Computer Methods and Programs in Biomedicine 22 (1986) 223–237
Elsevier

CPB 00767

First Prize

A diagnostic method that uses causal knowledge and linear programming in the application of Bayes' formula

Gregory F. Cooper

Medical Computer Science Group, Knowledge Systems Laboratory, TC-135, Stanford University Medical Center, Stanford, CA 94305, U.S.A.

Bayes' formula has been applied extensively in computer-based medical diagnostic systems. One assumption that is often made in the application of the formula is that the findings in a case are conditionally independent. This assumption is often invalid and leads to inaccurate posterior probability assignments to the diagnostic hypotheses. This paper discusses a method for using causal knowledge to structure findings according to their probabilistic dependencies. An inference procedure is discussed which propagates probabilities within a network of causally related findings in order to calculate posterior probabilities of diagnostic hypotheses. A linear programming technique is described that bounds the values of the propagated probabilities subject to known probabilistic constraints.

Computer-aided diagnosis Probability inference algorithms Artificial intelligence Deep models of medical reasoning Operations research

1. Introduction

In the past 25 years, extensive use has been made in the field of computer-aided medical diagnosis [1–5] of the Bayes' formula:

$$P(\mathrm{H}_i|\mathbf{F}) = \frac{P(\mathbf{F}|\mathrm{H}_i)\times P(\mathrm{H}_i)}{P(\mathbf{F}|\mathrm{H}_i)\times P(\mathrm{H}_i) + P(\mathbf{F}|\text{not } \mathrm{H}_i)\times P(\text{not } \mathrm{H}_i)}$$

$$= \frac{P(\mathbf{F}|\mathrm{H}_i)\times P(\mathrm{H}_i)}{\sum_{j=1}^{N} P(\mathbf{F}|\mathrm{H}_j)\times P(\mathrm{H}_j)} \qquad (1)$$

where **F** is a *set* of findings and H_i is an hypothesis from among a set of N hypotheses. Each hypothesis H_i consists of a *set* of diseases. The N hypotheses correspond to the members of the power set of all diseases within a given diagnostic domain. Thus, the N hypotheses constitute every possible subset of the diseases in the domain. Therefore, within the domain the N hypotheses are exhaustive and mutually exclusive.

One advantage of a computer diagnostic system that is based on Bayes' formula is that it is a formal probabilistic system. In general there are several benefits that result from using a diagnostic method that is based on formal probability theory:

(1) Probability as a field of mathematics is well developed. This aids in designing computation methods for diagnosis, in understanding how they relate to previous research, and in communicating them to other researchers.
(2) Any assumptions made during the scoring of an hypothesis can be unambiguously expressed. This may aid the user in interpreting the resulting score. It may also help the designer of the diagnostic algorithm to determine which assumptions to make in designing the program.

(3) It is possible to utilize available statistical data *directly*.
(4) It is possible to interface the hypothesis score (a probability) directly with other decision making procedures such as decision analysis programs [6,7].

Thus, there is a great attraction to using a formal probability-based system.

In addition, implementations of Bayes' formula yield formal systems that provide a means of using probabilities representing sensitivity information (i.e. $P(\text{findings}\,|\,\text{disease})$) and prevalence * (i.e. $P(\text{disease})$) rather than posterior probabilities (i.e. $P(\text{disease}\,|\,\text{findings})$). The utility of the formula is due to the *availability* of sensitivity and prevalence probabilities as opposed to posterior probabilities. Conditional probabilities in medicine are largely available as sensitivities rather than posterior probabilities. Thus, the literature is more likely to have data in the form of $P(\mathbf{F}\,|\,\mathrm{H}_i)$ than the form $P(\mathrm{H}_i\,|\,\mathbf{F})$, where the set $\mathbf{F}$ would usually contain only one finding in this setting. Additionally, physicians are often more comfortable relating subjective estimates of $P(\mathbf{F}\,|\,\mathrm{H}_i)$ than of $P(\mathrm{H}_i\,|\,\mathbf{F})$. In a like manner, $P(\mathrm{H}_i)$ may be either available from known population statistics or can be estimated for a given population. However, implementations of Bayes' formula almost invariably make the following two assumptions:

(1) *The conditional probabilities of findings given a diagnostic hypothesis are independent.* This means that $P(\mathbf{F}\,|\,\mathrm{H}_i)$, which is used in both the numerator and denominator of the last term in Eq. 1, is approximated by $P(\mathrm{f}_1\,|\,\mathrm{H}_i) \times \ldots \times P(\mathrm{f}_m\,|\,\mathrm{H}_i)$, where $\mathrm{f}_1, \ldots, \mathrm{f}_m$ are the individual findings in the set $\mathbf{F}$.
(2) *A diagnosis consists of a single disease from among a given set of diseases, which are exhaustive and mutually exclusive.* This is the assumption that the patient's clinical condition corresponds to one and only one disease from among a given set of diseases.

The problem with these assumptions is that they are often invalid. The first assumption is particularly questionable. Although in some domains the assumption of conditional independence may result in acceptable diagnostic accuracy [5], there are those in which this is not the case [8]. It seems clear that the assumption of conditional independence cannot be relied upon to yield accurate diagnoses across all fields of medicine under all possible conditions.

The second assumption is also often not valid, particularly in complex cases in which multiple disease diagnoses are likely. It is just these complex situations in which a physician is most likely to seek a consultation.

In summary, most computer programs which implement Bayes' formula have the advantages of being based on formal probability theory and using more readily available statistics. However, in implementing Bayes' formula they make assumptions that are often invalid. This paper discusses a method that has been implemented in a computer program called NESTOR * which avoids these assumptions, when this is possible, in the application of Bayes' formula [9].

NESTOR has been developed to aid physicians in determining the most likely diagnostic hypothesis to account for a set of patient findings. The domain of hypercalcemic disorders was used to test solution methods that should be applicable to other medical areas. This paper discusses the key concepts of NESTOR's diagnostic scoring algorithm in general terms with only minor reference to a specific medical domain. These key concepts have been implemented as an INTERLISP computer program on a DEC PDP 20/60. This paper describes an extension to the methods developed in my Ph.D. dissertation [9]. In particular, linear programming has been incorporated into the diagnostic algorithm in place of special-case inference techniques. The remainder of this paper will discuss this new diagnostic method. The focus will be on the avoidance of assumption (1) above regarding the conditional independence of findings.

* The term prevalence is used here to represent the probability of encountering a given disease or set of diseases within a given medical setting.

* The term NESTOR is taken from the name of a character in Greek mythology who provided well respected advice.

2. The knowledge representation

To perform the calculation in Eq. 1 it is apparent that the probability $P(\mathbf{F}|H_i) \times P(H_i)$ must be calculated for each hypothesis H_i. This probability will be called the *score* of the diagnostic hypothesis H_i given the findings in set **F**. * In this paper we will assume that $P(H_i)$, the prior probability of diagnostic hypothesis H_i, is available as data or can be calculated by methods analogous to those described below. Thus, the remaining task is to compute $P(\mathbf{F}|H_i)$, which is the probability of a particular set of patient findings given hypothesis H_i. The key idea in avoiding the uniform assumption of conditional independence of the findings in set **F** is to explicitly represent any probabilistic dependencies among those findings. This leads to a graph structure connecting findings to each other and to the etiologies (of the diseases of the diagnostic hypothesis) under consideration. The primary assumption in using such a graph is that it represents all the significant probabilistic relationships among the findings and etiologies. Although NESTOR can be used as a probabilistic inference procedure that does not assume that links are causal, in this paper we will assume they are.

