Discrete Abstractions of Continuous Systems – an Input/Output Point of View

J. RAISCH*

ABSTRACT

This contribution proposes a hierarchy of discrete abstractions for a given continuous model. It adopts an input/output point of view, and starts from the continuous system behaviour $\mathcal{B}_c$ (i.e. the set of all pairs of input and output signals which are compatible with the continuous model equations). The first step is to construct a sequence of behaviours $\mathcal{B}_l$, $l = 0, 1, \ldots$, such that $\mathcal{B}_0 \supseteq \mathcal{B}_1 \supseteq \ldots \supseteq \mathcal{B}_c$. In a second step, nondeterministic Moore automata A_l are generated as minimal realizations for the behaviours $\mathcal{B}_l$. Hence, the continuous base system and its discrete abstractions A_l form a totally ordered set of models, where ordering is in the sense of set inclusion of model behaviours or, equivalently, in terms of approximation accuracy. Within this set, there exists a uniquely defined "coarsest" (and therefore least complex) model which allows a given set of specifications to be enforced by discrete feedback. The ordering property implies that this discrete feedback also forces the continuous base system to obey the specifications.

Key words: abstractions, approximations, automata, behaviours.

1 INTRODUCTION

The motivation for this work stems from a standard problem in hybrid control systems theory – the design of discrete controllers (supervisors) for a continuous or hybrid plant with discrete control inputs and quantized measurements[1]. By approximating the continuous part of the plant by a discrete model, the

*Max-Planck-Institut für Dynamik komplexer technischer Systeme, Leipziger Str. 44, D-39120 Magdeburg, Germany, email: raisch@mpi-magdeburg.mpg.de

[1]This paper is set in a discrete-time framework, i.e. the domain of all signals is $\{t_0,\ t_1,\ \ldots\}$, with $t_k - t_{k-1}$ being constant. This understood, the adjectives "discrete" and "continuous" will in the sequel only be used to refer to the *codomain* of signals: the codomain of a discrete, or discrete-valued, signal is a countable (and for our purposes finite) set of symbols; the codomain of a continuous, or continuous-valued, signal are the real numbers. The state of a continuous model evolves in $\mathbb{R}^n$, whereas the state of a discrete model "lives" on a (finite) set of symbols.

hybrid control problem is converted into a purely discrete one, which can subsequently be solved using standard tools from DES (discrete event systems) theory (e.g. [25]). The discrete approximation, or abstraction[2], has to meet two key requirements, which can be formulated conveniently in terms of *behaviours*[3]: 1. the approximation behaviour must contain the continuous model behaviour; 2. the approximation behaviour must be "sufficiently small" (the abstraction "sufficiently accurate") to allow a given set of specifications to be met. Condition 2 ensures that there exists a discrete controller which is able to force the specifications upon the approximation. Condition 1 guarantees that the underlying continuous or hybrid base system also satisfies the specifications when subjected to the same (discrete) feedback law. In other words: if conditions 1 and 2 hold, design of (supervisory) feedback control for continuous or hybrid plants can be based on discrete abstractions.

We adopt an input/output philosophy to solve the approximation problem: in a first step, we choose a set of behaviours $\mathcal{B}_l$, $l = 0, 1, \ldots$, such that $\mathcal{B}_0 \supseteq \mathcal{B}_1 \supseteq \ldots \supseteq \mathcal{B}_c$ (where $\mathcal{B}_c$ denotes the behaviour of the underlying continuous system). The approximation behaviour $\mathcal{B}_{l+1}$ is said to be *at least as accurate* as $\mathcal{B}_l$, because the number of "spurious solutions" (elements in $\mathcal{B}_{l+1} \setminus \mathcal{B}_c$) is less or equal to the corresponding number in $\mathcal{B}_l$. In a subsequent step, we "build" discrete minimal realizations (state models) A_l for the behaviours $\mathcal{B}_l$. It turns out that nondeterministic Moore automata provide a "natural solution" for the realization problem. This approach therefore generates a totally ordered set of abstractions A_l, where the order is in the sense of set inclusion of the associated behaviours or, equivalently, in terms of approximation accuracy. It is then straightforward to search this set for the "coarsest" approximation that allows the specifications to be met.

Approximating continuous systems by discrete models for the purpose of feedback design has been suggested, for example, by *Antsaklis*, *Lemmon*, and *Stiver* [27, 2], and by *Lunze* [10, 11, 12]. They propose a discrete model which corresponds to the least accurate element in our hierarchy of abstractions. This, however, may well turn out to be too coarse for a given set of specifications to be met. Hence, unless one can change "the rules of the game" and create a more accurate approximation by simply altering quantization levels in the measurement signal, one has "run out of luck". This motivated *S. O'Young* and the author of the present paper to suggest a modified approximation scheme [20, 21, 22, 23], which provides a *set* of abstractions (ordered in the sense of approximation accuracy) and hence the possibility to adjust approximation accuracy to suit various specifications. As this approach is set in discrete time

[2]The expressions "(discrete) abstraction" and "(discrete) approximation (of a given continuous or hybrid system)" are used synonymously throughout the paper.

[3]The behaviour of a model is the set of pairs of input/output signals which are compatible with the model equations.

(with the sampling instants being fixed a-priori), time is trivially retained on the abstraction level: both the continuous "base" model and its abstractions evolve on the same (discrete) time axis. The present paper builds on, and improves, these results: without changing the state sets of the abstractions, their approximation accuracy is increased by "trimming" transition structures. This improvement follows from the input/output point of view adopted in this paper. The theoretical aspects in the present paper can be formally interpreted in terms of *strongest l-complete approximations*. Details can be found in [17].

Another approach which allows adjusting approximation accuracy without changing measurement quantization levels has been proposed by *Lichtenberg* and *Lunze* [9]: they take a complementary (state-based) view and partition the state space of the continuous model. Each cell of the partition corresponds to a state of their discrete model. Their discrete approximation is also a nondeterministic automaton. In contrast to our abstractions, it may not be observable; increasing partition granularity may therefore not imply that "tighter" specifications can be met.

Beside the references given above, the following papers deal with closely related topics: *Pappas* and *Sastry* [19] consider the (mathematically much more intricate) problem of finding *continuous* abstractions of continuous dynamical systems. Interesting work on discrete abstractions of continuous systems has been published by *Caines* and *Wei* [5, 6]: they consider the lattice of dynamically consistent partition machines associated with a given continuous system.

Of related interest is also the work by *Niinomi*, *Krogh*, and *Cury* [18]: they design supervisory control on the basis of approximating automata. In contrast to our work, their discrete approximations are purely logical, i.e. timing information (apart from the temporal order of events) is *not* retained in the approximation. *Stursberg*, *Kowalewski*, and *Engell* [28] improve on that by retaining a notion of time in their approximation: they provide lower and upper bounds for residence times in discrete model states. The continuous "base" models in the latter papers are given in continuous time, hence the problem considered there is much more difficult to solve than the one addressed in the present paper.

Finally, it goes without saying that the way we interprete our results is almost entirely in terms of *J. C. Willems'* "behavioural systems theory" ([29] and the references therein). Behaviours (in *Willems'* sense) are a compact and illustrative way of thinking about a variety of dynamical systems and their abstractions.

The paper is organized as follows: in Section 2, the continuous plant model is introduced. Section 3 contains a brief review of the concept of behaviours of dynamical systems; it states – in terms of behaviours – the main requirements that any abstraction has to meet. Section 4 describes how to build discrete abstractions for the continuous model by first approximating its behaviour

and then generating realizations for the approximate behaviours. It is shown that the state equations of these discrete abstractions are in canonical form and minimal by construction. In Section 5, it is pointed out that the discrete abstractions can be interpreted as observers for the continuous "base" system. Finally, in Section 6, two examples are presented. The first example is purely academic, its sole purpose being to illustrate the main ideas in this paper. The second example deals with the synthesis of a discrete supervisory controller for the start-up procedure of a distillation column; it is much more demanding and is meant to demonstrate that the approach described in this paper can be used to solve nontrivial technical problems.

