Appears in: Working Papers of the 14th International Workshop on Qualitative Reasoning, Morelia, Mexico (2000), pp. 144-153.

Automated Determination of Qualitative Distinctions: Theoretical Foundations and Practical Results

Martin Sachenbacher[1] and Peter Struss[1,2]

[1]Technische Universität München
Department of Computer Science
Orleansstr. 34, 81667 Munich, Germany
sachenba@in.tum.de

[2]OCC'M Software GmbH
Gleissentalstr. 22
82041 Deisenhofen, Germany
struss@occm.de

Abstract

Automating the generation of qualitative models at a level that is tailored to support a particular task is crucial to the deployment of model-based systems technologies in practical applications, because reusability of models is of vital importance. If this task cannot be solved, models in a library will either be ineffective, because they are too coarse for solving a particular problem, or inefficient, because they are too fine-grained. The key question to be answered is, "*what are the distinctions in the domains of the system variables that are both necessary and sufficient to achieve a particular goal in a certain context and under given conditions?*". In our approach, the goal is defined by a set of target partitions of the domains of selected variables (e.g. output variables), the context is given by the structure of the modeled system, and the conditions are represented by a set of initial variables and their possible distinctions (e.g. possible observations). The task includes problems such as determining the appropriate qualitative values of variables in order to enable prediction at the desired level or discrimination for diagnosis, deciding whether or not changes can be modeled as discontinuous ones, and determining when a deviation of a parameter can be considered significant. We have analyzed and formalized the task for relational behavior models, implemented an (incomplete) algorithm, and carried out first experiments. The paper first elaborates on previous theoretical foundations, defining the goal and the specification of algorithmic solutions. We then outline the implemented algorithm and present and discuss some experimental results of applying this prototype. We conclude with some open problems and discuss alternative approaches.

Introduction

Automated modeling, model abstraction and model composition has been studied in the area of qualitative modeling for quite some time (e.g. [Addanki et al. 91], [Iwasaki 92], [Falkenhainer Forbus 91], or [Struss 92]). Very often, the motivation for this work was mainly academic, and there are hardly any tools that are designed and implemented to serve serious applications. With model-based systems technology intruding industrial applications, the creation of appropriate models gains practical importance, and strong support to this task or automating it may well be decisive to the success of model-based systems.

For instance, in our work on applying model-based diagnosis, prediction, and fault analysis to industrial problems (prominently car subsystems, see [Sachenbacher et al. 00]), we are thrown back to very fundamental tasks and theoretical problems the closer we get to the stage of actual use of the technology in industry.

Where do the models come from? is the most frequently asked question, because most of our solutions are based on qualitative models which look quite different from the existing (differential) equation models used in engineering, e.g. for numerical simulation. *Can't we use these models or, at least, start from them?* Our answer is, in principle, yes, because qualitative models are abstractions of the equations. But, "in principle" is not enough for convincing management - it has to be for real, well-supported, and preferably automated.

What is the appropriate level of a qualitative model? is a second question, which we often ask ourselves, and experience says "it depends". It depends on what you need the model for, and what problem you want to solve. This is anything but surprising, but it is very critical: Much of the feasibility and economic attractiveness of model-based solutions stems from the extensive re-use of model fragments taken from a domain-specific library across different systems and tasks. The contradiction between the genericity of models (to be re-usable) and their task-dependent specificity (to be effective and efficient) can be fatal for the industrial success of model-based systems. If we use too coarse-grained models (e.g. with signs only), a diagnostic system may be unable to detect certain symptoms. On the other hand, if the model is too fine-grained, it may be inefficient w.r.t. time and space requirements. For instance, the real-time performance of our on-board diagnosis system described in [Sachenbacher et al. 00] crucially depends on the qualitative abstraction of the models and the observed signals. Consequently, unless we find ways to automatically transform a generic model from a library into a model tailored to a particular task, the applicability of model-based techniques will be limited.

A Tiny Example

In order to illustrate the problem and the key ideas of our work, we introduce a simple example. The system comprises a reservoir (which is assumed to be never empty and not shown in the figure), filled with liquid with pressure $T_1.p$. It is connected via a valve with maximal diameter A_{max} to an outlet pipe that fills a container with bottom area B and vertical walls (see Figure 1). The task is to use a model in order to design the control scheme that opens and closes the valve in order to fill the container up to a given height h with a precision of $\Delta h > 0$.

The example does not appear to be an industrial application at first glance. However, consider it to be a simplification of a controlled injector (i.e. a valve) that is to supply a certain amount of diesel fuel to the combustion chamber (i.e. a container) of a car engine. Below h, the fuel mixture will be too lean, above $\Delta h + h$, there will be too much fuel in the cylinder to burn it completely, a situation which should be avoided in any case.

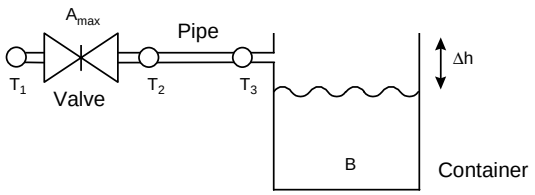

Figure 1: Filling a container

All the components in the system are fairly standard, and we expect to find their behavior model fragments in a library in order to compose a model of the example system. For instance, the valve model will have to associate the obvious equation

$$T_1.q = A \cdot sgn(T_1.p - T_2.p) \, (| T_1.p - T_2.p |)^{0.5}$$

with the states open and closed of the valve. But what about the transitions between the states? Do we have to explicitly model the opening and closing state, during which some amount of liquid is pouring into the container? Or can we model the transition from $A = A_{max}$ to $A = 0$ as a discontinuous change, i.e. neglect the duration of the valve closing operation? The answer, of course, depends on the targeted precision Δh and on the characteristics of the entire configuration, namely $T_1.p$, $T_2.p$, A_{max}, and B. Perhaps, the container is so large that the increase of the height during closing of the valve is negligible. Perhaps, the time required for closing the valve is quite significant, because the required precision Δh is tight and/or because the pressure difference $T_1.p - T_2.p$ is high. Given the contextual conditions of the task, we are able to compute a boundary for durations of the opening that are insignificant w.r.t. the required precision Δh. Based on the result, we can decide whether or not it is appropriate to approximate the valve model by one with discontinuous transitions. Note that the result will be influenced by another factor: the precision of the inputs to the calculation. For instance, if we just know that the diameter is between zero and A_{max} during opening of the valve, we obtain a boundary on the duration which is smaller than the one calculated for an opening that is described as linear within a certain tolerance. In the same way, the precision of the pressure values influences the distinctions on the duration. Finally, note that the decision about the appropriate distinctions in the valve model affects other component models as well. If the opening has to be modeled in a detailed way, then this requires the pipe model and the container model to include certain distinctions of the flow value and of the volume and the level, respectively.

