Design of reward structures for sequential decision-making processes using symbolic analysis

Manuel Mazo Jr. and Ming Cao

***Abstract*— In this paper we construct a symbolic mathematical model for a deterministic decision-making process, called the win-stay lose-switch model, that has been used recently to study human behaviors in a class of two-alternative, sequential, decision-making tasks. The human decisions in these tasks are conditioned by the provision of certain rewards after each taken decision. Our modeling approach allows us to use symbolic analysis techniques to verify or design reward structures guaranteeing that the process of decision making respects pre-specified temporal properties. More importantly, it enables the study of properties going beyond the asymptotic behavior of the decision process. The proposed approach could prove useful to facilitate the interaction of a human user learning to interact with smart machines.**

I. Introduction

While many artificial systems are designed to function completely autonomously without human intervention, many others are designed to interact with human operators [1]. These operators might be experts making use of a tool, or may be the recipients of a service provided by an assistant artificial system. Many of these systems require that the human actions stay within a certain set of acceptable behaviors to either guarantee the safe operation of the overall system, or to ensure that the user is interacting with the system in an optimal way, *e.g.* in the case of rehabilitation devices [2]. It is therefore of paramount importance both for safety and correct operation of artificial systems to have design techniques capable of taking into account the human factor. Due to the reasons just exposed, there has been a recent spark of interest in the control systems community to study the properties of human decision-making models proposed in the psychology literature. This interest has been reflected in a recent special issue of *the Proceedings of the IEEE*, especially devoted to the topic of interactions between humans and smart machines [3].

One central issue in such studies is to build a reasonable and tractable model to describe human decision-making processes, especially in face of uncertainties and social feedback [4]. Various models have been studied in the field of cognitive psychology [5], [6] and theoretical convergence analysis has been carried out for a special class of human decision-making tasks, called the two-alternative sequential forced-choice tasks, in the past few years [7], [8], [9]. Various efforts have also been made trying to apply such analytical results to integrate both human and robot decision-making dynamics to design a better interface between humans and smart machines [7], [4]. However, it is still an open question as to how to formalize systematically methodologies to specify the interactions between humans and smart machines.

In this paper, as in [8] we adopt the deterministic Win-Stay Lose-Switch model [10], [11] to describe the decision-making process. The model aims at explaining the process by which a decision-maker selects one between two possible choices, given a certain observed reward that depends on the past choices. Our approach to the study of such models is based on discrete event systems and formal verification and synthesis techniques [12]. Namely, similar to the observation made in [13], [9] we note that the Win-Stay Lose-Switch model can be seen as a finite state machine and thus computational techniques can be employed in its analysis. Furthermore, this also enables us to not only analyze the decision-making process under different reward structures, but also to design reward structures that enforce a certain behavior of the decision-maker. The contribution of the paper is first, a more precise model of the system under study; second, enabling the analysis of more complex properties of the system, including transient behavior; and finally and most relevant, enabling the automatic design of reward functions for given specifications.

II. Preliminaries

A. Notation

We denote by $\mathbb{N}$ the natural numbers including zero and by $\mathbb{N}^+$ the strictly positive natural numbers. With $\mathbb{R}^+$ we denote the strictly positive real numbers, and with $\mathbb{R}_0^+$ the positive real numbers including zero. Given a vector $x \in \mathbb{R}^n$ we denote by x_i the i–th element of x; by $x_{[i,\dots,j]}$ the vector obtained by selecting the i-th to j-th elements of x.We denote by $\Pi_X : X \times U \times X \to X$ the projection sending $(x, u, x') \in X \times U \times X$ to $x \in X$. Given a function $H : X \to Y$ we overload the notation and denote by $H(S) = \bigcup_{x \in S} H(x)$, for any set $S \subseteq X$. Finally, for a relation $R \subseteq A \times A$, we denote as $R^{-1} = \{(a_1, a_2) \in A \times A \,|\, (a_2, a_1) \in R\}$.

