Qualitative Induction

Dorian Šuc and Ivan Bratko
Faculty of Computer and Information Science
University of Ljubljana, Slovenia
{dorian.suc, ivan.bratko}@fri.uni-lj.si.

Abstract

We consider the problem of automatic construction of qualitative models by inductive learning from quantitative examples. We present an algorithm QUIN (QUalitative INduction) that learns *qualitative trees* from a set of examples described with numerical attributes. At difference with decision trees that are often used in machine learning, the leaves of qualitative trees contain qualitative functional constraints. A qualitative tree defines a partition of the attribute space into the areas with common qualitative behaviour of the chosen class variable.

We demonstrate the use of qualitative trees by their application to the reconstruction of human skill to control container cranes. The induced qualitative trees define qualitative control strategies that provide good insight in the human operator's control skill and enable also the reconstruction of individual differences in control styles of different operators.

Introduction

Building a qualitative model for a complex system requires significant knowledge and is a time-consuming process. For this reason, many researchers have addressed the problem of automatic generation of a qualitative model. One approach is to build models from existing libraries of model fragments (Forbus 1984). Another approach is to learn a model of a physical system from behaviours using existing knowledge of processes and mechanisms commonly found in physical systems (Doyle 1988). Less knowledge-intensive approaches (Coiera 1989; Bratko, Mozetič, & Lavrač 1989; Bratko, Muggleton, & Varšek 1991; Richards, Kraan, & Kuipers 1992; Džeroski & Todorovski 1993) use inductive learning to induce qualitative differential equations or logical models from a set of qualitative behaviours.

In this paper we present algorithm QUIN for automatic generation of a qualitative model by inductive learning from quantitative examples. We apply QUIN to the learning of qualitative models of human control skill. Qualitative models have already been used in dynamic system's control, including deriving controllers by qualitative reasoning about differential equations models (Makarovič 1991), applying qualitative simulation to the analysis of heterogeneous fuzzy controllers (Kuipers & Åström 1994), using a qualitative explanation of the control skill (DeJong 1994) and deriving a qualitative control rule from a qualitative differential equations model of the controlled system (Bratko 1995).

The comprehensibility of qualitative models offers also the possibility of their use in the *reconstruction of the human control skill*. The motivation for skill reconstruction is in understanding of the operator's skill and its use in the development of an automatic controller. This is particularly interesting in the case of complex dynamic systems, as a plane or a crane, that are controlled by skilled operators who acquired their skill in years of experience. Typically such a control skill is sub-cognitive and hard to reconstruct through introspection. The operators cannot completely describe their skill, but can demonstrate it. Therefore an attractive approach to the reconstruction of the human control skill involves machine learning from the logged data from skilled, human operators, i.e. operator's execution traces. This approach is also called *behavioural cloning* (Michie 1993).

Behavioural cloning has been used in problem domains as pole balancing (Chambers & Michie 1969; Michie, Bain., & Hayes-Michie 1990), piloting (Sammut *et al.* 1992; Bain & Sammut 1999; Camacho 2000) and container cranes (Urbančič & Bratko 1994). These experiments are reviewed in (Bratko, Urbančič, & Sammut 1998). Controllers, also called clones, were usually induced as a mapping from the system's states to control actions in the form of trees or rule-sets. Although such clones do provide some insight into the control strategy, they are often large and difficult to understand. Often, the induced trees have several hundreds of leaves and are, as such, not suitable as explanatory models of human skill (Bratko & Urbančič 1999).

Our experiments in controlling a crane (Šuc & Bratko 1999a; 1999b) and double pendulum called acrobot (Šuc & Bratko 2000b) showed that qualitative abstractions of induced quantitative control strategies are comprehensible and therefore suitable as models of the operator's skill. These qualitative abstractions offer also a space for controller optimization. These experiments

motivate the learning of qualitative strategies directly from execution traces.

The structure of the paper is as follows. First we give the learning problem description for induction of qualitative trees and define monotonicity constraints, called qualitatively constrained functions. Then we describe how qualitatively constrained functions and qualitative trees are learned from a set of numerical examples. The developed algorithm QUIN is then applied to the reconstruction of the human crane control skill. Finally, we discuss some points of interest and give conclusions.

Learning Problem Description

Qualitative Trees

We consider the usual setting of classification learning, but in our case the hypothesis language involves *qualitative constraints*.

Let there be N learning examples. Each example is described by $n+1$ continuous variables $X_1, \ldots, X_{n+1}$ with values $x_{i,1}, \ldots, x_{i,n+1}$, $i = 1, \ldots, N$. The variable X_{n+1} is called the *class*, and the others are called *attributes*. In the context of modelling dynamic systems, an example can be a state in the system's state space.

Given the learning examples, our problem is to learn a hypothesis that separates the areas of attribute space which share a common qualitative behaviour of the class variable. We learn such hypotheses in the form of *qualitative trees*. A qualitative tree is a binary tree with internal nodes called splits and *qualitatively constrained functions* in the leaves. The splits define a partition of the attribute space into areas with common qualitative behaviour of the class variable. A split consists of a split attribute and a split value. Qualitatively constrained functions (abbreviated QCFs) in leaves define qualitative constraints on the class variable.

