First-Order Diagnosis by Propositional Reasoning: A Representation-Based Approach*

Helmut Prendinger and **Mitsuru Ishizuka**
Department of Information and Communication Engineering
School of Engineering, University of Tokyo
7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
E-mail: {helmut,ishizuka}@miv.t.u-tokyo.ac.jp

Abstract

We present an approach to diagnosis that first compiles a first-order system description into a propositional representation, and then solves the diagnostic problem as a linear programming instance. The compilation method employs techniques from relevance reasoning and may exploit structural features of the system. To solve diagnostic problems, we use cost-based hypothetical reasoning as an intermediate problem representation. The diagnostic engine is based on a 0-1 integer linear programming method that generates a near-optimal diagnosis in polynomial time. A probabilistic interpretation of the solution diagnoses is possible as well. Preliminary empirical results indicate that our approach has great potential to efficiently solve diagnosis problems traditionally represented in first-order Horn logic.

Introduction and Motivation

A long-standing problem of model-based diagnosis is the complexity underlying the diagnostic reasoning task. Recently, it was shown that the efficiency of the diagnostic engine is highly sensitive to *structural* properties of the system description. For instance, Stumptner and Wotawa (1997) show that diagnosis achieves high speed for tree-structured systems. If the system description is not already in a form that can be processed efficiently, Darwiche (1998) proposes to compile the system description to a sentence called *consequence*, which has provably good computational behavior and guarantees on its size. Another property that gained much less attention is a *representational* property of system descriptions, i.e., the expressivity of the descriptions. This is due to the fact that most approaches to diagnosis only consider propositional representations in the first place. However, many domains are more naturally described in a more expressive language.

In this paper, we allow for system descriptions expressed in first-order Horn logic (without function symbols). In order to preserve the advantageous property of 'propositionality', we compile the first-order description to a propositional one. Our approach is similar in spirit to the work of Williams and Nayak (1997), which starts out with a temporal logic specification of a system, and produces an essentially propositional representation after compilation (by prime implicant generation). By contrast, we start with a first-order specification, and our compilation phase does not rely on the possibly expensive generation of prime implicants.

To achieve efficient propositional representations, techniques from relevance reasoning (Levy, Fikes, & Sagiv 1997) are used to filter out the part of the system that is 'relevant' for certain observation types, and to economically instantiate variables by appropriate constants (usually denoting values of system attributes). As a result, we may retain the compactness of first-order descriptions, while being able to process an efficient propositional theory.

Given a propositional system description, we will use the Networked Bubble Propagation (NBP) mechanism (Ohsawa & Ishizuka 1997), a high-speed hypothetical ('abductive') reasoner, as the diagnostic engine. Focusing on least-expensive (most-probable) diagnoses is realized by assigning a numerical weight to each hypothesis (possible system fault). The NBP mechanism tries to find an optimal solution, i.e., a diagnosis with a minimal sum of individual faults' weights, and actually generates a near-optimal solution in polynomial time of approximately $\mathcal{O}(n^{1.4})$, where n the number of hypotheses in the problem formulation. Since the efficiency of the NBP mechanism relies on methods from 0-1 integer linear programming, diagnosis problems will be translated to instances of linear programming problems. Williams and Nayak (1996) follow a similar approach, by recasting their model-based configuration manager as a combinatorial optimization problem.

The rest of the paper is organized as follows. In the following section, we explain the notions of 'diagnostic problem' and 'diagnosis'. Then we introduce a general framework for extracting information relevant to finding diagnoses, relative to certain observation types and the structure of the system. Moreover, we discuss a rather sophisticated instantiation method based on relevance reasoning. Next, we explain how hypothetical reasoning problems (corresponding to diagnostic prob-

* This paper is a modified version of Prendinger and Ishizuka (1999b), with focus on the reformation phase.

lems) can be seen as problems of integer linear programming. We also report on some preliminary experimental results obtained in testing our approach. Finally, we briefly discuss related work and draw some conclusions.