Let us summarize the key insights from this example that form the starting point for our work:

- The distinctions to be made within a (component) model (w.r.t. system variables, time, and states) depend on the task, the context, and conditions on the inputs.
- The **task** was characterized by some **target distinctions** to be made (in our case, Δh).
- The **context** is given by the structure of the system (e.g. a second valve and inlet would influence the model of the first one).
- The **conditions** appear as certain **initial distinctions** that are possible (reflecting, for instance, how precisely certain variables can be measured).
- Resulting from this are certain **induced distinctions** on other variables (in our example, whether the finite duration of the opening has to be distinguished from a zero duration).

The thrust of the work reported here is to turn these intuitions into a formal theory and algorithms in order to automate the task of goal-dependent qualitative abstraction. Although our results are still preliminary, they extend, refine, and partially correct the theoretical foundations presented in [Struss Sachenbacher 99], including now a prototype system and results carried out based on its implementation.

The following section summarizes the formal foundations and extends the notion in [Struss Sachenbacher 99] by including the role of initial distinctions and by analyzing the impact of different purposes of the model (consistency check vs. prediction), the uniqueness of solutions, and completeness of proposed algorithms. Then we describe our current implementation (of an approximation algorithm) and present experimental results on the introductory example and the pedal position sensor example that was outlined in [Struss Sachenbacher 99]. Finally, we discuss possible modifications and extensions of the current algorithms as well as alternative approaches.

Foundations

Basic Definitions and Concepts

In this section, we present the theoretical foundations of our work, trying to turn the intuitive ideas extracted from the

example into formal concepts. The theory we presented in [Struss Sachenbacher 99] will be summarized as briefly as possible. We will indicate when we omit the repetition of a formal definition from that paper, and we try to highlight the extensions and modifications.

Behavior Models. Our theory is based on **relational** models, i.e. the behavior of a component or system, S, is given by a relation

$$R_S \subset DOM(\underline{v}_S) ,$$

where $\underline{v}_S = v_1 \times v_2 \times ... \times v_n$ is the vector of all parameters and variables in the system. This allows us to treat symbolic values, discrete states, behavior modes, and non-deterministic models rather than only real-valued functions which are subsumed as a special case. The relation R_S can be given implicitly as the composition of several component models, as in our example.

Domains. We assume that there exists a base domain $DOM_0(v_i)$ for each variable v_i, from which the desired task-specific domains can be obtained by means of abstraction, and a base model

$$R_{S,0} \subset DOM_0(\underline{v}_S) = DOM_0(v_1) \times ... \times DOM_0(v_n).$$

Distinction. This central concept of the problem is defined as a partition of the base domain, i.e. a set of non-empty disjoint subsets

$$\Pi_i = \{P_{i,k}\} \subset P(DOM_0(v_i))$$

that together cover the entire domain (P(X) is the power set of X). Each $P_{i,k}$ is a qualitative value which represents a set of values that are not distinguished from each other.

Target distinctions. As pointed out in the discussion of the introductory example, the target distinctions characterize the goal of a particular task to be performed based on a model and are represented as a set of target partitions[1]

$$\Pi_{targ,i} = \{P_{targ,i,k}\} \subset P(DOM_0(v_i))$$

If we are not interested in any distinctions of a variable vi, then we can express this by the trivial partition

$$\Pi_{targ,i} = \{DOM_0(v_i)\}.$$

Initial Distinctions. Analogously, the conditions of the model-based task are expressed as a set of initial partitions $\Pi_{obs,i}$. They represent what will be the granularity of inputs to the model, e.g. observations or partially described hypothetical situations. All the other variables receive the trivial partition as their initial one.

Qualitative Domain Abstraction. Finding qualitative values for all variables which can then replace the original, fine-grained domain is the goal of our enterprise. Given a set of qualitative values (i.e. a partition), each value of the fine grained domain can be mapped to its respective qualitative value. We call such a mapping

$$\tau: DOM_0(v_i) \rightarrow DOM_\alpha(v_i) \subset P(DOM_0(v_i))$$

a domain abstraction. Dependent on the kind of partition that induces a domain abstraction, we will give τ an index "targ" for target distinctions or "obs" for initial distinctions. There is a dualism between the domain partitions and qualitative abstractions, because each domain abstraction that produces mutually exclusive values induces a partition, which is given by

$$\Pi_{\tau,i} = \tau(DOM_0(v_i)) .$$

We can apply the concepts of **refinement** of a partition (i.e. a partition that makes further distinctions compared to the elements of another partition) and **merge** of partitions (i.e. the maximal partition which is a refinement of both) to domain abstractions, as well. For a rigorous treatment and definitions of these operations, see [Struss Sachenbacher 99].

