MODEL-BASED PROCESS CONTROL VIA FINITE MARKOV CHAINS

Enso Ikonen *

* *Systems Engineering Laboratory, Department of Process and Environmental Engineering, FIN-90014 University of Oulu, Finland*

Abstract: Predictive and optimal process control using finite Markov chains is considered. A basic procedure is outlined, consisting of space discretization; model conversion; specification of costs; computation of control policy; and, analysis of the closed-loop system behavior. A simulation illustrates the fiability of the approach using a standard office PC. Discussion of nonlinear process control emphasizing in on-line learning from uncertain data ends the paper.

Keywords: Markov decision process, generalized cell-to-cell mapping

1. INTRODUCTION

Many engineering problems can be modelled as controlled Markov chains (Puterman, 1994) (Häggström, 2002) (Poznyak *et al.*, 2000). The basic idea is simple. The system state space is quantized (discretized, partitioned, granulated) into a finite set of states (cells), and the evolution of system state in time is mapped in a probabilistic (frequentist) manner. With controlled Markov chains, the mappings are constructed from each state-action pair.

Once equipped with such a model, a control action for each state can be deduced by minimizing a cost function defined in a future horizon, based on specification of immediate costs for each state-action pair. Specification of immediate costs allows versatile means for characterising the desired control behavior. Dynamic programming, studied in the field of Markov decision processes (MDP), offers a way to solve various types of expected costs. However, applications of MDP in process control have been few, as pointed out in (Lee and Lee, 2004) (see also (Ikonen, 2004) (Negenborn *et al.*, 2005)); instead, the model predictive control paradigm is very popular in the process control community. Whereas not-so-many years ago the computations associated with finite Markov chains were prohibitive, the computing power available using cheap office-pc's motivates a re-exploration of these techniques.

As the basic ideas are old (Riordon, 1969), well-known, and widespread, relevant literature can be found from many fields and under different keywords: generalized cell-to-cell mapping (Hsu, 1987), qualitative modelling (Lunze *et al.*, 2001), and reinforcement learning (Kaelbling *et al.*, 1996), for example. A large part of the current literature on MDP examines means to overcome the problem curse-of-dimensionality, e.g., by means of function approximation (neuro-dynamic programming, Q-learning, etc.). The main obstacle in such approaches is in that the unknown properties introduced by the mechanisms of function approximation make void the fundamental benefit of applying finite Markov chains: a straightforward and elegant means to describe and analyse the dynamic characteristics of a stochastic nonlinear system. Recall how linear systems are limited by the extremely severe assumption of linearity (affinity), yet they have turned out to be outmost useful for control design purposes. In

a similar way, the finite Markov chains are fundamentally limited by the resolution of the problem presentation (discretization of state-action spaces). The hypothesis of this work is that keeping in mind this restriction (just as we keep in mind the assumption of linearity) the obtained results can be most useful. The practical validity of this statement is in the focus of the research.

The work described in this paper aims at building a proper basic framework for examining the possibilities of controlled finite Markov chains in nonlinear process control. In particular, the field of process engineering is in our main concern, with applications characterized by: availability of rough process models, slow sampling rates, nonlinearities that are either smooth or appear as discontinuities, expensive experimentation (large-scale systems running in production), and substantial on-site tuning due to uniqueness of products. Clearly, this type of requirements differ from those encountered, e.g., in the field of economics (lack of reliable models), robotics (very precise models are available), consumer electronics (mass production of low cost products), telecommunication (extensive use of test signals, fast sampling), or academic toy problems (ridiculously complex multimodal test functions).

Due to systematical errors, noise, and lack of accuracy in measurements of process state variables, among many other reasons, there is a urgent need for extended means of learning and handling of uncertainties. The finite Markov chains provide straightforward means for dealing with both of these issues.

This paper is organized as follows: The process models are considered in section 2, control design in section 3, open and closed loop system analysis in section 4. The MATLAB toolbox, and an illustrative example is provided in section 5. Discussion on aspects relevant to learning under uncertainties, and conclusions, are given in the final section.

