Approximate Model-Based Diagnosis using Preference-Based Compilation

Gregory Provan

Computer Science Department, University College Cork, Cork, Ireland
g.provan@cs.ucc.ie

Abstract. This article introduces a technique for improving the efficiency of diagnosis through approximate compilation. We extend the approach of compiling a diagnostic model, as is done by, for example, an ATMS, to compiling an approximate model. Approximate compilation overcomes the problem of space required for the compilation being worst-case exponential in particular model parameters, such as the path-width of a model represented as a Constraint Satisfaction Problem. To address this problem, we compile the subset of most "preferred" (or most likely) diagnoses. For appropriate compilations, we show that significant reductions in space (and hence on-line inference speed) can be achieved, while retaining the ability to solve the majority of most preferred diagnostic queries. We experimentally demonstrate that such results can be obtained in real-world problems.

1 Objective

One of the most influential approaches to model-based diagnosis (MBD) consists of compiling the diagnostic model into a representation, Θ, from which diagnoses can be more efficiently computed. This approach has been adopted within a number of approaches, e.g., [1–3]. The advantage of this approach is that the computational task is linear in the size of the compiled representation. However, the disadvantage with compiling a large model is the space required for the compilation; for example, for a model represented as a Constraint Satisfaction Problem (CSP), e.g., as a causal network [4], this space is worst-case exponential in the path-width of the CSP [5]. For real-world problems (which have large path-width or thousands of variables), the size of the compiled representation is typically too large for practical inference.

To address the large size of a compiled diagnostic model Θ, we compile a subset of the space of diagnoses, namely the most preferred subset of diagnoses, using a valuation function to specify the most preferred diagnoses. The most common valuation function is the likelihood of a fault, which can be specified in terms of a probability (e.g., [6]) or order-of-magnitude probability (e.g., [4]) assigned to failure modes. We address two well-known diagnostic compilation approaches for which valuations can be assigned to each compiled diagnosis, prime implicants [1] and consequences in d-NNF [2]. We are interested in the tradeoff between the proportion of the most-preferred diagnoses represented in a partial compilation Θ_φ versus the space saved by Θ_φ, relative to that of Θ. We use two measures to analyse this tradeoff for a partial compilation: (1) χ measures the relative fraction of important diagnoses that are generated by the partial compilation Θ_φ, relative to the space of the full compilation; and (2) λ measures the proportion of the space that a partial compilation Θ_φ requires, relative to the space of Θ.[1] We provide a theoretical bound that can be used to predict the tradeoff parameters (χ, λ) for a partial compilation, and show experimental results that such bounds are relevant for real-world problems.

We frame our analysis using the general diagnostic framework of constraint optimisation using CSPs [7]. This framework describes the diagnostic model using the CSP framework, with valuations over the CSP described using a c-semiring. For the class of CSPs we have addressed, our partial compilation results are encouraging. For partial compilations in which all failure modes are unlikely or in which some failure modes are much more likely (preferred) than others, we can produce order-of-magnitude space savings, with little loss of deductive coverage; in other words, we can have compilations with χ close to 1 and $R \ll 1$. Under these scenarios, the most likely diagnoses comprise a small fraction of the number of total diagnoses, with the majority of remaining diagnoses being significantly less likely.

This article makes two main contributions. First, it describes a general framework for MBD in which a variety of valuations and compilation techniques can be adopted. Second, it describes the conditions under which approximate preference-based compilation can significantly speed up diagnostic inference with little loss of diagnostics coverage.

2 Notation and Representation

This section introduces our notation for CSPs, for compilation, and for valuations of solutions to CSPs.

2.1 CSP Problem Formulation

We assume the CSP diagnostic formulation of [7]:

Definition 1 (Constraint Satisfaction Problem (CSP)). *A Constraint Satisfaction Problem (CSP) $\Pi = \langle \mathcal{X}, \mathcal{D}, \mathcal{C}, \mathcal{H} \rangle$ over $\{\top, \bot\}$ consists of:*

- *a set of variables $\mathcal{X} = \{x_1, ..., x_n\}$;*
- *for each variable x_i, a finite set $\mathcal{D}_i$ of possible values (its domain);*
- *and a set $\mathcal{C}$ of constraints restricting the values that the variables can simultaneously take. A constraint c_i is a relation defined on a subset X' of the variables, that is, $c_i \subseteq \times_j \{x_j : x_j \in X'\}$.*

The constraints c_j can be considered as functions defined over the variables in c_j, $V(c_j)$, where allowed tuples have value $\top$ and disallowed tuples have value $\bot$.