In this formalism, we can define our goal precisely. Whatever model we use, the only thing that matters is the information it can infer about the target partitions from the possible initial partitions. We can express this interest by abstracting each result of inference to the level of target partitions by the respective domain abstraction. Let

$$\tau_{targ} = (\tau_{targ,1}, \tau_{targ,2}, ... \tau_{targ,n}):$$
$$DOM_0(\underline{v}_S) \rightarrow \Pi_{targ} := \Pi_{targ,1} \times \Pi_{targ,2} \times ... \times \Pi_{targ,n} .$$

If we supply the base model, $R_{S,0}$, with some information, i.e. some restrictions on system variables expressed in terms of initial partitions,

$$R_{obs} \subset \Pi_{obs} := \Pi_{obs,1} \times \Pi_{obs,2} \times ... \times \Pi_{obs,n},$$

then we are interested in

$$\tau_{targ}(R_{obs} \cap R_{S,0}) ,^2$$

and our goal is to find qualitative domain abstractions that do not change this result. Hence, we obtain the following definition of our goal concept:

Definition (Distinguishing Qualitative Domain Abstraction)

Let $R_{S,0} \subset DOM_0(\underline{v}_S)$ be the base model of a system S, and Π_{targ} and Π_{obs} the target partitions and initial partitions, respectively. A qualitative domain abstraction

$$\tau: DOM_0(\underline{v}_S) \rightarrow P(DOM_0(\underline{v}_S))$$

is distinguishing w.r.t. $\{\Pi_{targ} , \Pi_{obs}\}$ iff it is a refinement of τ_{targ} and

$$\forall R_{obs} \subset \Pi_{obs} :$$
$$\tau_{targ}(\tau(R_{obs}) \cap \tau(R_{S,0})) = \tau_{targ}(R_{obs} \cap R_{S,0}). \quad (1)$$

[1] We changed terminology compared to [Struss Sachenbacher 99], because we felt that the term "primary partitions" used in this paper is misleading as it suggests they would be the starting point of the task.

[2] To simplify the notation, here and in the following we implicitly map the initial partitions to DOM_0 whenever necessary, i.e. we identify $R_{obs} \subset \Pi_{obs}$ with $R'_{obs} = \tau^{-1}_{obs}(R_{obs})$. Furthermore, we implicitly extend each mapping, τ, defined on some set, S, to the power set P(S) by defining $\forall X \subset S \ \tau(X) := \cup \{\tau(x) \mid x \in X\}$.

A distinguishing abstraction, τ, is **maximal** iff there exists no other distinguishing abstraction, τ', that is a refinement of τ.

Condition (1) requires that the abstraction preserves information about the **tuples** of target partitions. However, a weaker and often sufficient goal for some applications is to postulate that **for each single target variable**, the possible distinctions are maintained under the abstraction. If proj(v_1, v_2, ..., v_k) denotes projection on a set of variables v_1, v_2, ..., v_k, then this can be expressed as

$$\forall R_{obs} \subset \Pi_{obs}, \forall v_{targ}:$$
$$\text{proj}(v_{targ})(\tau_{targ}(\tau(R_{obs}) \cap \tau(R_{S,0}))) =$$
$$\text{proj}(v_{targ})(\tau_{targ}(R_{obs} \cap R_{S,0})). \quad (1')$$

The Scope of R_{obs}

In contrast to the corresponding definition in [Struss Sachenbacher 99], we will capture the set of initial restrictions, i.e. the starting point of the model-based computation, and their impact on the resulting qualitative abstractions in a more rigorous and appropriate way. In that paper, we allowed arbitrary initial restrictions $R_{obs} \subset DOM_0(\underline{v}_S)$, called "$R_{ext}$". But in practice, the initial restrictions are limited in basically two ways:

- only a subset of the variables form a potential input to the model (e.g. because only certain variables are observable),
- the input values usually have limited precision which can be coarser than DOM_0. This may be due to the precision of a sensor or a partial specification of some input to the model (e.g. when analyzing a circuit's behavior "with a voltage source of more than 6V").

Uniqueness Properties of Maximal Distinguishing Abstractions

It turns out that without further restrictions on R_{obs}, maximal distinguishing abstractions are not uniquely defined. Consider a very simple example that consists of two equality constraints

$$x = y = z .$$

Let us assume that for all variables, the base domain is the set of integers, that x and z are observable at this granularity, and that the only target distinction is to determine the sign of y:

$$\Pi_{targ,y} = \{\{(-\infty, 0)\}, \{0\}, \{(0, \infty)\}\} .$$

For unique initial distinctions (i.e. there is always a value given for both x and z), there are obviously two maximal distinguishing abstractions, given by the partitions

$$\Pi_{\tau,x} = \{\{(-\infty, 0)\}, \{0\}, \{(0, \infty)\}\} , \Pi\tau,y = \{\{(-\infty, \infty)\}\}$$

and

$$\Pi'_{\tau,x} = \{\{(-\infty, \infty)\}\} , \Pi'\tau,y = \{\{(-\infty, 0)\}, \{0\}, \{(0, \infty)\}\},$$

which simply reflects the fact that one of x and z is not needed for determining the target distinction. In contrast, if the initial restrictions of x and y are made independently of each other, this includes situations in which one of the variables is not restricted, leading to a unique distinguishing abstraction

$$\Pi_{\tau,x} = \{\{(-\infty, 0)\}, \{0\}, \{(0, \infty)\}\},$$
$$\Pi'\tau,y = \{\{(-\infty, 0)\}, \{0\}, \{(0, \infty)\}\} .$$

To reflect this difference w.r.t. restrictions on R_{obs}, we further refine our definition. A distinguishing abstraction τ is called **maximal for independent initial restrictions**, if R_{obs} in (1), (1') is of the form

$$R_{obs} = R_{obs,1} \times R_{obs,2} \times ... \times R_{obs,n} \subset \Pi_{obs}.$$

The case of independent initial restrictions captures a very common case of model-based reasoning, namely to perform prediction (or consistency checking) after restricting a number of variables to a certain value (set) independently of each other. This is the case, for instance, if the input is the result of measuring certain system variables. The only important exception we can think of is the case when time is included as a variable: if the input is a time-stamped set of variable values, one has to use the general definition.

A still further specialization is obtained, when there are unique initial distinctions, i.e. each R_{obs} is a tuple of single qualitative values:

$$R_{obs} \in \Pi_{obs} .$$

An example is on-board diagnosis, where at each instance in time, unique values (at the level of Π_{obs}) are present for a fixed set of variables. In contrast, the general case of independent initial distinctions captures the situation where there is a set of variables that are observable, but we are looking for an abstraction that allows to draw the strongest conclusions also if only a subset of the variables have actually been measured (the non-measured ones receive no restriction).