Causal knowledge in NESTOR is in the form of causal links that interconnect nodes which represent states or processes. Later we will see how a causal graph is created for a set of findings. This graph connects the etiologies (of the diseases of the diagnostic hypothesis being scored) to the findings through possibly many intermediate causal states.

A fundamentally important aspect of a causal representation is that it can greatly limit the number of possible probabilistic influences on any process or state. For example, Fig. 1 shows a causal graph connecting the disease hyper-

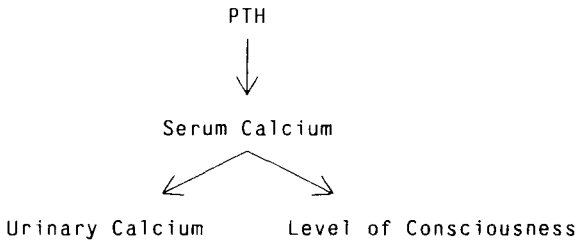

Fig. 1. Causal relationships among four nodes in hyperparathyroidism.

parathyroidism, namely the parathormone (PTH) level, to three of the variables it influences. For a given link, the tail of the arrow indicates the causal node and the head of the arrow its effect. Later we will attach probabilities to such links and they will represent $P(\text{effect}|\text{cause})$.

Note that Fig. 1 makes immediately clear the possible influences of the four nodes on each other. For example, it is apparent that the urinary calcium level cannot affect the serum calcium level or the PTH level. In general an effect is probabilistically dependent on its direct causes. Thus, the urinary calcium (UC) level is probabilistically dependent on the serum calcium (SC) level, or more formally, there exists a value_1 and a value_2 such that $P(\text{UC} = \text{value}_1) <> P(\text{UC} = \text{value}_1 | \text{SC} = \text{value}_2)$, where value_1 and value_2 are variable values (e.g. increased, normal, or decreased) and $<>$ designates inequality. An effect *may* also be probabilistically dependent on its sibling nodes depending on the assumptions of local conditional independence being made. For example, urinary calcium and level of consciousness (LOC) are sibling effects of serum calcium. These two effects may be probabilistically dependent given a value of serum calcium, or more formally, there exists a value_1, a value_2, and a value_3 such that $P(\text{UC} = \text{value}_1 \ \& \ \text{LOC} = \text{value}_2 | \text{SC} = \text{value}_3) <> P(\text{UC} = \text{value}_1 | \text{SC} = \text{value}_3) \times P(\text{LOC} = \text{value}_2 | \text{SC} = \text{value}_3)$. NESTOR allows the user to specify whether or not such sibling nodes are conditionally independent. To summarize, NESTOR uses a causal network of nodes in which a node is probabilistically dependent only on its direct causes and *possibly* on its siblings. This representation of the influences among

* Chapter 5 of Cooper [9] demonstrates that, in general, the most probable diagnostic hypothesis can be determined by calculating the scores of only a small subset of all possible hypotheses. Chapter 6 describes a technique for bounding the posterior probability of the most probable hypothesis, again by calculating the scores of only a small subset of all possible hypotheses.

findings allows NESTOR to be sensitive to their probabilistic dependence when calculating $P(\mathbf{F}|H_i)$. Thus, NESTOR allows known dependencies to be represented, and it therefore performs best in areas of medicine where such dependencies are known. However, we will see shortly that even when causal knowledge is not available, NESTOR yields diagnostic results that are no less accurate than computer programs that uniformly assume conditional independence of the findings.

Causal knowledge has been used in a number of previous medical diagnostic programs in order to structure the probabilistic dependency of findings [10–14]. Patil's program uses causal knowledge, but does not use probabilistic knowledge. Weiss et al. and Ludwig have developed programs which use both causal and probabilistic knowledge, but their inference algorithms are ad hoc and do not necessarily yield the formal probabilities of diagnostic hypotheses. The programs developed by Lemmer and Rouseau also use causal and probabilistic knowledge, and they calculate formal probabilities of diagnostic hypotheses. However, both programs make strong local conditional independence assumptions in the absence of known probabilities among locally interacting nodes. As will be described in detail below, NESTOR is able to avoid uniformly making such local conditional assumptions by using a probability constraint satisfaction method to *bound* probabilities rather than attempt to calculate them exactly.

There are 3 types of nodes in a causal graph in NESTOR: etiology nodes, finding nodes and intermediate nodes. Each node is represented by a single variable which has a range of discrete values (e.g. a binary variable with values true and false). In the case of a continuous variable (e.g. serum calcium level) the values of the variable are divided into discrete intervals (e.g. one interval of the serum calcium level might be 10.5 to 12 mg/100 ml). The granularity of interval values for a continuous variable is assumed to be small enough so that different values within a given interval are not clinically significant. By discretizing continuous variables NESTOR is able to reason uniformly with only discrete valued variables. This simplifies the knowledge representation and the diagnostic algorithm.

An **etiology node** represents some state or process which, when assigned a particular value, represents the etiology of a disease. In Fig. 1 when the etiology node 'PTH' has a value of *increased* it defines the disease hyperparathyroidism. A **finding node** is a variable which may represent observable information about the presentation of a disease. A finding in a particular patient case is represented by assigning a finding node a particular value. An example of a finding is the finding node 'serum calcium' when it is assigned the value *increased*. An **intermediate node** is any node without a known value which causally connects other nodes. For example, a finding node without a known value which connects other nodes in a causal graph is called an intermediate node. In Fig. 1, if the value of serum calcium were not known, then it would be an intermediate node. Intermediate nodes need not be finding nodes, but may represent states or processes which are known to exist, but are not typically classified as finding nodes because of the difficulty or cost of measuring their values.

3. The diagnostic algorithm

The calculation of the score of an hypothesis H_a, namely $P(\mathbf{F}|H_a) \times (P(H_a)$, will be used as a generic example of how $P(\mathbf{F}|H_i) \times P(H_i)$ is calculated. The example will show how an hypothesis is scored in a three-step process. The sample case consists of two findings, f_1 and f_2,

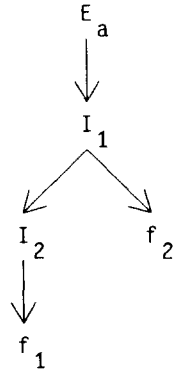

Fig. 2. Step 1: Causal graph generated for the example.

that constitute the set **F**. In this paper we will assume that $P(H_a)$ is known from available statistics or expert estimates. However, techniques similar to those described below can be used to bound the prior probabilities of hypotheses even when etiologies are known to be causally dependent and complete statistical data are not available [9]. Thus, given that $P(H_a)$ is known or can be calculated, the remaining task is to compute $P(\mathbf{F}|H_a)$. In general, an hypothesis H_i is a set of one or more diseases with each disease defined in terms of one or more etiology nodes. For the purposes of the example, assume that H_a consists of one disease D_a which has one etiology E_a.

3.1. Step 1: Create a patient-specific causal graph

The goal of the first step is to create a causal graph that links the etiologies of the diseases of the hypothesis being scored to the patient findings. Recall that findings are just finding nodes that are assigned specific values. In the example, the first step in the scoring process is to construct a causal graph, as shown in Fig. 2, which connects f_1 and f_2 to the etiological node E_a. The graph is generated by starting with each finding and chaining backward to the etiological nodes of the diseases of the hypothesis being scored. The resulting graph contains every possible causal linkage (known to NESTOR) from E_a to f_1 and f_2. I_1 and I_2 are intermediate causal nodes.