Figure 1 shows an example of qualitative tree induced from a set of example points for the function $z = x^2 - y^2$. This tree is usually written in a text form as:

$$
\begin{array}{l}
x \leq 0 \\
\quad | \quad y \leq 0 : z = M^{-,+}(x,y) \\
\quad | \quad y > 0 : z = M^{-,-}(x,y) \\
x > 0 \\
\quad | \quad y \leq 0 : z = M^{+,+}(x,y) \\
\quad | \quad y > 0 : z = M^{+,-}(x,y)
\end{array}
$$

Qualitatively Constrained Functions

Qualitatively constrained functions are inspired by the qualitative proportionality predicates Q_+ and Q_- as defined by Forbus (Forbus 1984) and are also a generalization of the qualitative constraint M^+, as used in (Kuipers 1986). We use QCFs to define qualitative constraints on the class variable. A QCF constrains the qualitative change of the class variable in response to the qualitative changes of the attributes.

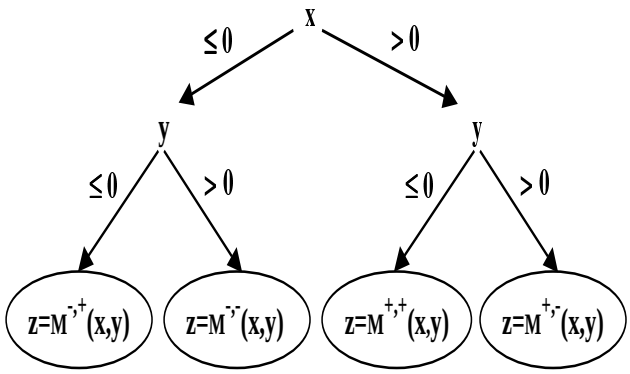

Figure 1: A qualitative tree tree induced from a set of example points for the function $z = x^2 - y^2$. The rightmost leaf, applying when attributes x and y are positive, says that z is monotonically increasing in its dependence on x and monotonically decreasing in its dependence on y.

A *qualitatively constrained function* $M^{s_1,\ldots,s_m}$: $\Re^m \mapsto \Re$, $s_i \in \{+,-\}$ represents an arbitrary function with $m \leq n$ continuous attributes that respect the qualitative constraints given by signs s_i. The qualitative constraint given by sign $s_i = +$ ($s_i = -$) requires that the function is strictly increasing (decreasing) in its dependence on the i-th attribute. We say that the function is *positively related* (negatively related) to the i-th attribute. $M^{s_1,\ldots,s_m}$ represents any function which is, for all $i = 1, \ldots, m$ positively (negatively) related to the i-th argument, if $s_i = +$ ($s_i = -$).

Note that the qualitative constraint given by sign $s_i = +$ only states that when the i-th attribute increases, the QCF will also increase, *barring other changes*. It can happen that a QCF with the constraint $s_i = +$ decreases even if the i-th attribute increases, because of a change in another attribute. For example, consider the behaviour of gas pressure in a container: $Pres \times Vol/Temp = const$. We can express the qualitative behaviour of gas by QCF $Pres = M^{+,-}(Temp, Vol)$. This constraint allows that the pressure decreases even if the temperature increases, because of a change in the volume. Notice however, that the qualitative behaviour of gas is not consistent with the constraint $Pres = M^+(Temp)$.

QCFs are concerned with qualitative changes and qualitative change vectors. *Qualitative change* q_j is the sign of change in continuous variable X_j, where $q_j \in \{pos, neg, zero\}$, corresponding to *positive*, *negative* or *zero* change. A *qualitative change vector* is a vector of qualitative changes of the variables. We define *QCF-prediction* $P(s_i, q_i)$ as:

$$
P(s_i,q_i)=\begin{cases}
\text{pos, if } (s_i{=}+ \wedge q_i{=}pos) \vee (s_i{=}- \wedge q_i{=}neg) \\
\text{neg, if } (s_i{=}+ \wedge q_i{=}neg) \vee (s_i{=}- \wedge q_i{=}pos) \\
\text{zero, otherwise}
\end{cases}
$$

A qualitative change vector $q = (q_1, \ldots, q_{n+1})$ is *consis-*

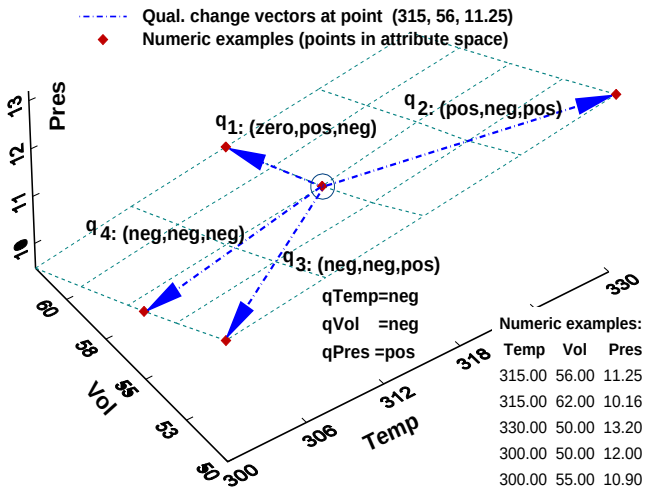

Figure 2: Gas in the container example: five numerical examples (the table at the bottom right), described with attributes $Temp$, Vol and class $Pres$, are represented as points in the attribute space. The arrows denote the qualitative change vectors at the circled point e=($Temp$=315, Vol=56, $Pres$=11.25). For example, the point ($Temp$=300, Vol=50, $Pres$=12.00) gives qualitative change vector q_3=(q_{Temp}=neg, q_{Vol}=neg, q_{Pres}=pos) with point e.