Different Tasks: Consistency Check vs. Prediction

The definition of distinguishing abstractions nicely expresses that we are looking for qualitative abstractions that do not reduce the information about the target variables. In [Struss Sachenbacher 99], we have shown that sometimes, every abstraction entails a loss, and one is stuck with the base domain. We discuss one fundamental reason for this.

Condition (1) includes the case where R_{obs} is **inconsistent** with the model, $R_{S,0}$. This could well be a situation in practice; e.g. in model-based diagnosis or in design verification, we are interested in determining whether or not a set of observations is in conflict with a behavior model. In this case,

$$R_{obs} \cap R_{S,0} = \varnothing$$

holds, and condition (1) implies

$$\tau(R_{obs}) \cap \tau(R_{S,0}) = \varnothing \Leftrightarrow R_{obs} \cap R_{S,0} = \varnothing.$$

Note that this imposes a condition on τ which is **independent of the target partitions** and reflects only $R_{S,0}$ and the initial distinctions, represented by R_{obs}. If the latter have the granularity of the base domain, the very "shape" of $R_{S,0}$ may exclude an abstraction, because the above implies

$$\underline{v}_0 \in R_{S,0} \Leftrightarrow \tau(\underline{v}_0) \subset \tau(R_{S,0}) ,$$

and for any $\underline{v}_1 \in R_{S,0}$ with

$$\tau(\underline{v'}_0) = \tau(\underline{v}_0) ,$$

we obtain

$$\begin{aligned} \underline{v}_0 \in R_{S,0} &\Leftrightarrow \tau(\underline{v}_0) \subset \tau(R_{S,0}) \\ &\Leftrightarrow \tau(\underline{v'}_0) \subset \tau(R_{S,0}) \\ &\Leftrightarrow \underline{v'}_0 \in R_{S,0} , \end{aligned}$$

This means all values that are mapped onto $\tau(\underline{v}_0)$ are in $R_{S,0}$, and since $\tau(\underline{v}_0)$ contains exactly these values, we have

$$\underline{v}_0 \in R_{S,0} \Leftrightarrow \tau(\underline{v}_0) \subset R_{S,0}.$$

Hence, a distinguishing abstraction can never "cross the boundaries" of $R_{S,0}$. Since

$$\tau(\underline{v}_0) = \tau(v_1) \times ... \times \tau(v_n) \in R_{S,0} ,$$

τ is limited by the existence of "rectangular building blocks" within $R_{S,0}$. Figure 2 provides an abstract illustration: $R_{S,0}$ allows an abstraction in case a), but not in case b).

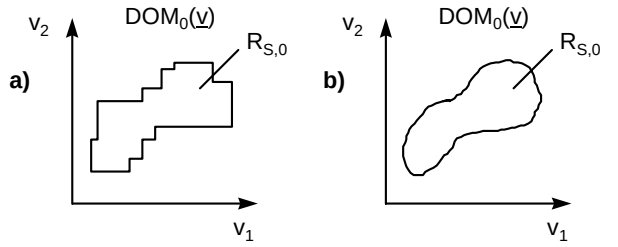

Figure 2: Different $R_{S,0}$ with different potential for abstraction

In other applications, the model is not intended to be used for consistency check with $R_{S,0}$, but only to compute consequences of the initial restrictions which are assumed to be consistent with the model. Under this assumption, condition (1) can be weakened as follows

$$\begin{aligned} &\forall\, R_{obs} \subset \Pi_{obs} : \\ &R_{obs} \cap R_{S,0} \neq \varnothing \Rightarrow \tau_{targ}(\tau(R_{obs}) \cap \tau(R_{S,0})) = \\ &\tau_{targ}(R_{obs} \cap R_{S,0}). \quad (2) \end{aligned}$$

This means that τ, based on initial values that are consistent with the model, manages to predict the target distinctions properly, but it is also allowed to map certain inconsistent tuples to consistent ones. However, also in this case, we will sometimes not be able to move away from the real numbers as a base domain, e.g. for a simple multiplication constraint on real numbers, whereas we obtained a nice abstraction when starting with a discretized domain (see [Struss Sachenbacher 99]). This highlights the importance of a definition and algorithms that reflect initial partitions (e.g. granularity of observations), because they introduce a discretization. Also discrete states and operating modes may induce partitions on continuous variables, as will be illustrated later by the examples.

Characterizing Distinguishing Domain Abstractions

While (2) defines the desired property of the abstractions, i.e. our goal, we need some constructive characterization as a starting point for designing algorithms to compute them.
Exploiting the duality between domain abstractions and domain partitions pointed out in the previous section, we will characterize τ by constructing the appropriate partition of the observable domains.
To simplify the presentation, we restrict ourselves in the following to the special situation where the observed (or otherwise given) initial restrictions have a granularity equal to the base domain $DOM_{0,}$ i.e. $\Pi_{obs,i}$ is the identical mapping. We will return to the more general case later.
The question can be stated as "*Which values can be joined in a partition (i.e. abstracted to the same qualitative value under τ) without violating (2)*?". The intuitive answer, based on (2) is: If two values $v_{j,1}, v_{j,2}$, of a variable v_j, when combined with at least one consistent initial restriction for other variables, are consistent with different sets of target values, then we cannot join them without losing information on the level of the target distinctions. Otherwise, we can, because it is then guaranteed that there exists no initial restriction that leads to different predictions on the level of target distinctions if we drop the distinction between $v_{j,1}$ and $v_{j,2}$. Formally, we can introduce an equivalence relation for $v_{j,1}, v_{j,2} \in proj_j(R_{S,0})$ by:

$$\begin{aligned} &v_{j,1} \approx_j v_{j,2} :\Leftrightarrow \\ &\forall\, R_{obs,-j} = R_{obs,1} \times ... \times R_{obs,j-1} \times R_{obs,j+1} \times ... \times R_{obs,n} \subset R_{S,0}: \\ &\tau_{targ}(\text{select}\ (v_j = v_{j,1})\ (R_{obs,-j} \cap R_{S,0})) = \\ &\tau_{targ}(\text{select}\ (v_j = v_{j,2})\ (R_{obs,-j} \cap R_{S,0})) \quad (3) \end{aligned}$$

Here, $\text{select}(v_j = v_{j,1})(R)$ is the restriction of a relation R to the set of tuples that satisfy the condition $v_j = v_{j,1}$.