Any additional direct causal links to the nodes in this graph (except the etiology node(s)) are added to the graph. This does not occur in Fig. 2, because each of the findings and the intermediate nodes in that example are assumed to have no causal influences other than those shown. A temporary modification of the example will demonstrate the point. Suppose finding f_2 is directly influenced by a node I_3, then a link from I_3 to f_2 would be added to those already in Fig. 2. Suppose further that I_3 does not have a causal pathway that originates from any of the other nodes already in the graph in Fig. 2. In this case I_3 would exist in the graph without any causal predecessors. It would be treated as an etiology node, which means that $P(I_3)$ would have to be calculated if it were not already known.

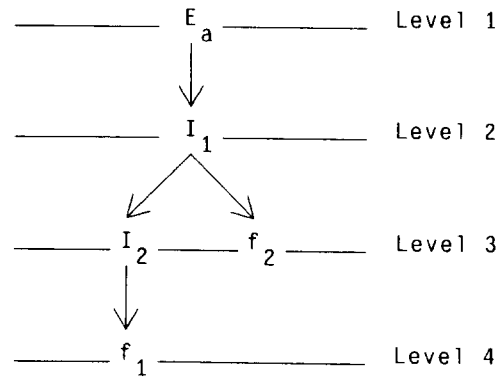

Fig. 3. Step 2: Assigning nodes to levels in the example.

If there is more than one disease in the hypothesis being scored, the etiology nodes of the hypothesis consist of the union of all the disease etiologies. In this way, a case-specific multi-disease causal graph can be constructed. Also, an etiology of one disease in the hypothesis being scored can be causally affected by the etiology of another disease in the hypothesis, in the same way that a finding can be causally linked to another finding.

3.2. Step 2: Segment the causal graph into levels

The second step in scoring the sample hypothesis is to divide the causal graph into levels as shown in Fig. 3. The purpose of this step is to place each of the nodes generated in step 1 into one of several possible levels in the graph. The algorithm classifies node x as belonging to level $n + 1$ if and

E_a ---> I_1:

$P(I_1 \mid E_a) = 0.8$ $P(-I_1 \mid E_a) = 0.2$

$P(I_1 \mid -E_a) = 0.3$ $P(-I_1 \mid -E_a) = 0.7$

I_1 ---> I_2:

$P(I_2 \mid I_1) = 0.4$ $P(-I_2 \mid I_1) = 0.6$

$P(I_2 \mid -I_1) = 0.1$ $P(-I_2 \mid -I_1) = 0.9$

I_1 ---> f_2:

$P(f_2 \mid I_1) = 0.6$ $P(-f_2 \mid I_1) = 0.4$

$P(f_2 \mid -I_1) = 0.2$ $P(-f_2 \mid -I_1) = 0.8$

I_2 ---> f_1:

$P(f_1 \mid I_2) = 0.7$ $P(-f_1 \mid I_2) = 0.3$

$P(f_1 \mid -I_2) = 0.1$ $P(-f_1 \mid -I_2) = 0.9$

Fig. 4. Conditional probabilities used in the example.

only if n is the longest path-length from any etiological node to node x. Although not shown in Fig. 3, a node can be causally affected by multiple nodes and these nodes can exist on the same level or on different levels. In general, then, NESTOR represents a causal graph as a lattice of directed links.

The important property of this ordering is that a node at level j can only be causally influenced by nodes at levels 1 to $j-1$ (i.e. above the jth level). The levels are used in step 3 to order the sequence of calculations.

3.3. Step 3: Compute the score from the segmented causal graph

The initial calculation of step 3 in the scoring process is to iterate over all the possible values of the intermediate nodes computing the following:

P(node values at level $j+1$ | node values at levels 1 to j).

This is the critical calculation of step 3. In what follows we will see how this readily leads to the calculation of $P(\mathbf{F}|\mathrm{H}_a)$. In this example, for sim-

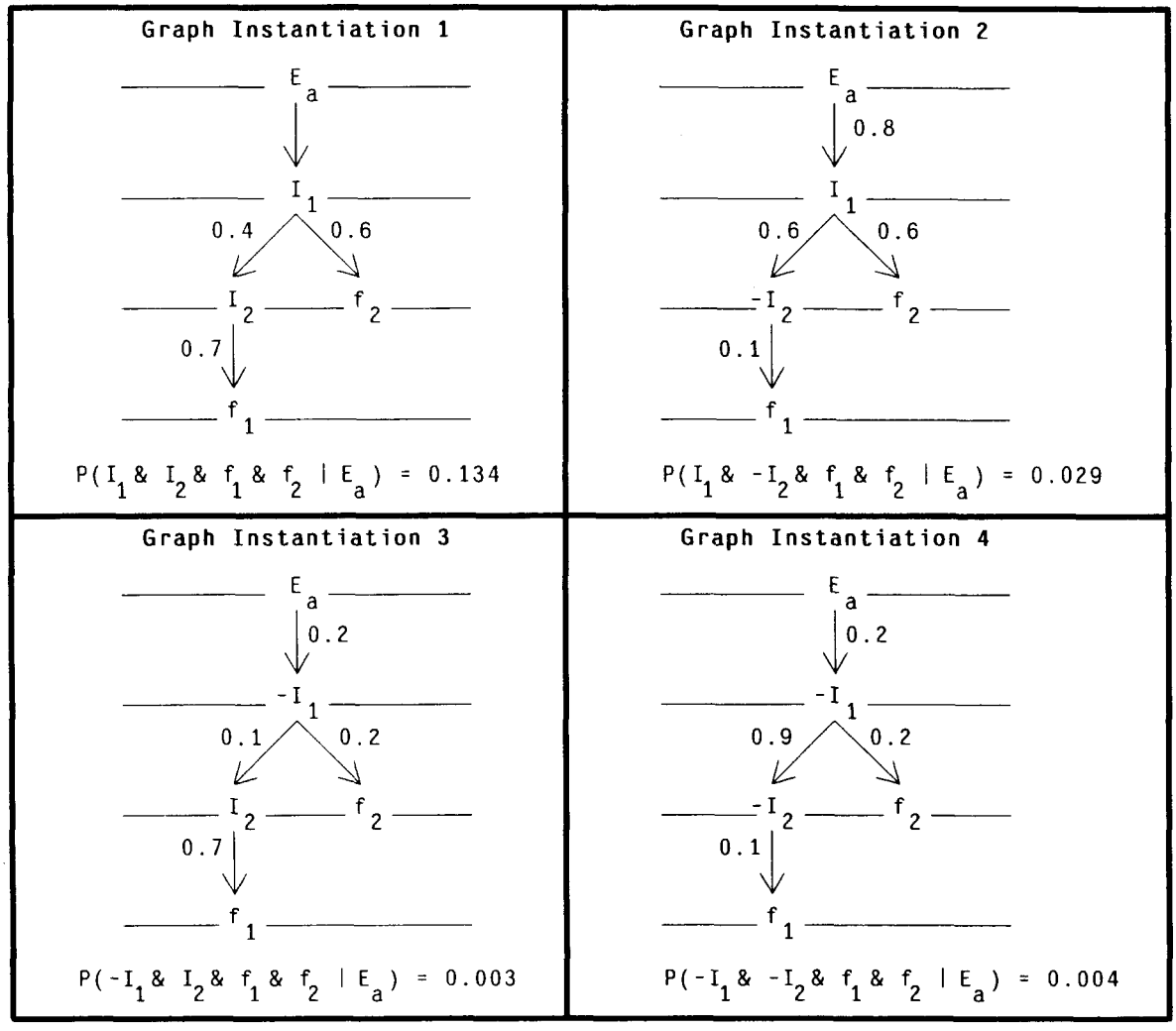

Fig. 5. Step 3: Calculating the probability of causal graphs with specific value instantiations for all intermediate nodes.

plicity, all variables are assumed to be binary. The *value of a level* in the graph will be used to refer to the values of the nodes at that level.

