A Semantic Theory of Abstractions

P. Pandurang Nayak
Recom Technologies,
NASA Ames Research Center, MS 269-2
Moffett Field, CA 94035.
nayak@ptolemy.arc.nasa.gov

Alon Y. Levy
AT&T Bell Laboratories
AI Principles Research Department
600 Mountain Avenue, Room 2C-406
Murray Hill, NJ 07974.
levy@research.att.com

Abstract

In this paper we present a semantic theory of abstractions based on viewing abstractions as model level mappings. This theory captures important aspects of abstractions not captured in the syntactic theory of abstractions presented by Giunchiglia and Walsh [1992]. Instead of viewing abstractions as syntactic mappings, we view abstraction as a two step process: first, the intended domain model is abstracted and then a set of (abstract) formulas is constructed to capture the abstracted domain model. Viewing and justifying abstractions as model level mappings is both natural and insightful. This basic theory yields abstractions that are *weaker* than the base theory. We show that abstractions that are *stronger* than the base theory are model level mappings *under certain simplifying assumptions.* We provide a precise characterization of the abstract theory that exactly implements an intended abstraction, and show that this theory, while being axiomatizable, is not always finitely axiomatizable. We present an algorithm that automatically constructs the strongest abstract theory that implements the intended abstraction.

1 Introduction

Abstractions and approximations are pervasive in human common-sense reasoning and problem-solving. Abstractions have been used in a variety of problem-solving settings including planning [Sacerdoti, 1974], theorem proving [Plaisted, 1981], diagnosis [Davis, 1984; Genesereth, 1984; Struss, 1992], compositional modeling [Falkenhainer and Forbus, 1991], constraint satisfaction [Ellman, 1993], and automatic programming [Lowry, 1989]. Until recently there has been no unifying account of these disparate forms of abstractions. However, in the last few years, there has been an explosion of interest in understanding the underlying principles of abstractions and approximations [Ellman, 1992; Lowry, 1992; van Baalen, 1994].

A comprehensive theory of the principles underlying abstractions is useful for a number of reasons. Such a theory can provide the means for clearly *understanding* the different types of abstractions and approximations used in past work. It can provide semantic and computational *justifications* for using abstractions and approximations. Furthermore, such justifications can be used to automatically *construct* useful abstractions and approximations. Finally, an understanding of different abstractions within a common framework can allow the *transfer* of techniques between disparate domains.

Recently, Giunchiglia and Walsh have presented an elegant theory of abstractions that unifies past work and provides a vocabulary to discuss different types of abstractions [Giunchiglia and Walsh, 1992]. Their theory characterizes abstractions as syntactic mappings between formulas of formal systems. They classify abstractions according to whether the set of theorems of the abstract theory are a subset, superset, or equal to the set of theorems of the base theory (TD, TI, and TC abstractions, respectively). Their theory is very good at capturing an important aspect of many abstractions, viz., many abstractions result directly from syntactically manipulating formulas. Moreover, problem solvers ultimately reason by applying inference rules to formulas, and hence understanding the properties of abstractions as mappings between formulas is essential.

However, viewing abstractions as syntactic mappings captures only one aspect of abstractions. Consider the following example.

Example 1 *Predicate abstractions* [Plaisted, 1981; Tenenberg, 1990] are a class of abstractions based on the observation that the distinctions between a set of predicates $P_1,\ldots,P_n$ in a theory are often irrelevant. An abstract theory can be constructed by replacing all occurrences of the P_i's in the base theory by a single abstract predicate P. For example, consider the following base theory:

$$\begin{array}{ll} JapaneseCar(x) \Rightarrow Car(x) & \\ EuropeanCar(x) \Rightarrow Car(x) & (1) \\ Toyota(x) \Rightarrow JapaneseCar(x) & \\ BMW(x) \Rightarrow EuropeanCar(x) & \end{array}$$

The distinction between *JapaneseCar* and *EuropeanCar* is often irrelevant (e.g., when trying to answer a query *Car(A)),* and therefore these predicates can be replaced by *ForeignCar,* yielding the following simpler abstract theory:

ForeignCar(x) =>■ Car(x)
Toyota(x) => ForeignCar(x) (2)
BMW(x) =» ForeignCar(x)

However, suppose the base theory also includes the following:

EuropeanCar(x) => Fast(x) (3)
JapaneseCar(x) => Reliable(x)

Applying the same mapping to axioms 3 would result in the following:

ForeignCar(x) => Fast(x) (4)
ForeignCar(x) => Reliable(x) (4)

However, adding these axioms to the axioms in (2) leads to false proofs [Plaisted, 1981], which may be undesirable. For example, one can infer that Toyotas are fast, and BMWs are reliable—inferences not sanctioned by the base theory.