Our notation may seem a bit involved, but really is completely straightforward: we distinguish discrete-valued, or symbolic, signals from continuous-valued signals by using the subscript "d". Signals are represented by lower case letters, their codomains by the corresponding upper case letters. For example, $u_d : \{t_0, t_1, \dots\} \to U_d$ is the (discrete-valued) control signal, which is defined on the sampling grid $\{t_0, t_1, \dots\}$ and "lives" (takes values) in the (discrete) set U_d. The signal $w : \{t_0, t_1, \dots\} \to W$ is continuous-valued, its codomain W a dense subset of Euclidean space. The codomains of all discrete-valued signals are assumed to be finite, their elements (the possible values the signal can take at each sampling instant) are characterized by superscripts: the i-th element in the set U_d, for example, is denoted by $u_d^{(i)}$.

2 THE CONTINUOUS PLANT MODEL

The continuous "base" model of the plant is described by

$$x(t_{k+1}) = f(x(t_k), w(t_k), u_d(t_k)), \tag{1}$$
$$y_d(t_k) = q_y(x(t_k)), \tag{2}$$

where $k \in \{0, 1, 2, \dots\}$ is the time index, $x(t_k) \in \mathbb{R}^n$ the continuous state at time t_k, and $w(t_k) \in \mathbb{R}^r$ an unknown but bounded disturbance:

$$w(t_k) \in W := \{w \mid w \in \mathbb{R}^r,\ \|w\|_\infty \leq 1\}, \tag{3}$$

with $\|w\|_\infty := \max_i |w_i|$. $u_d(t_k) \in U_d$ and $y_d(t_k) \in Y_d$ are control input and measurement symbols, respectively. The sets U_d and Y_d are assumed to be finite:

$$U_d = \{u_d^{(1)}, \dots, u_d^{(\alpha)}\},$$
$$Y_d = \{y_d^{(1)}, \dots, y_d^{(\gamma)}\}.$$

$f : \mathbb{R}^n \times \mathbb{R}^w \times U_d \to \mathbb{R}^n$ is the state transition map, $q_y : \mathbb{R}^n \to Y_d$ the output map. Without loss of generality, the latter is required to be onto (if it were not, we could simply throw away the superfluous elements in Y_d).

Remark: It is straightforward to introduce a second output signal $z_d \in Z_d$ to capture design specifications which cannot be expressed in terms of inputs or measurement symbols – this is standard practice, for example, in H_∞-control theory. If safety were our only concern, we could choose an output map $q_z : \mathbb{R}^n \to Z_d$ with codomain $Z_d = \{\texttt{forbidden}, \texttt{legal}\}$. This would effectively declare part of the continuous state space a "no go area". In order to keep exposition as simple as possible, we do not treat this extension here. Details can be found in [24].

Remark: A second extension is also completely straightforward: suppose the plant "base" model is hybrid, i.e. it consists of a continuous part (modelled by a set of difference equations) interacting with a discrete part (modelled by a sequential automaton). Then, one simply generates a set of (ordered) discrete abstractions for the continuous part of the plant model; combining these abstractions with the discrete part of the plant model gives discrete abstractions for the overall hybrid "base" model. It has been shown in [24] that the ordering property is not affected by this operation. Again, for reasons of expositional simplicity, we do not explicitly treat this case. Details can be found in [24].

3 BEHAVIOURS

Let $T \subset \mathbb{R}$ be the chosen sampling grid, i.e. $T = \{t_0, t_1, \dots\}$. Denote the set of all functions from T into $(U_d \times Y_d)$ by $(U_d \times Y_d)^T$. Then, the behaviour $\mathcal{B}_c \subseteq (U_d \times Y_d)^T$ is the set of all pairs of control and measurement signals which are compatible with the the continuous model (1) – (3). For a survey on "behavioural" systems theory see [29].

Clearly, a *conditio sine qua non* for *any* discrete abstraction is that its behaviour $\mathcal{B}_l$ must contain the discrete-time behaviour $\mathcal{B}_c$ of the underlying continuous model:

$$\mathcal{B}_c \subseteq \mathcal{B}_l \tag{4}$$

implies that every sequence of input/measurement symbols that the continuous plant model can generate, can also be produced by the discrete approximation. If this condition were violated, the continuous system could respond to a given input signal with an unacceptable measurement signal which would not be predictable by the discrete approximation. Hence, this unacceptable phenomenon

could not be suppressed by a control strategy based on the discrete approximation – the approximation would be useless as far as control systems design is concerned. If the controller's job is to force the closed loop behaviour to be a subset of some specification behaviour, this "abstraction condition" is also sufficient for the purposes of control systems design. This is illustrated in Fig. 1: suppose we design a causal (discrete) feedback controller (with input signal y_d

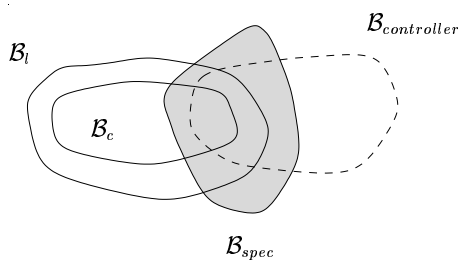

Fig. 1. Visualization of "abstraction condition".

and output signal u_d) and hook it up to the discrete abstraction. Obviously, the controller behaviour $\mathcal{B}_{controller}$ is also a subset of $(U_d \times Y_d)^T$; the feedback loop consisting of controller and abstraction exhibits behaviour $\mathcal{B}_l \cap \mathcal{B}_{controller}$ – only those pairs of input/output signals "survive" that are compatible with both abstraction and controller equations. From Fig. 1, it is immediately clear that the "abstraction condition" (4) implies:

$$\mathcal{B}_l \cap \mathcal{B}_{controller} \subseteq \mathcal{B}_{spec} \Longrightarrow \mathcal{B}_c \cap \mathcal{B}_{controller} \subseteq \mathcal{B}_{spec}; \tag{5}$$

in other words: if the controller forces the discrete approximation to obey the specifications, the feedback loop consisting of discrete controller and continuous "base" model will also meet the specifications[4].

It is worth while pointing out that "blocking" ($\mathcal{B}_c \cap \mathcal{B}_{controller} = \emptyset$) cannot occur within the proposed framework: as we work on a fixed sampling grid T, "blocking" would imply that connecting controller and continuous plant model would, literally, stop time. In mathematical terms, "blocking" cannot occur within our framework, as any feedback loop composed of a strictly causal system (the continuous plant model) and a causal system (the controller) has a solution trajectory on T.

If (4) holds, the "size" of the difference $\mathcal{B}_l \setminus \mathcal{B}_c$ is an indicator for the accuracy of the discrete approximation: the "smaller" $\mathcal{B}_l \setminus \mathcal{B}_c$, the smaller

[4]Invariance of the set inclusion (4) under feedback has been proven formally in [24].

the loss in "prediction power" when replacing the continuous model by its discrete abstraction. The trivial abstraction (characterized by the fact that, regardless of previous control inputs, every measurement symbol can occur at each sampling instant) 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).

4 DISCRETE ABSTRACTIONS

In this section, we will introduce suitable discrete abstractions for the continuous plant model (1) – (3). First, we will specify the behaviours that the abstractions are required to exhibit; then, in a second step, we will come up with minimal realizations for these behaviours.

