COMPLEX SYSTEMS – SIMPLE MODELS?

Jörg Raisch

Max-Planck-Institut für Dynamik komplexer technischer Systeme
Magdeburg

Abstract: In this contribution, the relation between the intended purpose of a model and its required degree of accuracy is investigated. It is based on an intuitive definition of accuracy from J. C. Willems’ “behavioural systems theory”. We outline a procedure that generates a strictly ordered set of abstractions, or approximations, for a given detailed model, where ordering is in the sense of approximation accuracy. It is particularly useful for the purpose of control synthesis: within the set of abstractions, there exists a unique coarsest, i.e. least accurate, model that allows a given specification to be met. The resulting control scheme is guaranteed to “work properly” for the underlying detailed model. The approach is illustrated by a number of examples from process control. Copyright © 2000 IFAC

Keywords: Abstractions, approximations, discrete-event systems, behavioural theory.

1. INTRODUCTION

In most areas of science and engineering, it is an accepted fact that modelling is a purpose-driven act. Namely, the required accuracy of any mathematical process model depends on the application problem which is to be solved on the basis of the model. In particular, models that are to be used for open-loop applications (e.g., simulation) need to be much more accurate than models that are built for solving closed-loop problems (e.g., the design of feedback control): a closed-loop information structure allows one to actively combat model uncertainty.

It is obvious that, in general, there exists a trade-off between model accuracy and model simplicity: to achieve a more accurate process description, one may switch from a static to a dynamic model (by taking into account time), one may increase the order of the model (by modelling additional features), one may replace a linear model (which, in general, is only adequate in the vicinity of a single operating point) by a nonlinear model (which covers a wide operating region). In short: improving model accuracy implies increasing model complexity.

As a natural consequence, we want a process model to be as precise (and therefore as complex) as necessary to solve a given application problem, but also as simple (and therefore as inaccurate) as possible without sacrificing solvability. This does not only please our sense of beauty — after all, Ockham’s razor (“Entities should not be multiplied beyond necessity”) constitutes a cornerstone of engineering aesthetics — but it also minimizes the workload in any subsequent (model-based) step.

To address this tradeoff properly, we need to formalize the notion of “model accuracy”. This is easily accomplished within the framework of J. C. Willems’ “behavioural systems theory” (see Willems, 1989; Willems, 1991). It is worth re-

1 Support by Deutsche Forschungsgemeinschaft (DFG) through "Sonderforschungsbereich 412" is gratefully acknowledged.

2 This “principle of parsimony” became associated with the “More than Subtle Doctor” W. Ockham (1285-1347); it characterizes his philosophical conclusions although it was apparently never actually phrased by him.
marking that the basic principles of “behavioural systems theory” are in perfect accordance with K. POPPER’s central ideas on the logic of science (Popper, 1934): a dynamic model, in WILLEMS’ sense, is characterized by the set of all external signals it allows to occur; this set is called the model behaviour. Clearly, a model defined in this way is falsifiable and hence, in POPPER’s terminology, a scientific statement: the model is falsified by any observed signal which is not an element in the model behaviour. Now consider two models which have not yet been falsified; model A is said to be more accurate than model B, if its behaviour is a proper subset of the behaviour of model B. This is a natural and very intuitive definition of accuracy, as model A predicts the future evolution in a more precise manner (i.e. predicts it to be in a smaller set) than model B. It is also consistent with POPPER’s terminology.

We can now turn to the more specific problem of synthesizing feedback control for a given chemical process. Any model which the chemical engineering community accepts as a “decent representation” of the real process can serve as a starting point. However, in all likelihood, this model will be too complex for the purposes of feedback synthesis – control systems design using the known “tools of the trade” might either be too time-consuming or even plain impossible. Hence, we replace the given model by a less accurate and therefore less complex model. This reduction in accuracy/complexity can be taken up to a point where we can still find a controller that enforces the (given) specifications, but where any further reduction would imply loss of solvability of the control problem. We then synthesize a suitable controller for the simplified model. Of course, we need to guarantee that this controller also works properly when connected to the original (detailed) model. It is the beauty of the described approach that this property comes for free: if the simplified model is less accurate than the original one (in the precise sense defined above), our controller will also solve the problem for the latter model.

As long as the above requirements are met, one can go to any extreme. In particular, it is possible to replace an n-th order ODE-model (involving n state variables that can each take an infinite number of values) by a finite state machine, or automaton (involving one state variable which can only take a finite number of values). This is particularly helpful when dealing with hybrid systems, which are characterized by the interaction of continuous and discrete-event components. Hybrid control problems, both verification and synthesis, have proven to be intrinsically difficult, if not impossible, to solve. This explains why abstraction based techniques have enjoyed considerable popularity (e.g., (Antsaklis et al., 1993; Cury et al., 1998; Lunze, 1995; Raisch and O’Young, 1997; Moor and Raisch, 1999)). By approximating the continuous component by a discrete system, the overall problem is translated into a purely discrete problem; this, in turn, can subsequently be approached using established methods from computer science or discrete-event systems theory. Any “positive outcome” of the problem investigation on the discrete level translates back into a positive result on the hybrid level: if we can verify that a given discrete controller “works properly” for a discrete abstraction, we know it will also “do the job” if connected to the underlying continuous or hybrid plant model; if we succeed in synthesizing such a controller for the discrete abstraction, we have also solved the control synthesis problem for the original problem. Simplification demands a price, however. It is inconclusiveness in cases of negative outcome: for example, if there is no solution to the control synthesis problem on the abstraction level, we cannot infer that the underlying problem cannot be solved.

The main purpose of this paper is to demonstrate feasibility of approximation based approaches to hybrid control systems synthesis. This, by implication, will show that simple models may indeed suffice to treat control problems for not so simple processes and will hence affirmatively answer the question posed in the title of this contribution. We will only touch upon theoretical issues where absolutely necessary. Instead, we will resort to a series of examples to illustrate the basic ideas.