Definition of Diagnosis

A *diagnostic problem* is characterized by a set of observations to be explained, given a behavioral model of some system (device). The behavioral model of a system describes its normal and/or faulty behavior. A solution to a diagnostic problem (a *diagnosis* for short) is a set of hypotheses which, if assumed, would 'explain' the observations. The following definition borrowed from Console and Torasso (1991) subsumes both abductive and consistency-based definitions of diagnosis. Integrity constraints are not present in the original formulation.

Definition 1 *A* diagnostic problem DP *is a quadruple* $DP = \langle T, I, CXT, \langle O^+, O^- \rangle\rangle$*, such that*

- T *is a set of Horn clauses that contains no cycles, i.e., a state must not be a direct or indirect cause of itself.* T *is called a* behavioral model*. Base predicates in* T *denote either contextual ('boundary', input) conditions or states that can be assumed as hypotheses. We use* $\mathcal{H}$ *to denote the set of hypotheses.*
- I *is a set of* inconsistency constraints*, i.e., a set of Horn clauses of the form*
$$\perp \leftarrow q_1(\bar{X}_1) \wedge ... \wedge q_n(\bar{X}_n)$$
where each q_i *denotes a context condition or a hypothesis, and the symbol* $\perp$ *denotes the logical constant* falsum*.*
- CXT *is a set of variable-free (ground) atoms that denote context conditions.*
- $\mathcal{O}^+$ *denotes the set of observations that have to be covered by the solution;* $\mathcal{O}^-$ *is the set of observations that 'contradict' (or 'conflict with') the observations.*

Console and Torasso (1991) observe that a diagnostic problem (as defined above) has the same structure as a hypothetical reasoning problem (HRP). Consequently, the notion of 'diagnosis' can be explained by giving a definition of a *solution* to a HRP. The following definition extends their definition.

Definition 2 *Let* $HRP = \langle T, I, CXT, \langle \mathcal{O}^+, \mathcal{O}^- \rangle\rangle$ *be a hypothetical reasoning problem, and* $\mathcal{H}$ *a set of hypotheses. A set* $H \subseteq \mathcal{H}$ *is a* solution hypotheses set *for HRP if and only if*

- *for each* $m \in \mathcal{O}^+$*:* $T \cup CXT \cup H \vdash m$*;*
- $T \cup CXT \cup H \cup \mathcal{O}^- \not\vdash \perp$*;*
- $I \cup CXT \cup H \not\vdash \perp$*.*

The first condition says that elements from $\mathcal{O}^+$ must be covered by the solution, whereas the second condition says that $\mathcal{O}^-$ has to be consistent with the solution. We also account for the case where certain solution sets are not admissible, given the inconsistency constraints and system inputs (third condition). This condition is not present in (Console & Torasso 1991).

function $factorize(T)$ **return** a partition $\mathcal{P}$ of T.
Input theory T and $\mathcal{P} = \{\{Q\}\}$, where Q is the query.

- { **for each** $C \in T$
  where C resolves with $\mathcal{D}_1, ..., \mathcal{D}_k \in \mathcal{P}$
  - **if** $k = 0$ **then** $\mathcal{P} = \mathcal{P} \cup \{\{C\}\}$;
  - **if** $k \geq 1$ **then** $\mathcal{P} = (\mathcal{P} \backslash \{\mathcal{D}_1, ..., \mathcal{D}_k\}) \cup \{\mathcal{D}_1 \cup ... \cup \mathcal{D}_k \cup \{C\}\}$;
  - $T = T \backslash \{C\}$;
- **return** $\mathcal{P}$ }

Figure 1: Theory factorizing procedure.