Diagnostic applications typically consider the case where (1) we assume a subset of distinguished unary constraints $\mathcal{H} \subseteq \mathcal{C}$ referred to as assumptions, and (2) we can measure a set $O \subseteq \mathcal{X}$ of variables, called observables. Given this framework, we can specify a diagnosis as follows:

[1] λ provides a measure of the relative complexity of approximate compiled inference versus using the full model.

Definition 2 (Diagnosis). *Let $\Pi = \langle \mathcal{X}, \mathcal{D}, \mathcal{C}, \mathcal{H} \rangle$ be a CSP and O an observation, i.e., a constraint on variables in $\mathcal{X}$. A diagnosis of O on Π is a subset of constraints $E \subset \mathcal{H}$ such that $\mathcal{C} \cup O \cup E \not\models \perp$, or equivalently, there exists an assignment of values in $\mathcal{D}$ to $\mathcal{X}$ consistent with $\mathcal{C}$ and O.*

2.2 Valuation

In this article, we assign a valuation to unary constraints (i.e., assumptions), and then use this valuation to compute most preferred diagnoses. A valuation denotes the importance of a constraint. We represent a valuation of a constraint c using $\upsilon(c)$. Hence, we have a weighted-pair (c_i, υ_i) for each constraint and valuation. We formalise this general notion of valuation in terms of c-semiring operations [8]. Note that this formalism covers, among others, the probabilistic valuation of [6] and the order-of-magnitude probabilistic valuation of [4].

Definition 3 (c-semiring). *A c-semiring is a tuple $(A, +, \times, 0, 1)$ such that*

– *A is a set and $\{0, 1\} \in A$;*
– *$+$ is a commutative, associative and idempotent operation with unit element 0 and absorbing element 1;*
– *$\times$ is a commutative, associative and idempotent operation with unit element 1 and absorbing element 0;*
– *$\times$ distributes over +.*

Definition 4 (c-semiring constraint system). *A constraint system over a c-semiring is a constraint system where the constraints $c_j \in \mathcal{C}$ are functions defined over the variables in c_j assigning to each tuple a value in A.*

Definition 5. *An objective function υ maps tuples $Z \subseteq \mathcal{X}$ to a set A with a partial order $\preceq_A$ that forms a complete lattice.*

In the probabilistic case [6] (see Section 3.2), A is the interval $[0, 1]$ with total order $\leq$, and υ associates a probability with each failure mode assignment.

We can define an optimization task over a constraint system in terms of c-semiring operations provided that the objective function is $\times$-separable.

2.3 Compilation

Diagnosis can be formalised as a type of Consistency Maintenance Algorithm, and a number of techniques have been developed for compiling this type of task. These techniques include prime implicates [1], d-NNF [2], OBDD [9], and cluster-trees [10].

Given a set $\mathcal{C}$ of constraints, we compile the constraints after partitioning them into a constant part $\mathcal{C}_c$ and a varying part $\mathcal{C}_v$. The constant part is then replaced by an equivalent, but computationally more efficient, compiled representation $\mathcal{C}'_c$. Thus given an entailment problem for determining consequences α of $\mathcal{C}$, i.e., $\mathcal{C}_c \cup \mathcal{C}_v \models \alpha$, we can compile $\mathcal{C}_c$ into C'_c and express this as

$$\mathcal{C}'_c \models \alpha \vee \bigvee_{\xi \in \mathcal{C}_v} \neg \xi.$$

Prior approximate compilation techniques typically weaken the problem representation. For example, papers by Selman and Kautz [11] and by del Val [12] have studied approximating propositional and First-Order formulae by Horn lowest upper bound (LUB) representations, as well as their generalisations.