tent with a given QCF $M^{s_1,\ldots,s_m}$, if the QCF does not reject the qualitative change of the class variable, that is, if either (a) class qualitative change is zero, (b) all attribute's QCF-predictions are zero, or (c) there exists an attribute whose QCF-prediction is equal to the class's qualitative change.

A QCF does not always uniquely predict the class's qualitative change given the qualitative changes of the attributes. *Qualitative ambiguity*, i.e. ambiguity in the class's qualitative change appears whenever there exist both positive and negative QCF-predictions or whenever all QCF-predictions are zero. In this case any qualitative class change is consistent with the QCF.

Note that *any* class change (not just zero class change) is consistent with a QCF when all QCF-predictions are zero. This is a weaker definition of consistency than one might expect. The reason for such definition is in learning QCFs, where each possible QCF is penalized according to inconsistent qualitative change vectors: small changes in all attributes (all zero QCF-predictions) are not a strong evidence against large (nonzero) class change.

Learning Qualitatively Constrained Functions

When learning a QCF from a set of numerical examples we are interested in a QCF that is consistent with most of the examples, i.e. in the *"minimal cost"* QCF. For this reason we define *error-cost* $E(g)$ of a QCF g (defined later in Eq. 3) that penalizes g with inconsistent

QCF	*Inconsis.*	*Ambig.*
$Pres$=$M^{+}(Temp)$	q_3	q_1
$Pres$=$M^{-}(Temp)$	q_2, q_4	q_1
$Pres$=$M^{+}(Vol)$	q_1, q_2, q_3	/
$Pres$=$M^{-}(Vol)$	q_4	/
$Pres$=$M^{+,+}(Temp, Vol)$	q_1, q_3	q_2
$Pres$=$M^{+,-}(Temp, Vol)$	/	q_3, q_4
$Pres$=$M^{-,+}(Temp, Vol)$	q_1, q_2	q_3, q_4
$Pres$=$M^{-,-}(Temp, Vol)$	q_4	q_2

Table 1: Gas in the container example: the first column gives all possible QCFs using attributes $Temp$ and Vol. The second and the third column give qualitative change vectors at point e (see Figure 2) that are respectively inconsistent and ambiguous with the corresponding QCFs.

and ambiguous qualitative change vectors at every example. The "minimal cost" QCF is learned from a set of numerical examples by first forming qualitative change vectors from examples and then minimizing error-cost of a QCF over all possible QCFs.

First, every pair of examples e and $f \neq e$ is used to form a qualitative change vector with qualitative changes $q_{(e,f),j} \in \{pos, neg, zero\}$, $j = 1, \ldots, n+1$ defined as:

$$q_{(e,f),j} = \begin{cases} \text{pos, if } x_{f,j} > x_{e,j} + Tzero_j \\ \text{neg, if } x_{f,j} < x_{e,j} - Tzero_j \\ \text{zero, otherwise} \end{cases} \quad (1)$$

where $Tzero_j$ denotes a user-defined steady threshold defining negligible changes of j-th attribute. The default value of $Tzero_j$ is 1% of the difference between maximal and minimal value of j-th attribute. Typically, many pairs of examples map into the same qualitative change vector. A qualitative change vector is either consistent or not consistent with a given QCF. Note that a consistent qualitative change vector can also be ambiguous for a given QCF.

We illustrate the method to find the "minimal cost" QCF by an example of gas in the container. Figure 2 gives five numerical examples described with attributes $Temp$ and Vol and class $Pres$, giving gas temperature, volume and pressure according to equation $Pres = 2\,Temp/Vol$. There are five numerical points, each with four qualitative change vectors with respect to other points. Figure 2 illustrates qualitative change vectors q_1, q_2, q_3 and q_4 at the circled point e=($Temp$=315, Vol=56, $Pres$=11.25). To find the "minimal cost" QCF at point e, qualitative change vectors that are inconsistent and ambiguous with each possible QCF are counted. Consider for example QCF $Pres = M^{+}(Temp)$. Qualitative change vector q_3=(q_{Temp}=neg, q_{Vol}=neg, q_{Pres}=pos) is not consistent with this QCF. Qualitative change vector q_1=(q_{Temp}=$zero$, q_{Vol}=pos, q_{Pres}=neg) is ambiguous with respect to this QCF.

Table 1 gives qualitative change vectors at point e that are inconsistent with and ambiguous for each possible QCF. QCF $M^{+,-}(Temp, Vol)$ is the only QCF consistent with all qualitative change vectors and is the "minimal cost" QCF. It also minimizes the error-cost (defined in the next paragraph) over all QCFs. Note that this QCF is also ambiguous for q_3 and q_4. If the error-cost would prefer simpler QCF with one inconsistent qualitative change vector over QCF with two ambiguous qualitative change vectors then QCF $M^{-}(Vol)$ would be selected as the "minimal cost" QCF.