System Reformation

System reformation consists of two steps: first the theory is *factorized* and then *instantiated* by constants. The aim of factorizing is to remove clauses that cannot contribute to the solution of some query (observation) type from the system description. Those clauses are also called *strongly irrelevant* clauses (Levy, Fikes, & Sagiv 1997; Schurz 1999). Factorizing for tree-structured systems has to be applied only once, whereas for acyclic theories, the factorizing procedure has to be invoked for all (combinations of) atomic query types. Instantiation involves binding variables to constants. By using the *query-tree* idea (Levy, Fikes, & Sagiv 1997), this can be done efficiently and economically.

System Factorizing

A theory T is called *tree-structured* if the directed graph corresponding to T consists of subtrees $S_1, ..., S_n$, i.e., each S_i has exactly one top node and there exists only one path from every node in S_i to the top node. Here, the idea of theory factorizing is to split a theory T into disjoint subtheories $T_1, ..., T_n$ such that no clause C in a given subtheory T_i resolves with some clause D from a different subtheory T_j. This means that the search space for a given atomic query type $p(\bar{X})$ can be restricted to a single subtheory T_i (assuming that $p(\bar{X})$ resolves with some clause in T_i). The algorithm is given in Fig. 1.

If the system is *acyclic*, factorizing is performed by means of an algorithm that computes all clauses that are 'reachable' from a query type. Informally, a clause C is *reachable* from a query type $p(\bar{X})$ if there exists some path from $p(\bar{X})$ to the head of C. The set of clauses reachable from $p(\bar{X})$ is denoted by T_p.

Theorem 1 *Let* T *be a first-order Horn theory, and* $p(\bar{X})$ *an admissible query type in* T*.*

- *Suppose* T *is tree-structured and* $\mathcal{P} = \{T_1, ..., T_n\}$ *the partition of* T*. Then any instance of* $p(\bar{X})$ *can be answered by considering exactly one independent subtheory* $T_i \in \mathcal{P}$*. All theories* T_j *with* $i \neq j$ *are strongly irrelevant to any instance of* $p(\bar{X})$ *w.r.t.* T*.*
- *If* T *is acyclic, then any instance of* $p(\bar{X})$ *can be answered by only considering the set* T_p*.*

Thus, in either case, factorizing can be done by only considering the query *types* such as $p(\bar{X})$, i.e., independent of particular instantiations (e.g., $p(a)$ or $p(b)$). The resulting (smaller) theories are now input to the instantiation procedure, to be described in the following subsection.

System Instantiation

In this section we show how the theory is actually instantiated. By employing the query-tree idea (Levy, Fikes, & Sagiv 1997), we obtain exactly the set of ground clauses relevant to a query type, together with all instantiations of the query type that have a solution w.r.t. the theory. In short, a *query-tree* is a compact representation of a search tree for first-order Horn theories and has the form of an AND-OR tree with goal-nodes and rule-nodes. Since we do not allow recursion in clauses, our construction of the query tree is simpler than the original one in (Levy, Fikes, & Sagiv 1997). On the other hand, we allow that some leaves of the query-tree (typically denoting hypotheses) are uninstantiated.

Example 1 *Consider the following theory T where the atom with predicate h denotes a hypothesis.*

(r_1) $p(X,Y) \leftarrow q1(X,Y) \wedge q2(X,Y)$.
(r_2) $q1(X,Y) \leftarrow r1(X,Y) \wedge h(X,Y)$.
(r_3) $q2(X,Y) \leftarrow r2(X,Y)$.
(r_4) $q2(X,Y) \leftarrow r3(X,Y)$.
(f_1) $r1(a,b)$. (f_2) $r1(a,c)$.
(f_3) $r2(a,b)$. (f_4) $r2(c,d)$. (f_5) $r3(b,d)$.

The query tree algorithm consists of two phases (a more detailed description is given in (Levy, Fikes, & Sagiv 1997).