B. Human Decision Models

There are several models in the literature trying to describe how humans take decisions when confronted with different

M. Mazo Jr is with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands m.mazo@tudelft.nl

M. Cao is with the Faculty of Mathematics and Natural Sciences, University of Groningen, The Netherlands m.cao@rug.nl

This work was partially supported by INCAS^3 and grants from the Dutch Organization for Scientific Research (NWO), the Dutch Technology Foundation (STW) and the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement 257462 HYCON2 Network of Excellence.

tasks. A popular task of study is that known as "two-alternative forced-choice" task (TAFC). Arguably, two of the most popular models to explain human behavior in such a task are the Win Stay Loose Switch (WSLS) model and the Drift Diffusion model [8]. In what follows we concentrate on the WSLS model, due to its deterministic nature, while extensions to the stochastic Drift Diffusion model are left for future study. Being a bit more specific, in a TAFC task a human must sequentially[1] select an option, denoted $d(n)$, among two competing choices which we denote as 1 and 0. After taking a decision a reward is provided: $r(n)$, which is a function of both the average number of 1 choices in the the last N decisions; and of the last decision $d(n)$. In this context, the WSLS model states that a human takes at time n the same decision as at time $n-1$ if (and only if) the reward at time $n-1$ was no smaller than the reward at time $n-2$, *i.e.* $r(n-1) \geq r(n-2) \Rightarrow d(n) = d(n-1)$; and otherwise the decision taken is the other choice available. In Section III we formalize and describe this model in more detail.

C. Transition Systems

In what follows we rely on the abstract notion of system described below, which subsumes dynamical systems of very varied forms, *e.g.* ordinary differential equations, difference equations, finite state machines, etc.

Definition 2.1 (System [12]): A system S is a sextuple $(X, X_0, U, \longrightarrow, Y, H)$ consisting of:

- a set of states X;
- a set of initial states $X_0 \subseteq X$
- a set of inputs U;
- a transition relation $\longrightarrow \subseteq X \times U \times X$;
- a set of outputs Y;
- an output map $H : X \to Y$.

A system is said to be:

- *metric*, if the output set Y is equipped with a metric $\mathbf{d} : Y \times Y \to \mathbb{R}_0^+$;
- *countable*, if X is a countable set;
- *finite*, if X is a finite set.

We denote by $U(x)$ the set of inputs that can be applied at the state x, *i.e.* $\{u \in U : \exists\, x' \in X, (x, u, x') \in \longrightarrow\}$. We also denote by $\xrightarrow{q} = \{(x, u, x') \in \longrightarrow : u = q\}$, and we use $\mathrm{Post}_u(x)$ to denote the u-successors of x, *i.e.* the set: $\mathrm{Post}_u(x) = \{x' \in X | (x, u, x') \in \longrightarrow\}$.

We say that the system S is *deterministic* if $\forall x \in X$, $u \in U : |Post_u(x)| \leq 1$, and *non-deterministic* otherwise. Finally, if $\exists x \in X$ such that $U(x) = \emptyset$ we say the system is *blocking*, and otherwise *non-blocking*.

D. Preorders and partial orders

In our design procedure we use the notions of preorders and partial orders, which we formally define here:

Definition 2.2: A relation $R \subseteq X \times X$ is said to be a preorder if it satisfies:

1) $(x, x) \in R$ (reflexivity);
2) $(x, x'), (x', x'') \in R \Rightarrow (x, x'') \in R$ (transitivity);

If additionally it satisfies: $((x, x') \in R \wedge (x', x) \in R) \Rightarrow (x = x')$ (antisymmetry), the relation R is said to be a partial oder.

Starting with any relation $R \subseteq X \times X$, it is possible to construct a preorder computing the transitive, and reflexive closure of the relation, denoted $R^=$ and R^* respectively. While it is obvious that computing the reflexive closure can be done efficiently, it is not so trivial to see in the case of the transitive closure. Nevertheless, we note that computing the transitive closure $R^=$ can be performed in polynomial time, *e.g.* using the *Floyd-Warshall* algorithm [16].

III. Modeling of the TAFC/WSLS System

A. Discrete time formalization

Let us first formalize a mathematical model for the system described in Section II-B. We denote by $x(n) \in \mathbb{R}^{\min\{n, N+2\}}$ the vector containing the last $N+2$ or n choices, when $n \geq N+2$ and $n < N+2$ respectively, taken by the human subject at time $n \in \mathbb{N}$ *i.e.*:

$$\begin{aligned} x_1(n+1) &= d(n), \\ x_2(n+1) &= x_1(n) = d(n-1) \\ &\vdots \\ x_T(n+1) &= x_{T-1}(n) = d(n-T+1), \\ &\quad T = \min\{n, N+2\} \end{aligned} \tag{1}$$

where, we remind the reader that $d(n) \in U = \{0, 1\}$. For notational convenience, we consider that $x(0) = \emptyset$. Note that this is simply a shift register, and thus, this discrete time system moves in the finite set $\bigcup_{i=0}^{N} U^{i+2}$. We use an $N+2$ dimensional system (for $n \geq N+2$) instead of just an N dimensional system, as this allows us to formalize the whole TAFC/WSLS model as an output-feedback system without memory.