The syntactic theory of abstractions does distinguish between abstractions that yield false proofs (TI) and those that don't (TD). However, it gives no guidance in comparing TD abstractions to determine which is more natural. For example, the axioms in (2) and (4), when used independently, will not yield false proofs, but the former is more natural, considering the intended interpretation of ForeignCar.[1] Nor does the syntactic theory tell us how to construct the strongest such abstraction. For example, we will see that adding the axiom

ForeignCar(x) => (Fast{x) V Reliable(x)) (5)

to (2) yields the strongest theory that removes predicates JapaneseCar and EuropeanCar and still does not admit false proofs. D

The fundamental shortcoming of the syntactic theory is that while it captures the final result of an abstraction, it does not capture the underlying justification that leads to the abstraction. In this paper we present a semantic theory of abstractions that addresses this shortcoming. Our theory is based on the idea that knowledge representation involves using formulas to capture an intended domain model. From this perspective, we argue that an abstraction should be performed in two steps: first, the intended domain model is abstracted and then a set of (abstract) formulas is constructed to capture the abstracted domain model. Hence, we argue that the decision of what to abstract is made at the model level (using knowledge about relevant aspects of the domain), with the syntactic transformation being justified by this decision. In our example, the intended abstraction to the domain model is to replace the relations denoted by JapaneseCar and EuropeanCar by one relation representing their union, denoted by ForeignCar. As mentioned earlier, the strongest theory that implements this intended abstraction consists of the axioms in (2) and (5).

We introduce a class of model increasing (MI) abstractions, a strict subset of TD abstractions. Like TD abstractions, MI abstractions yield no false proofs. However, they have additional natural properties such as compositionality. We show that the abstract theory that precisely implements the intended model level abstraction, is exactly the strongest MI abstraction of a base theory. We show that if the base theory is axiomatizable, then so is its strongest MI abstraction. We present a procedure to automatically construct the strongest MI abstraction. Our work generalizes Tenenberg's treatment of predicate abstractions [Tenenberg, 1990], and we disprove his conjecture that the predicate abstraction of a finite theory is always finitely axiomatizable.

Abstractions that admit false proofs are commonly used to speed up problem solving by guiding search, e.g., ABSTRIPS [Sacerdoti, 1974]. We show that all such abstractions can be viewed as MI abstractions in conjunction with a set of simplifying assumptions. For example, in ABSTRIPS, we first make the simplifying assumption that a predicate of lower criticality can always be achieved without affecting predicates of higher criticality, and then we construct an MI abstraction by dropping the appropriate preconditions. This formalization is insightful because it shows that an abstraction will yield false proofs only when the simplifying assumption is violated. This enables us to evaluate the utility of an abstraction depending on the reliability of the simplifying assumption.

2 Abstractions as model mappings

Our theory of abstractions applies to any language with a declarative semantics, e.g., propositional logic, constraint languages, first-order logic, modal logic. The declarative semantics of such languages is provided by interpretations of the language and the the notion of satisfaction. An interpretation, /, is a model of a set of sentences, E, (denoted / I= E) if and only if / satisfies each sentence in the set. A set of sentences T1 entails another set of sentences T2 (denoted T1 I= T2) if and only if every model of T1 is a model of T2.

2.1 Model increasing abstractions

Let Tbase and Tabs, be sets of sentences in languages Lbase and Labt > respectively. What does it mean for Tabs to be an abstraction of T base? If Lbase and Labs are the same language, natural definitions are possible (e.g., Tbase |= Tabs is one such option). However, if Lbata and Labs ate different, such a direct comparison is not possible since L base and Labs have no common interpretations. A comparison is possible only if there is a way of translating between the interpretations of the two languages. Such a translation can be specified by an abstraction mapping (Section 3.1 shows how to formally specify n):

IT : Interpretations(Lba$e) -> Interpretations Labs) (6)

The idea is that TT is a model level specification of how the interpretations of L base are to be abstracted to interpretations of Labs • Recall that we view abstractions as consisting of two steps: first, the intended domain

[1] Note that, since the axioms in (1) and (3) are mutually disjoint, the preorder C defined in [Giunchiglia and Walsh, 1992] provides no help in selecting between the two options.

model is abstracted and then the abstract theory is constructed to capture the abstracted domain model. Given only T_{base}, any of its models could be its intended model. Hence, T_{abs} is an abstraction of T_{base}, given the abstraction mapping π, if and only if T_{abs} captures all the abstracted models of T_{base}. This motivates the following definition of *model increasing* (MI) abstractions:

Definition 1 (Model increasing abstractions) *Let T_{base} and T_{abs} be sets of sentences in languages L_{base} and L_{abs}, respectively. Let π : Interpretations$(T_{base}) \rightarrow$ Interpretations(L_{abs}) be an abstraction mapping. T_{abs} is a model increasing abstraction of T_{base}, with respect to π, if for every model M_{base} of T_{base}, $\pi(M_{base})$ is a model of T_{abs}.*