In the *bottom-up* phase, a set of *adorned* predicates and rules is generated. An adorned predicate p^c is a predicate p with constraint c on its arguments. The adorned rules are the rules of the theory with predicates replaced by adorned predicates. We start with the base predicates of the theory, i.e., the predicates of facts and hypotheses. For instance, the adorned predicate $r1^c$ is obtained by completion: $r1(X,Y) \leftrightarrow (X = a \wedge Y = b) \vee (X = a \wedge Y = c)$. For convenience, the adornment of $r1$ is written as $c(X,Y) = \{\langle a,b\rangle, \langle a,c\rangle\}$. Let U be the set of all constants appearing in the theory. Then the adornment of the (uninstantiated) hypothesis h is $\{\langle X,Y\rangle : \langle X,Y\rangle \in U^2\}$. The adornments of predicates in head atoms of rules are generated by projecting the adornments of predicates in body atoms onto the head variables. For instance, the adornment of $q1$ is $\{\langle a,b\rangle, \langle a,c\rangle\}$. The bottom-up phase terminates when no new adornments are generated. In Example 1, the following further predicate adornments are generated: $r2^{\{\langle a,b\rangle,\langle c,d\rangle\}}$, $r3^{\{\langle b,d\rangle\}}$ $q2^{\{\langle a,b\rangle,\langle c,d\rangle,\langle b,d\rangle\}}$, and $p^{\{\langle a,b\rangle\}}$.

In the *top-down* phase, the predicate adornment of the query type is 'pushed down' to the base predicates. Starting with the node of the adorned query type q^c, we construct the query-tree such that each node g of a predicate p has a *label* $l(g)$. Initially the goal-node $l(g) = q^c$ is created (see Fig. 2). A goal-node g for a

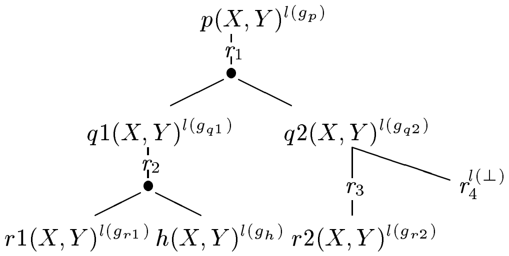

Figure 2: Query-tree for Example 1. The label for each goal-node g is $l(g) = \{\langle a,b\rangle\}$. For simplicity, labels of rule nodes are omitted. Note that expanding node $q2(X,Y)$ with rule r_4 would result in an inconsistent label.

predicate q^{c_h} can be unified with adorned rules r of the form $q^{c_h}(\bar{X}_{n+1}) \leftarrow c \wedge p_1^{c_1}(\bar{X}_1) \wedge ... \wedge p_n^{c_n}(\bar{X}_n)$. If $l(g) \wedge c$ is satisfiable, a rule-node g_r is created as a child of g, with $l(g_r) = l(g) \wedge c$ as its label. For every body atom $p_i^{c_i}$ in r, the rule-node $l(g_r)$ has a child goal-node whose label is the projection of $l(g_r)$ onto $\bar{X}_i$. The top-down construction halts, since nodes of base predicates and nodes with unsatisfiable label, denoted by $l(\perp)$, are not expanded. As shown in (Levy, Fikes, & Sagiv 1997), the complexity of building the query-tree is linear in the number of rules and possibly exponential in the arity of predicates.

The rationale for building the query tree is that *instantiation* of the theory can be performed as a by-product of the top-down construction of the query-tree: if $l(g_r)$ is the label of a rule r in the query-tree, the propositional version of r is obtained by performing all unifications appearing $l(g_r)$. As a result, the following propositional theory is obtained for Example 1.

(r'_1) $p(a,b) \leftarrow q1(a,b) \wedge q2(a,b)$.
(r'_2) $q1(a,b) \leftarrow r1(a,b) \wedge h(a,b)$.
(r'_3) $q2(a,b) \leftarrow r2(a,b)$.
(f'_1) $r1(a,b)$. (f'_3) $r2(a,b)$.

It is a consequence of the construction of the query-tree that $p(a,b)$ is the only instance of the query type $p(X,Y)$ with a solution (given that $h(a,b)$ is assumed). As output of the reformation process, we obtain either independent propositional theories (for tree-structured systems), or propositional theories indexed with query types (for acyclic systems).