The error-cost of a QCF is based on the minimum description length principle (Rissanen 1978; Quinlan & Rivest 1989). Basically it is defined as the number of bits needed to code the QCF plus the number of bits to code the inconsistent and ambiguous qualitative change vectors as follows. Let $\mathcal{C}_e(q)$ and $n_e(q)$ denote respectively the set and the number of all examples f that form, with e, qualitative change vector q:

$$\begin{aligned} \mathcal{C}_e(q) &= \{f | \forall j = 1, \ldots, n+1 : q_{(e,f),j} = q_j\} \\ n_e(q) &= |\mathcal{C}_e(q)| \end{aligned} \quad (2)$$

In the above gas in the container example $\mathcal{C}_e(q_3) = \{$(Temp= 300,Vol= 50,Pres= 12.00)$\}$ and $n_e(q_3) = 1$. The *error-cost* $E(g)$ of a QCF g that mentions m out of all n attributes is:

$$\begin{aligned} E(g) = &\log_2 n + m(\log_2 n + 1) + \\ &\log_2 N_{nonamb} + N_{reject}(\log_2 N_{nonamb}) + \\ &\log_2 N_{amb} + N_{amb} \end{aligned} \quad (3)$$

Here N_{reject} denotes the number of example pairs (e,f), $f \neq e$, that form a qualitative change vector that is not consistent with QCF g and is computed as the sum of $n_e(q)$ over all examples e and over all qualitative change vectors q that are not consistent with g. Similarly, N_{amb} and N_{nonamb} denote the sum of $n_e(q)$ of vectors q that are respectively ambiguous and not ambiguous for g. This error-cost is based on the following encoding: we code the QCF, the indexes of N_{reject} inconsistent qualitative change vectors (each index requires $\log_2 N_{nonamb}$ bits since ambiguous qualitative change vectors are always consistent) and one bit for each ambiguous qualitative change vector.

The "minimal cost" QCF is found by a simple exhaustive search algorithm that forms all possible QCFs and selects the one with the smallest error-cost. This requires the number of error-cost computations that is exponential in the number of attributes. Instead of the exhaustive search, QUIN uses a greedy heuristic algorithm that requires the number of error-cost computations that is quadratic in the number of attributes, but does not guarantee the "minimal cost" QCF. The idea is to start with the QCF that minimizes error-cost over all QCFs that use only one attribute, and then use error-cost to refine the current QCF with another attribute.

Learning Qualitative Trees

QUIN learns a qualitative tree from a set of numerical examples by a top-down greedy algorithm guided by error-cost of a QCF. This algorithm is similar to the decision tree learning algorithms.

Given the examples, QUIN chooses the best split by comparing the partitions of the examples they generate: for every possible split, it splits the examples into two subsets (according to the split), finds the "minimal cost" QCF in both subsets, and selects the split which minimizes the *tree error-cost* (defined below). It puts the best split in the root of the tree and recursively finds both subtrees for the corresponding example subsets, until the stopping condition is satisfied, i.e. until using the best split does not improve the tree error-cost.

The *tree error-cost* is computed as follows. The tree error-cost of a leaf is the error-cost $E(g)$ of the "minimal cost" QCF g that is induced from the examples in the leaf. The tree error-cost of an internal node is the sum of the error-costs of both subsets plus the cost of the split, i.e. the number of bits needed to encode the split:

$$\begin{aligned} E_{tree} &= E_{left} + E_{right} + SplitCost \\ SplitCost &= \log_2 n + \log_2(Splits_i - 1) \end{aligned}$$

Here E_{left} and E_{right} denote the error-costs in both subsets, n is the number of variables and $Splits_i$ is the number of possible splits for the split variable, i.e. the number of different values of the variable X_i.

The error-cost (Eq. 3) penalizes inconsistent and ambiguous qualitative change vectors formed from every pair of examples. This sometimes results in myopic selection of the "best" split, since the proximity of examples and the consistency of qualitative change vectors are not considered. We refer to the algorithm that uses this error-cost also as ep-QUIN algorithm (ep standing for every pair).

QUIN is a heuristic variant of ep-QUIN algorithm with the improved error-cost. QUIN's error-cost is similar to ep-QUIN's error-cost, but with qualitative change vectors weighted according to their confidence estimates. These confidence estimates take into account the locality and the consistency of qualitative change vectors.

A more elaborate description of the QUIN algorithm and the evaluation of ep-QUIN and QUIN algorithms on a set of artificial domains is given in (Šuc 2001). The empirical results show that QUIN usually gives better results and is more time efficient, can handle noisy data, and, on simple domains, produces qualitative trees that correspond to the human intuition.

QUIN in the Crane Domain

In this section QUIN is applied to the reconstruction of the crane control skill from operator's execution traces. An *execution trace* is a sequence of the system's states and the corresponding operator's actions that are (at some frequency) logged to a file.