2. GENERALIZED CELL MAPPING

Let the process under study be described by the following discrete-time dynamic system and measurement equations

$$\mathbf{x}(k) = f(\mathbf{x}(k-1), \mathbf{u}(k-1), \mathbf{w}(k-1)) \quad (1)$$

$$\mathbf{y}(k) = h(\mathbf{x}(k), \mathbf{v}(k)) \quad (2)$$

where $f : \Re^{n_\mathrm{x}} \times \Re^{n_\mathrm{u}} \times \Re^{n_\mathrm{w}} \to \Re^{n_\mathrm{x}}$ and $h : \Re^{n_\mathrm{x}} \times \Re^{n_\mathrm{v}} \to \Re^{n_\mathrm{y}}$ are nonlinear functions, $w_k \in \Re^{n_\mathrm{w}}$ and $v_k \in \Re^{n_\mathrm{v}}$ are i.i.d. white noise with probability density functions (pdf's) p_w and p_v, respectively. The initial condition is known via $p_\mathrm{X}(0)$.

Let the state space be partitioned into a finite number of sets called state cells, indexed by $s \in \mathcal{S} = \{1, 2, ..., S\}$. The index s is determined from

$$s = \arg\min_{s \in \mathcal{S}} \left\| \mathbf{x} - \mathbf{x}_s^{\mathrm{ref}} \right\|$$

where $\mathbf{x}_s^{\mathrm{ref}}$ are reference points (e.g., cell centers). In addition, let us define a 'sink cell', s_{sink}; a state is categorized into a sink cell if $\min_{s \in \mathcal{S}} \left\| \mathbf{x} - \mathbf{x}_s^{\mathrm{ref}} \right\| > \mathbf{x}^{\mathrm{lim}}$. Similarly, let the control action and measurement spaces be partitioned into cells indexed by $a \in \mathcal{A} = \{1, 2, ..., A\}$ and $m \in \mathcal{M} = \{1, 2, ..., M\}$, respectively, and determined using reference vectors $\mathbf{u}_a^{\mathrm{ref}}$ and $\mathbf{y}_m^{\mathrm{ref}}$. The partitioning results in $\mathcal{X} = \cup_{s=1}^{S} \mathcal{X}_s$, $\mathcal{U} = \cup_{a=1}^{A} \mathcal{U}_a$ and $\mathcal{Y} = \cup_{m=1}^{M} \mathcal{Y}_m$.

The evolution of the system can now be approximated as a finite state controlled Markov chain over the cell space (however, see (Lunze, 1998)). In simple cell mapping (SCM), one trajectory is computed for each cell. Generalized cell mapping (GCM) considers multiple trajectories starting from within each cell, and can be interpreted in a probabilistic sense as a finite Markov chain.

2.1 Evolution of states

Let the state pdf be approximated as a $S \times 1$ cell probability vector $\mathbf{p}_\mathrm{X}(k) = [p_{\mathrm{X},s}(k)]$ where $p_{\mathrm{X},s}(k)$ is the cell probability mass. The evolution of cell probability vectors is described by a Markov chain represented by a set of linear equations

$$\mathbf{p}_\mathrm{X}(k+1) = \mathbf{P}^{a(k)} \mathbf{p}_\mathrm{X}(k)$$

or, equivalently, $p_{\mathrm{X},s'}(k+1) = \sum_{s \in \mathcal{S}} p^a_{s',s} p_{\mathrm{X},s}(k)$, where $\mathbf{P}^a$ is the transition probability matrix under action a, $\mathbf{P}^a = \left[p^a_{s',s}\right]$, $p^a_{s',s} = \int_{\mathcal{X}_s} p(\mathbf{x}(k+1) \in \mathcal{X}_{s'} \mid \mathbf{x}(k) \in \mathcal{X}_s, u(k) \in \mathcal{U}_a)\, d\mathbf{x}$.

The likelihood of obtaining a measurement cell m, when the system state cell is s, is given by the likelihood matrix $\mathbf{L}$, $\mathbf{L} = [l_{m,s}]$(Ungarala and Chen, 2003), $l_{m,s} = \int_{\mathcal{Y}_m} p(\mathbf{y} \in \mathcal{Y}_m \mid \mathbf{x} \in \mathcal{X}_s) d\mathbf{y}$. Let a row in the likelihood matrix be denoted as a likelihood vector $\mathbf{l}_m$. Given the current likelihood vector and the previous posterior probability vector $\mathbf{p}_\mathrm{X}(k-1)$, a Bayesian estimate of the cell probability can be constructed,

$$\mathbf{p}_\mathrm{X}(k) \propto \mathbf{l}_m \otimes \mathbf{P}^{a(k)} \mathbf{p}_\mathrm{X}(k-1),$$

where $\otimes$ denotes component-wise multiplication.