Diagnosis as Linear Programming

Hypothetical reasoning problems (HRPs) corresponding to diagnostic problems can be efficiently solved as integer linear programming problems (LPPs). In this section we first outline the translation from HRP to a problem of integer linear programming. Next, the notion of 'best explanation' is explicated within the framework of linear programming. Then, we briefly describe the mechanism of Networked Bubble Propagation (NBP), an efficient hypothetical reasoning method for computing near-optimal solutions.

Hypothetical Reasoning as Integer Linear Programming

An *integer linear programming problem (LPP)* is defined by a set of variables, a set of linear constraints, and an objective function. The set of linear constraints consists of linear inequalities and equalities, and the objective function is a linear function on the variables. A solution for LPP is called *feasible* if it satisfies the constraints, and it is called *optimal* if it is feasible and maximizes (or minimizes) the objective function.

The following translation is similar to the one given by Santos (1994). V denotes the set of all propositional variables occurring in the problem, and $\mathcal{V}$ is the set of variables indexed by V, i.e., $\mathcal{V} = \{x_p : p \in V\}$. Clauses bodies in the behavioral model T are assumed to be either AND-related or OR-related, i.e., (AND) $p \leftarrow q_1 \wedge ... \wedge q_n$, or (OR) $p \leftarrow q_1, ..., p \leftarrow q_m$. The heads of clauses with AND-related and OR-related bodies are called AND-nodes and OR-nodes, respectively. If the body of an OR-node p is a conjunction of propositional variables $q_{k,1} \wedge ... \wedge q_{k,n_k}$, an auxiliary propositional variable $q_{k,0}$ is invented such that $p \leftarrow q_{k,0}$ and $q_{k,0} \leftarrow q_{k,1} \wedge ... \wedge q_{k,n_k}$. For convenience, we define a successor set S^p for each $p \in V$ as follows: $S^p = \{q : q \text{ occurs in an AND-related or OR-related body in clauses with head } p\}$. $|S^p|$ is the cardinality of S^p.

Definition 3 *Let $HRP = \langle T, I, CXT, \langle \mathcal{O}^+, \mathcal{O}^- \rangle \rangle$ be a hypothetical reasoning problem. An* integer linear programming problem (LPP) *corresponding to HRP is a pair $LPP = \langle \mathcal{V}, \mathcal{I} \rangle$, where $\mathcal{V}$ is a set of variables and $\mathcal{I}$ is a finite set of linear inequalities and equalities on $\mathcal{V}$.*
(1) Clauses with AND*-related and* OR*-related bodies are translated to linear inequalities as follows:*

- *Let p be an* AND*-node with successor set S^p.*

$$x_p \leq x_q \ (q \in S^p), \qquad \sum_{q \in S^p} x_q - |S^p| + 1 \leq x_p$$

- *Let p be an* OR*-node with successor set S^p.*

$$\sum_{q \in S^p} x_q \geq x_p, \quad x_p \geq x_q \ (q \in S^p)$$

(2) Inconsistency constraints of the form

$$\perp \leftarrow p_1 \wedge ... \wedge p_n$$

are a special cases of clauses with AND*-related body because their head is the constant $\perp$, where $x_\perp = 0$. Hence, we only need a single linear inequality*

$$\sum_{p \in S^\perp} x_p - |S^\perp| + 1 \leq 0$$

Since $\perp$ is (always) assigned false, some p in $S^\perp$ must be false. This condition guarantees that the inconsistency constraints are satisfied.
Example: *take the constraint "$\perp \leftarrow h_1 \wedge h_2$" and assume that each of the hypothesis is assigned the value true (i.e., each of the hypothesis is assumed). Then we have $1 + 1 - 2 + 1 \leq 0$, i.e., $1 \leq 0$, which violates the constraint.*
(3) For each $x_p \in \mathcal{V}$, x_p is either 0 or 1. In particular, for the variables associated with predicates denoting observations that have to be covered by the solution, or context conditions, we add the equation $x_p = 1$, whereas for variables associated with predicates denoting observations that conflict with observations, we add the equation $x_p = 0$.