In contrast to this approach, we are interested in using the prior valuations on assumptions to compile a subset of *most preferred* potential diagnoses. This is similar to the penalty logic framework introduced in [13], except that in this case we compile only a *subset* of the most preferred diagnoses, rather than the full set of ranked diagnoses. We compile the least-cost diagnoses to $\mathcal{C} \cup \mathcal{H}$ up to a threshold φ. In other words, we compile all diagnoses such that $\upsilon(E) \leq \varphi$. This approach is a general one, and can be applied to any compilation method. For example, with regard to the prime implicants (or labels) computed by an ATMS [1] or consequence generation [2], we ensure that no label (consequence) will have cost more than a bound φ.

3 Valuation-Based Diagnosis

We now introduce some well-known methods for valuations, and in later sections we will see the types of results that are possible given those valuations. We derive some theoretical results about such partial compilations, and then present experimental results for real-world models.

3.1 Valuation 1: Unary Integral Valuation

We first examine the valuation addressed in [4]. The valuation corresponds to a semiring $S_{\mathbb{N}}$ given by $\langle \mathbb{N} \cup \{\infty\}, min, +, \infty, 0\rangle$. This valuation, $\upsilon : \mathcal{H} \rightarrow \mathbb{N}^+$, is assigned to the assumptions, and is a totally ordered mapping over an diagnosis $E \subseteq \mathcal{H}$ such that the valuation for any diagnosis E is given by

$$\upsilon(E) = \sum_{H \in \mathcal{H}} \upsilon(H).$$

In other words, the valuation is a measure assigned to the assumptions (constraints) contained in E; i.e., it represents the likelihood of occurrence of the diagnosis E. Under this valuation, a 0-cost represents a normal system and increasing costs (greater than zero) correspond to increasingly unlikely (less-preferred) diagnoses. Hence, our inference objective is to compute minimum-cost diagnoses.

3.2 Valuation 2: Probabilistic Valuation

We now outline a valuation widely used in diagnosis [3, 6] and other areas of cost-based abduction. In this valuation we assign a probability p to each assumption: $Pr : \mathcal{H} \rightarrow [0, 1]$. The valuation of a diagnosis $E \subseteq \mathcal{H}$ is given by

$$Pr(E) = \prod_{H \in \mathcal{H}} \Pr(H),$$

where we assume that all assumptions are independent, such that we can compute the joint probability $Pr(E)$ by the products of the probabilities for $H \in \mathcal{H}$. The valuation corresponds to a semiring S_{Pr} given by $\langle [0,1], max, \cdot, 0, 1 \rangle$.

The semantics of this valuation are slightly different than those of Valuation 1. Starting from a maximum valuation of 1 (which represents a normal system), all valuations less than 1 correspond to solutions which are increasingly less likely (preferred). Hence, our objective is to compute *maximum*-probability diagnoses.

4 Valuations for Compilations

This section examines the valuations for compilations, and in particular looks at the tradeoffs of relative size of the compilation versus the total relative value of the compilation.

We pose an optimisation task for Valuation 1, that of computing the least-cost diagnoses, and then the compile the least-cost diagnoses up to a threshold $\varphi \in \mathbb{N}^+$. In other words, we compile all diagnoses $E \in \mathcal{E}^*$ such that $\upsilon(E) \leq \varphi$.

4.1 Relative Value of a Partial Compilation

The objective of our approximate compilation is to provide coverage for a fixed percentage of possible diagnosis queries. We use the following notation for specifying the relative value of a partial compilation:

Definition 6 (Constraint Set Valuation). *The valuation associated with a constraint set $\mathcal{H}$ (or equivalently, with a complete compilation Θ of $\mathcal{H}$), is given by the sum over all valuations:*

$$\upsilon(\Theta) = \sum_{E \in 2^{\mathcal{H}}} \upsilon(E).$$

Definition 7 (Partial Constraint Set Valuation). *The valuation of a partial compilation Θ_φ with valuation threshold φ is given by*

$$\upsilon(\Theta_\varphi) = \sum_{E \in 2^{\mathcal{H}}} \{\upsilon(E) | \upsilon(E) \leq \varphi\}.$$

We use these notions to define a key parameter for our experimental analysis, the valuation coverage ratio.