This paper is organized as follows: Section 2 recalls a few basic ideas from WILLEMS’ “behavioural systems theory”. In particular, it explains why these ideas provide an ideal framework to discuss approximations with regard to control systems synthesis. Section 3 suggests a straightforward approximation scheme which generates a hierarchy of discrete abstractions for a given continuous plant model. This hierarchy is ordered with respect to approximation accuracy, and any element within the hierarchy is a candidate to serve as a basis for control synthesis. Section 3 is based on (Raisch, 1998), and the reader is referred to this reference for details. Section 4 introduces an extremely simple example, its only purpose being to illustrate the approximation scheme from the previous section. Then, in Section 5, a more demanding example is introduced – synthesis of a safe shut-down procedure for a batch evaporator problem. It represents joint work with E. KLEIN and is taken from (Klein and Raisch, 1998). Finally, in Section 6, we consider the problem of synthesizing a closed-loop start-up strategy for a distillation column. This is based on joint work with E. KLEIN, A. KIENLE and A. ITIGIN (Klein et al., 2000).
2. BEHAVIOURS AND ABSTRACTIONS

Let $T \subseteq \mathbb{R}$ denote time. This covers both continuous time ($T = \mathbb{R}$ or $T = \mathbb{R}^+$) and discrete time ($T = \{\ldots, t_0, t_1, \ldots\}$ or $T = \{t_0, t_1, \ldots\}$). Suppose we want to model the temporal evolution of a (vector) variable “living” in some set $W$, where $W$ can be Euclidean space or just a set of symbols without any mathematical structure (e.g., the colours of a traffic light). Let $W^T$ represent the set of all functions mapping $T$ into $W$ or, in other words, the set of all signals defined on time $T$ taking values in $W$. Obviously, what a model needs to do is to discriminate between signals $w \in W^T$ which it deems possible and signals that, according to the model, cannot occur. Hence, in WILLEMS’ “behavioural systems theory” (Willems, 1989; Willems, 1991), a model is defined to be a triple $(T, W, B)$, where the behaviour $B$ is exactly the set of signals that the model allows to occur. Clearly, for any non-trivial model, $B$ will be a proper subset of $W^T$.

As an example, let’s say that we want to model the future evolution of temperature in the city of Magdeburg. Then $T = \mathbb{R}^+$, with $0$ representing the time of writing, and $W = [-273°C, \infty)$. The following is a very crude, though nontrivial model: $\Sigma = (T, W, B)$ where $B = \{w \in W^T | -50°C \leq w(t) \leq 50°C, t \in T\}$. Clearly, a dynamical model with unrestricted time cannot be verified, but any nontrivial model could be falsified (for example by a temperature measurement of more than $+50°C$ in Magdeburg at some time in the future).

Suppose that there are two models $\Sigma_1 = (T, W, B_1)$ and $\Sigma_2 = (T, W, B_2)$ explaining the same phenomenon and neither of these models has been falsified. $\Sigma_1$ is called at least as accurate as $\Sigma_2$ if $\Sigma_1 \subseteq \Sigma_2$. Accuracy (in the sense of “$\subseteq$”) imposes a partial order on the set of all models $\{(T, W, B) | B \subseteq W^T\}$. This definition of model accuracy is both intuitive and consistent with POPPER’S philosophy of science: “... the more they prohibit, the more they say” (Popper, 1934).

As behaviours are a usually infinite collection of signals, they are not directly suitable for computational purposes. To perform actual calculations, a finite representation of $(T, W, B)$ is needed. Behaviours are, however, an excellent way to visualize various aspects of systems and control. As an example, let us consider a standard feedback configuration: denote the control inputs of the plant by $u(t) \in U, t \in T$, its measurable outputs by $y(t) \in Y, t \in T$. Then $W := U \times Y$, and $\Sigma_p := (T, U \times Y, B_p \subseteq (U \times Y)^T)$ is a plant model. It is to be controlled by feedback $y$ to $u$ via a second system (“the controller”) with behaviour $B_c$. Then, the closed loop behaviour is given by $B_{pc} = B_p \cap B_c$ — only signal pairs $(u, y)$ that are compatible with the dynamics of both plant model and controller “survive” closing the loop. In the simplest case, closed loop specifications can be formulated as a “legal” set $B_{spec} \subseteq (U \times Y)^T$ of signal pairs. The control task is then to “force” $\emptyset \neq B_{pc} \subseteq B_{spec}$ by finding (and realizing) a suitable $B_c$ (Fig. 1). Now, suppose that controller

![Fig. 1. Control and abstraction.](image)

synthesis for a system $\Sigma_p = (T, U \times Y, B_p)$ is inconvenient (because, for example, realizations of $\Sigma_p$ are tricky to handle). Hence, we want to perform the synthesis step on the basis of an approximation, or abstraction, $\Sigma_a = (T, U \times Y, B_a)$. Clearly, a condition sine qua non for $\Sigma_a$ is that

$$B_p \subseteq B_a.$$  

(1)

If this condition were violated, $\Sigma_p$ could respond to a given input signal with an unacceptable measurement signal which would not be predictable by the abstraction. Hence, this unacceptable phenomenon could not be suppressed by a control strategy based on $\Sigma_a$—the abstraction would be useless as far as control synthesis is concerned. As illustrated in Fig. 1, the “abstraction condition” (1) implies

$$B_a \cap B_c \subseteq B_{spec} \implies B_p \cap B_c \subseteq B_{spec}. \quad (2)$$

One also needs to ensure that $B_p \cap B_c \neq \emptyset$ or, in other words, that $\Sigma_p$ and $\Sigma_c$ are “nonblocking” (they can agree on at least one common pair of signals on $T$)\(^3\). If this can be achieved, any controller which enforces the specifications for the abstraction $\Sigma_a$ will also make the “base” model $\Sigma_p$ obey the specifications.