Definition (Induced Partitions)
Let $\approx_j$ be the equivalence relation on $DOM_0(v_j)$ defined by (3). The sets of partitions $\Pi_{ind,j}$ for $DOM_0(v_j)$ given by the equivalence classes of the relations $\approx_j$,

$$\Pi_{ind,j} := DOM_0(v_j) \,|\, \approx j ,$$

are called the partitions induced by the target distinctions.

This defines a distinguishing domain abstraction τ_{ind}:

Lemma (Characterization of distinguishing domain abstractions)
The induced partition

$$\Pi_{ind} := \times \Pi_{ind,j}$$

defines a distinguishing qualitative abstraction τ_{ind} w.r.t. Π_{targ} .

Conjecture (Maximality of the induced partition)
The induced partition is a maximal distinguishing qualitative abstraction.

Analogously, we obtain a characterization of distinguishing qualitative abstractions under the weaker condition (1') which states that determining the distinctions for each single target variable suffices:

$v_{j,1} \approx_j v_{j,2} :\Leftrightarrow$
$\forall R_{obs,-j} = R_{obs,1} \times ... \times R_{obs,j-1} \times R_{obs,j-+1}. \times ... \times R_{obs,n} \subset R_{S,0}$,
$\forall v_i$:
$proj(v_i)\ \tau_{targ}(select\ (v_j = v_{j,1})\ (R_{obs,-j} \cap R_{S,0})) =$
$proj(v_i)\ \tau_{targ}(select\ (v_j = v_{j,2})\ (R_{obs,-j} \cap R_{S,0}))$ (3')

Approximations to Induced Partitions

Conditions (3) and (3') formulate that a distinction between values of v_j is made if and only if this leads to a target distinction, potentially in conjunction with other observations.
From a computational point of view, this is problematic, since it requires to consider mutual combinations of values, plus all the possible (combinations of) observations, to determine if they have to be distinguished.
In this section, we develop a characterization that avoids these difficulties by starting from the target distinctions instead. As we will see, this characterization is more amenable to computation, but at the price that the resulting partitioning represents only a subset of the necessary distinctions.

Definition (Approximation of Induced Partitions)
For $v_{j,1}, v_{j,2} \in proj(v_j)(R_{S,0})$, let

$v_{j,1} \approx_4 v_{j,2} :\Leftrightarrow$
$\forall R_{targ} := R_{targ,1} \times ... \times R_{targ,n} \subset \tau_{targ}(R_{S,0})$:
$v_{j,1} \in proj(v_j)(R_{targ,} \cap R_{S,0}) \Leftrightarrow v_{j,2} \in proj(v_j)(R_{targ,} \cap R_{S,0})$ (4)

The partition defined by the equivalence relation (4) is called approximation of the induced partition, and τ^*_{ind} denotes the corresponding domain abstraction.

Again, we can formulate a corresponding approximation for the weaker conditions (1'), (3').
The approximation τ^*_{ind} is not necessarily a distinguishing abstraction. Since (4) is only a necessary condition of (3), it can fail to preserve all possible target distinctions. However, the distinctions of the approximation are necessary ones, and, hence, all distinguishing abstractions are refinements of the approximation. Formally, this is captured by the following proposition.

Proposition: τ_{ind} is a refinement of τ^*_{ind} .

Proof:
Assume that $v_{j,1}$, $v_{j,2}$ are distinguished in τ^*_{ind} , i.e. $v_{j,1} \approx_4 v_{j,2}$ does not hold. Then, there exists at least one initial restriction $R_{targ,}$ such that either $v_{j,1} \in proj(v_j)(R_{targ,})$ and $v_{j,2} \notin proj(v_j)(R_{targ,})$, or $v_{j,1} \notin proj(v_j)(R_{targ,})$ and $v_{j,2} \in proj(v_j)(R_{targ,})$. Since $R_{targ,} \subset \tau_{targ}(R_{S,0})$, $v_{j,1}$ combined with R_{targ} yields a restriction on the level of target distinctions that is different from the restriction resulting from combining $v_{j,2}$ with R_{targ}, i.e. $\tau_{targ}(select\ (v_j = v_{j,1})\ (R_{targ} \cap R_{S,0})) \neq \tau_{targ}(select\ (v_j = v_{j,2})\ (R_{targ} \cap R_{S,0}))$. From condition (3), it follows that $v_{j,1}$, $v_{j,2}$ must also be distinguished in τ_{ind}. □

So far, we have assumed that all distinctions in DOM_0 can be observed. Now, we consider the case that Π_{obs} is not given by the identical mapping.

Observable Distinctions

Obviously, what distinctions can be provided by the initial (observable) distinctions crucially affects the distinctions that can be derived for other variables. It does not make sense to introduce distinctions for some intermediate variables that would help to determine the target distinctions, but that never become effective because the initial distinctions are not precise enough to obtain them. Intuitively, two values of a variable v_j cannot be distinguished by the observable distinctions, if for any observable tuples, they are either both consistent or both inconsistent with this tuple:

Definition (Induced observability)
For $v_{j,1}, v_{j,2} \in proj(v_j)(R_{S,0})$, let

$v_{j,1} \approx_5 v_{j,2} :\Leftrightarrow$
$\forall R_{obs} := R_{obs,1} \times ... \times R_{obs,n} \subset \tau_{obs}(R_{S,0})$:
$v_{j,1} \in proj(v_j)(R_{obs,} \cap R_{S,0}) \Leftrightarrow v_{j,2} \in proj(v_j)(R_{obs,} \cap R_{S,0})$ (5)

Let $\tau_{ind,obs}$ denote the domain abstraction corresponding to the partitions induced by (5).