Fig. 4 shows the conditional probabilities associated with the links in the graph of the example. A minus sign before a node label means that the node has the value *false*, otherwise its value is *true*.

Fig. 5 shows one graph for each step in the iteration over the values of the levels, where the goal here is to calculate $P(f_1 \ \& \ f_2 \mid E_a)$. Each of these graphs contains a unique instantiation of intermediate node values. Note that the etiology node E_a and the finding nodes f_1 and f_2 have a constant value of *true*. The relevant conditional probabilities are shown next to the links in the graphs. To score one of these graphs requires calculating the probability that the nodes in the graph will have the values they are assigned, given that the etiology exists. The sum of all four graph scores is 0.169, which is the value of $P(f_1 \ \& \ f_2 \mid E_a)$. The product of $P(f_1 \ \& \ f_2 \mid E_a)$ with $P(E_a)$, where $P(E_a) = P(H_a)$, is the *score* for H_a given f_1 and f_2.

The probability of a particular instantiation of values in a graph given the etiology is called P_{in}. If there are N levels in a causal graph, then P_{in} is calculated as follows:

$$P_{in} = \prod_{j=1}^{N-1} P(\text{node values at level } j+1 \mid \text{node values at levels 1 to } j)$$

Fig. 5 shows the P_{in} value for each possible instantiation of the graph. In order to demonstrate the method of calculating a P_{in}, the P_{in} for Graph 1 in Fig. 5 will be calculated. If the above equation is applied to Graph 1 in Fig. 5, the following equation results:

$$P_{in} = P(\text{level 2 values} \mid \text{level 1 values}) \times P(\text{level 3 values} \mid \text{level 1 to level 2 values}) \times P(\text{level 4 values} \mid \text{level 1 to level 3 values})$$

The value of each of these three conditional probabilities is derived as follows:

$$P(\text{level 2 values} \mid \text{level 1 values}) = P(I_1 \mid E_a) = 0.8$$

Explanation: This is a given probability (see Fig. 4).

$$P(\text{level 3 values} \mid \text{level 1 to level 2 values}) = P(I_2 \ \& \ f_2 \mid I_1) = P(I_2 \mid I_1) \times P(f_2 \mid I_1) = 0.4 \times 0.6 = 0.24$$

Explanation: I_2 and f_2 are assumed to be independent given I_1. This is a form of default reasoning in which the effects of a cause are assumed independent unless some explicit causal connection between them is represented in the knowledge-base. NESTOR gives the user the option of specifying whether it should make this default assumption or use other techniques (to be described below) to bound the value of the joint conditional probability. The conditional independence assumption is used at this point for simplicity of illustration. Note that this local conditional independence assumption differs from a uniform, global assumption of conditional independence among all findings. It differs in that causal knowledge has been used to control for any known dependencies. Thus, the assumption is that the causal knowledge is valid and complete, rather than the assumption that all the findings in a given case are conditionally independent. If the causal knowledge is not complete, and local conditional independence is incorrectly assumed, then the resulting calculation will be at least as accurate, and commonly more accurate, than if conditional independence had been uniformly assumed among all the findings.

$$P(\text{level 4 values} \mid \text{level 1 to level 3 values}) = P(f_1 \mid I_2) = 0.7$$

Explanation: This is given probability (see Fig. 4).

Thus, the P_{in} for graph instantiation 1 is equal to $0.8 \times 0.24 \times 0.7 = 0.134$. The P_{in} values for the other three instantiated graphs are shown in Fig. 5 and are 0.029, 0.003 and 0.004. The four instantiated graphs represent every possible way in which disease D_a can cause the findings f_1 and f_2. The sum of the four P_{in} values is 0.169, which is the value of $P(f_1 \ \& \ f_2 \mid E_a)$. If $P(E_a) = 0.1$, then, since in this case $P(H_a) = P(D_a) = P(E_a)$, it follows that $P(\mathbf{F} \mid H_a) \times P(H_a) = 0.169 \times 0.1 = 0.0169$, which is the score of H_a.

Thus, the purpose of step 3 is to use the segmented causal graph from step 2 to compute $P(\mathbf{F} \mid H_i)$, which then leads to the calculation of $P(\mathbf{F} \mid H_i) \times P(H_i)$. Recall that, given an etiology (or etiologies), the probability of a *particular* assignment of values in a graph with N levels is calculated as follows:

$$P_{in} = \prod_{j=1}^{N-1} P(\text{node values at level } j+1 \mid \text{node values at levels 1 to } j).$$

The values of nodes at a given level in the graph can be more explicitly expressed by using a function VALUES(i, j), which specifies the ith value assignment to the nodes at level j. Here we are assuming that all possible value assignments to the nodes at a given graph level have been totally ordered (i.e. conceptually they are in a list). It does not matter how they are ordered in this list. For example, in Fig. 5 the list of value assignments to level 3 is $((I_2 = T \ f_2 = T)(I_2 = F \ f_2 = T))$. The assignment of values for the nodes at level 3 of graph instantiation 4 in Fig. 5 is specified by VALUES(2, 3), corresponding to $I_2 = F$ and $f_2 = T$.

Using the function VALUES, the calculation of $P(\mathbf{F} \mid H_i)$ can be expressed as follows:

$$P(\mathbf{F} \mid H_i) = \sum_{i_1=1}^{V_1} \cdots \sum_{i_N=1}^{V_N} \prod_{j=1}^{N-1} P(\text{VALUES}(i_{j+1}, j+1) \mid \text{VALUES}(i_1, 1) \ \& \ldots \& \ \text{VALUES}(i_j, j))$$

where:
N is the number of levels in the graph.
V_k is the total number of possible unique value assignments to the nodes at level k. These value assignments are assumed to be ordered from 1 to V_k. In Fig. 5 V_3 is equal to 2 as described above.
VALUES(i_j, j) is the i_jth unique value assignment to the nodes at level j.

Applying this general formula to the example in Fig. 5 results in the following calculation:

$$P(f_1 \ \& \ f_1 \mid E_a) = \sum_{i_1=1}^{1} \sum_{i_2=1}^{2} \sum_{i_3=1}^{2} \sum_{i_4=1}^{1} \prod_{j=1}^{3} P(\text{VALUES}(i_{j+1}, j+1) \mid \text{VALUES}(i_1, 1) \ \& \ldots \& \ \text{VALUES}(i_j, j))$$

where:
VALUES(1, 1) = (E_a = T)
VALUES(1, 2) = (I_1 = T)
VALUES(2, 2) = (I_1 = F)
VALUES(1, 3) = (I_2 = T f_2 = T)
VALUES(2, 3) = (I_2 = F f_2 = T)
VALUES(1, 4) = (f_1 = T)

The above computation of $P(\mathbf{F} \mid H_i)$ as a sum of products is convenient from a notational standpoint, but is not computationally efficient since the time complexity of the calculation increases exponentially as a function of the number of intermediate nodes in a given causal graph. A much more efficient calculation technique, based on storing subcalculation results, is discussed in Section 4.6.1 of Cooper [9], and the interested reader is referred there for details.