If (1) holds, the “size” of the difference $B_a \setminus B_p$ is an indicator for the accuracy of the approximation: the “smaller” $B_a \setminus B_p$, the smaller the loss in “prediction power” when replacing $\Sigma_p$ by its abstraction $\Sigma_a$. The trivial abstraction (with $B_a = (U \times Y)^T$) has no “prediction power” whatsoever. It is obvious, that no controller can enforce the specifications on the abstraction level, if the plant approximation is “too coarse” (if, for example, the trivial abstraction is chosen).

\(^3\) In the following important scenario, this property comes for free: $T$ represents an equidistant sampling grid, i.e. $T = \{t_0, t_1, \ldots\}$ with $t_{i+1} - t_i = \text{constant}, i = 0, \ldots; \ldots$; the plant model $\Sigma_p$ can be realized by a strictly causal system (with $u$ being the input and $y$ the output), and the controller is realized by a causal system (with the role of $u$ and $y$ reversed). As any such feedback connection has a solution on $T$, nonblocking is guaranteed.
3. DISCRETE ABSTRACTIONS FOR CONTINUOUS SYSTEMS

We now switch to a slightly different perspective and concentrate on a special case. Consider the following (minimal) realization of a plant:

\[ x(t_{k+1}) = f(x(t_k), u_d(t_k)), \]
\[ y_d(t_k) = q_p(x(t_k)), \]  
where \( T = \{t_0, t_1, \ldots \} \) represents an equidistant sampling grid, and \( x(t_k) \in \mathbb{R}^n \) is the continuous state at time \( t_k \). Both the control input, \( u_d(t_k) \in U_d \), and the measured output, \( y_d(t_k) \in Y_d \), can only take a finite number of discrete values, i.e. \( U_d \) and \( Y_d \) are finite sets of symbols. \( f : \mathbb{R}^n \times U_d \rightarrow \mathbb{R}^n \) is the state transition map, \( q_p : \mathbb{R}^n \rightarrow Y_d \) the output map. Without loss of generality, the latter is required to be onto. If required, nondeterminism can be covered by introducing \( f \) as \( f : \mathbb{R}^n \times U_d \rightarrow 2^{\mathbb{R}^n} \). Alternatively, a realization of the plant model could be given by a set of differential equations (or inclusions) in \( \mathbb{R}^n \), with a sampling device for the outputs and a hold device for the inputs. In both cases, we deal with a continuous realization (the state is “continuous-valued”) although the external behaviour \( B_p \subseteq (U_d \times Y_d)^T \) is clearly discrete.

We now try to find abstractions \( \Sigma_a = (T, U_d \times Y_d, B_a) \) which can be realized by finite state machines and satisfy \( B_a \supseteq B_p \). Slightly abusing terminology, we say that we look for discrete abstractions for the given continuous plant model (3), (4). The motivation for doing so is clear from the previous section: if we succeed, control systems synthesis can be performed on the basis of the much simpler model \( \Sigma_a \), and the resulting controller will work “properly” for \( \Sigma_p \).

In order to specify desired abstraction behaviours, we need a bit of additional notation: recall that \( T = \{t_0, t_1, \ldots \} \) is the sampling grid. Denote the intervals \( \{t_0, \ldots, t_k\} \) and \( \{t_{k+1}, \ldots, t_k\} \) by \( T_k \) and \( T_{k+} \), respectively. If \( t_k \) refers to the present sampling instant, \( T_k \) comprises “the future”, and \( T_{k+} \) “the past and present”. Define \( B_{p, k} \subseteq (U_d \times Y_d)^{T_{k+}} \) to be the restriction of the behaviour \( B_p \) to the interval \( T_k \):

\[
B_{p, k} := \{ b^k \in (U_d \times Y_d)^{T_{k+}} \mid \exists b^{k+} \in (U_d \times Y_d)^{T_{k+}} \text{ such that } [b^k, b^{k+}] \subseteq B_p \}.
\]

Finally, introduce the predicted output set \( Y_p(b^k) \subseteq Y_d \) as the set of all measurement symbols that the continuous model (3), (4) can possibly generate at time \( t_{k+1} \) if the string \( b^k \) has occurred. Then, the continuous system behaviour can be written iteratively as:

\[
B_{p, k+1}^n = \left\{ \left[ b^k, \left( y_d^{(i)}, u_d^{(i)} \right) \right] \mid b^k \in B_{p, k}^n, \right. \\
\left. u_d^{(i)} \in U_d, y_d^{(i)} \in Y_p(b^k) \right\}.
\]

Now we define less accurate prediction sets \( Y_l \), \( l = 0, 1, \ldots \), by using only “recent data”. By “recent data” (denoted by \( b^{k,l} \)), we mean the restriction of \( b^k \) to the interval

\[
T_{k,l} := \{ t_{k-l}, t_k \} \text{ if } k > l, \\
\{ t_0, \ldots, t_k \} \text{ if } k \leq l.
\]

Hence, \( b^{k,l} \) represents a string of input and measurement symbols reaching back to time \( t_{k-l} \) or, if \( k \leq l \), to the initial sampling instant \( t_0 \). Obviously,

\[
b^k = \{ b^{k-l-1, k} \} \text{ if } k > l, \\
b^{k,l} \text{ if } k \leq l.
\]