4.1 Abstraction behaviours

In order to specify desired abstraction behaviours, we need a bit of additional notation (sorry): recall that $T = \{t_0, t_1, \dots\}$ is the sampling grid. Denote the intervals $\{t_0, \dots, t_k\}$ and $\{t_{k+1}, \dots\}$ 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 $\mathcal{B}_c^k \subseteq (U_d \times Y_d)^{T_k}$ to be the restriction of the behaviour $\mathcal{B}_c$ to the interval T_k:

$$\mathcal{B}_c^k := \left\{ b^k \in (U_d \times Y_d)^{T_k} \mid \exists b^{k+} \in (U_d \times Y_d)^{T_{k+}} \text{ such that } [b^k, b^{k+}] \in \mathcal{B}_c \right\}.$$

Finally, introduce the *predicted output set* $\mathcal{Y}_c(b^k)$ as the set of all measurement symbols that the continuous model (1) – (3) 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 :

$$\mathcal{B}_c^{k+1} = \left\{ \left[b^k, \left(y_d^{(j)}, u_d^{(i)} \right) \right] \;\middle|\; b^k \in \mathcal{B}_c^k, u_d^{(i)} \in U_d, y_d^{(j)} \in \mathcal{Y}_c(b^k) \right\}. \qquad (6)$$

Now we define less accurate prediction sets $\mathcal{Y}_l$, $l = 0, 1, \dots$, by using only "recent data". By "recent data" (denoted by $b^{k,l}$), we mean the restriction of b^k to the interval

$$T_{k,l} := \begin{cases} \{t_{k-l}, \dots, t_k\} & \text{if } k > l, \\ \{t_0, \dots, t_k\} & \text{if } k \leq l. \end{cases}$$

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

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

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

$$\mathcal{Y}_l(b^{k,l}) := \begin{cases} \bigcup_{b^{k-l-1}} \mathcal{Y}_c(b^k) \text{ if } k > l, \\ \mathcal{Y}_c(b^k) \text{ if } k \leq l, \end{cases}$$

i.e. $\mathcal{Y}_l(b^{k,l})$ 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-1}) (see Fig. 2 for an illustration). Therefore,

$$\mathcal{Y}_0(b^{k,0}) \supseteq \mathcal{Y}_1(b^{k,1}) \supseteq \ldots \mathcal{Y}_l(b^{k,l}) \ldots \supseteq \mathcal{Y}_c(b^k). \tag{7}$$

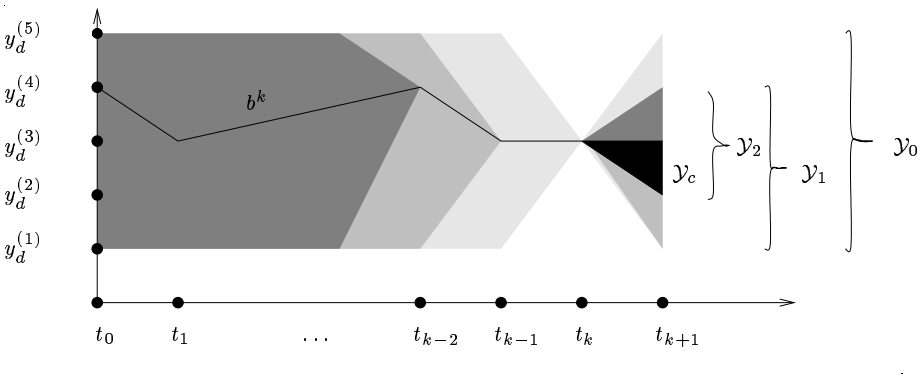

Fig. 2. Predicted output sets.

From the prediction sets $\mathcal{Y}_l$, the approximation behaviours $\mathcal{B}_l$ can be defined iteratively by

$$\mathcal{B}_l^0 := \mathcal{B}_c^0, \tag{8}$$

$$\mathcal{B}_l^{k+1} := \left\{ \left[[b^k, \left(y_d^{(j)}, u_d^{(i)} \right)] \;\middle|\; b^k \in \mathcal{B}_l^k, u_d^{(i)} \in U_d, y_d^{(j)} \in \mathcal{Y}_l(b^{k,l}) \right\}, \quad k = 0, 1, \ldots \tag{9}$$

From (6), (7), (8), and (9), it follows immediately that

$$\mathcal{B}_0 \supseteq \mathcal{B}_1 \supseteq \ldots \mathcal{B}_l \ldots \supseteq \mathcal{B}_c. \tag{10}$$

Now that we have specified the abstraction behaviours $\mathcal{B}_l$, $l = 0, 1, \ldots$, we only need to explain how to find discrete realizations (state models) that generate these behaviours, or, equivalently, the corresponding output prediction sets $\mathcal{Y}_l(b^{k,l})$.

4.2 Realization of abstraction behaviour $\mathcal{B}_l$

In this section we show how to build a discrete state model A_l with predicted output set $\mathcal{Y}_l(b^{k,l})$ and, consequently, behaviour $\mathcal{B}_l$. We first select the *state set* of A_l.

State set
Recall that, by definition, the state of any dynamical system summarizes all the information that, together with the current input, is needed to predict the future: Clearly, to compute the output prediction set $\mathcal{Y}_l(b^{k,l})$, we need to know $b^{k,l}$. Hence, we define the state x_d of the abstraction A_l at time t_k to consist of all the control and measurement symbols in the string $b^{k,l}$, with the exception of the current control 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:

$$x_d(t_k) := \begin{cases} [y_d(t_0)], & \text{if } k = 0, \\ ([y_d(t_0), \ldots, y_d(t_k)], [u_d(t_0), \ldots, u_d(t_{k-1})]), & \text{if } k = 1, \ldots, l, \\ ([y_d(t_{k-l}), \ldots, y_d(t_k)], [u_d(t_{k-l}), \ldots, u_d(t_{k-1})]), & \text{if } k > l. \end{cases} \tag{11}$$

This choice is reminiscent of observer canonical realizations in (continuous) control theory (e.g. [8]) – the state is made up of known quantities, and hence is trivially observable. As both U_d, the set of control symbols, and Y_d, the set of measurement symbols, are finite, the state set X_d of the discrete abstraction is finite. An upper bound for the number of elements of X_d is

$$\bar{N}_v = \sum_{i=0}^{l} \gamma^{i+1} \alpha^i ,$$

where $\gamma = |Y_d|$ and $\alpha = |U_d|$; this is the number of different strings (11) one gets by exhaustive permutation of measurement and control symbols.

It is obvious, however, that the continuous system (1) – (3) cannot generate *all* these strings. In the next step, one therefore eliminates all strings which are not compatible with the continuous model (1), (2), and the disturbance assumption (3): let $1 \leq \rho \leq l$ and denote $f(x, u_d^{(i)}, w)$ by $f_i(x, w)$. Then, $\left(\left[y_d^{(j_{k-\rho})}, \ldots, y_d^{(j_k)}\right], \left[u_d^{(i_{k-\rho})}, \ldots, u_d^{(i_{k-1})}\right]\right)$ is an element in the state set X_d of the discrete abstraction if and only if

$$y_d^{(j_k)} = q_y\left(f_{i_{k-1}}\left(f_{i_{k-2}}\left(\ldots f_{i_{k-\rho}}(\tilde{x}, w_{k-\rho}), \ldots, w_{k-2}\right) w_{k-1}\right)\right) \quad (12)$$

$$y_d^{(j_{k-1})} = q_y\left(f_{i_{k-2}}\left(\ldots f_{i_{k-\rho}}(\tilde{x}, w_{k-\rho}), \ldots, w_{k-2}\right)\right) \quad (13)$$

$$\vdots$$

$$y_d^{(j_{k-\rho+1})} = q_y\left(f_{i_{k-\rho}}(\tilde{x}, w_{k-\rho})\right) \quad (14)$$