Consider the rewards that the human subject receives as an output of this system:

$$r(n) = F(d(n), a(x(n))),$$

where $d(n)$ is the last decision, $a(x(n))$ is the average of the last N decisions, and $F : U \times A \to \mathbb{R}_0^+$ has the following structure:

$$F(d(n), a(x(n))) := d(n) f_1(a(x(n))) + (1 - d(n)) f_0(a(n)), \tag{2}$$

where $f_{0,1} : A \to \mathbb{R}_0^+$ fully determine the reward structure. The function $a : X \cup \emptyset \to A$, being precise, computes the average of the last $\min\{n, N\}$ decisions. This function is given, when $n > 0$, by:

$$a(x(n)) := \frac{1}{\min\{n, N\}} \sum_{i=1}^{\min\{n,N\}} x_i(n),$$

and otherwise, *i.e.* when $n = 0$, by:

$$a(\emptyset) = -1.$$

[1]The pace at which decisions are made is not necessarily constant so that, depending on how the experiment is set, the human subject could spend more or less time between decisions at his/her will.

Finally, the human decision model WSLS is given by:

$$\begin{array}{lll} d(n) = x_1(n) & \text{iff} & r(n-1) \geq r(n-2) \\ d(n) = (1 - x_1(n)) & \text{iff} & r(n-1) < r(n-2). \end{array} \quad (3)$$

Note that on this model, contrasting with the view in [14], the human information set is solely:

$$I(n) = \{r(n-1), r(n-2), x_1(n)\}.$$

The human information set, necessary to take a decision, can be completely determined from the state x:

$$\begin{array}{rcl} r(n-1) & = & F(x_1(n), a(x_{[2,\ldots,N+1]}(n))) \\ r(n-2) & = & F(x_2(n), a(x_{[3,\ldots,N+2]}(n))). \end{array}$$

It is clear now that extending the state space to be of dimension $N+2$ allows for a feedback formulation not requiring extra memory[2].

Defining as output the information set $y(n) = I(n)$ the human decision model can thus be expressed as an output feedback controller for the shift register model.

B. TAFC/WSLS as a finite transition system

In the following, we provide a particular embedding of the dynamics presented in the previous section in the form of a transition system. That embedding makes even more apparent the finite nature of the system at hand.

We construct a finite system S representing the shift register model describing the dynamics of x when affected by a WSLS human decision model. The system S is given by the sextuple $S = (X, X_0, U, \longrightarrow, Y, H)$, where:

- $X = \bigcup_{i=0}^{N} U^{i+2}$,
- $X_0 = U^2$,
- $U = \{0, 1\}$,
- $Y = (U \times A)^2$,
- $H(x) = [(x_1,\ a(x_{[2,\ldots,s(x)-1]}));\ (x_2,\ a(x_{[3,\ldots,s(x)-2]}))]$
- $\longrightarrow = \{(x, u, x') \mid s(x') = \min\{N+2,\ s(x)+1\} \wedge x'_{[2,\ldots,s(x')]} = x_{[1,\ldots,s(x')-1]} \wedge ((u = 1 \wedge x'_1 = x_1) \vee (u = 0 \wedge x'_1 \neq x_1))\}$,

where $s(x)$ denotes the length of the vector x. This system captures the dynamics of the shift register depending on an input u representing wether the current decision was the same as the previous one ($u = 1$) or not ($u = 0$). The state of this system is the last $\min\{n, N+2\}$ decisions, and the output contains two pairs of values. The first pair in the output at a given state contains the last input (at time $n-1$) in the first entry and the average of the previous $\min\{n, N\}$ decisions in the second. Similarly, the second pair in the output contains the decision taken two steps before, and the average at that time of the $\min\{n-1, N\}$ decisions preceding that one.

Under our modeling approach, we have four possible initial states: 00, 01, 10 and 11. These four initial states correspond to four possible initial couple of decisions from the human. Note that we model these first two decisions as initial states, as they are taken irrespective of the rewards received.