Example 2 Consider the predicate abstraction in Example 1. Let the axioms in (1) and (3) be the base theory. Given any model of the base theory, the abstraction mapping π defines the extension of the abstract predicate *ForeignCar* to be the union of the extensions of *JapaneseCar* and *EuropeanCar*. One can verify that the axioms in (2) and (5) form an MI abstraction of the base theory, i.e., the image of every model of the base theory is a model of (2) and (5). However, the axioms in (4) are not part of any MI abstraction of the base theory, e.g., the image of a model of the base theory which has an unreliable European car is not a model of these axioms. □

A variety of commonly used abstractions are MI abstractions. For example, sign algebras [Williams, 1991] and quantity spaces [Kuipers, 1986] are MI abstractions of the real algebra. Structural and behavioral abstractions used in model-based diagnosis [Genesereth, 1984; Hamscher, 1991] are MI abstractions. Ground abstractions, where a clause is replaced by (some of) its ground instances, used in theorem proving [Plaisted, 1981] are MI abstractions. These applications exploit the following important property of MI abstractions:

Proposition 1 *Let T_{abs} be an MI abstraction of T_{base}. If T_{abs} is inconsistent, then T_{base} is inconsistent.*

Proof: If T_{base} were consistent, it would have a model, and the image of this model would be a model of T_{abs}, making T_{abs} consistent. □

In other words, to prove the inconsistency of a base theory, it suffices to prove the inconsistency of a (potentially simpler) MI abstraction. Another important property of MI abstractions is *compositionality*:

Proposition 2 *Let T_{abs} and S_{abs} be MI abstractions of T_{base} and S_{base}, respectively, with respect to π. Then $T_{abs} \cup S_{abs}$ is an MI abstraction of $T_{base} \cup S_{base}$, with respect to π.*

Proof: Let M_{base} be any model of $T_{base} \cup S_{base}$. Hence, M_{base} is a model of both T_{base} and S_{base}, and hence $\pi(M_{base})$ is a model of both T_{abs} and S_{abs}, and hence a model of $T_{abs} \cup S_{abs}$. □

Compositionality is exploited in diagnosis with multiple theories [Nayak, 1994b], and in compositional modeling [Falkenhainer and Forbus, 1991; Nayak, 1994a; Nayak and Joskowicz, 1996; Iwasaki and Levy, 1994] where theories are built by composing knowledge from different sources. Moreover, one may argue that independent of these applications, theory compositionality is intrinsic to the very notion of abstractions.

When T_{abs} is an MI abstraction of T_{base}, T_{abs} can have models that are *not* the image of any model of T_{base}. Therefore, an MI abstraction can be *weaker* than the intended model-level abstraction. This motivates the definition of the *strongest* MI abstraction, which exactly implements the intended model-level abstraction:

Definition 2 (Strongest MI abstraction) *T_{abs} is the strongest MI abstraction of T_{base} under π if*

$$T_{abs} = \{\sigma : \textit{for all models } M_{base} \textit{ of } T_{base}, \pi(M_{base}) \models \sigma\}$$

2.2 MI Abstractions with simplifying assumptions

While many common abstractions are MI abstractions, a large class of abstractions described in the literature are not. In particular, abstractions that admit false proofs are commonly used to speed up problem solving by guiding the search (e.g., [Ellman, 1993; Imielinski, 1987; Sacerdoti, 1974]). In this section we argue that such abstractions can be, and are best viewed as MI abstractions in conjunction with a set of *simplifying assumptions.*

Example 3 Consider Imielinski's *domain abstractions* [1987]. Given a base theory and an equivalence relation over the set of constants of the base language, he constructs an abstract theory by replacing each occurrence of each constant in the base theory by a representative of the constant's equivalence class. This is not, in general, an MI abstraction of the base theory. For example, let $\{P(a,b), \neg P(c,d)\}$ be the base theory, and let a and c be equivalent (with representative a) and b and d be equivalent (with representative b). Hence, the domain abstraction of this theory is the inconsistent theory $\{P(a,b), \neg P(a,b)\}$. Since the base theory is consistent, Proposition 1 is violated, and hence it can't be an MI abstraction. However, suppose that the equivalence relation was a *congruence*, i.e., for every n-ary relation P and terms t_i, t'_i, $1 \leq i \leq n$, such that t_i and t'_i are equivalent, the base theory entails $P(t_1, \ldots, t_n) \Leftrightarrow P(t'_1, \ldots, t'_n)$. In this case, one can see that the domain abstraction is, indeed, an MI abstraction. To put it another way, domain abstractions are MI abstractions under the simplifying assumption that the equivalence relation is a congruence. If the domain abstraction admits false proofs (as in the above case where the abstraction was inconsistent), it is precisely because the simplifying assumption is violated: the equivalence relation is not a congruence. □

A number of other commonly used abstractions can be viewed in this fashion. As mentioned earlier, the simplifying assumption made in ABSTRIPS [Sacerdoti, 1974] is that the literal dropped from preconditions of actions can always be achieved without affecting the truth value of literals with higher criticality. Davis's work on diagnosis uses an abstract theory that is an MI abstraction of the base theory under the simplifying assumption that there are no *bridge faults* [Davis, 1984]. The heuristic underlying the use of *Connection graphs* in [Chang, 1979] is to use an MI abstraction of the base theory, under the simplifying assumption that different literals in a clause share no common variables, e.g., a clause *P(x)* V *Q(x)* is revised to the stronger clause *P(y)* V *Q(z).* An MI abstraction of the revised theory is constructed using a ground abstraction that preserves all possible unifications.