The abstraction $\tau_{ind,obs}$ captures all distinctions that can in principle be revealed by initial restrictions, for example, be observed. The general case of arbitrary initial partitions $\Pi_{obs,i}$ is thus captured by replacing $R_{S,0}$ with $\tau_{ind,obs}(R_{S,0})$ in definitions (3) to (4).

Note that (5) has the same form as (4), with $R_{targ,}$ replaced by R_{obs}. The partition on v_j that results from τ^*_{ind} and $\tau_{ind,obs}$ can be reformulated as a merge operation

$merge\ (proj(v_j)\ (select(\underline{v_S} = P)R_{S,0})$

running over all tuples

$P := P_{targ,1} \times ... \times P_{targ,n} \in \tau_{targ}(R_{S,0})$

in the case of τ^*_{ind} , and

$P := P_{obs,1} \times ... \times P_{obsn} \in \tau_{obs}(R_{S,0})$

in the case of $\tau_{ind,obs}$. This means that both τ^*_{ind} and $\tau_{ind,obs}$ can be computed using the same basic algorithm, as just the different sets of tuples used in the select statement account for the difference.
Now, computing distinguishing domain abstractions under a given granularity of initial distinctions has reduced to computing and intersecting projections. Yet, they are still

described as global computations on the system relation $R_{S,0}$. This will be the topic of the following section.

Computing Induced Abstractions for Composed Relations

The problem of determining significant distinctions arises, in particular, due to composing generic behavior models taken from a library of model fragments. Therefore, we extend the concepts of computing induced distinctions to relations that are composed of several individual behavior model fragments. In the following, the relation $R_{S,0}$ is considered to be a composed relation of the form

$$R_{S,0} = R_{c1,0} \cap R_{c2,0} \cap ... \cap R_{cn,0} ,$$

where the $R_{ci,0}$ represent the relational model for component C_i. If necessary, the $R_{ci,0}$ include also mode variables representing different behavior modes or fault modes of the components.

The interaction between a variable v_j and targeted or initial distinctions in other variables v_i is then not direct, but instead mediated by a sequence of variables in a set of component relations. This leads to the idea of determining the induced qualitative values from the primary distinctions by propagation through the structure of the model. In [Struss Sachenbacher 99], we described the foundations for an approach to propagate distinctions to other parts of the system and then to compute the distinctions for one component relation locally.

For this, two problems have to be overcome that were not covered in detail in [Struss Sachenbacher 99].

First, it is possible that tuples occuring in the relations $R_{ci,0}$ are inconsistent with the aggregate system model $R_{S,0}$. The problem is that the presence of inconsistent tuples can both lead to results which are too coarse and to results which are too fine-grained. Intuitively, inconsistent parts of the relation could make it appear either more "rectangular" or less "rectangular" (in the sense of Figure 2), which either enables or blocks possibilities for domain abstractions. Thus, in order to ensure correctness of the results, a complete elimination of inconsistent tuples is required. This shows that determining significant distinctions is an inherently hard problem, because as a consequence of the above, the problem of determining whether a given distinction is significant or not w.r.t. the target distinctions is as least as hard as checking satisfiability of the involved behavior constraints, which is NP-hard in general.

The second issue is that the components C_i have to exchange information about their distinctions, since a distinction in the domain of one components' variable could be necessary to make a desired distinction in another component. In order to avoid cycles in the local computation (which would mean determining distinctions in one variable which aim at target distinctions for the same variable), it is necessary to provide additional information about the origin of each propagated distinction. We accomplished this by creating a label for each partition element, which contains a reference the target partition element it was introduced for. Propagation rules then ensure that loops in the computation are eliminated and the introduced induced distinctions reflect indeed only the target distinctions (details are described in [Kutscha 00]).

Furthermore, it is quite clear that the local propagation algorithm is incomplete, with the consequence that it will in general not compute τ^*_{ind} and $\tau_{ind,obs}$, but approximations thereof. For the result, however, the proposition stated in the last chapter still holds: a distinction that has been derived by the propagation algorithm is guaranteed to be a necessary distinction, although not all such distinctions might be found.

The next section describes the implemented prototype in more detail.

A Prototype System for Automated Qualitative Model Abstraction

Following the theory outlined above, we implemented a prototpyic system that automatically computes approximations of maximal distinguishing abstractions for a given compositional system model.

The prototype (described in detail in [Kutscha 00]) uses software components from the commercial model-based systems framework Raz'r ([OCC'M 00]). The Raz'r development system allows to compose a system model from a library of model fragments. The prototype reads in the resulting system description, which consists of a definition of domains, constraint types and behavior models.

The task-dependent initial and target partitions Π_{obs}, Π_{targ} are given as XML documents. Alternatively, they can be defined interactively using a domain partition editor.

In a first step, tuples inconsistent with $R_{S,0}$ will be eliminated from each component model. Based on this, the prototype uses the propagation algorithm outlined in the previous section to iteratively compute the abstraction $\tau_{ind,obs}(R_{S,0})$. In a third step, this result is used as a starting point for the computation of qualitative distinctions based on the definition of τ^*_{ind}.

The local computation steps for the $R_{ci,0}$ are based on a constraint system that represents the relational behavior models (more precisely, their characteristic functions) as ordered binary decision diagrams (OBDDs, see [Bryant 92]). This data structure allows for fairly efficient realization of the required manipulations of the constraints, such as join or projection on a subset of variables. However, this also results in the limitation that the system can currently only transform behavior models that have finite domains.

The resulting qualitative domain abstraction τ^*_{ind} can be applied to the system model, leading to a transformed system description. Variables in the model that are found to have no induced distinction after computation (i.e. whose domain consists of one element only) can optionally be eliminated form the model. If all variables of a component

have been eliminated, this may eventually lead to the elimination of a component from the model.