[2]The memory that otherwise would be necessary is in fact embedded on the extra dimensions of the state.

The first decision is taken before any reward was obtained, while the second one when only one reward was received and therefore the WSLS strategy is still undetermined. Depending on the values of $f_{0,1}(\emptyset)$, $f_{0,1}(0)$, and $f_{0,1}(1)$ these states determine what the first WSLS-motivated human action is.

In what remains we refer to this kind of systems as TAFC/WSLS systems, and refer to N as their order. We say a reward structure F is compatible with the system S if the horizon used to compute the average in the second entry of F is the same as the order of the system S.

IV. Design of Rewards

We can now consider the system S as the system we wish to control by an adequate design of reward functions. It is worth noting also that the actual values of the rewards are actually not important, according to the WSLS model, as the human making decisions only uses ordering information. Thus, all that one needs to specify when designing rewards is a preorder for the set $U \times A$. We represent the preorder $\geq_Y$ as a relation $G \subseteq Y$, such that for a pair $(y_1, y_2) \in Y$, $y_1 \geq_Y y_2$ if and only if $(y_1, y_2) \in G$. Then, we can construct from this preorder also a partial ordering of the image of F:

$$\begin{array}{rcl} \forall (y, y') \in Y,\ y \geq_Y y' & \Leftrightarrow & F(y) \geq F(y'), \\ (y' \geq_Y y) \wedge (y' \geq_Y y) & \Leftrightarrow & F(y) = F(y'), \end{array} \quad (4)$$

where the inequality and equality on the righthand side of the equivalences must be interpreted under the usual ordering of the real numbers, in which we assume the rewards live. Thus, the design of a reward function can be reduced to the design of a preorder relation G such that at each state $x \in X$, the corresponding output $H(x)$ is in G if the input to be applied at that state needs to be $u = 1$, and $(H_2(x), H_1(x)) \in G$ if the input needs to be $u = 0$, to satisfy the specification. We remind the reader that u does not denote the decision taken, but rather if the current decision was the same as the previous one (1) or not (0).

A. State-feedback controller synthesis limitations

The TAFC/WSLS model for human decision making is constructed under the assumption that the human never stops taking decisions. This is reflected in our modeling in the form of a system S that is non-blocking, *i.e.* for every reachable state $x \in X$ there exists at least one possible transition: $(x, u, x') \in \longrightarrow$. This, in turn, limits the kind of problems we can solve to those admitting non-blocking solutions.

In essence, the design of reward structures (or their verification) is an exercise of transforming state-feedback controllers for the system S into a particular output-feedback controller structure, namely the WSLS strategy with rewards F. By its nature, this output-feedback structure requires that the state-feedback controllers designed are also memoryless. Nevertheless, because the system S is itself a memory register, increasing the order N of the model allows for more complex specifications, alleviating this limitation.

Taking into account the limitations mentioned above, we concentrate our efforts on the design/verification of reward structures for three kind of specifications:

- safety - the system must remain in a safe set of states;
- reach and stay - the system eventually must enter in a target set of states and stay in thereafter.
- reach and stay while stay - the system eventually must enter in a target set of states and stay in thereafter, and during the transient should always remain in some other safe set.

The first of these specifications can be expressed in terms of temporal logics, see *e.g.* [15], by the formula: $\Box Z$, while the second one is described by the formula $\Diamond\Box Z$ and the third by $\Diamond\Box Z \wedge \Box W$, where $Z, W \subseteq X$.

These problems admit maximally permissive controllers without memory [12]. This last statement essentially means that one can obtain a controller for the system that only "disables transitions" of the original system that could lead to violations of the formal specification and no more than those.

Note that, while the simple safety problem $\Box Z$ requires a non-blocking solution, *i.e.* solutions in which the system/human keeps taking transitions/decisions, if one considers a simple reachability specification, *i.e.* $\Diamond Z$, this would not be the case. In the case of simple reachability problems, a controller would, in general, disable any further transitions once the set Z is reached. By considering, instead, the specification $\Diamond\Box Z$ this problem is avoided.

A controller for this kind of problems can be defined as a system $S_c = (X, \bar{X}_0, U, \xrightarrow[c]{}, Y, H)$, where $\xrightarrow[c]{} \subseteq \xrightarrow{}$ and $\bar{X}_0 \subseteq X_0$. System S_c, essentially, indicates at each state which inputs can be applied to satisfy the given specification. For more details, we refer the interested reader to [12]. Note that, for the problem at hand, one would generally only accept as solutions to the problem controllers for which $\bar{X}_0 = X_0$. We will not show in the current paper how controllers for these problems are synthesized, instead we refer the interested reader again to the book [12], and we will concentrate our attention in how to synthesize or verify reward structures assuming that we can synthesize the state-feedback controllers.