$$y_d^{(j_{k-\rho})} = q_y(\tilde{x}) \quad (15)$$

subject to

$$\|w_{k-m}\|_\infty \leq 1, \quad m = 1, \ldots \rho, \quad (16)$$

has a non-empty solution set for $[\tilde{x}', w_{k-\rho}', \ldots, w_{k-1}']'$. This is a formal way of saying that there exists a continuous state variable $\tilde{x}$ and disturbances w_{k-m}, $m = 1, \ldots, \rho$, such that applying the string $\left[u_d^{(i_{k-\rho})}, \ldots, u_d^{(i_{k-1})}\right]$ of control symbols makes the continuous model respond with the string $\left[y_d^{(j_{k-\rho})}, \ldots, y_d^{(j_k)}\right]$ of measurement symbols. Checking this set of general nonlinear equations for the existence of solutions is of course non-trivial. For an important special case, however, this reduces to a numerically straightforward procedure, namely checking whether a set of *linear inequalities* possesses a non-empty solution set. This special case is characterized by the fact that the right hand side of (1) is affine in the state $x(t_k)$ and the unknown disturbance vector $w(t_k)$, and the measurement map has the form $q_y = Q_y \circ C_y$ (where $C_y : \mathbb{R}^n \to \mathbb{R}^p$ is a linear map, and the "quantizer" $Q_y : \mathbb{R}^p \to Y_d$ partitions $\mathbb{R}^p$ into finitely many rectangular boxes with edges parallel to the coordinate axes). The computational procedure for this case is described in [24].

This answers the question whether a string $\left(\left[y_d^{(j_{k-\rho})}, \ldots, y_d^{(j_k)}\right], \left[u_d^{(i_{k-\rho})}, \ldots, u_d^{(i_{k-1})}\right]\right)$ is an element in the state set of the discrete abstraction for $\rho = 1, \ldots, l$. What about the case $\rho = 0$, i.e. strings of the form $\left[y_d^{(j)}\right]$? As we do not assume any a-priori knowledge on the continuous system state, it could be anywhere in $\mathbb{R}^n$ at time t_0 (when we "start looking at the system"). Hence, any measurement symbol in Y_d can occur at time t_0 and $X_{d0} := Y_d$ is the *set of possible initial states* of the abstraction.

This finishes the construction of the abstraction state set X_d. The number of elements in X_d is denoted by N_v:

$$X_d := \{x_d^{(1)}, \ldots, x_d^{(N_v)}\}. \tag{17}$$

Up to now, we have only defined the state set of the abstraction A_l. We now describe the transition structure and show that the resulting state model is minimal.

State transition structure

Denote the strings of control and measurement symbols associated with a particular $x_d^{(i)}$ by $u_d^*(x_d^{(i)})$ and $y_d^*(x_d^{(i)})$, 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 $k \geq l$:

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

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

Now, writing down the transition structure of the discrete abstraction is easy – at least conceptually: $(x_d^{(i)}, u_d^{(j)}, x_d^{(k)})$ is a transition in A_l if and only if there exists a $y_d^{(m)} \in Y_d$ such that

1. $y_d^*(x_d^{(k)}) = \left[\mathcal{F}(y_d^*(x_d^{(i)})),\ y_d^{(m)}\right]$ and $u_d^*(x_d^{(k)}) = \left[\mathcal{F}(u_d^*(x_d^{(i)})),\ u_d^{(j)}\right]$,
2. $\left(\left[y_d^*(x_d^{(i)}),\ y_d^{(m)}\right], \left[u_d^*(x_d^{(i)}),\ u_d^{(j)}\right]\right)$ is compatible with the continuous system model (1) – (3).

The first condition can be verified by simple visual inspection of the abstraction states $x_d^{(i)}$ and $x_d^{(k)}$: do they "overlap" in the indicated manner? The second condition can be checked by setting up a set of equations as in (12) – (16) and testing it for the existence of solutions. Suppose both conditions hold, i.e. $(x_d^{(i)}, u_d^{(j)}, x_d^{(k)})$ turns out to be a transition in A_l. Then, $x_d^{(i)}$ and $x_d^{(k)}$ are called the exit state and the entrance state of the transition; the control 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 $y_d^*(x_d^{(i)})$. Hence, for each nonnegative integer l, we get a finite Moore automaton as a discrete abstraction for the continuous plant model. It is clear that for a given state $x_d^{(i)}$ and a given control symbol $u_d^{(j)}$, 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 abstraction

into more than one successor state (each carrying a different output symbol) – the resulting automaton is nondeterministic. This will be further illustrated in Section 5. By construction, the resulting state model is a realization of the output prediction set $\mathcal{Y}_l$.

Remark: In [21, 23, 24], we proposed slightly different abstractions $\tilde{A}_l$: both $\tilde{A}_l$ and A_l share the same state set, but the transition structure of $\tilde{A}_l$ is more conservative: $(x_d^{(i)}, u_d^{(j)}, x_d^{(k)})$ is declared to be a transition of $\tilde{A}_l$, if condition 1 holds. The omission of condition 2 gives less accurate approximation behaviours, but evidently does not affect their ordering. It can be argued that trimming the transition structure by adding condition 2 is a direct consequence of the input/output view adopted in the present paper: deriving the state model A_l as a realization of the output prediction set $\mathcal{Y}_l$ naturally leads to conditions 1 *and* 2.

Remark: It is understood that the discrete abstractions evolve on the same sampling grid as the underlying continuous system: the time needed for a transition is the sampling interval $t_{i+1} - t_i$. This explains why the notion of time is trivially retained at the abstraction level.

4.3 Minimality

It only remains to show that the proposed state model A_l is a *minimal* (i.e. observable and reachable) realization of $\mathcal{Y}_l$, hence $\mathcal{B}_l$.

As the abstraction state $x_d(t_k)$ is made up of known quantities (see equation (11)), A_l is guaranteed to be observable. It is also straightforward to prove that every element in X_d can be reached from an initial state $x_d \in X_{d0}$: reachability of an initial state itself is trivial; consider therefore any "non-initial" state

$$x_d^{(k)} = \left(\left[y_d^{(j_k-\rho)}, \ldots, y_d^{(j_k)}\right], \left[u_d^{(i_k-\rho)}, \ldots, u_d^{(i_k-1)}\right]\right) \in X_d \setminus X_{d0}, \quad 1 \leq \rho \leq l.$$

Then,

$$x_d^{(i)} = \begin{cases} \left(\left[y_d^{(j_k-1)}\right]\right) & \text{if } \rho = 1 \\ \left(\left[y_d^{(j_k-\rho)}, \ldots, y_d^{(j_k-1)}\right], \left[u_d^{(i_k-\rho)}, \ldots, u_d^{(i_k-2)}\right]\right) & \text{if } \rho = 2, \ldots, l \end{cases} \in X_d,$$

as solvability of (12) – (16) implies solvability of (13) – (16) (if a string of symbols constitutes an abstraction state, this is also true for its substrings). Hence, $\left(x_d^{(i)}, u_d^{(i_k-1)}, x_d^{(k)}\right)$ is a transition of the abstraction A_l, and $x_d^{(k)}$ can be reached from $x_d^{(i)}$. Repeating this procedure shows that every abstraction state can be reached from an initial state.

5 DISCRETE ABSTRACTIONS AS OBSERVERS FOR THE CONTINUOUS PLANT STATE

By construction, X_d, the state set of the discrete abstraction, contains only elements (i.e. strings of control and measurement symbols) which are compatible with the continuous model equations (1), (2), and assumption (3) regarding the unknown disturbance signal w. In other words, $x_d^{(i)} = \left(\left[y_d^{(j_{k-\rho})}, \dots, y_d^{(j_k)}\right], \left[u_d^{(i_{k-\rho})}, \dots, u_d^{(i_{k-1})}\right]\right)$ is an element in X_d if and only if there exists a non-empty solution set $\mathcal{S}(x_d^{(i)})$ for $[\tilde{x}', w'_{k-\rho}, \dots, w'_{k-1}]'$ in (12) – (16). Denote the projection of this solution set onto its first n components by $\widetilde{X}(x_d^{(i)})$. This is the set of continuous model states at time $t_{k-\rho}$, that allows the continuous model to respond with an output string $\left[y_d^{(j_{k-\rho})}, \dots, y_d^{(j_k)}\right]$ if an input string $\left[u_d^{(i_{k-\rho})}, \dots, u_d^{(i_{k-1})}\right]$ is applied. Hence, the set $\widetilde{X}(x_d^{(i)})$ can be interpreted as a *delayed estimate* of the continuous model state.