Formally, a *variable assignment* for LPP $= \langle \mathcal{V}, \mathcal{I} \rangle$ is a function ϕ from $\mathcal{V}$ to $\{0, 1\}$. ϕ is a *0-1 solution* for LPP if ϕ satisfies all the constraints in $\mathcal{I}$. A *0-1 solution hypotheses set* $H_{\text{0-1}}$ for LPP consists of all $x_p \in \mathcal{V}_{\mathcal{H}}$ which are assigned 1 in the 0-1 solution for LPP. The following theorem due to Santos (1994) also holds when the problem formulation includes inconsistency constraints.

Theorem 2 *H is a solution hypotheses set for HRP if and only if $H_{\text{0-1}}$ is a 0-1 solution hypotheses set for LPP.*

Optimal 0-1 Solutions

In *cost-based* hypothetical reasoning, each hypothesis has an associated numerical weight, and the weight of a solution hypotheses set is simply the sum of the weights associated with hypotheses in the set (Santos 1994; Ohsawa & Ishizuka 1997). A solution hypotheses set is *optimal* if the sum is minimal.

In diagnostic reasoning, it is often desirable to obtain an optimal solution or best explanation; a fault can be said to have low weight if the fault is easy to repair, or even, a fault with low weight is more probable (see (de Kleer 1991) and the discussion below). The idea to concentrate on the preferred (more probable, less expensive) diagnoses is also known as *focusing*. A formal notion of optimal solution (best explanation) is given by defining a function w from $\mathcal{H}$ to the set of natural numbers. The *objective function* being minimized is as follows:

$$\Psi_{LPP} = \sum_{h \in \mathcal{V}_{\mathcal{H}}} x_h w(h)$$

Definition 4 *An* optimal 0-1 solution hypotheses set *for LPP is a 0-1 solution hypotheses set (for LPP) that minimizes Ψ_{LPP}.*

Following the terminology of (Eiter & Gottlob 1995), optimal solution sets correspond to *minimum penalty* solutions, while in the case where all hypotheses are uniformly assigned a default weight, an optimal solution set corresponds to a *minimum cardinality* solution.

Networked Bubble Propagation (NBP)

In practice, it is more advantageous (in terms of efficiency) to search for a *near-optimal* solution rather than for the optimal one. NBP is a method for cost-based hypothetical reasoning that can be used to compute near-optimal diagnoses (Ohsawa & Ishizuka 1997). The search mechanism draws inspiration from 0-1 linear integer programming and improves the behavior of the so-called 'pivot and complement' (PC) method, which is known to find a near-optimal solution close to the optimal solution in polynomial time. The optimal/near-optimal tradeoff is shown experimentally in (Ohsawa & Ishizuka 1997): in all but three out of 250 problems, the obtained solutions were among the best three solutions. In about 90%, the obtained solution is a (strictly) optimal one.[1]

In addition, NBP exploits structural features of the search tree (or network) in order to restrict the pivoting operation to a small subregion of the entire network. Thereby, the costly pivoting operation of the PC method is replaced by the propagation of so-called 'bubbles'. NBP can find a near-optimal solution in polynomial time of approximately $\mathcal{O}(n^{1.4})$, where n is the number of hypotheses in the problem formulation (Ohsawa & Ishizuka 1997). An interesting subclass of acyclic theories are 'singly-connected' theories (also called *polytrees*), well-known from the Bayesian network literature (Charniak 1991). A theory T is called *singly-connected* if the undirected graph corresponding to T has at most one path between any two nodes. Ohsawa and Ishizuka (1997) report on the efficiency gain effected by this structural property of a theory. Note that in the case of singly-connected theories, there cannot exist inconsistencies between hypotheses. The same is true for tree-structured systems. At present, however, we have no results that validate the conjecture that NBP processes tree-structured systems faster than singly-connected systems.