Definition 4.1: Given a reward function F compatible with a TAFC/WSLS system

$$S = (X, X_0, U, \xrightarrow{}, Y, H),$$

we denote by $S(F)$ the controller induced by F, *i.e.* the system

$$S(F) = (X, X_0, \xrightarrow[F]{}, Y, H)$$

with $\xrightarrow[F]{}$ given by:

$$\begin{aligned}\xrightarrow[F]{} \quad = \quad & \{(x, u, x') \in \xrightarrow{} \mid \\ & ((F(H_1(x)) \geq F(H_2(x))) \wedge u = 1) \\ & \vee ((F(H_1(x)) < F(H_2(x))) \wedge u = 0)\}.\end{aligned}$$

Intuitively, the system $S(F)$ is the system describing the sequence of decisions a human would exhibit when receiving the rewards indicated by F in a TAFC task if he/she follows the WSLS strategy.

B. Transforming controllers into static reward functions

Assume now that a memoryless, maximally permissive, non-blocking controller S_c is available to solve one of the aforementioned specifications. Given such a controller, we want to implement it through a reward function like the ones described earlier for the WSLS human decision model. Before proceeding further, we need to introduce some auxiliary sets to make the notation more concise.

Definition 4.2: Given a controller S_c for a TAFC/WSLS system S we define the following auxiliary sets:

$$\begin{aligned}
Q_1 &:= H(\Pi_X(\xrightarrow[c]{1})), && (5)\\
Q_0 &:= H(\Pi_X(\xrightarrow[c]{0})), && (6)\\
Q_{0,1} &:= H(\Pi_X(\xrightarrow[c]{0}) \cap \Pi_X(\xrightarrow[c]{1})), && (7)\\
Q_{e1} &:= Q_1 \backslash Q_{0,1}, && (8)\\
Q_{e0} &:= Q_0 \backslash Q_{0,1}. && (9)
\end{aligned}$$

In order to achieve our goal, first it needs to be checked if this controller can be implemented as an output feedback, that is if the following holds:

$$H(x) = H(x') \Rightarrow U(x) \cap U(x') \neq \emptyset. \tag{10}$$

Using the sets introduced above, this is equivalent to checking $Q_{e0} \cap Q_{e1} = \emptyset$.

If the controller is indeed implementable as an output feedback controller, next we need to check that a preorder G can be constructed from S_c. In order to be able to construct a preorder, one first needs to check if reflexivity can be respected. This is done by verifying the following:

$$\forall (y_1, y_2) \in Y \text{ s.t. } y_1 = y_2 \ : \ (y_1, y_2) \notin Q_{e0}. \tag{11}$$

Note that we do not require that such pairs are in Q_1 but rather that they are not in Q_{e0} as, depending on the specification, certain outputs in Y might never be seen, *i.e.* the states generating such outputs are not reachable.

Finally, one must check if transitivity can also be respected. In order to perform this test, one can construct the transitive closure of the sets Q_{e0} and Q_{e1}, *i.e.* $Q_{e0}^{=}$ and $Q_{e1}^{=}$, and then check if these sets are compatible, *i.e.*:

$$Q_{e0}^{=} \cap Q_{e1}^{=} = \emptyset. \tag{12}$$

We would like to remark that (12) is automatically violated if (10) is not satisfied, and therefore, one can just check this last condition. If this last condition is also satisfied then a preorder G can be constructed as:

$$G := ((Q_{e0}^{=})^{-1} \cup Q_{e1}^{=})^{*}, \tag{13}$$

where we remind the reader that by R^* we denote the reflexive closure of the relation R.

To construct from the preorder G a concrete numerical reward function, one can use any efficient ordering algorithm, as *e.g.* Quicksort [16], to place in order (according to G[3]) the elements of $U \times A$. Then, once the set $U \times A$ is ordered

[3]Replacing every comparison of the form $a \geq b$ by $(a, b) \in G$ in the ordering algorithm selected.

one can give numerical values to the image of each of the elements in $U \times A$.