In the above examples, the simplifying assumption is added to the base theory by simply adding in additional axioms. However, there are common situations in which the simplifying assumption is *inconsistent* with the base theory, so that merely adding in the simplifying assumption makes the theory inconsistent. In such cases, adding a simplifying assumption to a base theory is better viewed as a *belief revision* operation: the base theory is revised to make sure that the simplifying assumption holds, while ensuring that the revised theory is consistent. The revised theory is then abstracted.

Example 4 Most approximations in engineering involve simplifying assumptions that contradict the base theory. For example, consider two railroad cars connected by a linkage. Say that the base theory describing the linkage models it as a spring with a very large spring constant (i.e., as a very stiff spring). It is common to assume that such linkages are rigid, i.e., the spring constant is infinite. Clearly, the simplifying assumption that the spring constant is infinite is inconsistent with the base theory; the base theory must be revised by retracting the axiom specifying the large spring constant of the linkage, and then adding in the simplifying assumption. The revised theory can now be abstracted by combining the two railroad cars into a single, composite rigid body. The *fitting approximations* in [Weld, 1992] are all of this form. □

Viewing abstractions as a combination of a set of simplifying assumptions and an MI abstraction has two key advantages. First, the simplifying assumptions underlying the abstraction are made *explicit,* and therefore can be used in reasoning, as has been done in compositional modeling [Falkenhainer and Forbus, 1991; Iwasaki and Levy, 1994] and diagnosis [Davis, 1984; Nayak, 1994b; Struss, 1992]. Second, we can show that an abstraction will yield false proofs only if the simplifying assumptions are inappropriate. In particular, a simple corollary of Proposition 1 is that if an abstraction of a consistent base theory is inconsistent then it is because the simplifying assumptions are inconsistent with the base theory. This enables us to evaluate the utility of an abstraction depending on the reliability of the simplifying assumption.

3 Abstracting first-order theories

The semantic account of abstractions developed thus far applies to arbitrary languages with a declarative semantics, and to arbitrary model level abstraction mappings 7r. In this section we restrict our attention to first-order languages, and show how abstraction mappings can be specified using *interpretation mappings* [Enderton, 1972]. We use this development to precisely characterize the strongest MI abstraction of a base theory. We show that if the base theory is axiomatisable, then the strongest MI abstraction is also axiomatisable, though not always finitely axiomatisable.

3.1 Interpretation mappings

Let T_{base} be a base theory in language L_{base}, and let L_{abs} be an abstract language. A first-order interpretation consists of a universe of discourse and denotations of the object, function, and relation constants within this universe. Hence, specifying the abstraction mapping π amounts to specifying how the abstract universe and denotations for the abstract object, function, and relation constants are constructed using a base model. This is done using interpretation mappings. A key notion used in specifying interpretation mappings is the relation defined by a well-formed formula (wff) in an interpretation.

Definition 3 (Defined relation) *Let ϕ be a wff in language L with n free variables $v_1, v_2, \ldots, v_n$, and let I be an interpretation of L. The n-ary relation defined by ϕ in I is:*

$$\{\langle a_1, a_2, \ldots, a_n\rangle : \ I \models \phi[a_1/v_1, \ldots, a_n/v_n]\}$$

i.e., the tuple $\langle a_1, a_2, \ldots, a_n\rangle$ is in the defined relation iff I is a model of ϕ with a variable assignment that assigns a_i to v_i, $1 \leq i \leq n$.

The above notion is used in specifying interpretation mappings by finding appropriate formulas in L_{base} that define, in a base model, the abstract universe, and denotations for the abstract object, function, and relation constants. More formally, an interpretation mapping π that maps a model M_{base} of T_{base} to an interpretation $\pi(M_{base})$ of L_{abs} consists of the following (if ϕ has free variables $v_1, \ldots, v_n$, and $x_1, \ldots, x_n$ are any variables, we denote by $\phi(x_1, \ldots, x_n)$ the result of replacing all free occurrences of v_i by x_i, $1 \leq i \leq n$, in ϕ):

1. a wff $\pi_\forall$ with one free variable, v_1, that defines the abstract universe. The idea is that, given any model, M_{base}, of T_{base}, $\pi_\forall$ defines the universe of $\pi(M_{base})$ to be the set defined by $\pi_\forall$ in M_{base}.
2. for each n-ary relation R in L_{abs}, a wff π_R with n free variables, $v_1, \ldots, v_n$, that defines R. The idea is that, given a model M_{base} of T_{base}, π_R defines an n-ary relation in M_{base}. The denotation of R in $\pi(M_{base})$ is this relation restricted to the universe of $\pi(M_{base})$.