3. CONTROL DESIGN

Using a GCM model of the plant, an optimal control action for each state can be solved by minimizing a cost function. In both optimal (Kaelbling

et al., 1996) and predictive control (Ikonen and Najim, 2002) the cost function is defined in a future horizon, based on specification of immediate costs for each state-action pair. Whereas optimal control considers (discounted) infinite horizons and solves the problem using dynamic programming, nonlinear predictive control approaches rely on computation of future trajectories (predictions) and exhaustive search.

3.1 Optimal control

In optimal control, the control task is to find an appropriate mapping (optimal policy or control table) π from states ($\mathbf{x}$) to control actions ($\mathbf{u}$), given the immediate costs $r(\mathbf{x}(k), \mathbf{u}(k))$. The infinite-horizon discounted model attempts to minimize the geometrically discounted immediate costs

$$J(\mathbf{x}_0) = \sum\nolimits_{k=0}^{\infty} \gamma^k r(\mathbf{x}(k), \pi(\mathbf{x}(k)))$$

under initial conditions $\mathbf{x}(0) = \mathbf{x}_0$. The optimal control policy π^* is the one that minimizes J. The optimal cost-to-go is given by $J^* = \min_\pi J$.

Bellman's principle of optimality states that

$$J^*(\mathbf{x}_0) = \min_{\mathbf{u}} [r(\mathbf{x}_0, \mathbf{u}) + \gamma J^*(f(\mathbf{x}_0, \mathbf{u}))]$$

i.e., the optimal solution (value) for state $\mathbf{x}$ is the sum of immediate costs r and the optimal cost-to-go from the next state, $J^*(f(\mathbf{x}_0, \mathbf{u}))$. Application of the Bellman equation leads to methods of dynamic programming.

3.1.1. Value iteration In value iteration, the optimal value function is determined by a simple iterative algorithm derived directly from the Bellman equation. Let the immediate costs be given in matrix $\mathbf{R} = [\mathbf{r}^a]$, with column vectors $\mathbf{r}^a = [r_s^a]$, and collect the values of the cost-to-go at iteration i into a vector $\mathbf{J}^*(i) = [J_s^*(i)]$. Given arbitrary initial values $J_s^*(0)$, the costs are updated for $i = 0, 1, 2, ...$:

$$\begin{aligned} Q_s^a(i) &= r_s^a + \gamma \sum_{s' \in \mathcal{S}} p_{s',s}^a J_{s'}^*(i) \\ J_s^*(i+1) &= \min_{a \in \mathcal{A}} Q_s^a(i) \end{aligned}$$

$\forall s, a$, until the values of $J_s^*(i)$ converge. Denote the converged values by J_s^*. The optimal policy is then obtained from

$$\pi_s^* = \arg\min_{a \in \mathcal{A}} \left[r_s^a + \gamma \sum\nolimits_{s' \in \mathcal{S}} p_{s',s}^a J_{s'}^* \right].$$

3.2 Predictive control

Given a system model and the associated costs, we can easily set up a predictive control type of a problem. In predictive control, the costs are minimized in an open loop in a fixed horizon

$$\begin{aligned} &J(\mathbf{x}(k), ..., \mathbf{x}(k+H_\mathrm{p}), \mathbf{u}(k), ..., \mathbf{u}(k+H_\mathrm{p})) \\ &= \sum\nolimits_{h=0}^{H_\mathrm{p}} r(\mathbf{x}(k+h), \mathbf{u}(k+h)) \end{aligned}$$

under initial conditions $\mathbf{x}(k)$. In practice it is useful to introduce a control horizon, where it is assumed that the control action will remain fixed after a given number of steps, H_c.