Definition 8 (Valuation Coverage Ratio). *We define the* valuation coverage ratio χ *of a partial compilation Θ_φ, with valuation threshold φ, as the fraction of the complete system valuation provided by Θ:*

$$\chi = \frac{\upsilon(\Theta_\varphi)}{\upsilon(\Theta)}. \qquad (1)$$

Our approach cannot use a valuation with an unbounded maximum value, e.g., ∞ for Valuation 1, as we are interested in computing ratios of cumulative valuations. For such valuations we must construct an inverse valuation, which we call a loss function, in order to compute an appropriate measure for χ. We adopt a standard decision-theoretic loss function $\mathcal{L}$, defined for constraint c as $\mathcal{L} : v(c) \rightarrow [0, 1]$.[2] Using a loss function, we can modify Equation 1 into: $\chi = \frac{\mathcal{L}(\Theta_\varphi)}{\mathcal{L}(\Theta)}$. We say that a partial compilation is *effective* if it has a high coverage ratio.

There are a variety of methods that we can use to map valuations with unbounded maximum values into a loss function. For example, we can define two classes of loss function for Valuation 1 as follows. First, we could adopt an appropriate *log* transformation of the form: $v(e) \rightarrow log_\zeta(\zeta - v(e))$,[3] for a set of diagnoses with maximum valuation ζ. A second method of representing a loss function uses a parameterised equation of the form: $\mathcal{L}(c) = \gamma\epsilon^{v(c)}$, for constant γ and parameter ϵ.[4]

4.2 Relative Memory of a Partial Compilation

The second key parameter in which we are interested is the relative memory of a partial compilation, which we can define as follows. Let $|\Theta|$ be a measure for the size of the original compiled CSP, and $|\Theta_\varphi|$ be a measure for the size of the CSP compiled based on valuation threshold φ. For simplicity, we assume that all diagnoses (solutions) take up equal memory, and define a ratio based only on the relative number of solutions.

Definition 9 (Memory Reduction Factor). *The memory reduction of partial compilation, with respect to compiling the full CSP, is given by* $\lambda = \frac{|\Theta_\varphi|}{|\Theta|}$.

4.3 Analysis of Different Valuations

This section analyses the impact of two parameters on the size and effectiveness of a partial compilation: (1) *valuation distribution*, the relative proportion of different preferences; and (2) *valuation differential*, the difference in degree of preference between any two different valuations. A model may specify a preference ordering in which some assumptions are very strongly preferred, and others are not preferred; in that case the distribution specifies the *relative proportions* of highly preferred to not preferred assumptions, and the differential indicates the difference in *degree* of preference among the assumption valuations.

Example 1. Consider a simple example with a lattice defined over assumptions $\{a, b, c, d\}$ *and semiring* S_{Pr}. *In this case we assign probabilities describing the failure likelihoods of the variables. If we have a valuation distribution given by* $Pr(a) = Pr(b) = 0.09$,

[2] Note that Valuation 2 automatically satisfies this requirement, but Valuation 1 does not.

[3] Note that a complete mapping is more complicated than this example.

[4] This mapping corresponds to the semiring $S_\mathcal{L}$ given by $\langle [0, 1], max, \cdot, 0, 1 \rangle$, and is very close to the calculus proposed in [14].

and $Pr(c) = Pr(d) = 0.01$,[5] *then assume that we have a model in which a full compilation would generate a lattice containing the 13 diagnoses shown in Figure 1.*[6] *The lattice elements represent all possible consistent diagnoses for potential observations; the valuations on the diagnoses can be used to rank-order the diagnoses in terms of likelihood; for example, if we have diagnosis set* $\{cd \vee abd\}$ *for an observation* O*, the most-likely diagnosis will be* $\{cd\}$.

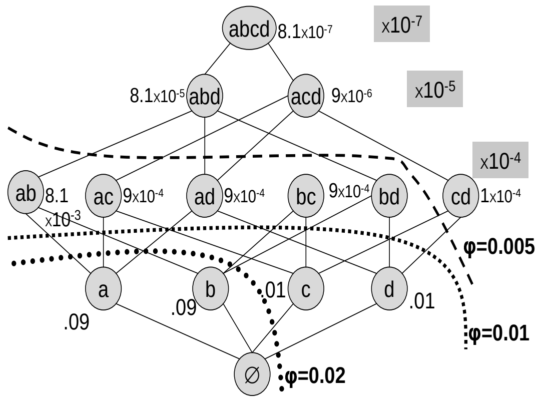

Fig. 1. Lattice for simple example, showing the probabilistic valuations to lattice elements, and the lattice elements (in lower left corner) with probability greater than 0.2, 0.1 and 0.05.