In addition, similar wffs are used to specify the denotations of abstract object and function constants (see [Enderton, 1972] for details).

Example 5 Consider Example 1 again. The abstraction mapping π preserves the universe of discourse. Hence, $\pi_\forall$ is the wff $(v_1 = v_1)$, which is satisfied by all elements. The extension of the predicate *ForeignCar* is the union of the extensions of *JapaneseCar* and *EuropeanCar*. Hence $\pi_{ForeignCar}$ is the wff $JapaneseCar(v_1) \vee EuropeanCar(v_1)$. The extension of the other predicates (except *JapaneseCar* and *EuropeanCar*, which are not in the abstract language) is unchanged. Hence, π_{Car} is just the wff $Car(v_1)$, etc. □

3.2 Strongest MI abstraction

Given the above specification of abstraction mappings as interpretation mappings, we now give a precise characterisation of the strongest MI abstraction of a base theory (defined in Definition 2). The characterization of this theory is based on a natural mapping f_π that maps an arbitrary wff ϕ in L_{abs} to a wff $f_\pi(\phi)$ in L_{base}.[2] Intuitively, in a sense to be made precise, $f_\pi(\phi)$ says the same thing as ϕ.

For any wff ϕ in L_{abs}, $f_\pi(\phi)$ is defined recursively as follows. If ϕ is an atom $P(x_1, \ldots, x_n)$ that contains no function or constant symbols, then $f_\pi(\phi)$ is $\pi_P(x_1, \ldots, x_n)$. When ϕ contains object or function constants, the translation is more complex and we refer the reader to [Enderton, 1972] for details. Non-atomic sentences are mapped in the natural way: $f_\pi(\neg\phi) = \neg f_\pi(\phi)$, $f_\pi(\phi \wedge \psi) = f_\pi(\phi) \wedge f_\pi(\psi)$. Finally, $f_\pi(\forall x \phi) = \forall x \pi_\forall(x) \Rightarrow f_\pi(\phi)$, i.e., the translation of quantifiers is restricted to just the elements that satisfy $\pi_\forall$.

Example 6 Continuing Example 5, the translation of the abstract sentence $\forall x ForeignCar(x)$ is $\forall x(x = x) \Rightarrow (JapaneseCar(x) \vee EuropeanCar(x))$. □

The following lemma, taken from [Enderton, 1972], is the key property of f_π. It formalizes the sense in which ϕ and $f_\pi(\phi)$ say the same thing:

Lemma 1 *Let M_{base} be any model of T_{base} and σ be a sentence in L_{abs}. Then*

$$M_{base} \models f_\pi(\sigma) \text{ iff } \pi(M_{base}) \models \sigma$$

i.e., M_{base} is a model of $f_\pi(\sigma)$ if and only if $\pi(M_{base})$ is a model of σ.

The above lemma associates with every sentence σ in L_{abs} an equivalent sentence $f_\pi(\sigma)$ in L_{base}. However, not every sentence in L_{base} has an equivalent sentence in L_{abs}. This is not surprising since, intuitively, L_{abs} is a more abstract language and there is no reason to believe that every base level sentence in L_{base} has an equivalent sentence in L_{abs}.

Theorem 1 *Let T_{abs} be the strongest MI abstraction of a base theory T_{base}, with respect to π, and let T' be a theory in L_{abs} such that*

$$T' = \{\sigma : T_{base} \models f_\pi(\sigma)\}$$

Then $T_{abs} = T'$.

Proof:

$$\begin{array}{lll} \sigma \in T' & \Leftrightarrow & T_{base} \models f_\pi(\sigma) \quad \text{by definition of } T' \\ & \Leftrightarrow & \text{For all models } M_{base} \text{ of } T_{base} \\ & & M_{base} \models f_\pi(\sigma) \\ & \Leftrightarrow & \text{For all models } M_{base} \text{ of } T_{base} \\ & & \pi(M_{base}) \models \sigma \quad \text{by Lemma 1} \\ & \Leftrightarrow & \sigma \in T_{abs} \quad \text{by Definition 2} \end{array}$$

□

Tenenberg's construction of the abstract theory for predicate abstractions is a special case of the above construction. An immediate corollary of the above theorem is that if T_{base} is axiomatisable, so is its strongest MI abstraction:[3]

Corollary 1 *Let T_{abs} be the strongest MI abstraction of T_{base}. If T_{base} is axiomatizable, then so is T_{abs}.*

Proof: A theory is axiomatisable iff it is effectively enumerable [Enderton, 1972]. Since T_{base} is axiomatizable, we can enumerate the theorems of T_{base}. Whenever we encounter a theorem of the form $f_\pi(\sigma)$, where σ is a sentence of L_{abs}, we add σ as a theorem of T_{abs}. Theorem 1 ensures that this will enumerate all the theorems of T_{abs}, and hence T_{abs} is axiomatisable. □

This corollary raises the following natural question. If T_{base} is *finitely* axiomatisable, i.e., T_{base} is the deductive closure of a finite knowledge base, is T_{abs} also finitely axiomatisable?