\( Y_l(b^{k,l}) \) is defined as the set of all measurement symbols that the continuous model (3), (4) can generate at time \( t_{k+1} \) if the string \( b^{k,l} \) has occurred during the time interval \( T_{k,l} \). Clearly,

\[
Y_l(b^{k,l}) := \bigcup_{b^{k-l-1}} \left\{ \left[ y_p(b^k) \right] \mid b^k \in B_{p, k}^l \right\}
\]

i.e. \( Y_l(b^{k,l}) \) is obtained as the union of the prediction sets of all data strings \( b^k \) which coincide on the interval \( T_{k,l} \) (but may differ during the “distant past” \( T_{k-l} \)) (see Fig. 2 for an illustration). Therefore,

\[
Y_0(b^{k,0}) \supseteq Y_1(b^{k,1}) \supseteq \ldots Y_l(b^{k,l}) \ldots \supseteq Y_p(b^k).
\]

From the prediction sets \( Y_l \), the approximation

\[
\text{Fig. 2. Predicted output sets.}
\]
behaviours \( B_l \) can be defined iteratively by

\[
B_0 := B_p^0, \\
B_{l+1} := \left\{ \left[ b^k, \left( y_d^{(i)}, u_d^{(i)} \right) \right] \mid b^k \in B_l, \right. \\
\left. u_d^{(i)} \in U_d, y_d^{(i)} \in Y_p(b^k) \right\}, \quad k = 0, 1, \ldots
\]

From (5) – (8), it follows immediately that

\[
B_0 \supseteq B_1 \supseteq \ldots B_l \ldots \supseteq B_p.
\]

We now show how to build a finite state machine \( A_l \) which realizes \( \Sigma_l = (T, U_d \times Y_d, B_l) \), \( l = 0, 1, \ldots \). Recall that, by definition, the state
of any dynamical system summarizes all the information which, together with the current input, is needed to predict the future: Clearly, to compute the output prediction set $\mathcal{Y}_d(b^{k,l})$, we need to know $b^{k,l}$. Hence, we define the state $x_d$ of the realization $A_d$ at time $t_k$ to consist of all the input and measurement symbols in the string $b^{k,l}$, with the exception of the current input $u_d(t_k)$. For $l = 0$, the result is trivial: $x_d(t_k) := y_d(t_k)$; for $l = 1,2,\ldots$, we get:

$$x_d(t_k) := \begin{cases} 
[y_d(t_0), \ldots, y_d(t_k)], [u_d(t_0), \ldots, u_d(t_{k-1})], & \text{if } k = 1, \ldots, l, \\
[y_d(t_{k-1}), \ldots, y_d(t_k)], [u_d(t_{k-1}), \ldots, u_d(t_{k-1})], & \text{if } k > l.
\end{cases}$$

Note that this choice makes the state instantly observable. It also implies that the state set $X_d$ of $A_d$ consists of strings of measurement and control symbols up to length $l$. More precisely, $X_d \subseteq Y_d$ for $l = 0$, and $X_d \subseteq Y_d \cup \{y_d^{l+1}, y_d^p \times (U_d)^{p-1}\}$ for $l = 1,2,\ldots$ As both $U_d$ and $Y_d$ are finite, $X_d$ is also finite for every $l \in \mathbb{N}$.

It is obvious that, for a fixed $l \geq 1$, not all elements of $Y_d \cup \{y_d^{l+1}, y_d^p \times (U_d)^{p-1}\}$ are reachable. In other words, if we feed a specific string of $\rho - 1$ control symbols into the continuous system, it can in general not respond with an arbitrary string of $\rho$ measurement symbols, $2 \leq \rho \leq l + 1$. We say that certain strings of control and measurement symbols are not compatible with the continuous system dynamics. To make the state set $X_d$ minimal, we remove all such strings. If the continuous system is given by (3),(4), this amounts to checking whether a set of (nonlinear) algebraic equations in $\mathbb{R}^n$ has a solution. For details, we refer to (Raisch, 1998).

We have now defined the state set of the realization $A_d$, but still need to describe its transition structure: denote the strings of input and measurement symbols associated with a particular $x_d^{(i)} \in X_d$ by $u^*(x_d^{(i)})$ and $y^*(x_d^{(i)})$, respectively, and introduce a “forgetting operator” $\mathcal{F}$ which deletes the “oldest” symbol from strings $y_d^*(x_d(t_k))$ and $u_d^*(x_d(t_k))$, if $k \geq 1$:

$$\mathcal{F}(y_d^*(x_d(t_k))) := \begin{cases} 
[y_d(t_0), \ldots, y_d(t_k)], & \text{if } k = 0,1,\ldots, l - 1, \\
y_d(t_k), \ldots, y_d(t_{k-1}), & \text{if } k \geq l.
\end{cases}$$

$$\mathcal{F}(u_d^*(x_d(t_k))) := \begin{cases} 
[u_d(t_0), \ldots, u_d(t_k)], & \text{if } k = 0,1,\ldots, l - 1, \\
u_d(t_k), \ldots, u_d(t_{k-1}), & \text{if } k \geq l.
\end{cases}$$

Now, writing down the transition structure of the discrete abstraction is straightforward: given two states $x_d^{(i)}, x_d^{(k)} \in X_d$ and a control symbol $u_d^{(j)} \in U_d$, the triple $(x_d^{(i)}, u_d^{(j)}, x_d^{(k)})$ represents a transition in $A_d$ if and only if

1. there exists a $y_d^{(m)} \in Y_d$ such that

$$y_d^*(x_d^{(k)}) = \mathcal{F}(y_d^*(x_d^{(i)}), u_d^{(j)}),$$

$$u_d^*(x_d^{(k)}) = \mathcal{F}(u_d^*(x_d^{(i)}), u_d^{(j)}),$$

2. $(y_d^*(x_d^{(i)}), y_d^{(m)}), (u_d^*(x_d^{(i)}), u_d^{(j)})$ is compatible with the continuous system (3),(4).