Consider the effect of introducing partial compilations with bound φ*. If we introduce a bound of* $\varphi = 0.02$*, then we would compile only 2 diagnoses (*$\{a, b\}$*) out of 13 total diagnoses; note that this would give valuation coverage ratio* χ *of 0.849, and relative memory* λ *of 0.15. In this case, we have used only 15% of the memory of the total compilation, and can answer roughly 85% of the diagnosis queries. Decreasing* φ *to 0.01 and 0.005 results in* (χ, λ) *pairs of (.943, .308) and (.999, .769) respectively. These results show that we obtain diminishing increases in* χ *as we compile more diagnoses.*

If we increase the differential in this valuation to roughly two orders of magnitude, i.e., $Pr(a) = Pr(b) = 0.09$*, and* $Pr(c) = Pr(d) = 0.001$*, then the relative coverage of an equivalent partial compilation increases significantly. For* $\varphi = 0.02$ *we obtain a* (χ, λ) *pair of (0.992, 0.15), in which 15% of the memory covers over* ***99****% of the most-preferred diagnosis queries.* $\square$

We can study the impact of valuation differential on partial compilation tradeoffs for Valuation 1 using the loss function $\mathcal{L}(c) = \gamma\epsilon^{\upsilon(c)}$, for $\epsilon \leq 1$. Figure 2 shows the impact of the value of ϵ on the types of tradeoff curves that are possible. At one extreme, the value $\epsilon = 1$ produces an equi-loss situation where we generate a flat lattice that does not even respect subset inclusion; hence, there is no value to compilation. The benefit

[5] Note that there is roughly an order-of-magnitude differential between a strong preference, e.g., $Pr(a)$, and a weak preference, e.g., $Pr(c)$.
[6] The lattice elements $\{abc\}$ and $\{bcd\}$ have been ruled inconsistent by the constraints.

of compilation improves as ϵ grows smaller, i.e., as the gap between different valuations increases.

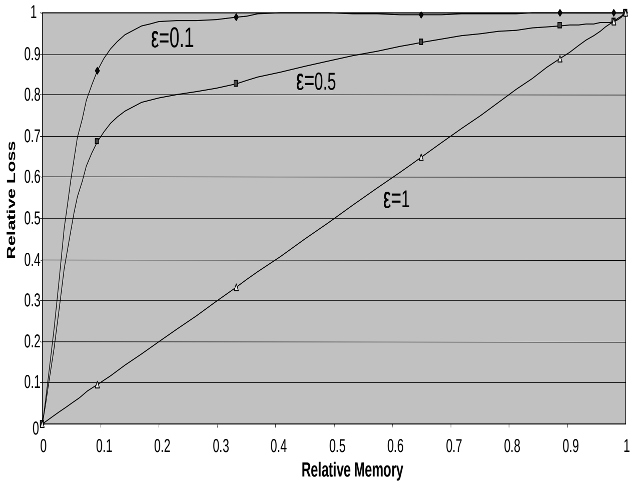

Fig. 2. Curves depicting the influence of the value of ϵ on the tradeoff curves.

In our analysis, we have found that the relative efficiency of a partial compilation, i.e., having a high query coverage with large reduction in memory, is directly related to the type of valuation. If a valuation is skewed, in the sense that some diagnoses are highly preferred and others are not at all preferred, then we can compute a very efficient partial compilation. If most diagnoses are relatively equally preferred, then little is gained by partial compilations.

Although it is too complicated to derive closed-form representations for the valuation tradeoff space in the general case, we can derive results for simple cases. For example, we can use the following result as a bound on the impact of partial compilation:[7]

Lemma 1. *Consider a model with n variables, where each diagnosis has identical loss of $l \in (0, 1]$. We generate a partial compilation, based on the maximum number q of variables in any diagnosis, with parameters given by:*

$$\chi = \frac{\sum_{i=1}^{q} \binom{n}{i} l^i}{(1+l)^n - 1}; \quad \lambda = \frac{\sum_{i=1}^{q} \binom{n}{i}}{2^n - 1}. \tag{2}$$

Equation 2 predicts a series of curves similar to those shown in Figure 2 for $\epsilon < 1$. Each predicted curve defines an upper bound for the expected (χ, λ) results of a partial

[7] If we start with valuation (as in Valuation 1) that generates an inverse χ, then we must map this into a loss function using an approach such as that described in Section 4.1.

compilation, i.e., it specifies the effectiveness of the compilation in the best case. We now present experimental results for real-world examples that show that the predictions of Equation 2 are relevant to real-world application problems.