It is important to note that the cost-based variant of hypothetical reasoning has great potential for incorporating notions of uncertain reasoning such as *probability*. Charniak and Shimony (1990) show that costs can be seen as –log probabilities. Poole (1993), on the other hand, shows the correspondence between probabilistic Horn abduction, with prior probabilities $P(h_i)$ associated with hypotheses h_i, and (discrete) Bayesian networks (Pearl 1988), under reasonable assumptions about the domain model. Those connections tie diagnosis (by cost-based hypothetical reasoning) closer to the exciting field of *learning* with Bayesian networks (Heckerman 1996; Poole 1998).

Preliminary Empirical Results

The reformation methods are implemented in Sicstus Prolog, the NBP method is implemented in C. For the experiments we use a Sun Ultra 2 workstation with 320 MB memory. Running times exclude the time needed for system reformation. For example, the relevant part of a theory discussed below (consisting of 1000 clauses) can be extracted in about 4 seconds.

Our experiment involves first-order (acyclic) Horn theories from 40 up to 1000 rules with a fixed percentage of hypotheses (about 30%) and integrity constraints (about 20%) and few facts. The theory of Ex. # 1 is systematically expanded to theories of larger size. Fig. 3 shows the inference time as a function of the number of hypotheses before and after reformation. In both cases, the same solution sets H are generated, each corresponding to a minimal explanation. Results are obtained by averaging over three different query types. The "before reformation" running times refer to the case where NBP is run on the straightforwardly instantiated original theories. The results show that the reformed theories can be processed more efficiently, usually in excess of a factor of 3. For a particular query type in this example, where 98% of the hypotheses are irrelevant, we measured a speedup effect of a factor of 196.

Ex. #	# hypotheses		time (sec)	
	before ref.	after ref.	before ref.	after ref.
1	13	6	0.02	0.01
2	39	17	0.08	0.04
3	78	33	0.21	0.07
4	156	67	0.63	0.16
5	312	129	2.14	0.46

Figure 3: Comparision for first-order Horn theories.

Clearly, this experiment only roughly indicates the efficiency gain of system reformation. The next step will be to compare our representation-based approach to other diagnostic engines that may handle first-order theories, in particular, ACCEL (Ng & Mooney 1992) and AID (Console, Portinale, & Duprè 1996). The efficiency of the NBP method is extensively documented in (Ohsawa & Ishizuka 1997).

Discussion and Conclusion

In this paper, we describe a new approach for generating diagnoses for observations. The diagnostic task is performed in two successive phases.

- *Off-line phase.* A first-order behavioral model is compiled into an equivalent propositional representation.
- *On-line phase.* The reformed system is used for efficient on-line processing. Integer linear programming is employed to generate a near-optimal solution to the diagnostic problem.

The diagnostic engine is representation-based since it only applies to propositional Horn theories. However, as shown in the section on system reformation, the original system description can be formalized in first-order Horn logic (without function symbols).

Our approach shares intuitions with the structure-based approaches of Stumptner and Wotawa (1997) and Darwiche (1998), as the structure of the system description has an impact on the efficiency of diagnostic reasoning. In addition, we also consider the expressivity of the description, by compiling first-order systems to

[1] We are not able to provide guarantees on near-optimal solutions, for instance: a near-optimal solution is at most k times the cardinality of the optimal one, where k is a constant factor.

propositional representations. Moreover, our approach is inspired by the success of the 'planning as satisfiability' approach of Kautz *et al.* (1996), where first-order planning problems are encoded as propositional satisfiability problems.

In a related paper (Prendinger & Ishizuka 1999a), we describe further reformation methods that allow to 'compress' the propositional theory by orders-of-magnitude. So-called *variable elimination* procedures reduce the number of different variables in a clause. This becomes crucial when the clause is to be instantiated by constants, since the instantiation is exponential in the number of different variables in a clause.