To estimate the *current state* of the continuous model, we only need to predict the set $\mathcal{S}(x_d^{(i)})$ by ρ time steps:

$$X(x_d^{(i)}) := \{x = f_{i_{k-1}}\left(f_{i_{k-2}}\left(\dots f_{i_{k-\rho}}(\tilde{x}, w_{k-\rho}), \dots, w_{k-2}\right) w_{k-1}\right) \mid [\tilde{x}', w'_{k-\rho}, \dots, w'_{k-1}]' \in \mathcal{S}(x_d^{(i)})\} \quad (18)$$

can be interpreted as *a set-valued estimate of the current state* $x(t_k)$ of the continuous system based on the observed data $u_d(t_{k-\rho}) = u_d^{(i_{k-\rho})}, \dots, u_d(t_{k-1}) = u_d^{(i_{k-1})}$ and $y_d(t_{k-\rho}) = y_d^{(j_{k-\rho})}, \dots, y_d(t_k) = y_d^{(j_k)}$. For the special case mentioned in the previous section (the right hand side of the continuous "base" model equation is affine in x and w, the output space is quantized in a "rectangular" fashion), the sets (18) are polyhedra in $\mathbb{R}^n$.

If we want a discrete abstraction to act as an observer for the continuous plant, we run both systems in parallel (Fig. 3); the continuous plant is fed with a string of control symbols and responds with a string of measurement symbols. Both u_d and y_d are provided as inputs for the discrete abstraction; these signals uniquely determine its current state $x_d(t_k) \in X_d$, and hence the set $X(x_d(t_k))$ – the estimate for $x(t_k)$.

Clearly, $\bigcup_{i=1}^{N_v} X(x_d^{(i)}) = \mathbb{R}^n$, and, in general, $X(x_d^{(i)}) \cap X(x_d^{(j)}) \neq \emptyset$. Hence, the sets $X(x_d^{(i)})$ form a *cover* of the continuous model state space. Intuitively, the smaller the sets $X(x_d^{(i)})$, the more accurate the discrete approximation. Obviously, a state set $X(x_d^{(i)})$ never increases (and, in general, decreases) in "size", if the control and measurement strings embodied in $x_d^{(i)}$ are extended

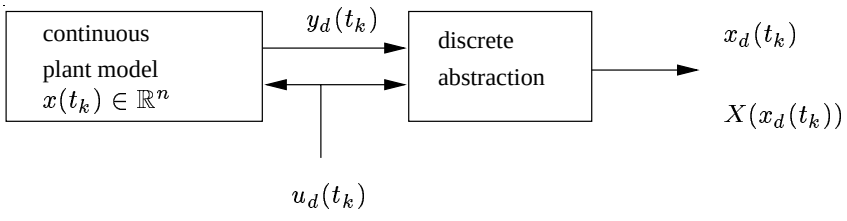

Fig. 3. Discrete abstraction acting as observer for the continuous plant.

in length – this is an immediate consequence of the "triangular" structure of (12) – (16). Increasing l, the maximum length of strings, is therefore equivalent to generating a finer "granularity" for the finite cover of $\mathbb{R}^n$ – the nonnegative integer l can be seen as a design parameter, which may be used to improve the accuracy of the discrete model. This of course implies that the number of states, and hence the complexity of the discrete model, also increases.

This mental picture is also helpful for understanding the intrinsic nondeterminism of discrete abstractions: if there were no unknown disturbances, a control input $u_d(t_k)$ would drive the state of the continuous plant model from $x(t_k)$ into a unique successor state $x(t_{k+1})$. A state $x_d(t_k) = x_d^{(i)}$ of the discrete abstraction, however, corresponds to a *set* $X(x_d^{(i)}) \subset \mathbb{R}^n$. An input $u_d(t_k) = u_d^{(j)}$ then maps this set onto another set. Only for very special continuous models[5] (1), (2), will the latter be contained in exactly one $X(x_d^{(k)})$ and not intersect any $X(x_d^{(m)})$, $m \neq k$, i.e. only in exceptional cases will the discrete state $x_d(t_k) = x_d^{(i)}$ have a unique successor state $x_d(t_{k+1}) = x_d^{(k)}$ under input $u_d^{(j)}$. Existence of unknown disturbances in the "base" model merely increases the "level of nondeterminism" in its discrete abstraction.

6 EXAMPLES

In this section, two examples are presented. The first one is a simple *Gedankenbeispiel*, meant for illustrational purposes only.

[5]Necessary and sufficient conditions in the autonomous, linear and time-invariant case $x(t_{k+1}) = Ax(t_k)$, $y(t_k) = x(t_k)$, $l = 0$, have been given in [11]. They imply that the matrix A has to be diagonal (up to permutation) and impose severe restrictions on its eigenvalues.

6.1 Simple water tank example

A water tank with cross sectional area $A = 100\text{cm}^2$ and height $\hat{x} = 30\text{cm}$ 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 from the tank at the same flow 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: $y_d(t) = E(mpty)$ 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. 4). After choosing a sampling interval of $t_{k+1} - t_k = 1\text{min}$, the following

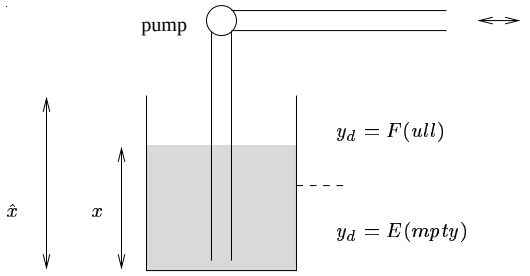

Fig. 4. Simple tank examle.

continuous-valued model can be easily written down:

$$x(t_{k+1}) = \begin{cases} x(t_k) + 10\text{cm} & \text{if } u_d(t_k) = \text{"+"} \text{ and } 0 \leq x(t_k) \leq 20\text{cm}, \\ 30\text{cm} & \text{if } u_d(t_k) = \text{"+"} \text{ and } 20\text{cm} < x(t_k) \leq 30\text{cm}, \\ x(t_k) - 10\text{cm} & \text{if } u_d(t_k) = \text{"−"} \text{ and } 10\text{cm} < x(t_k) \leq 30\text{cm}, \\ 0\text{cm} & \text{if } u_d(t_k) = \text{"−"} \text{ and } 0\text{cm} \leq x(t_k) \leq 10\text{cm}, \end{cases} \tag{19}$$

$$y_d(t_k) = \begin{cases} F & \text{if } 15\text{cm} < x(t_k) \leq 30\text{cm}, \\ E & \text{if } 0\text{cm} \leq x(t_k) \leq 15\text{cm}. \end{cases} \tag{20}$$

It is now straightforward to derive the coarsest abstraction A_0 by applying the procedure described in Section 4. The state set of 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. 5, where the output symbols associated with the states are indicated by dashed arrows. Clearly, $\mathcal{B}_c \subset \mathcal{B}_0$. Indeed, as expected, $\mathcal{B}_0$ contains pairs of control/measurement signals which do not make any physical sense and are therefore not contained in $\mathcal{B}_c$. For example, the abstraction A_0 deems it possible that the system responds with a string $FFFFF\ldots$ (i.e. the water level

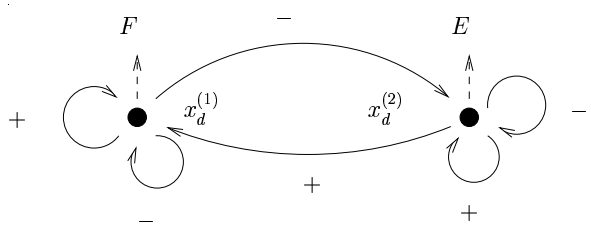

Fig. 5. Coarsest abstraction A_0.

remains above the 15cm-threshold), if an input string $----\ldots$ is applied (i.e. if water is perpetually removed from the tank).