5 Experimental Analysis

We have performed a set of empirical studies of compilation coverage. We represented the diagnostic models as causal networks [2], which is a CSP representation with propositional constraints and non-negative integer weights similar to Valuation 1. We then generated complete DNNF-compilations of all models, and used various thresholds φ to compute partial compilations from these. For each pair comprising a full and partial compilation, we posed identical queries, and compared the statistics of correct query-responses for the partial and full compilations. This section describes these studies, focusing first on the structural parameters, and then on the weights.

5.1 Structural Parameters

We have performed experiments on a collection of real-world models, including models of hydraulic, electrical and mechanical systems. Table 1 shows the basic parameters of the models we used for experimentation. The model classes are as follows: (a) the AC-v1 through AC-v4 models are for multiple aircraft subsystems; (b) the MCP models are for a control system. Each model class has multiple models, denoted by version numbers (e.g., v2), which denote the increased size and complexity over the basic model (v1).

Name	V	$\mathcal{H}$	O	Memory (KB)
AC-v1	12	5	8	1439
AC-v2	41	9	13	37,626
AC-v3	64	17	12	52,376
AC-v4	70	19	13	52,447
MCP	40	20	5	20.1
MCP-extended-v1	66	32	5	22.5
MCP-extended-v2	66	32	8	359.9

Table 1. Model Statistics for Real-world Models. We report data for the total number V of variables, number $\mathcal{H}$ of assumables, number O of observables, and memory for the full compiled model.

Table 1 shows data for a variety of models. One of the key factors to note is the memory required for the compiled model, displayed in the last column. In particular, the models cover memory values ranging from small (20.1KB) to large (52.4MB). As noted earlier, the memory of the compiled model is our metric for evaluation complexity, since evaluating a model is linear in the size of the compiled data. As a consequence, it is important to note that models with compiled data in excess of 20-30MB are computationally expensive to evaluate.

Model Size Parameters We have performed experiments to study the dependence of compiled-model performance on the parameters $\mathcal{C}$ and $\mathcal{H}$. Figure 3 shows the coverage versus relative memory for four different aircraft sub-system models of increasing size, AC-v1 through AC-v4. All models have identical failure-mode probabilities of 0.05. Note that every such coverage/memory curve has a similar shape, with the coverage asymptotically approaching 1 as memory increases. This particular graph shows how the curves are displaced downwards (meaning reduced coverage for any relative memory value) as the models grow in size and complexity.