Often only one step is allowed and the optimization problem reduces to the minimization of $J(\mathbf{x}(k), ..., \mathbf{x}(k+H_\mathrm{p}), \mathbf{u}(k))$. Under control action a, the costs are given by

$$\begin{aligned} J_a &= \sum\nolimits_{h=0}^{H_\mathrm{p}} [\mathbf{r}^a]^T \mathbf{p}_\mathrm{X}(k+h) \\ &= \sum\nolimits_{h=0}^{H_\mathrm{p}} [\mathbf{r}^a]^T [\mathbf{P}^a]^h \mathbf{p}_\mathrm{X}(k) \end{aligned}$$

where $\mathbf{r}^a = [r_s^a]$ is a column vector of immediate costs and $\mathbf{p}_\mathrm{X}(k)$ is current state cell pdf. In order to solve the problem, it suffices to evaluate the costs for all $a \in \mathcal{A}$ and select the one minimizing the costs. The prediction horizon H_p is a useful tuning parameter; a long prediction horizon leads to mean level type of control.

The control policy mapping $\pi^\diamond$ can be obtained by solving the above problem in each state s and tabulating the results:

$$\pi_s^\diamond = \arg\min_a J_a.$$

3.2.1. Control horizons greater than one For many practical cases, a good controller design can be obtained using either the optimal control approach, or the predictive control approach with $H_\mathrm{c} = 1$. Whereas the optimal control tends to result in "agressive" control actions in terms of the plant input (even if optimal in terms of the cost function), the predictive control approach provides a variety of responses as a function of the prediction horizon, H_p. With a small H_p, an agressive control is obtained. A large H_p results in mean level control where the closed loop shares the open loop plant dynamics. In some cases, however, an engineer may be interested in extending the controller design possibilities to larger control horizons. In principle, this is straightforward to realize in the GCM context: One simply creates A different sequences of control actions, simulates the system accordingly, and selects the sequence that minimizes the cost function.

With large H_c, the search space of the exhaustive search can become too large for practical

purposes, however. Luckily, in process engineering one is commonly interested in control sequences which fulfill rate constraints, one prefers to avoid jiggering, etc. With some simple rules, the set of appropriate control sequences can be reduced to a manageable size. Notice, however, that if sufficient information concerning the constraints is not contained in the cost description ($\mathbf{R}$), the optimal control policy (table) can not be constructed offline. If the optimal control action is solved online, one is free to select the potential control sequencies based on any available information.

4. SYSTEM ANALYSIS

The generalized cell-to-cell mapping is a powerful tool for analysis of nonlinear systems. In what follows, it is assumed that the system map (Markov chain) is described by transition probabilities $\mathbf{P}$. This may correspond to the process output under a fixed (open loop) control action a ($\mathbf{P} := \mathbf{P}^a$) or the systems closed loop behavior obtained from the construction of transition probabilities under $\mathbf{u} = \pi(\mathbf{x})$: $\mathbf{P}^\pi = \left[p^\pi_{s',s}\right]$, where $p^\pi_{s',s} = \int_{\mathcal{X}_s} p(\mathbf{x}(k+1) \in \mathcal{X}_{s'} \mid \mathbf{x}(k) \in \mathcal{X}_s, u(k) = \pi^*_s) d\mathbf{x}$.

An useful characterization of cells is obtained by studying the long term behavior of the Markov chain (Najim *et al.*, 2004). Decomposing the probability vector into recurrent cells ($i_\mathrm{r} \in \mathcal{I}_\mathrm{r}$) and transient cells ($i_\mathrm{t} \in \mathcal{I}_\mathrm{t}$), the Markov chain can be written as

$$\begin{bmatrix} \mathbf{p}_\mathrm{r}(k+1) \\ \mathbf{p}_\mathrm{t}(k+1) \end{bmatrix} = \begin{bmatrix} \mathbf{P}_\mathrm{rr} & \mathbf{P}_\mathrm{rt} \\ \mathbf{0} & \mathbf{P}_\mathrm{tt} \end{bmatrix} \begin{bmatrix} \mathbf{p}_\mathrm{r}(k) \\ \mathbf{p}_\mathrm{t}(k) \end{bmatrix}$$

The recurrent cells form communicating classes (closed subsets), where the cells within each communicating class (inter)communicate with each other, i.e., the probability of transition from one state to the other is nonzero, and do not communicate with other states. Each absorbing state only communicates with itself. A closed communicating class constitutes a sub-Markov chain, which can be studied separately.