$P(\text{VALUES}(i_{j+1}, j+1) \mid \text{VALUES}(i_1, 1) \ \& \ldots \& \ \text{VALUES}(i_j, j))$ is the joint conditional probability of a set of values for the nodes at level $j+1$ given a set of values for nodes at level 1 to level j. If this joint conditional probability were always known precisely, then by using the above techniques the computation of $P(\mathbf{F} \mid H_i)$ would be straightforward. Unfortunately, very few joint conditional probabilities (JCPs) are available as statistics and a JCP is difficult and tedious for an expert to estimate. Thus, NESTOR must use the typically sparse probability knowledge that *is* available, along with any known causal knowl-

edge, in order to bound the conditional probability as tightly as possible. The next section will explain how this is done.

3.4. Computing the probability of level $j+1$ given level 1 to level j

Fig. 6 is an example of causal links between two levels of a larger causal graph. The nodes in the graph will again be assumed to be binary variables in order to simplify the example, although the generalization to multi-valued variables is straightforward. An upper case letter will be used to designate the value of a node as *true* and a lower case used to designate its value as *false*. Thus, Fig. 6 shows all the nodes to be assigned the value *true*. From this point onward we will only be considering nodes with assigned values, and therefore the term 'node' will be used to designate both a node and its value. In Fig. 6, a node at level $j+1$ can be causally influenced only by nodes at level 1 to j. Similarly, although not shown, the nodes at level $j+1$ can causally influence other nodes at levels greater than $j+1$.

The goal for the example in Fig. 6 is to compute $P(V \,\&\, W \,\&\, X \,\&\, Y \,\&\, Z \mid Q \,\&\, R \,\&\, S \,\&\, T \,\&\, U)$. Notice that there are three separate groups of nodes in the figure, namely {Q, R, V, W}, {S, T, X}, and {U, Y, Z}. These sets are called local node groups. A local node group is formally defined as follows: for a given causal graph G, define set **S** to consist of the nodes at level $j+1$ plus the nodes at level 1 to level j which are directly causally connected to the nodes at level $j+1$. Set **S** can be partitioned into causally connected subsets of nodes. Such subsets are called local node groups.

Since NESTOR assumes that all significant causal relationships are represented, the lack of a causal interaction between the three local node groups in the example implies the following:

$$P(V \,\&\, W \,\&\, X \,\&\, Y \,\&\, Z \mid Q \,\&\, R \,\&\, S \,\&\, T \,\&\, U)$$
$$= P(V \,\&\, W \mid Q \,\&\, R) \times P(X \mid S \,\&\, T) \times P(Y \,\&\, Z \mid U)$$

Although NESTOR considers such local node groups to be probabilistically independent, it does not uniformly assume that the nodes within a given group are conditionally independent of each other. Making this latter assumption would be much riskier than the former one, since the causal relationships among the nodes within a group may lead to probabilistic dependencies among them. Often the available probabilities relating nodes within a given group will be insufficient to calculate the JCP for the group as a unique-valued probability. Thus, in NESTOR an upper bound (UB) and lower bound (LB) on the JCP is calculated, resulting in the following bounds of the probability of level $j+1$ given level 1 to level j for the example:

$$\mathrm{UB}[P(V \,\&\, W \,\&\, X \,\&\, Y \,\&\, Z \mid Q \,\&\, R \,\&\, S \,\&\, T \,\&\, U)]$$
$$= \mathrm{UB}[P(V \,\&\, W \mid Q \,\&\, R)] \times \mathrm{UB}[P(X \mid S \,\&\, T)] \times \mathrm{UB}[P(Y \,\&\, Z \mid U)]$$
$$\mathrm{LB}[P(V \,\&\, W \,\&\, X \,\&\, Y \,\&Z \mid Q \,\&\, R \,\&\, S \,\&\, T \,\&\, U)]$$
$$= \mathrm{LB}[P(V \,\&\, W \mid Q \,\&\, R)] \times \mathrm{LB}[P(X \mid S \,\&\, T)] \times \mathrm{LB}[P(Y \,\&\, Z \mid U)]]$$

Such bounds on the conditional probabilities among levels in the graph will result in bounds on $P(\mathbf{F} \mid H_i)$ and thus on $P(\mathbf{F} \mid H_i) \times P(H_i)$. This in turn will result in bounds on $P(H_* \mid \mathbf{F})$ when Bayes' formula is applied to calculate the the posterior probability of some diagnostic hypothesis H_* using a bounded calculation of $P(\mathbf{F} \mid H_i) \times P(H_i)$ for every H_i. When hypotheses have posterior probability ranges which overlap it is only possible to partially order them according to their likelihood. Although it would be preferable to know the unique-valued probability of the posterior probabilities, NESTOR is designed with the philosophy that it is better to represent (via a probability range) only what is known based on available knowledge rather than make assump-

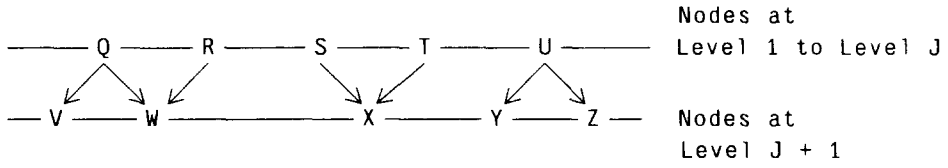

Fig. 6. An example of causal relationships between levels.

tions of conditional independence and risk deriving an inaccurate unique-valued probability.

The next step is to calculate the JCP for each group. From among the three groups in Fig. 6 the calculation of the bounds on $P(Y \& Z \mid U)$ (to be designated $P(YZ \mid U)$) will be used as an example. Fig. 7 shows this group along with the only probabilities assumed to be known relating nodes within the group, namely $P(Y \mid U) = 0.6$ and $P(Z \mid U) = 0.5$. This information can be used to derive upper and lower bounds on $P(YZ \mid U)$. An intuitive derivation for this example will be presented next, and then a general method will be described.

The probability of any group of events can be no greater than the least probable event in the group. Thus, $UB[P(YZ \mid U)] = \text{minimum}\ [P(Y \mid U), P(Z \mid U)] = 0.5$. The lower bound on $P(YZ \mid U)$ is derived by considering the co-occurrence of Y and Z given U to be minimal, subject to the constraints that $P(Y \mid U) = 0.6$ and $P(Z \mid U) = 0.5$. Given U, Y occurs 60% of the time. This leaves 40% of the time in which Z can occur given U and not co-occur with Y. However, since Z given U occurs 50% of the time, this leaves 10% of the time in which Y and Z *must* co-occur given U. Therefore, $LB[P(YZ \mid U)] = 0.1$. Thus, the two given conditional probabilities have led to bounding $P(YZ \mid U)$ between 0.1 and 0.5.