Theorem 2 *There exist finitely axiomatizable T_{base} such that its strongest MI abstraction is not finitely axiomatizable.*

Proof sketch: The proof is based on showing that while the theory of the structure $(N, 0, S, <, >)$, where N is the set of natural numbers, 0, $<$, and $>$ have their ordinary meaning, and S is the successor function, is finitely axiomatisable, the theory resulting from the predicate abstraction of $<$ and $>$, i.e., $\neq$, is not. □

Note that, in particular, the counterexample in the proof disproves Tenenberg's conjecture that the predicate abstraction of a finitely axiomatisable theory is finitely axiomatisable. A major consequence of Theorem 2 is that, in practice, it is not always possible to construct the strongest MI abstraction, since its axiomatization may be infinite. Hence, we must often settle for axiomatizations that yield weaker MI abstractions.

4 Automatically constructing abstractions

While model-level mappings and simplifying assumptions are usually very natural and easy to specify, it is not always easy to construct the abstract theories that best implement them. In this section we describe a procedure that automatically creates the strongest MI abstraction for a given model-level mapping. This procedure can be complemented with techniques from [Eiter and Gottlob, 1992] when incorporating the simplifying assumptions requires belief revision. We consider the case where *L abs* results from adding a set of new predicates and dropping some old predicates from *L base,* i.e., the object and function constants are unchanged. This covers various common model level abstractions including dropping predicate arguments, taking the union or intersection of a set of predicates, and selecting a subset

[2] Note that while the model mapping π goes from the base theory to the abstract theory, the sentence mapping f_π goes the other way.

[3] A theory T is axiomatizable iff there is a, possibly infinite, decidable set Σ whose deductive closure is T.

```
procedure Construct-abstraction($T_{base}, \mathcal{N}, \mathcal{P}$)
   Let $T$ be the set $T_{base} \cup \mathcal{N}$
   Let $T_{abs}$ be the set of clauses in $T_{base}$
      that contain no predicates in $\mathcal{P}$
   Use any complete resolution strategy on $T$
      to generate new clauses by resolving
      only on literals of predicates in $\mathcal{P}$
   Whenever a generated clause does not contain
      a predicate in $\mathcal{P}$, add the clause to $T_{abs}$
   return $T_{abs}$
end Construct-abstraction
```

Figure 1: Constructing MI abstractions of a base theory, T_{base}; $\mathcal{N}$ is the set of clauses defining the new predicates; $\mathcal{P}$ is the set of predicates to be dropped.

of a predicate's extension. For simplicity, we also assume that the abstract language does not include equality. This procedure is meant for off-line construction of abstractions that are later used at run-time.

Assume that the base theory, T_{base}, is specified as a set of clauses. Let the new predicates in L_{abs} be defined by the clauses in the set $\mathcal{N}$: if P is a new n-ary predicate, $\mathcal{N}$ contains the clauses resulting from the sentence $\forall v_1, \ldots, v_n P(v_1, \ldots, v_n) \Leftrightarrow \pi_P(v_1, \ldots, v_n)$. Let $\mathcal{P}$ be the set of predicates of L_{base} that are dropped in L_{abs}. Figure 1 shows the procedure to construct the resulting strongest MI abstraction. The idea is to first add in the clauses in $\mathcal{N}$ to T_{base}, and then to use any refutation complete resolution strategy, e.g., linear resolution, to generate clauses. Only resolutions on the predicates in $\mathcal{P}$ are considered. A generated clause that contains none of the predicates in $\mathcal{P}$ is added to T_{abs}. The following theorem tells us that this procedure is correct:

Theorem 3 *Procedure* **Construct-abstraction** *constructs the strongest MI abstraction of* T_{base}.

Proof sketch: Since the clauses in T_{abs} are a consequence of $T_{base} \cup \mathcal{N}$, it is easy to show that T_{abs} is an MI abstraction of T_{base}. To show that it is the strongest MI-abstraction of T_{base}, we define the class of *Herbrand-like* models, and show that every Herbrand-like model of T_{abs} is an image of a model of T_{base}. Then, we show that every model of T_{abs} is elementarily equivalent to a Herbrand-like model. (Two models are elementarily equivalent iff the same set of first-order sentences are true in both of them.) □

Clearly, procedure **Construct-abstraction** will not terminate if an infinite number of resolutions are possible, leading to an infinite T_{abs}. In such cases, the procedure can be terminated at any time to yield a weaker MI abstraction.