The first condition can be verified by simple visual inspection of the abstraction states $x_d^{(i)}$ and $x_d^{(k)}$. The second condition can be checked by resorting to the same methods that were used to eliminate non-reachable states from $X_d$. If $(x_d^{(i)}, u_d^{(j)}, x_d^{(k)})$ turns out to be a transition in $A_d$, $x_d^{(i)}$ and $x_d^{(k)}$ are called its exit and its entrance state; the input symbol $u_d^{(j)}$ is its transition label. Each state $x_d^{(i)}$ has an associated unique (measured) output, which is simply the rightmost symbol in $y^*(x_d^{(i)})$. Hence, for each nonnegative integer $l$, we get a finite Moore automaton as a realization for the abstraction $A_d$. If a-priori information on the continuous system state is not available, any measurement symbol from $Y_d$ may occur at time $t_0$. Hence, in that case, the initial state set of the automaton $A_d$ is $X_d = Y_d$. In (Moor and Raisch, 1999), the system $\Sigma = (T, U_d \times Y_d, B_d)$ realized by $A_d$ was shown to be the strongest $l + 1$-complete approximation of the underlying plant model $\Sigma_p = (T, U_d \times Y_d, B_p)$ (for a definition of $l$-completeness see (Willems, 1989); strongest $l$-complete approximations were introduced in (Moor and Raisch, 1999)).

Remark: Even if the underlying continuous system (3),(4) is deterministic, $A_d$ will in general be nondeterministic: for a given state $x_d^{(i)}$, more than one $y_d^{(m)} \in Y_d$ may satisfy conditions 1 and 2; then, applying $u_d^{(j)}$ at state $x_d^{(i)}$ may drive the realization $A_d$ into more than one successor state. This will be further illustrated in Section 4.

Remark: In (Moor and Raisch, 1999), the system $\Sigma = (T, U_d \times Y_d, B_d)$ realized by $A_d$ was shown to be the strongest $l + 1$-complete approximation of the underlying plant model $\Sigma_p = (T, U_d \times Y_d, B_p)$ (for a definition of $l$-completeness see (Willems, 1989); strongest $l$-complete approximations were introduced in (Moor and Raisch, 1999)).

4. AN EXTREMELY SIMPLE EXAMPLE

The sole purpose of this extremely simple example is to illustrate the abstraction procedure described in the previous section. A water tank with cross section $S = 1000 \text{cm}^2$ and height $z = 30 \text{cm}$ can be fed or drained by a pump. The pump can be switched between two modes: it either feeds water into the tank at a constant rate of 1l/min, or it removes water at the same rate. The pump is in feed mode if the control input is $u_d(t) = "+"$, and in removal mode if $u_d(t) = "-"$. The measurement signal can take two values: $y_d(t) = E(\text{mpy})$.
if the water level \( x(t) \) is less or equal to 15cm, and \( y_d(t) = F(ull) \) if the water level is above 15cm (Fig. 3). After choosing a sampling grid

\[
\begin{align*}
\text{if } & \ x(t) \leq 15\text{cm}, \\
\text{then } & \ y_d(t) = F(ull), \\
\text{if } & \ x(t) > 15\text{cm}, \\
\text{then } & \ y_d(t) = E(mpty).
\end{align*}
\]

After choosing a sampling grid

\[
\begin{align*}
\text{if } & \ x(t_k) \leq 15\text{cm}, \\
\text{then } & \ x(t_{k+1}) = x(t_k) + 10\text{cm}, \\
\text{if } & \ x(t_k) > 15\text{cm}, \\
\text{then } & \ x(t_{k+1}) = x(t_k) + 30\text{cm}, \\
\text{if } & \ 0 \leq x(t_k) \leq 20\text{cm}, \\
\text{then } & \ y_d(t_k) = F(ull), \\
\text{if } & \ 20\text{cm} < x(t_k) \leq 30\text{cm}, \\
\text{then } & \ y_d(t_k) = F(ull), \\
\text{if } & \ 10\text{cm} < x(t_k) \leq 30\text{cm}, \\
\text{then } & \ y_d(t_k) = F(ull), \\
\text{if } & \ 0 \leq x(t_k) \leq 10\text{cm}, \\
\text{then } & \ y_d(t_k) = E(mpty).
\end{align*}
\]

Fig. 3. Simple tank example.

\( T = \{ t_i | t_i = \text{imin}, i = 0, 1, \ldots \} \), a continuous-valued model can be easily written down:

\[
\begin{align*}
x(t_{k+1}) &= \begin{cases} 
  x(t_k) + 10\text{cm} & \text{if } u_d(t_k) = "+", \\
  30\text{cm} & \text{if } u_d(t_k) = "++", \\
  20\text{cm} < x(t_k) \leq 30\text{cm}, \\
  10\text{cm} < x(t_k) \leq 30\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
  0 \leq x(t_k) \leq 10\text{cm}, \\
\end{cases}
\end{align*}
\]

\[
y_d(t_k) = \begin{cases} 
  F & \text{if } 15\text{cm} < x(t_k) \leq 30\text{cm}, \\
  E & \text{if } 0 \leq x(t_k) \leq 15\text{cm}.
\end{cases}
\]

It is now straightforward to derive the coarsest abstraction \( \Sigma_0 = (T, U_d \times Y_d, B_0) \) by applying the procedure described in Section 3. The state set of the realization \( A_0 \) consists of two elements: \( x_d^{(1)} = ([F]) \) and \( x_d^{(2)} = ([E]) \). The transition structure of \( A_0 \) is depicted in Fig. 4, where the output symbols associated with the states are indicated by dashed arrows. Clearly, \( B_p \subseteq B_0 \). Indeed, as expected, \( B_p \subseteq B_1 \subseteq B_0 \). For example, unlike \( \Sigma_0 \), the abstraction \( \Sigma_1 \) does not allow the string \( FFFFF \ldots \) as a response to the input string \( - - - - - - \). In this particular case, the abstraction \( \Sigma_1 \) is actually extremely accurate: the restrictions of the behaviours \( B_p \) and \( B_1 \) to the interval \( T_{0+} = \{ t_1, t_2, \ldots \} \) are identical. Hence, with the exception of the initial sampling instant \( t_0 \), the abstraction \( \Sigma_1 \) is as good for output prediction and control purposes as the underlying continuous model \( \Sigma_p \) realized by (10), (11).