This particular example was solved by case-specific techniques. However, what is needed is a *general* probability constraint satisfaction algorithm. As shown in Fig. 8 such an algorithm would have two inputs. First, for any given local node group there are known probabilities, such as $P(Y \mid U) = 0.6$, that serve as constraints and are obtained either from human estimates or from database statistics or both. * The second input is the goal probability of interest, for example $P(YZ \mid U)$. The goal and the constraints serve as input to some probability constraint satisfaction algorithm which then produces the tightest possible bounds on the goal probability. The question still remains regarding how to implement such an algorithm. Fortunately, it can be framed as a linear programming problem for which there are standard solution procedures such as the simplex algorithm [15,16].

3.5. A linear programming approach to calculating probability constraints

Linear programming is the task of maximizing or minimizing a linear function, called the 'objective function', subject to a finite set of 'linear constraints' that consist of linear equations and linear inequalities [17]. The objective function and linear constraints contain real variables, which will be called the 'linear programming variables.'

Fig. 9 shows a simple example of a linear programming problem. There are two real variables x_1 and x_2, which, for simplicity, will be assumed to be positive. The first linear constraint is that $x_1 \geqslant x_2$, which is represented by the region

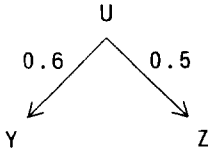

Fig. 7. A group of causally linked nodes from Fig. 6.

* Currently, NESTOR represents only probabilities among nodes *within* a local node group. In the future it may be useful to extend this to allowing the specification of probabilities among nodes that are in different local node groups. Nevertheless, the current representation seems adequate to express the large majority of known probabilistic relationships among causally related medical states or events.

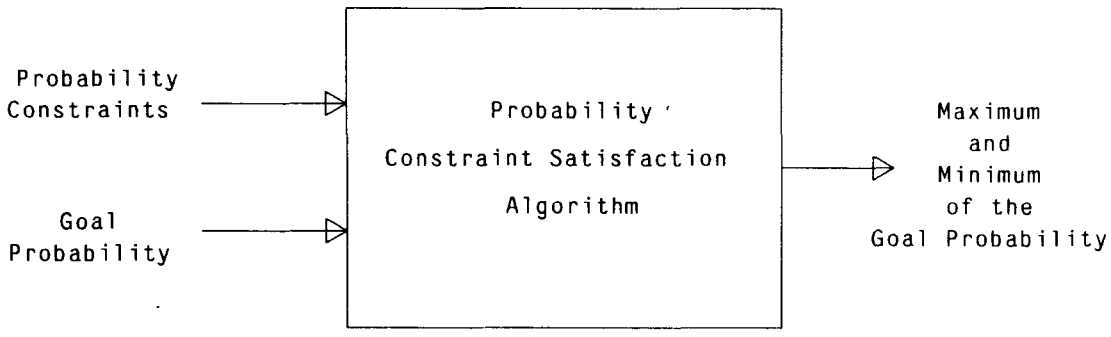

Fig. 8. Local node group calculations.

on the right. The second constraint is that $x_1 + x_2 \leqslant 2$, represented by the region on the left. Note that where the two regions overlap is where both constraints are satisfied. Any solution must lie in this region. The objective function could be any linear function of x_1 and x_2. In the example it is the simple function $3x_2 + 1$. The goal in this example is to maximize the function $3x_2 + 1$ subject to the two constraints. Alternatively, the goal could be to minimize this function. From the graph it is apparent that $x_2 = 1$ is the maximum value of x_2 in the region satisfying both constraints. Since $3x_2 + 1$ is maximum when x_2 is maximum, the objective function is maximized as $3 \times 1 + 1 = 4$.

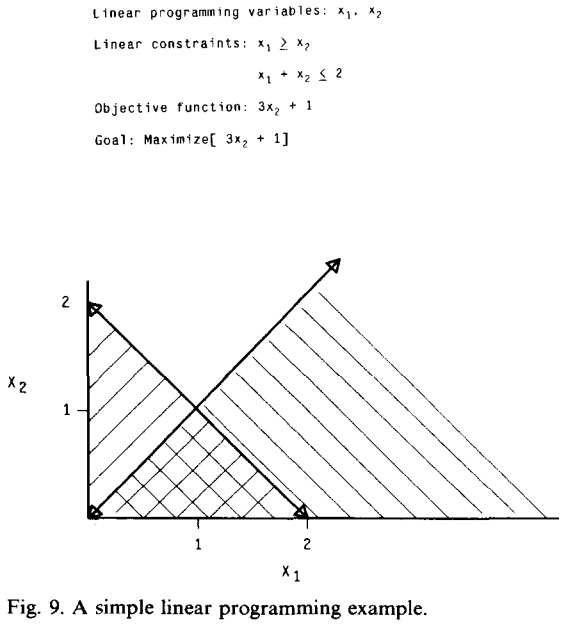

Fig. 9. A simple linear programming example.

A linear programming algorithm is just a method for efficiently solving the general form of problems such as this.

Linear programming has been noted by previous researchers as being applicable to the task of performing probability constraint satisfaction calculations [11,18]. However, to our knowledge our use of linear programming as described in this paper is the first application of this technique for probabilistic inference within a causal network.

We will present a method for using linear programming as a general technique for deriving bounds on a joint conditional probability subject to a set of probability constraints. However, in order to use linear programming some transform is needed to convert probability constraints into linear constraints. To perform a transformation, a probability event space is first defined for a group of N variables (nodes). If each of the N variables is assigned a value, this set of N (variable, value) pairs defines an event. In our example, {(U, *true*), (Y, *true*), (Z, *true*)} is an event, where U, Y and Z are variables and each variable has the value *true*. If upper case letters are used to designate assignments of the value *true* to a variable, and lower case letters denote the value *false* then the above event is represented as {U, Y, Z}.

The probability of a given event is the joint probability of the N (variable, value) pairs in the event. Thus, the probability of event {U, Y, Z} is P(U, Y, Z) or, more simply, P(UYZ). The event space for a group of N variables is the set consisting of all possible events for that group. For the three binary variables U, Y and Z there are $2^3 = 8$ events in the event space. The probability event

space (PES) is defined as the set consisting of the probabilities of all possible events in the event space. In the example, the PES is: $\{P(\mathrm{UYZ}), P(\mathrm{UYz}), P(\mathrm{UyZ}), P(\mathrm{Uyz}), P(\mathrm{uYZ}), P(\mathrm{uYz}), P(\mathrm{uyZ}), P(\mathrm{uyz})\}$.

In order to transform probability constraints into linear constraints, each element of the PES is viewed as a linear programming variable, for example $P(\mathrm{UYZ})$. This approach allows any marginal probability assignment to be represented as a linear constraint of the elements in the PES. For example, the marginal probability $P(\mathrm{UY})$ is defined by the linear function $P(\mathrm{UYz}) + P(\mathrm{UYZ})$. Thus, $P(\mathrm{UY}) = 0.3$ is represented by the linear constraint $P(\mathrm{UYz}) + P(\mathrm{UYZ}) = 0.3$. In general, if the variable–value pairs of some subset of N variables are designated μ, then $P(\mu)$ equals the sum of the probabilities of all events in the event space that have μ as a subset.