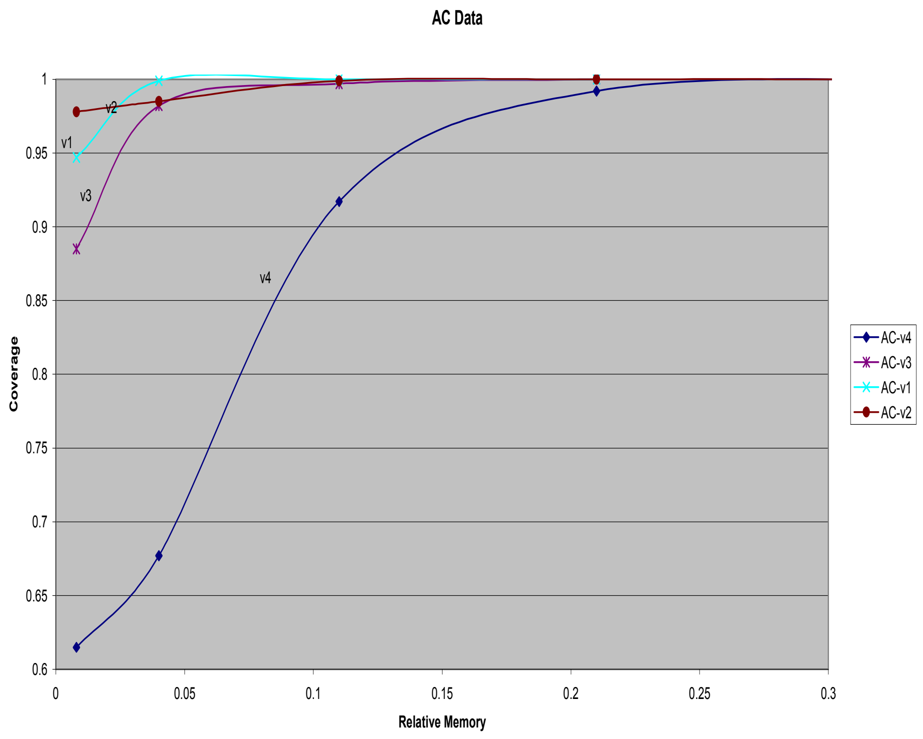

Fig. 3. Graph showing tradeoff of coverage versus relative memory for four different aircraft sub-system models, AC-v1 through AC-v4.

5.2 Valuations

We have performed a variety of experiments to study the influence of assumption probability on coverage. In these experiments, we assigned different probability values to the assumptions, and report our results using the mean probability, averaged over all probabilities assigned to failure-states in $\mathcal{H}$.

Figure 4 shows the effect of mean assumption probability values on the coverage for two control models, a basic model and an extension of that model. These figures show

how the mean probability value reduces the coverage values. For example, Figure 4(b) shows that for small probability values, the coverage asymptotes very quickly to coverage values near 1 at relatively small relative memory values, but as the mean probability gets larger, greater memory is needed to achieve high coverage values. These experimental results concur relatively well with predictions made by equation 2.

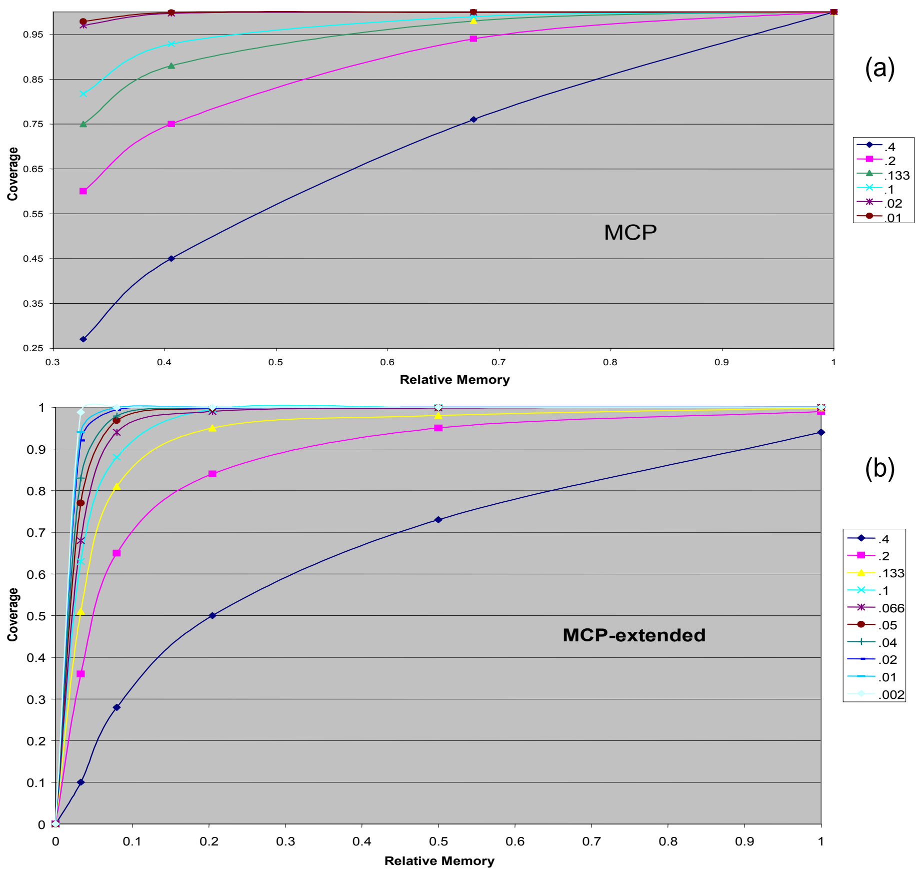

Fig. 4. Graph showing tradeoff of coverage versus relative memory for control sub-system model, for various mean loss values.