A much more accurate discrete model is obatined by computing the abstraction A_1. Applying the procedure from Section 4 gives the following result. The state set X_d of A_1 contains eight elements:

$$\begin{array}{ll} x_d^{(1)} = ([F]) & x_d^{(2)} = ([E]) \\ x_d^{(3)} = ([FF],[+]) & x_d^{(4)} = ([FF],[-]) \\ x_d^{(5)} = ([EF],[+]) & x_d^{(6)} = ([FE],[-]) \\ x_d^{(7)} = ([EE],[+]) & x_d^{(8)} = ([EE],[-]); \end{array}$$

the initial state set X_{d0} consists of the two elements $x_d^{(1)}$ and $x_d^{(2)}$. The strings $([FE],[+])$ and $([EF],[-])$ are not compatible with the continuous system dynamics; they correspond to unreachable states and are therefore removed from the state set of the abstraction. The resulting transition structure is shown in Fig. 6. To avoid cluttering the diagram, the sets of states associated with the measurement symbols $F(ull)$ and $E(mpty)$ have been collected in two shaded "areas". States in the left area generate $F(ull)$ as measurement symbol, states in the right area $E(mpty)$. Clearly, $\mathcal{B}_c \subset \mathcal{B}_1 \subset \mathcal{B}_0$. For example, unlike A_0, the abstraction A_1 does not allow the string $FFFFF\ldots$ as a response to the input string $----\ldots$. In this particular case, the abstraction A_1 is actually extremely accurate: the restrictions of the behaviours $\mathcal{B}_c$ and $\mathcal{B}_1$ 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 A_1 is as good for output prediction and control purposes as the underlying continuous model (19), (20).

Finally, let A_1 act as an observer for the continuous model (19), (20). As explained in Section 5, the sequences of control and measurement symbols fed into and generated by the system (19), (20) can be used as inputs for the automaton. They "drive" A_1 through its state set X_d. Each element $x_d^{(j)} \in X_d$

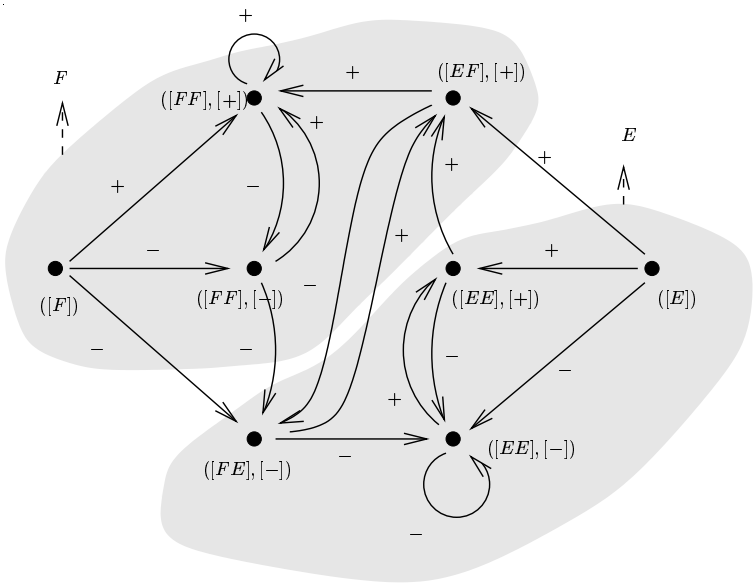

Fig. 6. Abstraction A_1.

corresponds to a set-valued estimate $X(x_d^{(j)})$ of the continuous state variable x:

$$X(x_d^{(1)}) = \{x|15\text{cm} < x \leq 30\text{cm}\} \qquad X(x_d^{(2)}) = \{x|0\text{cm} \leq x \leq 15\text{cm}\}$$
$$X(x_d^{(3)}) = \{x|25\text{cm} < x \leq 30\text{cm}\} \qquad X(x_d^{(4)}) = \{x|15\text{cm} < x \leq 20\text{cm}\}$$
$$X(x_d^{(5)}) = \{x|15\text{cm} < x \leq 25\text{cm}\} \qquad X(x_d^{(6)}) = \{x|5\text{cm} < x \leq 15\text{cm}\}$$
$$X(x_d^{(7)}) = \{x|10\text{cm} \leq x \leq 15\text{cm}\} \qquad X(x_d^{(8)}) = \{x|0\text{cm} \leq x \leq 5\text{cm}\}.$$

6.2 Automating the start-up procedure 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* and *A. Kienle*. Because of lack of space, it is only possible to give a short overview; details can be found in [14].

Distillation is one of the most important processes in the chemical industries. Its objective is to separate a mixture of chemical components. 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. 7). Its instrumentation satisfies industrial standards. Our application example is the separation of methanol and propanol.

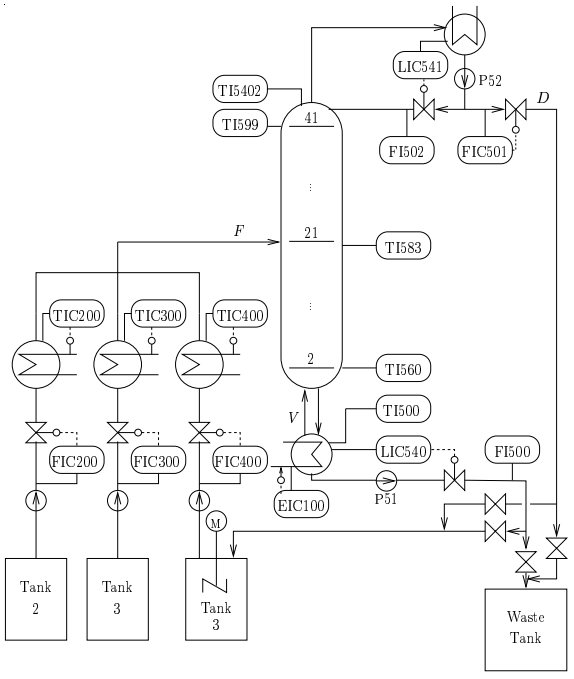

Fig. 7. Schematic representation of distillation column.

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. Thus, feed and product flows are switched off. At the end of this step, a single concentration front is established. The position of this front depends on the initial concentration distribution and typically varies from experiment to experiment. In a second step, the reflux, reboil, feed and product flows are switched to the desired steady-state values, and the initial front splits into two fronts, which usually move *very* slowly towards their desired steady state position. This is illustrated by the simulation results shown in Fig. 8; 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. Then, one switches to a conventional continuous control scheme for further adjustment and for disturbance rejection.

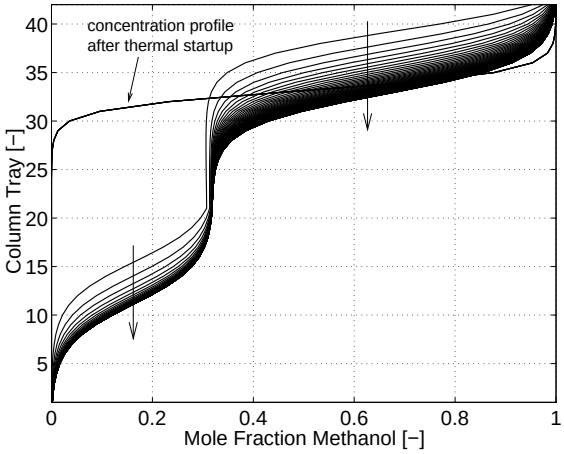

Fig. 8. Start-up without feedback control; time between shown profiles: $\Delta = 1h$.