Also, any conditional probability assignment among the N nodes can be expressed as a linear constraint. For example, the conditional probability $P(\mathrm{Y}\,|\,\mathrm{U}) = 0.6$ is transformed into a linear constraint as follows:

$$P(\mathrm{Y}\,|\,\mathrm{U}) = \frac{P(\mathrm{UY})}{P(\mathrm{U})} = 0.6$$

which implies that:

$$P(\mathrm{UY}) = 0.6\,P(\mathrm{U})$$

and, by expansion of the marginals $P(\mathrm{UY})$ and $P(\mathrm{U})$:

$$P(\mathrm{UYZ}) + P(\mathrm{UYz}) = 0.6[P(\mathrm{UYZ}) + P(\mathrm{UYz}) + P(\mathrm{UyZ}) + P(\mathrm{Uyz})]$$

which leads to the final linear constraint:

$$P(\mathrm{UYZ}) + P(\mathrm{UYz}) = 0.6\,P(\mathrm{UYZ}) + 0.6\,P(\mathrm{UYz}) + 0.6\,P(\mathrm{UyZ}) + 0.6\,P(\mathrm{Uyz})$$

Similarly, the other conditional probability in the example in Fig. 7, namely $P(\mathrm{Z}\,|\,\mathrm{U}) = 0.5$, is expressed as:

$$P(\mathrm{UYZ}) + P(\mathrm{UyZ}) = 0.5\,P(\mathrm{UYZ}) + 0.5\,P(\mathrm{UYz}) + 0.5\,P(\mathrm{UyZ}) + 0.5\,P(\mathrm{Uyz})$$

If $P(\mu_1)$ and $P(\mu_2)$ represent marginal probabilities defined on some PES and k is some probability constant, then the general method of transforming the conditional probability $P(\mu_1\,|\,\mu_2) = k$ into a linear constraint is to first express the conditional probability as $P(\mu_1\mu_2) = k \cdot P(\mu_2)$, where $\mu_1\mu_2$ designates the union of μ_1 and μ_2. Then, the marginals $P(\mu_1\mu_2)$ and $P(\mu_2)$ in the equation are expressed as linear functions, as described above.

There are several additional constraints which must be specified to express the general laws of probability theory. One constraint states that the sum of all the elements in a given PES equals 1. Other constraints specify that each element of a given PES not be less than 0.

The above linear constraints are linear equations. However, the same transformations convert probabilistic inequalities into linear inequalities. For example, the inequality $P(\mathrm{Y}\,|\,\mathrm{U}) \geqslant 0.6$ would result in the linear inequality:

$$P(\mathrm{UYZ}) + P(\mathrm{UYz}) \geqslant 0.6\,P(\mathrm{UYZ}) + 0.6\,P(\mathrm{UYz}) + 0.6\,P(\mathrm{UyZ}) + 0.6\,P(\mathrm{Uyz})$$

Thus, we now have a general means of representing marginal and conditional probability constraints as linear constraints.

Our goal is to use these constraints to calculate the upper and lower bound on some conditional probability $P(\mu_1\,|\,\mu_2)$, where $P(\mu_1)$ and $P(\mu_2)$ are marginal probabilities of some PES. In the example, the goal is to calculate $P(\mathrm{YZ}\,|\,\mathrm{U})$. The function $P(\mu_1\mu_2)/P(\mu_2)$, which is equivalent to $P(\mu_1\,|\,\mu_2)$, is the function to maximize and minimize subject to known constraints. However, it is not a linear function of the linear programming variables, but rather a ratio of linear functions. Thus, it cannot be directly used as the linear programming objective function. In order to convert this ratio into a linear function, a change of variables is made by dividing each variable or

constant in every known linear constraint by $P(\mu_2)$ and treating the ratio as a new variable. * Thus, in this new linear program $P(\mu_1\mu_2)/P(\mu_2)$ can be represented as a linear function of new variables since $P(\mu_1\mu_2)$ can be represented as a linear function of the old variables and, when each of these is divided by $P(\mu_2)$, a linear function of the new variables is created. ** This transformed linear function is then both maximized and minimized using a linear programming algorithm subject to the new linear constraints to obtain bounds on $P(\mu_1 \mid \mu_2)$.

In the example involving $P(\mathrm{YZ}\mid\mathrm{U})$ as the probability to be maximized and minimized, one of the constraints is converted via change of variables as follows:

$$P(\mathrm{UYZ}) + P(\mathrm{UYz}) = 0.6\,P(\mathrm{UYZ}) + 0.6\,P(\mathrm{UYz}) + 0.6\,P(\mathrm{UyZ}) + 0.6\,P(\mathrm{Uyz})$$

becomes:

$$\frac{P(\mathrm{UYZ})}{P(\mathrm{U})} + \frac{P(\mathrm{UYz})}{P(\mathrm{U})} = 0.6\frac{P(\mathrm{UYZ})}{P(\mathrm{U})} + 0.6\frac{P(\mathrm{UYz})}{P(\mathrm{U})} + 0.6\frac{P(\mathrm{UyZ})}{P(\mathrm{U})} + 0.6\frac{P(\mathrm{Uyz})}{P(\mathrm{U})}$$

where each of the ratios is considered to be a new variable.

The other known constraints are similarly transformed. A linear programming algorithm may then be used to bound $P(\mathrm{YZ}\mid\mathrm{U})$, which in the transformed system is equivalent to maximizing and minimizing the new linear programming variable $P(\mathrm{UYZ})/\mathrm{P(U)}$. The results are the same as those intuitively derived earlier, namely a lower bound of 0.1 and an upper bound of 0.5.

This linear programming method provides substantial generality in the kind of probabilistic constraints which can be specified. Fig. 10 emphasizes

* The general form of this technique is called fractional programming. See Wagner for more details [19].

** This change of variables creates a new linear constraint in which the linear function of old variables corresponding to $P(\mu_2)$ is divided by $P(\mu_2)$ to create a new linear function which must equal 1.

P(Y | U & Z)	P(Y | U & z)	P(Y | u & Z)
P(Y | u & z)	P(y | U & Z)	P(y | U & z)
P(y | u & Z)	P(y | u & z)	P(U | Y & Z)
P(U | Y & z)	P(U | y & Z)	P(U | y & z)
P(u | Y & Z)	P(u | Y & z)	P(u | y & Z)
P(u | y & z)	P(Z | Y & U)	P(Z | Y & u)
P(Z | y & U)	P(Z | y & u)	P(z | Y & U)
P(z | Y & u)	P(z | y & U)	P(z | y & u)
P(Y & U | Z)	P(Y & U | z)	P(Y & u | Z)
P(Y & u | z)	P(y & U | Z)	P(y & U | z)
P(y & u | Z)	P(y & u | z)	P(Y & Z | U)
P(Y & Z | u)	P(Y & z | U)	P(Y & z | u)
P(y & Z | U)	P(y & Z | u)	P(y & z | U)
P(y & z | u)	P(U & Z | Y)	P(U & Z | y)
P(U & z | Y)	P(U & z | y)	P(u & Z | Y)
P(u & Z | y)	P(u & z | Y)	P(u & z | y)
P(Y | U)	P(Y | u)	P(y | U)
P(y | u)	P(Y | Z)	P(Y | z)
P(y | Z)	P(y | z)	P(U | Y)
P(U | y)	P(u | Y)	P(u | y)
P(U | Z)	P(U | z)	P(u | Z)
P(u | z)	P(Z | Y)	P(Z | y)
P(z | Y)	P(z | y)	P(Z | U)
P(Z | u)	P(z | U)	P(z | u)
P(Y)	P(y)	P(U)
P(u)	P(Z)	P(z)
P(Y & U)	P(Y & u)	P(y & U)
P(y & u)	P(Y & Z)	P(Y & z)
P(y & Z)	P(y & z)	P(U & Z)
P(U & z)	P(u & Z)	P(u & z)
P(Y & U & Z)	P(Y & U & z)	P(Y & u & Z)
P(Y & u & z)	P(y & U & Z)	P(y & U & z)
P(y & u & Z)	P(y & u & z)	

Fig. 10. The 98 possible probability expressions for three binary variables.

this by listing all 98 possible probability terms for 3 binary variables that theoretically could be assigned probability values (or bounds) by an expert or from known statistical data. In general it is possible to specify the probability value of any marginal or conditional probability term defined by any of the nodes of a local node group. Of course, typically only a small subset of all the possible probability terms would be assigned probability values and thus would serve as probability constraints. The linear programming method uses all such known probability constraints when calculating the bounds on a joint conditional probability of a local node group.