5. SAFETY ENFORCEMENT FOR A BATCH EVAPORATOR

Next, we treat a batch evaporator benchmark problem suggested in (Kowalewski and Stursberg, 1998). A process flowchart is shown in Fig. 6. In (Kowalewski and Stursberg, 1998), the batch cycle is described as follows: a solution is filled into tank T1 and the solvent is evaporated until a desired concentration of the dissolved substance
is reached. During the evaporation stage, the condenser C1 is in operation and collects the steam coming from T1. When the desired concentration is reached, the material is drained from T1 into T2 as soon as T2 is available (i.e. emptied from the previous batch). We address the problem of coming up with a control program that guarantees safe shutdown of the process in case of a cooling breakdown in the condenser. Such a failure leads to dangerously high pressure in T1 if the evaporation process is continued for too long. To avoid this, the temperature in tank T1 is not allowed to exceed 385 K. On the other hand, switching the heating off immediately could cause another undesired scenario: if the temperature in T1 becomes too low, crystallization spoils the batch. Hence, temperature in tank T1 is only allowed to drop below 338 K when T1 is virtually empty. Furthermore, draining the batch from T1 into tank T2 can only begin, when T2 is empty because the two batches are not supposed to be mixed. Once draining has started, it should not be interrupted before T1 is empty. Finally, whenever tank T1 is empty, the heating must be switched off.

5.1 Detailed Model

We now describe a detailed process model. It exhibits a few minor differences when compared to the one suggested in (Kowalewski and Stursberg, 1998). Comments on the differences can be found in (Klein and Raisch, 1998). The model is realized by a set of ordinary differential equations,

\[
\frac{dx}{dt} = f(x(t), u_d(t)), \tag{12}
\]

where \( x := (T, m_1, m_2)' \) is the state vector, with \( T[K] \) the temperature in tank T1, \( m_1[100g] \) the total mass in tank T1, and \( m_2[100g] \) the mass in tank T2. \( u_d := (Q_h, Y_{V15}, Y_{V18})' \) is the control signal, with \( Q_h[W] \) the heat input, \( Y_{V15}[-] \) the position of valve V15, and \( Y_{V18}[-] \) the position of valve V18. During shut-down, the control signal can only be switched between four different values:

\[
u_d \in U_d := \{u_d^{(1)}, u_d^{(2)}, u_d^{(3)}, u_d^{(4)}\}, \tag{13}
\]

where

\[
\begin{align*}
u_d^{(1)} &= \begin{pmatrix} 5000 \\ 0 \\ 1 \end{pmatrix}, & u_d^{(2)} &= \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \\
u_d^{(3)} &= \begin{pmatrix} 5000 \\ 1 \\ 0 \end{pmatrix}, & u_d^{(4)} &= \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.
\end{align*}
\]

The realization (12) consists of energy and mass balances:

\[
\begin{align*}
dT &= -\frac{Q_h - k(A_1 + \tau D m_1)(T - T_s)}{c_p L m_1 - \hat{c}_p m_1 v - (T \hat{c}_p - \Delta h_{ev}) \frac{d m_1}{d T}}, \tag{14} \\
dm_1 &= -Y_{V15} A_R \sqrt{\frac{2 \rho_L m_1 g}{A_1}}, \tag{15} \\
dm_2 &= Y_{V15} A_R \sqrt{\frac{2 \rho_L m_2 g}{A_1}} - Y_{V18} A_R \sqrt{\frac{2 \rho_L m_2 g}{A_2}}, \tag{16}
\end{align*}
\]

where \( \hat{c}_p = c_p, L - c_p, V \) and

\[
m_v = \frac{p M_v (\frac{m_1(t)}{\rho_L}) + V_v (t_v - \frac{m_1}{\rho_L})}{R_m (T - \frac{p M_v}{\rho_m \rho_L})}, \tag{17}
\]

\[
\frac{dm_v}{dT} = -\frac{m_v}{T - \frac{p M_v}{R_m \rho_L}} + \frac{T m_v}{T - \frac{p M_v}{R_m \rho_L}} \cdot \frac{B_p}{(T + C_p)^2}, \tag{18}
\]

The following physical property relations have been used: \( p \), the saturation pressure of water, is modelled by the Antoine equation

\[
p = e^{A_0 - T \frac{B_0}{T}}. \tag{20}
\]

\( \Delta h_{ev} \), the specific enthalpy of vapourization, by

\[
\Delta h_{ev} = b_1 - b_2 T. \tag{21}
\]

Parameters for the realization (14) - (21) are collected in Table 1. Fig. 7 shows an (open loop) simulation of the emergency situation described above: crystallization can only be avoided by heating tank T1 for some small period of time (dash-dotted trajectory); otherwise, \( T \) will drop below 338.0 K before \( m_1 \) reaches the threshold where T1 can be regarded as empty (trajectory shown as solid line).