A stationary probability distribution satisfies $\overline{\mathbf{p}}_\mathrm{X} = \mathbf{P}\overline{\mathbf{p}}_\mathrm{X}$ and, consequently, the distribution must be an eigenvector of $\mathbf{P}$; for the distribution to be a probability distribution, the eivenvalue must be one. Therefore, the recurrent cells are found by searching for the unit amplitude eigenvalues of $\mathbf{P}$; the nonzero elements of the associated eigenvectors $\overline{\mathbf{p}}_\mathrm{X}$ point to the recurrent cells.

Examination of the behavior of transient cells as they enter the recurrent cells reveals the dynamics of the nonlinear system. We have that $\mathbf{p}_\mathrm{r}(k+1) = \mathbf{P}_\mathrm{rr}\mathbf{p}_\mathrm{r}(k) + \mathbf{P}_\mathrm{rt}\mathbf{P}^k_\mathrm{tt}\mathbf{p}_\mathrm{t}(0)$, where $\mathbf{P}_\mathrm{rt}\mathbf{P}^k_\mathrm{tt}$ represents the conditional probability that a solution starting from a transient cell will pass into an recurrent cell at time $k+1$. The probability that this will eventually happen, $\mathbf{P}_\mathrm{t2r}$, is given by

$$\mathbf{P}_\mathrm{t2r} = \sum\nolimits_{k=0}^{\infty} \mathbf{P}_\mathrm{rt}\mathbf{P}^k_\mathrm{tt} = \mathbf{P}_\mathrm{rt}\left(1 - \mathbf{P}_\mathrm{tt}\right)^{-1}$$

Each recurrent cell belongs to a communicating class, for absorbing cells this class consists of a single cell. The probability of transition into a particular communicating class is obtained by summing (column-wise) the entries in $\mathbf{P}_\mathrm{t2r}$.

The sink cell (Hsu, 1987) is an absorbing cell that represents the entire region outside the domain of interest. A nonzero probability to enter the sink cell indicates unstability of the system (given the resolution of the model). In the experimental section, the stationary probabilities of entering the sink cell are examined.

High probability cells determine the basin-of-attraction. The 'size of the basin-of-attraction' for each state was characterized by taking the sum of probabilities for entering a recurrent cell (from any transient cell, or from any recurrent cell) and weighting based on probability of occurance within a communicating class (i.e., multiplying this with the stationary mapping $\mathbf{P}_\infty$):

$$\mathbf{B} = \mathbf{P}_\infty \left[\sum_{i \in \mathcal{I}_\mathrm{t}} \left[\mathbf{P}_\mathrm{t2r}\right]_{j,i} + \sum_{i \in \mathcal{I}_\mathrm{r}} \left[\mathbf{P}_\mathrm{rr}\right]_{j,i} \right]$$

where $\mathbf{P}_\infty$ is a mapping to stationary distribution: $\mathbf{P}_\infty = \lim_{n\to\infty} \frac{1}{n} \sum \mathbf{P}^n_\mathrm{rr}$, and $[\mathbf{x}]_{a,b}$ denotes an element of $\mathbf{x}$ in a'th row and b'th column. Elements of $\mathbf{B}$ take values in the interval $[0, S]$, $\sum_{i \in \mathcal{I}_\mathrm{r}} [\mathbf{B}]_i = S$.

5. SIMULATION EXAMPLE

The following control design problem set-up was envisioned: A nonlinear state-space model of the plant is available (a set of ordinary differential equations, for example), and a decision on input, state, and output variables has been made. A controller is now seeked for, such that desired transitions between plant output set points would be optimal.

Let us consider a simple example of a two-tank MIMO system (for details, see (Åkesson *et al.*, 2006) and references there). The objective is to keep the temperature in the second tank (y) at it setpoint, while keeping the levels of both tanks within preset limits. The system is controlled by a valve for the first tank input flow, a pump between the two tanks, a heater in the second tank, and a valve for the second tank output flow. The heater (u_1) is constrained to values in the interval between 0 and 560 kW, the pump (u_2)

has three operational levels {off, medium, high}, the valves are binary {on/off}.