The transformed system description can be fed back into the Raz'r framework and be used for task such as behavior prediction, diagnosis, or test generation. Additionally, it is possible to specify domain abstractions that link the elements of DOM_0 to finer domain, in particular to a subset of the real numbers. By concatenating such domain mappings, this allows to maintain a mapping to the real numbers for the induced qualitative values. This information is used by a signal transformation component that accomplishes the transformation of (real-valued) sensor readings to the abstraction level of the qualitative model.

The next section presents results of running the implementation on two small examples. Note that since τ^*_{ind} characterizes an upper approximation of the induced partitions only, the computed sets of distinctions will necessarily be incomplete.

Computational Results

Container Filling Example

We first present the application of our theory and implementation to an instance of the container filling problem that was presented in the introduction.

Initial distinctions: The domain for all pressure, flow and parameter variables were chosen to be a 19-valued domain that consists of open intervals and points between them:

Π_{obs} = {{(-∞, -4)}, {-4}, {(-4, -3)}, {-3}, ..., {(3,4)}, {4}, {(4, ∞)}}.

The model formulated in this domain consists of 12 variables and 2 mode variables, which means it has a tuple space (i.e. size of $DOM_0(\underline{v})$) of $19^{12} \cdot 2^2 \approx 8.9 \cdot 10^{15}$.

Target distinction: There are two behavior modes for the container component, one that captures the situation where the container level is within the specified range (h + Δh) and one that represents overflow. Thus, the target distinction for the example can be stated by giving a distinction for the mode variable of the tank component.

Induced distinctions: What we are after is an answer to the question whether or not we have to include the transition delay of the valve's closing operation, denoted Δt, in the model. The following result is obtained for parameters A_{max} , Δh, B, and T_1.p set as

Amax = {(0,1)},
Δh = {(2,3)},
B = {1},
T1.p ∈ {{(2,3)}, {3}, {(3,4)}}.

After computation of the composed relation $R_{S,0}$, 21036 tuples remain consistent for the valve component. The algorithm determines the following induced distinctions for the valve delay parameter:

$\Pi_{ind, valve.\Delta t, 1}$ = {(-∞,-4), -4, (-4,-3), -3, (-3,-2), -2, (-2,-1), -1, (-1,0), 0, (0,1), 1}
$\Pi_{ind, valve.\Delta t, 2}$ = {(1,2), 2, (2,3), 3, (3,4), 4, (4, ∞)}

The first partition element corresponds to situations where overflow is not possible. The second partiton element corresponds to situations where overflow of the container is possible (though will not necessarily occur). This means that all situations where the delay during closing the valve is part of the first partition element (i.e. is equal to or smaller than 1) are equivalent to no (zero) delay. In other words, the transition delay can be neglected in the model of the valve if it is part of the first qualitative value, and must be considered in the model if it is part of the second qualitative value.

To give an idea of the problem size, a multiplication constraint that uses the base domain consists of 713 tuples out of a tuple space of $19^3 = 6859$ (which amounts to a ratio of 10.4% consistent combinations of values; for a domain that consists just of signs, the respective constraint would have had 9 tuples out of a tuple space of $3^3 = 27$, i.e. a ratio of 33.3% consistent tuples). The constraint describing the valve equation mentioned in the introduction, for instance, has 23075 tuples. Compared to $|DOM_0(\underline{v}))| \approx 8.9 \cdot 10^{15}$ for the original behavior model, the abstracted behavior model has a tuple space $|\tau^*_{ind}(DOM_0(\underline{v})|$ of only $1.2 \cdot 10^4$.

We can also start with a given parameter for the delay and ask for a partition for the mode variable of the valve component, given the same target distinction for the container. Depending on the magnitude of the delay, it will turn out to be necessary or unnecessary to explicitly distinguish the two behavior modes of the valve ("closing" and "closed"). For the given parameters, we obtain one partition element,

$\Pi_{ind, valve.mode}$ = {closing, closed}

i.e. the distinction between the two behavior modes becomes irrelevant, if and only if the parameter for delay is restricted to the first qualitative value.

Pedal Position Sensor Example

As a second example, we list results for a problem that was originally presented in [Struss Sachenbacher 99]. The device in Figure 3 shows a pedal position sensor in a passenger car. Its purpose is to deliver information about the position of the accelerator pedal to the electronic control unit (ECU) of the engine management system. The position is sensed in two ways, via the potentiometer as an analogue signal, v_{pot}, and via the idle switch as a binary signal, v_{switch}. The idle switch changes its state at a particular value of the mechanically transferred pedal position. The reason for the redundant sensing of the pedal position is that the signals from the potentiometer and the switch are cross-checked for plausibility by the on-board control software of the ECU. The two possible values of v_{switch} correspond to two ranges of v_{pot} separated by a particular voltage value.

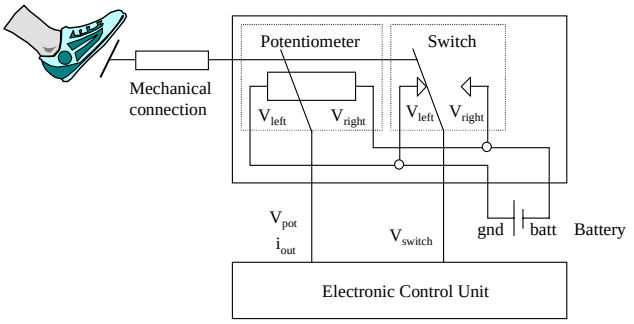

Figure 3: The Pedal Position Sensor

If we use qualitative models which distinguish only e.g. between voltage "gnd", "between" and "batt" as convenient for many applications, we are unable to perform tasks such as diagnosis or design verification, because these tasks refer to the redundancy which purposefully has been implemented in the system. The problem is that the particular distinction in the domain of v_{pot} cannot be anticipated in a generic model of the potentiometer, because the voltage landmark would not make any sense in a different structure.

We would like to have a composition of component models that make just the right distinctions required by the other components and the task the model is used for. To this end, we can use approach and the software component outlined above to compute an approximation of the necessary qualitative distinctions.