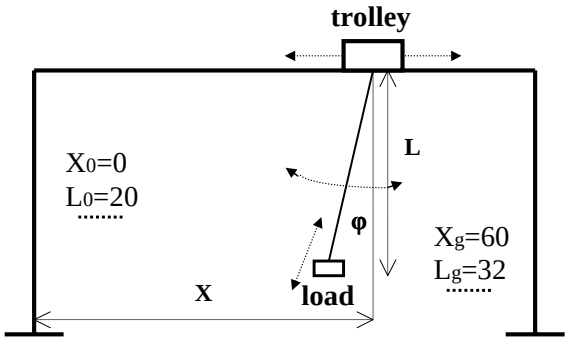

Figure 3: Container crane: the state of the system is specified by six variables: trolley position X and its velocity $\dot{X}$, rope inclination angle ϕ and its angular velocity $\dot{\phi}$, rope length L and its velocity $\dot{L}$. The system is controlled through force to the trolley in the horizontal direction and force in the direction of the rope.

First we describe the domain of container cranes, then give some notes on experiments and present qualitative control strategies induced from execution traces of two operators. The induced qualitative strategies provide good insight into the operators' control skill and also enable to reconstruct the individual differences in the control styles of different operators.

Container Crane

To transport a container (see Fig. 3) from the shore to a target position on the ship, two operations are to be performed: (1) positioning of the trolley, bringing it above the target load position (X_g), and (2) rope operation, bringing the load to the desired height (L_g). The performance requirements include basic safety, stop-gap accuracy and as high capacity as possible. The last requirement means that the time for transportation is to be minimized. Consequently, the two operations are to be performed simultaneously. The most difficult aspect of the task is to control the swing of the rope. When the load is close to the goal position, the swing should ideally be zero.

A crane simulator was used in our experiments. The parameters of the system (lengths, heights, masses, etc.) are the same as those of the real cranes in Port of Koper in Slovenia. The state of the system is specified by six variables: trolley position X and its velocity $\dot{X}$, rope inclination angle ϕ and its angular velocity $\dot{\phi}$, rope length L and its velocity $\dot{L}$. Two control forces are applied to the system: force to the trolley in the horizontal direction and force in the direction of the rope. The task is to transport the load from its start position ($X_0 = \dot{X}_0 = \phi_0 = \dot{\phi}_0 = 0,\ L_0 = 20,\ \dot{L}_0 = 0$) to the goal position ($X_g{=}60, \dot{X}_g{=}\phi_g{=}\dot{\phi}_g{=}0,\ L_g{=}32,\ \dot{L}_g{=}0$).

We used experimental data from manually controlling the crane from a previous study (Urbančič & Bratko 1994). In that study, six students volunteered to learn to control the simulator. Remarkable individual differences were observed regarding the characteristics of the strategy they used. Some operators tended towards fast and less reliable operation, others were more conservative and slower, in order to avoid large rope oscillations. One goal of our skill reconstruction was also to reconstruct the individual differences and similarities between the operators in the style of controlling the crane. For this reason we used traces of two operators that use, at least quantitatively, very different control styles. The two operators are named S and L.

Operator S uses very conservative strategy that is very reliable but requires more time to accomplish the control task. He uses small accelerations in order to avoid large rope oscillations. Operator L, on the other hand, uses a faster and more complex strategy. He is able to afford large initial swing caused by large acceleration, and later skillfully reduce the swing. Typical traces of both operators are given in Figure 4.

Notes on Experiments

Controlling the crane requires a rule to control the trolley and a rule to control the rope. The obvious choice would be to induce the constraints on the control actions, i.e. to select the force to the trolley and the force to the rope as the class variables. However, the experiments in (Šuc & Bratko 2000a; Šuc 2001) show that the learning of the operators control trajectories usually results in more robust and comprehensible controllers than learning of the operators actions. Here, the operator's trajectory is a sequence of system's states from the execution trace. For this reason, QUIN was used to induce the constraints on the operator's trajectories, not the actions. To simply the system's control according to the induced constraints and to gain comprehensible strategies, one or more state variables can be selected as the dependent trajectory variables. The dependent trajectory variable is the class variable and the other state variables are the attributes for learning.

We decided to induce the desired trolley velocity $\dot{X}_{des}$ and desired rope length L_{des}, that is qualitative trees predicting $\dot{X}$ and L in the next state as the function of the current state. We selected these class variables since they are close to human thinking about controlling the trolley and the rope.

Controlling the Rope Length

We used QUIN to induce the desired rope length, i.e. L_{des} from the traces of operator's S and L. L_{des} was the class and the other state variables were the attributes.