The state space was discretized by forming a grid quantized as $\{0, 1, 2, ..., 9\}$ [m] for tank levels; $\{17, 18, 19\}$ and $\{17, 18, ..., 24\}$ [C] for tank temperatures, respectively. The input flow temperature (disturbance) was discretized into three values: $\{17, 18, 19\}$; and the heating action in five values: $\{0, 140, ..., 560\}$. The immediate costs were set based on the Euclidean norm between desired and reference temperatures, $\|w - y_s\|$ and deviation from nominal controls for u_2, u_3 and u_4 at 1 with weights 0.1, 2 and 2 respectively (see (Åkesson *et al.*, 2006)). For the states where reference points exceeded either upper or lower limits for the tank level (at 0.5m and 8.5m), an additional large cost was added (ten times larger than other immediate costs). A 100 times larger cost was set for the sink cell.

The selected discretization resulted in a finite state–action space of 7201 states and 60 actions, including the sink cell. A GCM model was built by evaluating the state transitions five hundered times for each possible state-action pair (s, a). The starting state was generated from a random uniform distribution from within the state hypercube. While most of the computing time was spent on solving the ode, the computations took a couple of hours (PC: 3GHz Pentium 4 CPU, 1GB RAM, MATLAB R12). Clearly this presented a significant burden both in terms of computing power and memory, but not excessive at all. Given the GCM model, a predictive controller was designed using $H_{\mathrm{p}} = 5$.

Due to space restrictions, we only briefly summarize the closed-loop analysis:

- The probabilities for entering the sink cell from any other cell were zero (system was stable).
- The basin-of-attraction was empty for projections to levels 0 or 9 (constraints on tank levels were fulfilled)
- For set points 19°C... 23°C, the basin-of-attraction almost 100% transitions to the correct tank 2 temperature was obtained (set point control was successful). The smaller sizes of basins-of-attraction were explained by the lack of means to cool the incoming feed (for low temperatures); or by the observed open-loop transitions to sink state (for 24°C).
- In few cases (three initial states), the model predicted transitions that could be judged as impossible using physical arguments. The remedy for this problem is to increase either the sampling rate, or the number of evaluations.

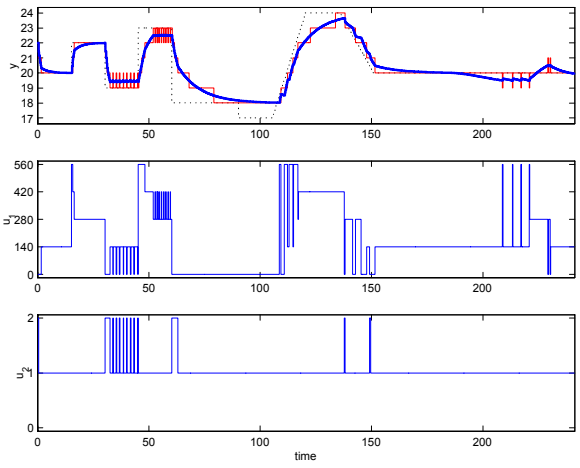

Fig. 1. Closed-loop simulation. The set point trajectory consists of a series of steps and ramps. A the end of simulation, an input disturbance affects the system.

- For a particular communicating class, or a state in it, very precise information can be obtained, such as the expected time of absorbtion from a particular system state. Unfortunately, it is not feasible to examine all states with this much care.

Figure 1 illustrates a trajectory following simulation with (a known) input disturbance. It can be concluded that convenient tools for design and analysis of the closed loop system were found, including examination of stability and steady state performance.

6. DISCUSSION AND CONCLUSIONS

The GCM transition matrices $\mathbf{P}^a$ can be learned from data (measured or simulated), by counting the number of observed transitions. For one update, all that is required are two successive state vectors and a control: $\mathbf{x}(k), \mathbf{u}(k), \mathbf{x}(k+1)$, for a given sampling time T_{s}. For likelihood matrix $\mathbf{L}$, a similar procedure can be applied. Therefore, all past or on-line recorded data-triplets can be efficiently used.

Update of a large map may require an excessive number of samples, however, and parts of the model space may never be visited in real life, due to constraints in time and plant operation. This rises up the question of efficient and practical identification procedures. A huge variety of discussions on this problem, and different procedures for solving it, have been published. Typically, depending on the problem set up, local updating procedures can be found and efficiently implemented. These are, however, always based on some prior knowledge on the plant characteristics, such as smoothness or other structural information on f. Accompagned with careful design of experiments

(statistics, interpolation / approximation techniques, focusing on control-relevant properties), the number of experiments required on real plant can be greatly reduced. A majority of the recent MDP related literature has focused on these issues. Results on findings of efficient methods have been reported. It is the authors opinion, however, that these methods introduce a substantial complexity (e.g., multilevel learning algorithms) and additional uncertainties (e.g., interpolation and smoothing) into the overall system.