In summary, start-up as described above is extremely slow, with the second step in the start-up procedure being the "main culprit" (Fig. 8). This is especially annoying in a research environment, where one usually wants to perform quite a few experiments within a limited time, and start-up has therefore to be repeated fairly frequently. Hence, we try to speed up the second step of the start-up procedure by introducing a suitable control strategy. To do this, we switch between discrete values of the control inputs and, during start-up, rely on highly quantized measurement information. Hence, our approach to start-up of a distillation column can be interpreted as a special hybrid control problem – the plant state "lives" in $\mathbb{R}^n$, whereas control inputs and measurement signals are discrete-valued, or symbolic.

As can be seen from Fig. 8, the front positions are crucial for the product compositions at the bottom and the top of the column. Furthermore, the front positions indicate reliably whether the plant is close enough to the desired operating point [13]. In the present case, start-up can be considered to be finished once the lower front of the concentration profile is between trays 8 and 12 and the upper front resides between trays 29 and 33. As the front positions can easily be determined from physical measurements, they can also be considered as (secondary) measurement signals. Furthermore, for the purposes of start-

up, it is sufficient to rely on a highly quantized version y_d of this measurement information. We have found it adequate to use five quantization intervals for each of the front positions (see Fig. 9, where s_r represents the front position in the rectifying section, and s_s the front position in the stripping section of the column). This gives $5 \times 5 = 25$ measurement symbols $y_d^{(1)} \ldots y_d^{(25)}$. For the case where one of the front positions is outside the area indicated in Fig. 9, an additional symbol $y_d^{(d)}$ is introduced ($y_d = y_d^{(d)}$ if $s_s < 4$ or $s_s > 16$ or $s_r < 25$ or $s_r > 37$), and $Y_d := \left\{ y_d^{(1)} \ldots y_d^{(25)}, y_d^{(d)} \right\}$.

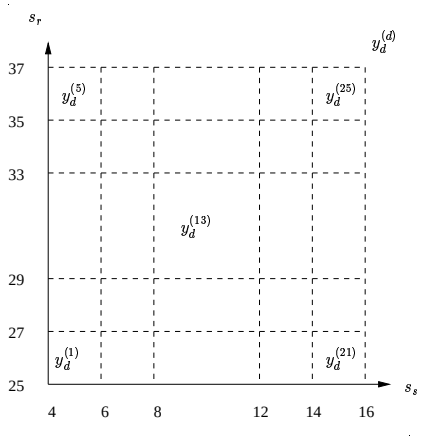

Fig. 9. Measurement quantization.

Ideally, one would want to manipulate the front positions and their propagation velocities w_s, w_r directly, in order to force the system as fast as possible towards the desired steady state. It is a well-known fact [16] that the propagation velocities are related to the ratios of the *internal* convective flow rates A_s and A_r of the vapour and the liquid phase in the rectifying (index 'r') and the stripping section (index 's'), respectively. A_s and A_r are easily adjusted by manipulating the *external* (distillate and vapour) flow rates D and V. The latter, in turn, is adjusted by manipulating the heating duty of the reboiler. Based on the relations between D, V and w_s, w_r, a number of discrete values of the external flow rates D and V have been determined such that the propagation velocity in the rectifying and the stripping section equals $\pm$ 3 trays per sampling interval (the sampling interval Δt is chosen to be 10 minutes). This procedure gives a set of nine control symbols $U_d := \left\{ u_d^{(1)}, \ldots, u_d^{(9)} \right\}$, in which the control signal "lives" during the second start-up step (see Table 1).

w_s	w_r	D [mol/h]	V [mol/h]	symbol
-3	-3	35.8070	188.2433	$u_d^{(1)}$
-3	0	59.3318	158.6412	$u_d^{(2)}$
-3	3	82.8566	129.0391	$u_d^{(3)}$
0	-3	46.8782	217.8455	$u_d^{(4)}$
0	0	70.4030	188.2433	$u_d^{(5)}$
0	3	93.9278	158.6412	$u_d^{(6)}$
3	-3	57.9494	247.4476	$u_d^{(7)}$
3	0	81.4742	217.8455	$u_d^{(8)}$
3	3	104.9990	188.2433	$u_d^{(9)}$

Table 1. Control symbols (w_s and w_r are given in [trays/10 min]).

We can now apply the approximation scheme from Section 4 to generate the "coarsest" element A_0 in our hierarchy of discrete abstractions. There is only one slight modification which needs to be taken into account: we work with a reduced set of measurement symbols $Y_d^{red} := \left\{y_d^{(1)}, \ldots, y_d^{(25)}\right\}$, hence the automaton A_0 will consist of 25 states. It therefore only describes the plant behaviour (in an approximate manner), if the front position s_s is between trays 4 and 16, and s_r is between trays 25 and 37. If this is not the case, we do not apply feedback control, but enter a predefined control symbol ($u_d^{(5)}$) that will eventually take s_s and s_r into the area where A_0 is a valid approximation of the underlying continuous plant model. Transitions between the abstraction states are also determined by the procedure in Section 4. It turns out that A_0 has 309 transitions.

We now need to formalize specifications: from the way we introduced our control symbols, it is reasonable to expect that start-up can be finished within 20 minutes (i.e. within two sampling intervals), once the front locations are within the area indicated by the measurement symbols $y_d^{(1)}, \ldots, y_d^{(25)}$. Recall that until this area is reached, we are not in a closed loop situation but apply a predefined control input. Reset time to t_0 once $y_d \in Y_d^{red}$; then, the closed loop specification behaviour is

$$\mathcal{B}_{spec} = \{b \in (U_d \times Y_d)^T \mid y_d(t_k) = y_d^{(13)}, k \geq 2\},$$

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 $y_d^{(13)}$, indicating that we require the front positions to be in the desired range from

the third sampling instant on. It is straightforward to realize this behaviour by an automaton A_{spec}.

It is now possible to apply a slightly modified[6] version of *Ramadge*'s and *Wonham*'s supervisory control theory [25, 26] to check whether there exists a supervisor that forces A_0 to obey the specifications. This is indeed the case, and a least restrictive supervisor is determined. It can be interpreted as another simple automaton that tracks the strings of measurement and control symbols and, at each sampling instant, disables all control symbols that might allow A_0 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. Nevertheless, for illustration purposes, we show a simulation result for the closed loop system consisting of continuous plant model and discrete controller (Fig. 10). Once thermal start-up is finished (i.e. after boiling point conditions have been established throughout the column), the first available measurement information is $y_d = y_d^{(d)}$. This implies that the automaton model is not yet a reliable approximation of the continuous plant model and supervisory control based on A_0 cannot yet be applied. Hence, $u_d = u_d^{(5)}$ is chosen as a constant control input until a measurement symbol in Y_d^{red} occurs. This happens after 4 hours (Fig. 10). Now, supervisory feedback

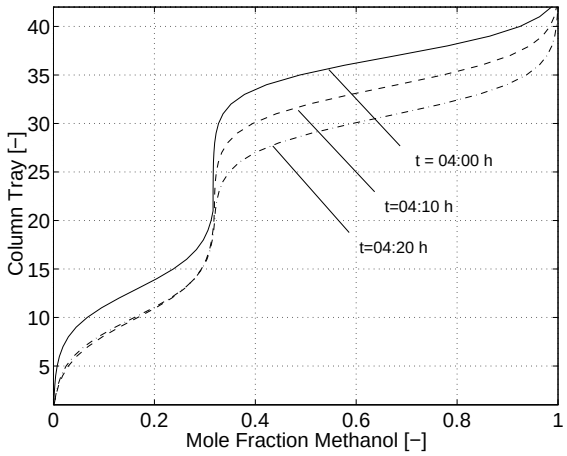

Fig. 10. Start-up under (discrete) control.