5.2 Measurement Quantization

The domain of the state variable \( x \) is given by

\[
X := \{(T, m_1, m_2)|283.0 \leq T \leq 450.0, 0 \leq m_1 \leq 60.12 \cdot 100g, 0 \leq m_2 \leq 60.12 \cdot 100g\},
\]
where the lower boundary of $T$ is ambient temperature, and the upper boundaries of $m_1$ and $m_2$ correspond to T1 and T2 being “full”. We use quantized measurement information for all three state variables; hence the quantization map $q$, partitions X into rectangular parallelepipeds. Quantization thresholds for $m_1$ and $m_2$ are 2.91·100g and 5.82·100g, respectively. Below these thresholds, T1 and T2 can be regarded as “empty”. There are four thresholds for temperature quantization, partitioning the domain of $T$ into five intervals: 385 K and 338 K may be interpreted as “explosion” and “crystallization thresholds”; we introduce two additional threshold values at $T = 375$ K and $T = 341$ K, which provide alarm messages before temperature (and hence pressure) becomes unacceptably high or crystallization sets in. Proper choice of the latter value is a bit tricky; it depends on the sampling interval $\Delta = t_{i+1} - t_i$, and its rationale is summarized in Fig. 8, where scenarios for three different sampling intervals are visualized: the case $\Delta = 14$ sec is represented by solid lines, the cases $\Delta = 18$ sec and $\Delta = 25$ sec by dash-dotted and dashed lines, respectively. In each case, heating must be switched off if $(m_1, T)$ is above the top line; otherwise the 385 K threshold could be reached within one sampling interval. Underneath the lower line, heating has to be switched on; otherwise $T$ could drop below the crystallization threshold within one sampling instant. Clearly, to operate the process, it is necessary that the upper and the lower line intersect for $m_1 < 2.91·100g$ (where tank T1 can be regarded as empty). This is true for $\Delta = 14$ sec, and it still holds if the solid lower line is replaced by a straight line at $T = 341$ K. Hence, $\Delta = 14$ sec is chosen as the sampling interval and $T = 341$ K as an additional threshold for temperature quantization. This leaves us with 2·2·5 measurement symbols. They are summarized in Table 2.

5.3 Discrete Abstraction and Control Synthesis

Continuous model equations, measurement quantization and sampling interval completely determine the abstraction set $\{\Sigma_l, l = 0, 1, \ldots\}$ and
its realizations $A_1$. We first compute $A_0$, a finite Moore automaton with 20 states (i.e. one state variable that can take 20 values); we then formalize the specifications by writing down a finite state machine realizing the specification behaviour. Using a slightly modified version of RAMADGE’s and WONHAM’s supervisory control theory (Ramadge and Wonham, 1987; Ramadge and Wonham, 1989), we check whether there exists a discrete control scheme enforcing the specifications for $\Sigma_0$. It turns out that this is not the case: $\Sigma_0$ is “too coarse”, and we have to resort to a more accurate abstraction.

We therefore compute the automaton $A_1$, which realizes $\Sigma_1$. $A_1$ has 307 states (i.e. one state variable that can take 307 values) and 3463 transitions. All safety requirements that were previously described can now be formulated in terms of “forbidden states” and “forbidden transitions” of $A_1$:

1. The requirement “Temperature must not exceed 385 K” translates into “all abstraction states $x_d^{(i)}$ with measurement symbol $y_d^{(17)}$, ..., $y_d^{(20)}$ are forbidden”.
2. The requirement “Temperature must not fall below 338 K while tank T1 is non-empty” (i.e. while $m_1 > 2.91 \cdot 100g$) translates into “all abstraction states $x_d^{(i)}$ with measurement symbol $y_d^{(3)}$ or $y_d^{(4)}$ are forbidden”.
3. “Heating not allowed while T1 empty” becomes “abstraction states $\left([y_d^{(i)}, y_d^{(j)}], u_d^{(k)}\right)$ are forbidden if $(k \in \{1, 3\}) \land (i \in \{1, 2, 5, 6, 9, 10, 13, 14, 17, 18\})$”.
4. The requirement “Do not start draining tank T1 unless T2 is empty” translates into: transition $x_d^{(i)} \rightarrow x_d^{(k)}$ is not allowed if
   $$j \in \{3, 4\} \land \left(\left(x_d^{(i)} = \left([y_d^{(i)}, y_d^{(j)}], u_d^{(k)}\right)
   \begin{array}{c}
   \text{with } i_2 \in \{2, 4, 6, ..., 20\} \land i_3 \in \{1, 2\} \\
   \lor (x_d^{(i)} = y_d^{(i)} \text{ with } i \in \{2, 4, 6, ..., 20\})
   \end{array}\right)$$
5. The requirement “Once draining of T1 has begun, do not stop until T1 is empty” translates into: transition $x_d^{(i)} \rightarrow x_d^{(k)}$ is not allowed if
   $$j \in \{1, 2\} \land \left(x_d^{(i)} = \left([y_d^{(i)}, y_d^{(j)}], u_d^{(k)}\right)
   \begin{array}{c}
   \text{with } i_2 \in \{3, 4, 7, 8, 11, 12, 15, 16, 19, 20\} \land \\
   i_3 \in \{3, 4\}
   \end{array}\right)$$

Again, using the slightly modified version of RAMADGE’s and WONHAM’s supervisory control theory mentioned above, we synthesize the least restrictive controller that guarantees safe shut-down of the abstraction $\Sigma_1$. A solution exists and control action can be summarized as follows: the supervisor infers the present state $x_d(t_k)$ of the automaton $A_1$ from the past measurement and control symbols, $y_d(t_{k-1})$ and $u_d(t_{k-1})$, and the present measurement symbol $y_d(t_k)$. For each value $x_d^{(i)}$ of the discrete state variable, a subset of symbols from $U_d$ (but never the entire set $U_d$) is disabled. This restricts the number of possible transitions and eliminates certain elements from the discrete state set $X_d$ by making them unreachable. In particular, all forbidden states and transitions are removed. $A_1$ under supervision consists of 100 states and 539 transitions. As the abstraction behaviour $B_1$ – by construction – is a superset of the continuous plant model behaviour $B_p$, safety is also guaranteed for the continuous plant model under supervision.