Example 7 Consider constructing the strongest MI abstraction of the axioms in (1) and (3). The axiom defining *ForeignCar* is

$$ForeignCar(x) \Leftrightarrow JapaneseCar(x) \vee EuropeanCar(x)$$

Adding this axiom to (1) and (3) and resolving on predicates *JapaneseCar* and *EuropeanCar* yields the axioms in (2) and (5). This is the strongest MI abstraction. □

We conclude this section with an application of the procedure in an analysis of ABSTRIPS. This analysis shows how the simplifying assumption and the model-level mappings justify the commonly used syntactic mapping, implementing the abstraction.

Example 8 ABSTRIPS solves planning problems by first solving abstract problems using abstract operators, and then using the abstract solution to guide the search in the base level problem. Each predicate is assigned a *criticality level*. Operators are abstracted at level i by dropping preconditions whose criticality is less than i. We now discuss the precise semantic steps involved in constructing this abstraction. We use Green's formulation of planning problems [Green, 1969]: actions are represented by axioms of the form $q_1(s) \wedge \ldots \wedge q_n(s) \Rightarrow r(do(A, s))$, where q_i are literals representing preconditions and r the postcondition of operator A, and $do(A, s)$ is the state resulting from executing A in state s.

Let $q_1(s) \wedge q_2(s) \wedge p(s) \Rightarrow r(do(A, s))$ be an operator, and suppose that the criticality of p is less than the current abstraction level. Intuitively, this means that p can be achieved in any state without affecting the truth values of predicates at a higher level of criticality. We formalize this simplifying assumption using the axiom

$$\forall s \exists a (p(do(a, s)) \wedge \phi) \qquad (7)$$

where ϕ denotes the frame axioms expressing the fact that the action a does not affect the truth value of any other predicate, e.g., $q_1(s) \Leftrightarrow q_1(do(a, s))$.

Next, we add in this simplifying assumption to the base theory. One of the clauses resulting from the simplifying assumption is $p(do(f(s), s))$, where f is a skolem function. Note that, because of the frame axioms, the only possible difference between a state s and a state $do(f(s), s)$ is in the truth value of p.

Now we use procedure **Construct-abstraction** to drop the predicate p, by resolving on p in all possible ways. For example, resolving the simplifying assumption with the above operator leads to the operator

$$q_1(do(f(s), s)) \wedge q_2(do(f(s), s)) \Rightarrow r(do(A, do(f(s), s)))$$

Since now the language does not contain the predicate p, it follows that the same set of facts are true at the states s and $do(f(s), s)$, i.e., they are congruent. Hence, the simplifying assumption underlying a domain abstraction which makes s and $do(f(s), s)$ equivalent is satisfied (see Example 3). Hence, we can replace all occurrences of $do(f(s), s)$ with s resulting in the axiom $q_1(s) \wedge q_2(s) \Rightarrow r(do(A, s))$. This is exactly the abstract operator used by ABSTRIPS. □

5 Discussion

Giunchiglia and Walsh [1992] characterize abstractions as syntactic mappings between theories. While it is true that ultimately abstractions are syntactic mappings between theories, they are an overgeneral characterization of abstractions. Hence, this view of abstractions provides little constraint on what abstractions actually are. On the other hand, our theory provides a more meaningful characterization of abstractions. It captures the fact that an abstraction is a model level mapping of the original theory in conjunction with a simplifying assumption. This difference is made clearer by the following comparison of TD abstractions to MI abstractions.

Since for every theorem of an MI abstraction, $f_\pi(\sigma)$ is in the base theory, all MI abstractions are TD abstractions. However, MI abstractions are a strict subset of TD abstractions. To see this, consider a base language with propositions p and q, and an abstract language with proposition r. Consider a syntactic mapping that maps p to r and q to $\neg r$, the base theories $B_1 = \{p\}$ and $B_2 = \{q\}$, and the abstract theories $A_1 = \{r\}$ and $A_2 = \{\neg r\}$. Clearly, A_1 and A_2 are TD abstractions of B_1 and B_2, respectively. However, $A_1 \cup A_2$ is inconsistent, while $B_1 \cup B_2$ is not. Hence, it follows that this TD abstraction is not compositional, and hence not an MI abstraction.

It is worth noting that the strongest MI abstraction of T base is not, in general, a TC abstraction. It is TC abstraction only if there is also an abstraction mapping p in the opposite direction, e.g., the mappings between polar coordinates and rectilinear coordinates. In this latter case, the base and abstract theories are equivalent, and there is really no model level abstraction going on. However, a switch from one theory to another may still be motivated by computational considerations.

Levy [1994] has outlined another method for constructing MI abstractions. His method is based on identifying sentences in T base that are independent of the abstraction, and hence can be syntactically abstracted. Intuitively, sentences of the form $f_\pi(\sigma)$, or sentences that entail sentences of the form $f_\pi(\sigma)$,, are independent, and such sentences can be abstracted to a. Identifying independent sentences can often be done easily, resulting in an efficient algorithm for constructing abstract theories.