In the near future, we want to test our approach on more diverse and more interesting first-order systems. We also plan to develop more specialized reformation tools for systems exhibiting a certain structure.

References

Charniak, E., and Shimony, S. E. 1990. Probabilistic semantics for cost based abduction. In *Proceedings 8th National Conference on Artificial Intelligence*, 106–111.

Charniak, E. 1991. Bayesian networks without tears. *AI Magazine* 12(4):50–63.

Console, L., and Torasso, P. 1991. A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence* 7(3):133–141.

Console, L.; Portinale, L.; and Duprè, D. T. 1996. Using compiled knowledge to guide and focus abductive diagnosis. *IEEE Transactions on Knowledge and Data Engineering* 8(5):690–706.

Darwiche, A. 1998. Compiling devices: a structure-based approach. In *Proceedings 6th International Conference on Knowledge Representation and Reasoning (KR-98)*, 156–166.

de Kleer, J. 1991. Focusing on probable diagnosis. In *Proceedings 9th National Conference on Artificial Intelligence (AAAI-91)*, 842–848.

Eiter, T., and Gottlob, G. 1995. The complexity of logic-based abduction. *Journal of the ACM* 42(1–2):3–42.

Heckerman, D. 1996. A tutorial on learning with Bayesian networks. Technical report, Microsoft Research, Advanced Technology Division, Microsoft Corporation.

Kautz, H.; McAllester, D.; and Selman, B. 1996. Encoding plans in propositional logic. In *Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR-96).*

Levy, A. Y.; Fikes, R. E.; and Sagiv, Y. 1997. Speeding up inferences using relevance reasoning: a formalism and algorithms. *Artificial Intelligence* 97:83–136.

Ng, H. T., and Mooney, R. J. 1992. Abductive plan recognition and diagnosis: A comprehensive empirical evaluation. In *Proceedings 3rd International Conference on Knowledge Representation and Reasoning (KR-92)*, 499–508.

Ohsawa, Y., and Ishizuka, M. 1997. Networked bubble propagation: a polynomial-time hypothetical reasoning method for computing near-optimal solutions. *Artificial Intelligence* 91:131–154.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems.* Santa Mateo: Morgan Kaufmann.

Poole, D. 1993. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence* 64:81–129.

Poole, D. 1998. Learning, Bayesian probability, graphical models, and abduction. In Flach, P., and Kakas, A., eds., *Abduction and Induction: essays on their relation and integration.* Kluwer. Forthcoming.

Prendinger, H., and Ishizuka, M. 1999a. Preparing a first-order knowledge base for fast inference. In *Proceedings 12th International FLAIRS Conference (FLAIRS-99).* To appear.

Prendinger, H., and Ishizuka, M. 1999b. Qualifying the expressivity/efficiency tradeoff: Reformation-based diagnosis. In *Proceedings 16th National Conference on Artificial Intelligence (AAAI-99).* To appear.

Santos, E. 1994. A linear constraint satisfaction approach to cost-based abduction. *Artificial Intelligence* 65:1–27.

Schurz, G. 1999. Relevance in deductive reasoning: A critical overview. In Schurz, G., and Ursic, M., eds., *Beyond Classical Logic.* St. Augustin: Academia Press.

Stumptner, M., and Wotawa, F. 1997. Diagnosing tree structured systems. In *Proceedings 15th International Conference on Artificial Intelligence (IJCAI-97)*, 440–445.

Williams, B. C., and Nayak, P. P. 1996. A model-based approach to reactive self-configuring systems. In *Proceedings 13th National Conference on Artificial Intelligence (AAAI-96)*, 971–978.

Williams, B. C., and Nayak, P. P. 1997. A reactive planner for a model-based executive. In *Proceedings 15th International Conference on Artificial Intelligence (IJCAI-97)*, 1178–1185.