless decomposition

$$neg_n = rel(R_1), \bowtie, \ldots, \bowtie rel(R_n) \tag{12}$$

where $rel(R_i)$ is given in figure 9b.

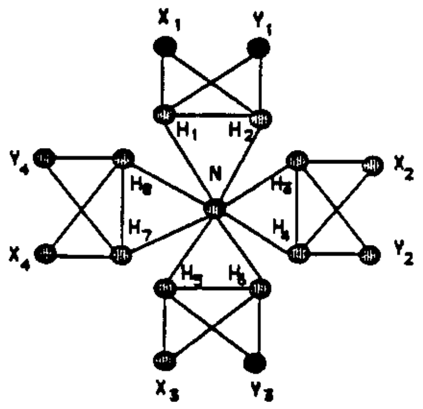
Two 3-valued hidden variables can be used for each $rel(R_i)$, resulting in a total of $2n$ 3-valued hidden variables. The constraint graph of this decomposition is given in Figure 9c.

$$neg_3 = \left\{ \begin{array}{ccccccc} N & X_1 & X_2 & X_3 & Y_1 & Y_2 & Y_3 \\ 0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&1 \\ 0&0&1&0&0&1&0 \\ 0&0&1&1&0&1&1 \\ 0&1&0&0&1&0&0 \\ 0&1&0&1&1&0&1 \\ 0&1&1&0&1&1&0 \\ 0&1&1&1&1&1&1 \\ 1&0&0&0&1&1&1 \\ 1&0&0&1&1&1&0 \\ 1&0&1&0&1&0&1 \\ 1&0&1&1&1&0&1 \\ 1&1&0&0&0&1&1 \\ 1&1&0&1&0&1&0 \\ 1&1&1&0&0&0&1 \\ 1&1&1&1&0&0&0 \end{array} \right\}$$

(a)

$$R_i = \left\{ \begin{array}{ccc} N & X_i & Y_i \\ 0&0&0 \\ 0&1&1 \\ 1&0&1 \\ 1&1&0 \end{array} \right\}$$

(b)



(c)

Figure 9: (a) The *neg* $_3$ relation, (b) Subrealtion R_i, (c) A network decomposition

7. Conclusions

We have shown that any relation can be expressed as a network of binary relations if augmented by hidden variables having three values, while no expressional power is gained by hidden variables having only two values. Specifically, a constructive scheme is presented that decomposes any relation, ρ, into a layered network using $\frac{|\rho|-2}{k-2}$ k-valued hidden variables when $k>2$. We also showed that the scheme is worse-case optimal, meaning that some relations require that many hidden variables. We extended the scheme to exploit an initial lossless decomposition of the relation, if one is available.

Comparing our decomposition scheme with current techniques used in the neural networks community we should consider two systems; those based on the Hebbian rule and those using feedforward networks. The former are restricted to orthogonal vectors, and thus our scheme is more general. The latter have no established theoretical guarantees and often require a long time to converge. In contrast our scheme is complete and it works in time linear in the size of the initial relation. Its drawback, however, is that it requires an a-priori knowledge of the entire relation. Nonetheless, understanding the basic theoretical limitations of architectures using hidden variables should facilitate the development of effective generalizing scheme based on partial relations.

References

[1] Collin, Z. and R. Dechter, ''A distributed solution to the network consistency problem,'' Technion, Haifa, Israel, 1990.

[2] Dechter, R., ''Design of networks with hidden variables,'' Technion, Haifa, Israel, 1990.

[3] Hinton, G.E. and T.J. Sejnowski, ''Learning and relearning in Boltzman machines,'' in *Parallel Distributed Processing*, D.E. Rumelhart, J.L. McClelland, Ed. Cambridge Ma, London England: MIT Press, 1988.

[4] Minsky, M and S. Papert, *Perceptrons*, Cambridge Ma: MIT Press, 1969.

[5] Montanari, U., ''Networks of constraints :fundamental properties and applications to picture processing,'' *Information Science,* Vol. 7, 1974, pp. 95-132.

[6] Rumelhart, D.E., G.E. Hinton, and R.J. Williams, ''Learning internal representation by error propagation,'' *Parallel Distributed Processing , Vol 1,*, 1988.

[7] Rumelhart, D.E. and J.L. McClelland, *Parallel Distributed processing*, Cambridge Massachusetts, London England: MIT Press, 1988.