Our theory of abstractions raises several directions for future work. First, it raises the question of finding restricted, but useful, settings within which the strongest MI abstraction is finitely axiomatizable and can be constructed efficiently. When the strongest MI abstraction cannot be constructed efficiently, an important issue is finding methods for constructing weaker, though still useful, MI abstractions. Second, our theory of abstractions is only a logical account of the process, and does not address the issue of the computational benefits of using abstractions. A better understanding of when model level abstractions lead to computational savings is needed. Third, we are developing probabilistic methods for reasoning about how likely it is that a simplifying assumption holds.

References

[Chang, 1979] Chang, C. L. 1979. Resolution plans in theorem proving. In Pros, of IJCAI-79. 143-148.

[Davis, 1984] Davis, R. 1984. Diagnostic reasoning based on structure and behavior. Artif. In tell. 24:347-410.

[Eiter and Gottlob, 1992] Eiter, T, and Gottlob, G. 1992. On the complexity of propositional knowledge base revision, updates and counterfactuala. Artif. Intell. 57:227-270.

[Ellman, 1992] Ellman, T., ed. 1992. Procs, of the Workshop on Approximation and Abstraction of Computational Theories.

[Ellman, 1993] Ellman, T, 1993. Abstraction via approximate symmetry. In Procs of IJCAI-93. 916-921.

[Enderton, 1972] Enderton, H. B. 1972. A Mathematical Introduction to Logic. Academic Press, Inc.

[Falkenhainer and Forbus, 1991] Falkenhainer, B. and Forbus, K. D. 1991. Compositional modeling: Finding the right model for the job. Artif. Intell. 51:95-143.

[Genesereth, 1984] Genesereth, M. R. 1984. The use of design descriptions in automated diagnosis. Artif. Intell. 24:411-436.

[Giunchiglia and Walsh, 1992] Giunchiglia, F. and Walsh, T. 1992. A theory of abstraction. Artif. Intell. 57(2-3):323-389.

[Green, 1969] Green, C. 1969. Application of theorem proving to problem solving. In Proa, of IJCAI-69. 219-239.

[Hamscher, 199l] Hamscher, W. C. 1991. Modeling digital circuits for troubleshooting. Artif. Intell. 51:223-271.

[Imielinski, 1987] Imielinski, T. 1987. Domain abstraction and limited reasoning. In Procs, of IJCAI-87. 997-1003.

[IwaBaki and Levy, 1994] Iwasaki, Y. and Levy, A. Y. 1994. Automated model selection for simulation. In Procs, of AAAI-94.

[Kuipers, 1986] Kuipers, B. 1986. Qualitative simulation. Artif. Intell. 29:289-338.

[Levy, 1994] Levy, A. Y. 1994. Creating abstractions using relevance reasoning. In Procs. of AAAI-94-

[Lowry, 1989] Lowry, M. R. 1989. Algorithm Synthesis Through Problem Reformulation. Ph.D. Dissertation, Stanford University.

[Lowry, 1992] Lowry, M. R., ed. 1992. Procs. of the Workshop on Change of Representation and Reformulation.

[Nayak and Joskowicz, 1996] Nayak, P. P. and Joskowicz, L. Efficient Compositional Modeling for Generating Causal Explanations. Artif. Intell. To appear.

[Nayak, 1994a] Nayak, P. P. 1994a. Causal approximations. Artif. Intell 70:277-334.

[Nayak, 1994b] Nayak, P. P. 1994b. Diagnosis with multiple theories. In Procs. of the Fifth International Workshop on Principles of Diagnosis.

[Plaisted, 198l] Plaisted, D. 1981. Theorem proving with abstraction. Artif. Intell. 16:47-108.

[Sacerdoti, 1974] Sacerdoti, E. 1974. Planning in a hierarchy of abstraction spaces. Artif. Intell. 5:115-135.

[Struss, 1992] Struss, P. 1992. What's in SD? Towards a theory of modeling for diagnosis. In Hamscher, W.; Console, L.; and de Kleer, J., eds. 1992, Readings in Model-Based Diagnosis. Morgan Kaufmann. 419-449.

[Tenenberg, 1990] Tenenberg, J. D. 1990. Abstracting first order theories. In Benjamin, Paul, ed. 1990, Change of Representation and Inductive Bias. Kluwer, Boston, Mass.

[van Baalen, 1994] van Baalen, J., ed. 1994. Procs. of the Workshop on Theory Reformulation and Abstraction.

[Weld, 1992] Weld, D. S. 1992. Reasoning about model accuracy. Artif. Intell. 56(2-3):255-300.

[Williams, 1991] Williams, B. C. 1991. A theory of interactions: unifying qualitative and quantitative algebraic reasoning. Artif. Intell. 51:39-94.