This experimental analysis has shown that Equation 2 provides bounds that can predict results such as:

- For a given CSP size, it tells you the type of skewed loss function that guarantees an efficient partial compilation.
- Given an appropriate loss function, it can tell you the coverage value χ that is achievable.

– Given an appropriate loss function and required coverage value χ^*, it can predict the size of the partial compilation.

6 Summary

The article described a partial-compilation technique for improving the efficiency of model-based diagnosis, and more generally for any compilation-based inference with preferences. For DNNF-compilations, we showed that significant reductions in space (and hence on-line inference speed) can be achieved, while retaining the ability to solve the majority of diagnosis queries. We experimentally demonstrated that such results can be obtained in real-world problems. For example, under skewed preference structures, we have found that extremely good coverage can be provided by relatively small partial compilations. Given the general c-semiring CSP framework for compilation, we argue that this partial compilation approach will work for a variety of c-semiring CSPs, and for compilation methods in which valuations can be assigned to compiled diagnoses (e.g., ATMS labels).

These results imply that high-reliability systems need only a relatively small compiled model to guarantee high diagnostic coverage. In contrast, low-reliability systems need a relatively large compiled model, e.g., containing up to 10 simultaneous faults (depending on the system), to guarantee high diagnostic coverage.

This analysis needs to be extended to cover failure consequences, i.e., incorporate utility functions. For many applications, e.g., commercial aircraft and space shuttle missions, one is interested in the low-probability/high-consequence failures. Our future work plans to analyse such situations.

References

1. de Kleer, J.: An Assumption-based TMS. AI Journal **28** (1986) 127–162
2. Darwiche, A.: A compiler for deterministic, decomposable negation normal form. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI), Menlo Park, California, AAAI Press (2002) 627–634
3. Console, L., Portinale, L., Dupre, D.T.: Using Compiled Knowledge to Guide and Focus Abductive Diagnosis. IEEE Trans. on Knowledge and Data Engineering **8(5)** (1996) 690–706
4. Darwiche, A.: Model-based diagnosis using structured system descriptions. Journal of Artificial Intelligence Research **8** (1998) 165–222
5. Bodlander, H.: Treewidth: Algorithmic techniques and results. In: Proc. 22nd Intl. Symp. on Mathematical Foundations of Computer Science, MFCS'97. Volume 1295 of Lecture Notes in Computer Science. Springer-Verlag (1997) 29–36
6. de Kleer, J.: Focusing on Probable Diagnoses. In: Proc. AAAI. (1991) 842–848
7. Sachenbacher, M., Williams, B.: Diagnosis as semiring-based constraint optimization. In: Proceedings of ECAI'04, Valencia, Spain (2004)
8. Bistarelli, S., Montanari, U., Rossi, F.: Semiring-based constraint logic programming: syntax and semantics. ACM TOPLAS **23** (2002)
9. Bryant, R.E., Meinel, C.: Ordered binary decision diagrams. In Hassoun, S., Sasao, T., eds.: Logic Synthesis and Verification. Kluwer Academic Publishers (2001)

10. Pargamin, B.: Extending Cluster Tree Compilation with non-Boolean Variables in Product Configuration. In: IJCAI. (2003)
11. Selman, B., Kautz, H.: Knowledge compilation and theory approximation. Journal of the ACM **43** (1996) 193–224
12. del Val, A.: An analysis of approximate knowledge compilation. In: Proc. IJCAI. (1995) 830–836
13. Darwiche, A., Marquis, P.: Compilation of weighted propositional knowledge bases. In: Proceedings of the Workshop on Nonmonotonic Reasoning, Toulouse, France (2002)
14. Spohn, W.: Ordinal conditional functions: A dynamic theory of epistemic states. In Harper, W.L., Skyrms, B., eds.: Causation in Decision, Belief Change, and Statistics. Reidel, Dordrecht, Netherlands (1988) 105–134