[6]The modification accounts for nondeterminism and time in the abstracted plant transition structure. It has been described in [22].

control is switched on and, as predicted, the controlled detailed plant model satisfies the specifications, i.e start-up is finished within another 20 minutes. Clearly, this represents an enormous improvement compared to the open-loop behaviour shown in Fig. 8.

7 CONCLUSIONS

We have proposed a hierarchy of discrete abstractions, or approximations, for a given continuous plant model. Our approach has been motivated by the problem of designing discrete feedback controllers for continuous (or hybrid) plants. By adopting a strict input/output point of view, we have been able to improve previous results and increase approximation accuracy without changing abstraction state sets. The first step in the suggested procedure is to construct a sequence of abstraction behaviours which is strictly ordered in terms of set inclusion or, equivalently, in terms of approximation accuracy. All abstraction behaviours are also required to cover the continuous model behaviour. In a second step, we generate non-deterministic Moore automata A_l, $l = 0, 1, \ldots$, as minimal realizations for these behaviours. Because of the ordering on the set $\{A_0, A_1, \ldots\}$, design of discrete feedback controllers for the underlying continuous plant becomes straightforward: using methods from Discrete Event Systems theory, one can search the hierarchy of discrete abstractions from the top for the "coarsest" abstraction A_i that allows a given set of specifications to be met. Any controller that "does the job" on the level of abstraction A_i will also "work properly" (enforce the specifications) when hooked up to a more accurate approximation A_j, $j > i$, or the continuous plant model. The actual controller synthesis problem on the (discrete) abstraction level has not been treated here. The reader is referred to [22] and [17] for a synthesis approach which is "tailor-made" for this kind of problem.

Two examples have been described: the first one is purely academic and only serves to illustrate the main ideas in this paper. The second example addresses the problem of controller synthesis for the start-up procedure of a distillation column. Another application example, controller synthesis for a batch evaporation benchmark problem, has been reported in [15].

ACKNOWLEDGEMENTS

The author wishes to thank *S. O'Young*, *T. Moor* and *J. Lunze* for many helpful discussions, and *E. Klein* and *A. Kienle* for their collaboration in the distillation column project. Support from Deutsche Forschungsgemeinschaft under Grant Ra 516/3-1 and through "Sonderforschungsbereich" SFB 412 is gratefully acknowledged.

REFERENCES

1. Alur, R., Henzinger, T. A., Sontag, E. D. (editors). *Hybrid Systems III*, Lecture Notes in Computer Science, Vol. 1066. Springer-Verlag, 1996.
2. Antsaklis, P., Stiver, J. A., Lemmon, M.: Hybrid system modelling and autonomous control systems. In [7], pages 366-392.
3. Antsaklis, P., Kohn, W., Nerode, A., Sastry, S. (editors): *Hybrid Systems II*, Lecture Notes in Computer Science, Vol. 999. Springer-Verlag, 1995.
4. Antsaklis, P., Kohn, W., Nerode, A., Sastry, S. (editors): *Hybrid Systems IV*, Lecture Notes in Computer Science, Vol. 1273. Springer-Verlag, 1997.
5. Caines, P. E., Wei, Y.-J.: On dynamically consistent hybrid systems. In [3].
6. Caines, P. E., Wei, Y.-J.: Hierarchical hybrid control systems: a lattice theoretic formulation. *IEEE Transactions on Automatic Control*, 43:501–508, April 1998, Special Issue on Hybrid Systems.
7. Grossman, R. L., Nerode, A., Ravn, A. P., Rischel, H. (editors): *Hybrid Systems*, Lecture Notes in Computer Science, Vol. 736. Springer-Verlag, 1993.
8. Kailath, T.: *Linear Systems*. Prentice-Hall, 1980.
9. Lichtenberg, G., Lunze J.: Observation of qualitative states by means of a qualitative model. *International Journal of Control*, 66:885–903, 1997.
10. Lunze, J.: Ein Ansatz zur qualitativen Modellierung und Regelung dynamischer Systeme. *at – Automatisierungstechnik*, 41:451–460, 1993.
11. Lunze J.: Qualitative modelling of linear dynamical systems with quantized state measurements. *Automatica*, 30:417–431, 1994.
12. Lunze J.: Stabilization of nonlinear systems by qualitative feedback controllers. *International Journal of Control*, 62:109–128, 1995.
13. Kienle, A.: Reduced models for multicomponent separation processes using nonlinear wave propagation theory. Proc. CHISA'98, Prague, 1998.
14. Klein, E., Kienle, A., Raisch, J.: Synthesizing a Supervisory Control Scheme for the Start-up Procedure of a Distillation Column – an Approach based on Approximating Continuous Dynamics by DES Models. Proc. LSS'98 - 8th IFAC Colloquium on Large Scale Systems, Patras, pp. 716–721, 1998.
15. Klein, E., Raisch, J.: Safety Enforcement in Process Control Systems - A Batch Evaporator Example. Proc. WODES'98 - International Workshop on Discrete Event Systems, Cagliari, Italy, pp. 327–333, 1998. IEE.
16. Marquardt, W.: Nichtlineare Wellenausbreitung – ein Weg zu reduzierten Modellen von Stofftrennprozessen, *VDI Fortschritt–Berichte* Nr. 8/161, VDI-Verlag, 1988.
17. Moor, T., Raisch, J., O'Young, S. D.: Supervisory Control of Hybrid Systems via l-Complete Approximations. Proc. WODES'98 - International Workshop on Discrete Event Systems, Cagliari, Italy, pp. 426–431, 1998. IEE.
18. Niinomi, T., Krogh, B. H., Cury, J. E. R.: Synthesis of Supervisory Controllers for Hybrid Systems Based on Approximating Automata. Proc. 34th IEEE Conference on Decision and Control, pp. 1461–1466, 1995.
19. Pappas, G.J., Sastry, S.: Towards continuous abstractions of dynamical and control systems. In [4], pp. 329–341.
20. Raisch, J., O'Young, S. D.: A DES approach to control of hybrid dynamical systems. In [1], pp. 563–574.
21. Raisch, J., O'Young, S. D.: Time-driven supervisory control of hybrid dynamical systems. Proc. 5th International Conference on CONTROL'96, Exeter, UK , pp. 716-721, 1996. IEE.
22. Raisch, J., O'Young, S. D.: Discrete approximation and supervisory control of continuous systems. *IEEE Transactions on Automatic Control*, 43(4):569–573, April 1998. Special Issue on Hybrid Systems.

23. Raisch, J.: Nondeterministic automata as approximations for continuous systems – an approach with an adjustable degree of accuracy. Proc. 2nd MATHMOD (International Symposium on Mathematical Modelling), Vienna, Austria, pp. 195–202, 1997. IMACS.
24. Raisch, J., O'Young, S. D.: A Totally Ordered Set of Discrete Abstractions for a given Hybrid or Continuous System. In [4], pp. 342–360.
25. Ramadge, P. J., Wonham, W. M.: Supervisory control of a class of discrete event systems. *SIAM J. Control and Optimization*, 25:206–230 1987.
26. Ramadge, P. J., Wonham, W. M.: The Control of Discrete Event Systems. *Proc. of the IEEE*, 77(1):81–98, 1989.
27. Stiver, J. A., Antsaklis, P.: Modeling and analysis of hybrid control systems. Proc. 31st IEEE Conference on Decision and Control, pp. 3748–3751. 1992.
28. Stursberg, O., Kowalewski, S., Engell, S.: Generating timed discrete models of continuous systems. Proc. 2nd MATHMOD (International Symposium on Mathematical Modelling), Vienna, Austria, pp. 203–209, 1997. IMACS.
29. Willems, J. C.: Paradigms and puzzles in the theory of dynamical systems. *IEEE Transactions on Automatic Control*, 36:259–294, 1991.