From a trace of operator S, a qualitative tree with a single leaf was induced:

$$L_{des} = M^{+}(X) \tag{4}$$

Although this tree is very simple, it is consistent with all the learning examples from the operator's trace and clearly describes the operator's strategy to control the

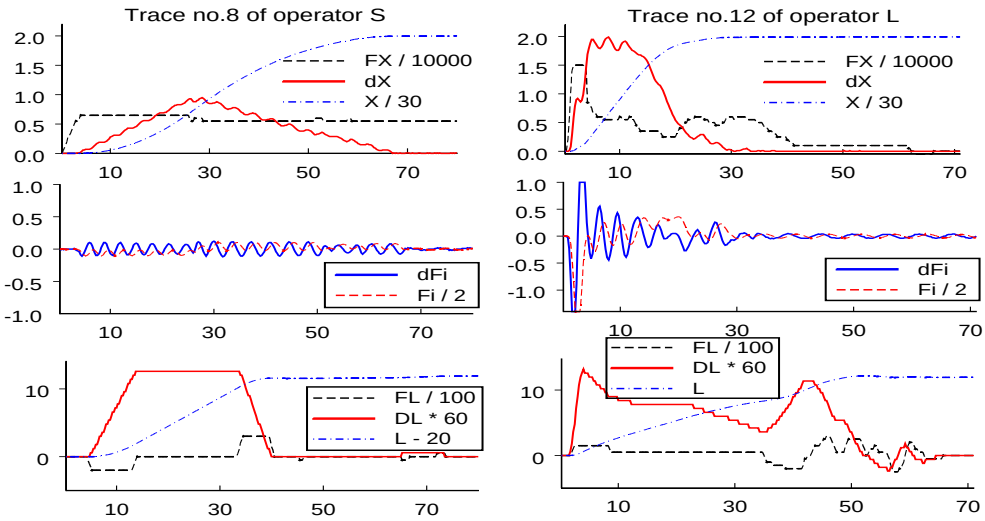

Figure 4: Typical traces of operators S and L: on the left is a trace of operator S and on the right a trace of operator L that uses a more complex strategy. Note the differences in the swing of the rope (variables Fi and dFi).

rope: the operator brings down the load as the trolley moves from its start to the goal position.

From other traces of both operators, similar trees were induced. Some of them were more complicated, but all of the induced trees had QCF $L_{des} = M^{+}(X)$ in the root. An example is a qualitative tree induced from another trace of operator S:

$$X \leq 60.5: \quad L_{des} = M^{+}(X)$$
$$X > 60.5: \quad L_{des} = M^{-}(\dot{X})$$

In this trace the operator failed to stop the trolley at the exact trolley goal position X_g=60. He uses negative trolley velocity to approach X_g from the other direction, but still increases the rope length.

Comparing the trees of both operators we can conclude that they use the same simple control strategy to control the rope length. They both increase the rope length as the trolley moves towards the goal. Usually this is all it takes to control the rope length. However, both sometimes miss the exact goal position and do some additional adjusting at the end.

Controlling the Trolley

Transporting the load from the shore to the ship requires positioning of the trolley above the goal position (X_g=60). Therefore the trolley velocity is to be first increased and then decreased, to precisely adjust the trolley at the goal. The task is difficult, because it requires that the swing of the rope is zero at the goal. Experienced operators try to reduce the swing by proper accelerations of the trolley.

We used QUIN to induce the desired trolley velocity, i.e $\dot{X}_{des}$ from the traces of operator's S and L. The following experiments with traces of operator S were a surprise for us. QUIN, in contrary to our expectations, discovered that the operator S is able to reduce the swing of the rope by proper acceleration of the trolley. Operator S controls the crane in a very conservative way and changes the control forces rarely as can be observed in Figure 4. His conservative control style, with small trolley accelerations, results in small swing of the rope. For this reason, our initial hypothesis was that he does not try to reduce it. Also in experiments with regression trees (Urbančič & Bratko 1994), the skill to reduce the swing could not be identified. Rope angle and velocity were typically not important attributes in the induced trees.

One reason that his skill to reduce the swing is hard to identify, is because he uses it rarely, typically just a few times in a trace, sometimes just at the very end. This is what QUIN induced from one of his less successful traces:

$$\begin{array}{l} X \leq 20.7: \quad \dot{X}_{des} = M^{+}(X) \\ X > 20.7: \\ \quad | \quad X \leq 60.1: \quad \dot{X}_{des} = M^{-}(X) \\ \quad | \quad X > 60.1: \quad \dot{X}_{des} = M^{+}(\Phi) \end{array} \tag{5}$$

This qualitative tree provides a good insight in the operator's control skill. The operator first increases the trolley velocity. At about half distance from the goal (X=20.7) he decreases the trolley velocity. At the goal position ($X > 60.1$) he uses the rule $\dot{X}_{des} = M^{+}(\Phi)$. This constraint, telling that the trolley velocity is positively related to the rope angle, shows his skill to reduce the swing of the rope. By acceleration of the trolley, when the rope angle increases, the angular velocity is decreased.

In (Urbančič & Bratko 1994), the human operators were also requested to describe their skills in the form of instructions for controlling the trolley. The surprising constraint $\dot{X}_{des} = M^{+}(\Phi)$ led us to reconsider the

instructions that the operator gave through introspection. Operator S described his skill of "final balancing" at the goal (when $X > 59.65$) by decelerating the trolley "a little before the maximum of $\dot{X}$" and accelerating "a little before the minimum of $\dot{X}$". Note that the oscillation of $\dot{X}$ is the consequence of swinging of the load and that $\dot{X}$ oscillates in the opposite direction than $\dot{\Phi}$. Since the minima (maxima) of $\dot{X}$ correspond to the points in the middle of the rope swing cycle when Φ is increasing (decreasing), the operators instructions amount to the same effect as the induced QCF: $\dot{X}_{des} = M^{+}(\Phi)$.