The examples in this paper have dealt only with binary variables, but the extension to variables with multiple discrete values is straightforward. In particular, the linear programming method is readily generalized. For example, suppose that variable U in Fig. 7 is a trinary variable with values Increased, Normal and Decreased. Furthermore, suppose that variables Y and Z are binary with values True and False. The PES for this case is as follows:

$\{P(\mathrm{U}=\mathbf{D}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{N}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{I}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{D}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{T}),$

$P(\mathrm{U}=\mathbf{N}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{T}),$

$P(\mathrm{U}=\mathbf{I}, \mathrm{Y}=\mathbf{F}, \mathrm{Z}=\mathbf{T}),$

$P(\mathrm{U}=\mathbf{D}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{N}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{I}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F}),$

$P(\mathrm{U}=\mathbf{D}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{T}),$

$P(\mathrm{U}=\mathbf{N}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{T}),$

$P(\mathrm{U}=\mathbf{I}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{T})\}$

Thus, for example the marginal probability $P(\mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F})$ is represented by the linear function $P(\mathrm{U}=\mathbf{D}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F}) + P(\mathrm{U}=\mathbf{N}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F}) + P(\mathrm{U}=\mathbf{I}, \mathrm{Y}=\mathbf{T}, \mathrm{Z}=\mathbf{F})$.

4. Summary

This paper has described a method which applies Bayes' formula to medical diagnosis without assuming that all findings are conditionally independent. The key idea involves using causal knowledge to represent the probabilistic dependencies among findings as a causal graph. When the findings in such a graph are divided into levels, then the computational task is reduced to calculating the probability of one level in the graph given the levels above it. This task was shown to reduce further to that of calculating conditional probabilities among nodes in local node groups. A linear programming technique was described to calculate upper and lower bounds on these local conditional probabilities. The technique computes the tightest bounds possible subject to a given set of probabilistic constraints among the nodes in a local node group. Thus, the overall technique provides a way of incorporating causal and probabilistic constraint knowledge in the calculation of the posterior probabilities of diagnostic hypotheses using Bayes' formula. In those areas of medicine where causal and probabilistic knowledge are known, the technique will generally yield more accurate diagnostic results than if conditional independence among findings is uniformly assumed. Therefore, in such contexts computer-aided medical diagnostic systems should benefit from the incorporation of the basic ideas in this technique.

Acknowledgements

I would like to thank Dr. Ted Shortliffe, Dr. Larry Crapo and Dr. Bruce Buchanan for their insightful discussions and feedback during my work on this research. Funding for this research was provided in part by the Medical Scientist Training Program under NIH grant GM-07365, the Office of Naval Research under ONR contract N00014-81-K-0004, the National Library of Medicine un-

der grant LM-03395, and a grant from the Henry J. Kaiser Family Foundation. The computation facilities were provided by SUMEX-AIM under grant RR-00785 from the Biomedical Research Technology Program of the NIH.

References

[1] M.C. Miller, III, M.C. Westphal, J.R. Reigart and C. Barner, Medical Diagnostic Models: A Bibliography, (University Microfilms International, Ann Arbor MI, 1977).

[2] H.R. Warner, A.F. Toronto, L.G. Veasy and R. Stephenson, A mathematical approach to medical diagnosis: Application to congenital heart disease, J. Am. Med. Assoc. 177 (1961) 177–183.

[3] G. Wagner, P. Tauta and U. Wolber, Problems of medical diagnosis – a bibliography, Meth. Inform. Med. 17 (1978) 55–74.

[4] D.J. Leaper, J.C. Horrocks, J.R. Staniland and F.T. de Dombal, Computer-assisted diagnosis of abdominal pain using estimates provided by clinicians, Br. Med. J. 4 (1972) 350–354.

[5] F.T. de Dombal, D.J. Leaper, J.C. Horrocks, J.R. Staniland and A.P. McCain, Human and computer-aided diagnosis of abdominal pain: Further report with emphasis on performance, Br. Med. J. 1 (1974) 376–380.

[6] W.B. Schwartz, G.A. Gorry, J.P. Kassirer and A. Essig, Decision analysis and clinical judgement, Am. J. Med. 55 (1973) 459–472.

[7] M.C. Weinstein, H.V. Fineberg, et al., Clinical Decision Analysis (W.B. Saunders, Philadelphia PA, 1980).

[8] M.J. Norusis and J.A. Jacquez, Diagnosis. I. Symptom nonindependence in mathematical models of diagnosis, Comput. Biomed. Res. 8 (1975) 156–172.

[9] G.F. Cooper, NESTOR: A Computer-Based Medical Diagnostic Aid that Integrates Causal and Probabilistic Knowledge (Ph.D. dissertation, Medical Information Sciences, Stanford University, 1984).

[10] W.F. Roussen, A method for computing probabilities in complex situations, (Technical report 6252-2, Stanford University Center for Systems Research, Stanford University, 1968).

[11] J.F. Lemmer, Algorithms for Incompletely Specified Distributions in a Generalized Graph Model for Medical Diagnosis (Ph.D. dissertation, University of Maryland, 1976).

[12] J.M. Weiss, C.A. Kulikowski, S. Amarel and A. Safir, A model-based method for computer-aided medical decision-making, Artif. Intell. 11 (1978) 145–172.

[13] D.W. Ludwig, INFERNET – A computer-based system for modeling medical knowledge and clinical inference, *in*: Proceedings of the Fifth Annual Symposium on Computer Applications in Medical Care, ed. H.G. Heffernan, pp. 243–249 (IEEE Computer Society Press, Los Angeles CA, 1981).

[14] R.S. Patil, Causal Representation of Patient Illness for Electrolyte and Acid-Base Diagnosis (Ph.D. dissertation, Department of Computer Science, M.I.T., 1981).

[15] R. Sedgewick, Algorithms (Addison–Wesley, Reading MA, 1984).

[16] M.J. Best and K. Ritter, Linear Programming: Active Set Analysis and Computer Programs (Prentice–Hall, Englewood Cliffs NJ, 1985).

[17] F.S. Hiller and G.J. Lieberman, Introduction to Operations Research (Holden–Day, Oakland, CA, 1980).

[18] K. Konolige, Bayesian methods for updating probabilities (Appendix D, Final Report, SRI Project 6415, Stanford Research Institute International, Palo Alto CA, 1979).

[19] H.M. Wagner, Principles of Operations Research (Prentice–Hall, Englewood Cliffs NJ, 1969).