5.4 Simulation Results

In this section, we give an example for the shut-down procedure of the supervised continuous plant model. Recall that a supervisory control scheme may enable several control symbols at each sampling instant. As all of them are “safe”, any one of them can be picked by some lower level mechanism. The initial state for the simulation of the detailed model is $x(t_0) = (376.5 K, 60 \cdot 100g, 60 \cdot 100g)$, denoted by “1” in Figs. 9 - 11. According to Table 2, this generates the measurement symbol $y_d(t_0) = y_d^{(16)}$. This symbol characterizes an emergency situation. To pre-
6. START-UP OF A DISTILLATION COLUMN

It is now demonstrated that the ideas described in this paper can be used to synthesize a supervisory control scheme for the start-up procedure of a distillation column. These results represent joint work with E. KLEIN, A. KIENLE and A. ITIGIN. Because of lack of space, it is only possible to give a short overview; details can be found in (Klein et al., 2000).

We consider a distillation column in pilot plant scale which is operated at the Institut für Systemdynamik und Regelungstechnik in Stuttgart. It is about 10m high, and consists of 40 bubble cap trays, a reboiler and a condenser (Fig. 12). Our application example is the separation of methanol and propanol.

The following steps can be distinguished during "conventional" column start-up: initially, the column trays are partially filled with liquid mixture from the previous experimental run. Further feed is added, and the column is heated until boiling point conditions are established in the whole column. During this start-up step, the column is operated at total reflux and reboil. At the end of this step, a single concentration front is established. The position of this front depends on the initial concentration distribution and varies from experiment to experiment. In a second step, the feed \( F \), reboil \( V \) and distillate flow rate \( D \) are adjusted to their desired steady state values, and the initial front splits into two fronts.

Now, in a third step, the two fronts move slowly towards their steady state position. This is illustrated by the simulation results shown in Fig. 13 (a); the simulation is based on a detailed plant model consisting of material balances for each tray, the reboiler and the condenser. Start-up is considered to be finished once the plant is "close" to the desired steady state.

We try to speed up the third step of the start-up procedure by introducing a suitable control strategy which switches between discrete values of the control inputs and relies on highly quantized measurement information. This represents a special hybrid control problem – the plant state "lives" in \( \mathbb{R}^n \) (with \( n = 42 \)), whereas control inputs and measurement signals are discrete-valued, or symbolic.

As can be seen from Fig. 13, the state variables (concentrations) \( x_z, z = 1, \ldots, 42 \), are not arbi-
Start-up is considered to be finished if 8 \leq s_r < 12, 29 \leq s_s < 33 and 0.315 \leq x_{21} < 0.325, i.e. if measurement symbol \( y_d^{(123)} \) occurs. Within the selected framework, it is reasonable to expect that – irrespective of the initial conditions – the task can be completed within 20 minutes (i.e. within two sampling intervals). This is formally expressed by the specification \( \Sigma_{\text{spec}} = (T, U_d \times Y_d, B_{\text{spec}}) \) with

\[
B_{\text{spec}} = \{ b \in (U_d \times Y_d)^T \mid y_d(u_k) = y_d^{(123)}, k \geq 2 \},
\]

i.e. there is no restriction with regard to the control symbols and the first two measurement symbols; all subsequent measurement symbols need to be \( y_d^{(123)} \), indicating that we require \( s_s, s_r \) and \( x_{21} \) to be in the desired range from the third sampling instant on. It is of course straightforward to realize \( \Sigma_{\text{spec}} \) by an automaton \( A_{\text{spec}} \).

We now apply the approximation scheme from Section 3 to generate the “coarsest” element \( \Sigma_0 \) in our hierarchy of discrete abstractions. A minimal realization \( A_0 \) consists of 246 states and 8249 transitions. However, when applying the discrete control synthesis procedure to \( A_0 \) and \( A_{\text{spec}} \), we find that no solution exists. As in the batch evaporator example, the coarsest abstraction in \( \{ \Sigma_i, i = 0, 1, \ldots \} \) proves to be too coarse. We therefore need a more accurate abstraction and turn to \( \Sigma_1 \). It is realized by a Moor automaton \( A_1 \) with 8495 states and 118936 transitions. This time, the control synthesis procedure comes up with a solution and generates a least restrictive supervisor for the start-up procedure. The supervisor can be interpreted as another simple automaton which tracks the strings of measurement and control symbols and, at each sampling instant, disables all control symbols that might allow the system to "escape" the specifications. All other control symbols remain enabled, and any of them can be picked without violating the specifications. Recall that, by construction, this control scheme is guaranteed to work for the underlying continuous model. This is illustrated by Fig. 13 (b): it shows a simulation of the closed loop system consisting of continuous plant model and discrete controller during the third step of start-up. As required, the closed loop system generates the desired measurement symbol after two sampling intervals. Clearly, this represents an enormous improvement compared to the open-loop behaviour shown in Fig. 13 (a).

7. CONCLUSIONS

In this paper, the question "complex systems – simple models?" has been investigated from a control engineering point of view. We have outlined a procedure that generates a hierarchy of discrete abstractions for a given continuous plant model. The abstractions are realized by finite automata and hence belong to an extremely simple class
of dynamical systems. The hierarchy of abstractions is ordered in the sense of approximation accuracy, with the “top element” being the least accurate but also the most simple abstraction. Any of these abstractions can be used as a basis for discrete control synthesis: if an adequate controller exists, it is guaranteed to “work properly” for the continuous “base” model. The approach has been illustrated by two examples from process control: safety enforcement in a benchmark batch evaporation problem and automatic start-up of a distillation column.

Acknowledgements: The results in this paper originate from collaborations with a number of people: theoretical aspects of the abstraction procedure were jointly investigated with T. Moor (ANU, Canberra); E. Klein (Robert Bosch GmbH) did most of the work for both the batch evaporator and the distillation example; A. Itigin (Universität Stuttgart) and A. Kienle (MPI Magdeburg) also contributed tremendously to the distillation example.

8. REFERENCES