In the same way the operator described his skill to reduce the swing also before reaching the goal. This was induced from another one of his traces:

$$\begin{array}{ll} X \leq 26.9: & \dot{X}_{des} = M^{+}(X) \\ X > 26.9: & \\ \mid \quad \Phi \leq -0.01: & \dot{X}_{des} = M^{-}(X) \\ \mid \quad \Phi > -0.01: & \dot{X}_{des} = M^{-,+,-}(X, \Phi, \dot{\Phi}) \end{array} \tag{6}$$

Here, the strategy is similar as before, but reducing the swing starts before reaching the goal ($X > 26.9$). Also, the skill to reduce the swing is here more refined. Let's observe just the part of the tree for $X > 26.9$. One interpretation is that the trolley velocity it to be increased when $\Phi > -0.01$ and Φ is increasing and $\dot{\Phi}$ is decreasing, i.e. when the rope swings through the vertical to the right. It is interesting that this part of the tree is a refinement of rule $\dot{X}_{des} = M^{+}(\Phi)$ and even more precisely corresponds to the operator's instructions to accelerate "a little before the minimum of $\dot{X}$". QUIN qualified "a little before" as $\Phi > -0.01$.

From a trace of operator L, the following qualitative tree was induced:

$$\begin{array}{ll} X \leq 29.3: & \dot{X}_{des} = M^{+,+,-}(X, \Phi, \dot{\Phi}) \\ X > 29.3: & \{\text{default rule: } \dot{X}_{des} = M^{-,+}(X, \Phi)\ \} \\ \mid \quad \dot{\Phi} \leq -0.02: & \dot{X}_{des} = M^{-}(X) \\ \mid \quad \dot{\Phi} > -0.02: & \dot{X}_{des} = M^{-,+}(X, \Phi) \end{array} \tag{7}$$

Similar to operator S, operator L first increases the trolley velocity and decreases it later to approach the goal. However, to approach the goal as fast as possible he uses large accelerations that cause large swing of the rope at the very start. He manages to skillfully reduce this large swing as can be observed in Figure 4. At a difference with operator S, he reduces the swing of the rope also in the first stage of control, while still increasing the trolley velocity. QUIN induced this skill as the QCF $M^{+,+,-}(X, \Phi, \dot{\Phi})$ that applies when $X \leq 29.3$. An interpretation of this QCF is that he decreases the trolley velocity when the rope angle Φ is decreasing and the rope velocity $\dot{\Phi}$ is increasing. This is, while the rope swings from the vertical to the left, until it reaches it local minimum.

Similarly to operator S, operator L also reduces the swing of the rope in the second part of control, that is when the trolley velocity is decreasing. However he does it more consistently. QUIN induced the default rule $\dot{X}_{des} = M^{-,+}(X, \Phi)$ in the second part of the tree (for $X > 29.3$). This rule, similar to the above mentioned rules, decreases the swing of the rope by increasing the trolley velocity when the rope angle increases. The operator's consistency also enabled QUIN to induce a simpler form (from QUIN's point of view) of this rule. Consider the leaf that applies when $\dot{\Phi} \leq -0.02$. $\dot{\Phi}$ is negative, so Φ is decreasing. Since X is increasing (as the trolley moves towards the goal) and Φ is decreasing, the default rule would require $\dot{X}_{des}$ to decrease. Since X is increasing, a simpler QCF $M^{-}(X)$ achieves the same result as the more complex default rule.

By comparing the induced trees we can conclude that both operators use qualitatively similar strategies to control the trolley, but there are differences. The most important difference is that operator L reduces the rope swing during the whole trace, whereas operator S tries to reduce the swing only at the end, while approaching the goal.

The experiments described in (Šuc 2001) show that, by transforming QCFs into real-valued functions, all the presented qualitative trees can be turned into operational controllers, that are successful and robust. In this way, qualitative trees define spaces for controller optimization. The success and efficiency of such controllers depend also on the operator's control style.

Conclusion

We presented QUIN algorithm for learning of qualitative trees from quantitative examples and demonstrated its use by an application in the human skill reconstruction. To our knowledge, no study has yet addressed the induction of similar tree-structured qualitative constraints from quantitative examples.

The results show that QUIN is able to detect very subtle aspects of the human control skill and enables the reconstruction of the individual differences in the control styles of different operators. Qualitative control strategies open also other new perspectives to the human skill reconstruction, such as automating the explanations of the induced strategies by qualitative simulation and automating qualitative or semi-qualitative reasoning that can verify whether, or under which conditions a qualitative strategy might be successful. In this case the induced strategy would be checked by qualitative simulation using the provided (or induced) constraints of the system's behaviour. An alternative would to modify the learning algorithm so as to consider the provided constraints of the system's behaviour during the learning of the strategy.

QUIN has been successfully used in skill reconstruction. The experiments in artificial domains described in (Šuc 2001) also show that QUIN can handle noisy data, and, at least in simple domains, produces qualitative trees that correspond to human intuition. We believe, QUIN can be applied to other domains as a

general tool for qualitative system identification.

Acknowledgements

This work was partially supported by the Slovenian Ministry of Science and Technology and European 5^{th} Framework project Clockwork.