In real applications, the system state may not be measurable, or the measurement may be severely corrupted by noise. If a system model ($\mathbf{P}^a$'s, $\mathbf{L}$) is available, a state estimator can be constructed. In some cases it can be more convenient to describe the system states based on delayed input–output measurements, leading to CMC–ARX models (Kárný *et al.*, 1998), thereby skipping the problem of state estimation, as if system inputs and outputs are measurable the states of this type of models are always measurable. However, the state may not be minimal, and if the measurements are noisy a state estimator (an observer or a filter) may provide useful.

6.1 Future directions

In our preliminary work we have focused on using Markov chains and MDP as a tool, the use of which is to be examined bearing in mind the resolution of the problem set up (discretization into a finite state space). Continuing in the same direction, the problem of identication is then related to keeping the original model up-to-date (the ode, for example), or –at least– approximating the original model using function approximation techniques, rather than looking for clever tricks to make counting feasible in the finite state space, doomed to be huge. If doable, the benefits are clear: physical interpretation of estimated parameters. In many process engineering problems, this may turn out to be more fruitful than pure machine learning approaches.

Instead, the problem of uncertainty in measurements can potentially be handled in a very elegant and efficient fashion using finite Markov chains. Given the finite state probabilistic description of the plant, it is straightforward to construct cost functions taking into account the uncertainty in the predictions (other than discounted conditional expectations). Under the predictive control paradigm, also uncertainties in current state can be taken into account in plant predictions (i.e., there's no need to restrict to ML estimates, etc.). Completing a literature review on these topics is a major direction in our future research.

REFERENCES

Åkesson, B M, M J Nikus and H T Toivonen (2006). Explicit model predictive control of a hybrid system using support vector machines. In: *Proc. IFAC ALSIS'06*. Helsinki-Stockholm, Finland-Sweden.

Häggström, O (2002). *Finite Markov Chains and Algorithmic Applications*. Cambridge University Press. Cambridge.

Hsu, C S (1987). *Cell-to-Cell Mapping - A Method of Global Analysis for Nonlinear Systems*. Springer-Verlag. New York.

Ikonen, E (2004). Learning predictive control using probabilistic models. In: *IFAC AFNC'04, Oulu, Finland*.

Ikonen, E and K Najim (2002). *Advanced Process Identification and Control*. Marcel Dekker Inc. New York.

Kaelbling, L P, M L Littman and A W Moore (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* **4**, 237–285.

Kárný, M, M Valecková and H Gao (1998). Mixtures of adaptive controllers based on Markov chains: A future of intelligent control?. In: *UKACC CONTROL '98*. IEE. pp. 721–726.

Lee, J M and J H Lee (2004). Approximate dynamic programming strategies and their applicability for process control: A review and future directions. *International Journal of Control, Automation, and Systems* **2**(3), 263–278.

Lunze, J (1998). On the Markov property of quantised state measurement sequences. *Automatica* **34**(11), 1439–1444.

Lunze, J, B Nixdorf and H Richter (2001). Process supervision by means of a hybrid model. *Journal of Process Control* **11**, 89–104.

Najim, K, E Ikonen and D Ait-Kadi (2004). *Stochastic Processes - Estimation , Optimization and Analysis*. Kogan Page Science. London.

Negenborn, R R, B De Schutter, M A Wiering and H Hellendoorn (2005). Learning-based model predictive control for Markov decision processes. In: *16th IFAC World Congress*.

Poznyak, A S, K Najim and E Gómez-Ramírez (2000). *Self-Learning Control of Finite Markov Chains*. Marcel Dekker. New York.

Puterman, M L (1994). *Markov Decision Processes – Discrete Stochastic Dynamic Programming*. Wiley et Sons. New York.

Riordon, J S (1969). An adaptive automaton controller for discrete-time Markov processes. *Automatica* **5**, 721–730.

Ungarala, S and Z Chen (2003). Bayesian data rectification of nonlinear systems with Markov chains in cell space. In: *Proc. ACC*. pp. 4857–4862.