Initial distinctions: We chose

$$\Pi_{obs,\ *.voltage} = \{\{[0,.2)\}, \{[.2,.4)\}, \{[.4,.6)\}, \{[.6,.8)\}, \{[.8,1.0)\}\}$$

for variables involving voltage and likewise to values

$$\Pi_{obs,\ *.position} = \{\{0\}, \{.2\}, \{.4\}, \{.6\}, \{.8\}, \{1.0\}\}$$

for variables involving position. The switch-over parameter of the idle switch component is specified as the value {.4}.

Target distinction: This is determined by the goal to distinguish between the ground voltage, corresponding to partition element {[0,.2)}, and the rest of the domain for terminal variable v_{switch} (termed switch.output.voltage in the model) of the control unit component:

$$\Pi_{targ,\ switch.output.voltage,\ 1} = \{[0,.2\}$$
$$\Pi_{targ,\ switch.output.voltage,\ 2} = \{[.2,.4), [.4,.6), [.6,.8), [.8,1.0)\}$$

Induced distinctions: The algorithm first determines that it is necessary to distinguish between the two values

$$\Pi_{ind,\ switch.state,\ 1} = \{right\}$$
$$\Pi_{ind,\ switch.state,\ 2} = \{left\}$$

for the state of the switch, and then induces two qualitative values

$$\Pi_{ind,\ pedal.position,\ 1} = \{.6, .8, 1.0\}$$

$$\Pi_{ind,\ pedal.position,\ 2} = \{0, .2, .4\}$$

for the domain of the pedal position. The resulting partition for v_{pot} (termed potentiometer.output.voltage in the model) consists of three partition elements:

$$\Pi_{ind,\ potentiometer.output.voltage,\ 1} = \{[0,.2), [.2,.4)\}$$
$$\Pi_{ind,\ potentiometer.output.voltage,\ 2} = \{[.4,.6)\}$$
$$\Pi_{ind,\ potentiometer.output.voltage,\ 3} = \{[.6,.8), [.8,1.0)\}$$

The first qualitative value corresponds to the situation where v_{switch} equals ground voltage, the second qualitative value corresponds to the situation where v_{switch} equals battery voltage, and the third qualitative value corresponds to the situations where the position of the switch and thus the voltage of v_{switch} is ambiguous.

The resulting abstracted behavior model that achieves the same target distinctions as the original model has a tuple space of 9216. The original model, in contrast, had a tuple space of $5.6 \cdot 10^7$.

Discussion and Future Work

The work presented here has provided us with a theoretical foundation to analyze the fundamental problem of automated qualitative modeling in the context of model-based systems, and a first implementation of an algorithm to compute an approximate solution. The first experimental results were encouraging, as even the incomplete algorithm produced useful model abstractions. More experiments will have to be done, and some will aim at producing models for tasks and problems in our application domains of car subsystems and process-oriented modeling and diagnosis.

Answering the question to what extent we can scale up requires, on the one hand, analysis and improvement of the operations on the underlying OBDD encoding of the models. On the other hand, further improvements of the propagation algorithm need to be explored. For instance, rather than starting with the granularity of the base domain, there could be an iterative, top-down approach for determining qualitative distinctions which starts with some small set of partitions and refines them if there is evidence for its utility. This evidence could be provided by the labels of the induced qualitative values. They indicate which partition elements might be the best candidates for "splitting" (e.g. qualitative values with maximum label size, since this means that a large number of target partition elements is consistent with this value, i.e. the value lacks "discriminating power").

Our deeper theoretical analysis compared to [Struss Sachenbacher 99] has shown that different tasks impose different requirements and definitions, e.g. consistency check vs. prediction, independent observations vs. observed tuples, and target distinctions vs. initial distinctions.

Another important technical aspect has not been discussed in this paper, namely the relation between an algorithm for computing significant distinctions and the algorithm that uses the resulting model for prediction or diagnosis. For instance, if we had an algorithm that would actually obtain

a maximal abstraction based on performing global consistency checking, a run time system that is performing value propagation on the resulting abstract model only is likely to be unable to take advantage of it and might miss some distinctions.

Acknowledgments

We would like to thank Alexander Kutscha, who worked on the implementation of the prototypic software in Visual Basic under Windows NT, and Oskar Dressler for helpful discussions and collaboration. This work was supported in part by the German Ministry of Education and Research (#01 IN 509 41).

References

[Addanki et al. 91] Sanjaya Addanki, Roberto Cremonini and J. Scott Penberthy: Graphs of models, Artificial Intelligence, 51 (1-3) (1991) pp. 145-177.

[Bryant 92] Randal E. Bryant: Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams, Carnegie Mellon University Technical Report CMU-CS-92-160, 1992.

[Falkenhainer Forbus 91] Brian Falkenhainer, Kenneth D. Forbus: Compositional Modeling: Finding the Right Model for the Job, Artificial Intelligence 51(1-3): 95-143, 1991.

[OCC'M 00] Occ'm Software GmbH, http://www.occm.de.

[Iwasaki 92] Iwasaki, Y.: Reasoning with Multiple Abstraction Models. In Faltings, B. and Struss, P. eds.: Recent Advances in Qualitative Physics, Cambridge, Mass., MIT Press, 1992.

[Kutscha 00] Kutscha, A.: Design and Implementation of a Component for Automated Abstraction of Domains in Qualitative Models (in German), Master Thesis, Technische Universität München, Department of Computer Science, February 2000.

[Sachenbacher et al. 00] Sachenbacher, M., Struss, P., Weber, R.: Advances in Design and Implementation of OBD Functions for Diesel Injection Systems based on a Qualitative Approach to Diagnosis, SAE 2000 World Congress, Detroit, USA, 2000.

[Struss 92] 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 Publishers, pp. 419-449.

[Struss Sachenbacher 99] Struss, P., Sachenbacher, M.: Significant Distinctions Only: Context-Dependent Automated Qualitative Modeling. In: 13th International Workshop on Qualitative Reasoning (QR99), Loch Awe, Scotland, 1999.