References

Bain, M., and Sammut, C. 1999. A framework for behavioural cloning. In Furukawa, K.; Michie, D.; and S., M., eds., *Machine Intelligence 15*, 103–129. Oxford University Press.

Bratko, I., and Urbančič, T. 1999. Control skill, machine learning and hand-crafting in controller design. In Furukawa, K.; Michie, D.; and S., M., eds., *Machine Intelligence 15*, 131–153. Oxford University Press.

Bratko, I.; Mozetič, I.; and Lavrač, N. 1989. *KARDIO: a Study in Deep and Qualitative Knowledge for Expert Systems, Chapter 5.* MIT Press.

Bratko, I.; Muggleton, S.; and Varšek, A. 1991. Learning qualitative models of dynamic systems. In *Proceedings of the 8th International Workshop on Machine Learning.*

Bratko, I.; Urbančič, T.; and Sammut, C. 1998. Behavioural cloning of control skill. In Michalski, R.; Bratko, I.; and Kubat, M., eds., *Machine Learning and Data Mining: Methods and Applications*, 335–351. John Wiley & Sons, Ltd.

Bratko, I. 1995. Deriving qualitative control for dynamic systems. In Furukava, K.; Michie, D.; and Muggleton, S., eds., *Machine Intelligence 14*, 367–386. Oxford University Press.

Camacho, R. 2000. *Inducing Models of Human Control Skills using Machine Learning Algorithms.* Ph.D. Dissertation, Engineering Faculty of Porto University, Portugal.

Chambers, R., and Michie, D. 1969. Man-machine co-operation in a learning task. *Computer Graphics: Techniques and Applications* 179–186.

Coiera, E. 1989. Generating qualitative models from example behaviours. Technical Report Technical report 8901, University of New South Wales.

DeJong, G. 1994. Learning to plan in continuous domains. *Artificial Intelligence* 65:71–141.

Doyle, R. 1988. *Hypothesizing Device Mechanisms: Opening Up the Black Box.* Ph.D. Dissertation, Massachusetts Institute of Technology.

Džeroski, S., and Todorovski, L. 1993. Discovering dynamics. In *Proceedings of the 10th International Conference on Machine Learning*, 97–103. Morgan Kaufmann.

Forbus, K. 1984. Qualitative process theory. *Artificial Intelligence* 24:85–168.

Kuipers, B, J., and Åström, K. 1994. The composition and validation of heterogeneous control laws. *Automatica* 30:233–249.

Kuipers, B, J. 1986. Qualitative simulation. *Artificial Intelligence* 29:289–338.

Makarovič, A. 1991. A qualitative way of solving the pole balancing problem. *Machine Intelligence 12* 241–258.

Michie, D.; Bain., M.; and Hayes-Michie, J. 1990. Cognitive models from subcognitive skills. In Grimble, M.; McGhee, J.; and Mowforth, P., eds., *Knowledge-Based Systems in Industrial Control*, 71–99. Peter Peregrinus.

Michie, D. 1993. Knowledge, learning and machine intelligence. In Sterling, L., ed., *Intelligent Systems*, 2–19. New York: Plenum Press.

Quinlan, J., and Rivest, R. 1989. Inferring decision trees using the minimum description length principle. *Information and Computation* 80:227–248.

Richards, B.; Kraan, I.; and Kuipers, B. 1992. Automatic abduction of qualitative models. In *Proceedings of the National Conference on Artificial Inteligence.* AAAI/MIT Press.

Rissanen, J. 1978. Modelling by shortest data description. *Automatica* 14:465–471.

Sammut, C.; Hurst, S.; Kedzier, D.; and Michie, D. 1992. Learning to fly. In *Proceedings of the 9th International Workshop on Machine Learning*, 385–393. Morgan Kaufmann.

Urbančič, T., and Bratko, I. 1994. Reconstructing human skill with machine learning. In Cohn, A., ed., *Proceedings of the 11th European Conference on Artificial Intelligence*, 498–502. John Wiley & Sons, Ltd.

Šuc, D., and Bratko, I. 1999a. Modelling of control skill by qualitative constraints. In *Proceedings of the 13th International Workshop on Qualitative Reasoning*, 212–220. University of Aberystwyth. Loch Awe, Scotland.

Šuc, D., and Bratko, I. 1999b. Symbolic and qualitative reconstruction of control skill. *Electronic Transactions on Artificial Intelligence, Section B* 3:1–22. http://www.ep.liu.se/ej/etai/1999/002/.

Šuc, D., and Bratko, I. 2000a. Problem decomposition for behavioural cloning. In Mántaras, R., and Plaza, E., eds., *Proceedings of the European Conference on Machine Learning*, 382–391. Springer-Verlag.

Šuc, D., and Bratko, I. 2000b. Skill modelling through symbolic reconstruction of operator's trajectories. *IEEE Transaction on Systems, Man and Cybernetics, Part A* 30(06):617–624.

Šuc, D. 2001. *Machine reconstruction of human control strategies.* Ph.D. Dissertation, Faculty of Computer and Information Sc., University of Ljubljana, Slovenia. http://ai.fri.uni-lj.si/dorian/MLControl/MLControl.htm.