Note that, when giving numerical values, one can try to impose the antisymmetry property on the image of F, as shown in equation (4), or ignore it. Given that the WSLS strategy produces the same outcome wether the last reward was strictly larger or larger or equal than the previous reward, imposing antisymmetry or not has no effect on the correctness of the reward structure.

Let us denote by $v \in (U \times A)^{|U \times A|}$ the ordered vector containing in each entry v_i the point $y \in U \times A$, such that: $v_{i-1} \geq_Y v_i \geq_Y v_{i+1}$. Then, the reward function can be constructed in its most general form, imposing also antisymmetry, following the simple algorithm 1.

```
**Data**: $v$, $\eta, G$
**Result**: $F$
$j = \eta$;
$F(v_1) = \eta$;
**for** $i = 2 : |U \times A|$ **do**
    **if** $(v_i, v_{i-1}) \notin G$ **then**
        $j = j + 1$;
    **end**
    $F(v_i) = j \cdot \eta$;
**end**
```

Algorithm 1: Concretization of a reward function.

We introduce in algorithm 1 the parameter η to control how large are the increments/decrements of reward between different states, similarly to what was proposed in [9]. Obviously, one could also shift the whole reward function by a constant factor.

Finally, note that in general there might be many possible vectors v respecting that $v_{i-1} \geq_Y v_i \geq_Y v_{i+1}$. This is the case because we have not imposed at any point that our preorder needs to be complete, and thus there might be pairs of elements in $U \times A$ that cannot be compared. In fact, those pairs that are not comparable, are irrelevant for the correctness of the controller because either they belong to $Q_{0,1}$ or should never appear as an output of a correct execution of the system. A typical implementation of the Quicksort algorithm with random selection of the pivoting element generates different v ordered vectors, and as a result also different reward structures F, all of which make $S(F)$ satisfy the prescribed specification. Another possibility to generate more possible reward structures is provided by the set $Q_{0,1}$. One could decide to move some elements from $Q_{0,1}$ to Q_0 or Q_1 and repeat the process to obtain yet more valid reward structures.

We can summarize now the main result of the paper in the following theorem:

Theorem 4.3: Assume that a memoryless, non-blocking, maximally permissive controller S_c is available to force a TAFC/WSLS system S to satisfy a given specification of the form: $\Box Z$, $\Diamond\Box Z$ or $\Diamond\Box Z \wedge \Box W$. If the controller S_c satisfies conditions (11) and (12), a reward function $F : U \times X \to \mathbb{R}_0^+$ of the form (2) can be constructed, as establish through (13) and algorithm 1, such that the system $S(F)$ is a controller for S enforcing the same specification as S_c.

Proof: First note that S_c is a maximally permissive controller, and $S(F)$ only offers one possible input at each state, *i.e.* $\forall x \in X$, $|U(x)| = 1$. Therefore, in order for $S(F)$ to satisfy the same specification as S_c is is enough to show that:

$$\forall x \in \Pi_X(\xrightarrow[c]{}), : (x, u, x') \in \xrightarrow[F]{} \Rightarrow (x, u, x') \in \xrightarrow[c]{} \qquad (14)$$

That is, we need to show that for every reachable state in S_c, $S(F)$ provides an input that is also available in S_c. By construction, at every $x \in \Pi_X(\xrightarrow[c]{})$ where the only available input is 1 $F(H_1(x)) \geq F(H_2(x))$ holds. Similarly $F(H_1(x)) < F(H_2(x))$ at those states of S_c where the only available input is 0. This means, that at least at those states where only 1 or 0 are possible inputs the condition (14) is satisfied. At any other state S_c offers either the two inputs as a possibility or the state is not reachable, and thus whatever the input that $S(F)$ provides, the condition (14) is respected. ■

As a straightforward corollary, we can also perform verification of a given reward function by simply constructing the system $S(F)$ and designing a maximally permissive controller S_c for $S(F)$ to satisfy the specification to be verified. If $\xrightarrow[c]{} = \xrightarrow[F]{}$ then the property is satisfied, and otherwise, *i.e.* $\xrightarrow[c]{} \subset \xrightarrow[F]{}$, the reward function F does not enforce the desired behavior on the decision-maker.

V. Example

We implemented the proposed approach in Matlab, and employed the toolbox Pessoa [17] to synthesize the state-feedback controllers. Following the design procedure outlined in Section IV we synthesized reward functions for a specification of the form $\Diamond\Box Z \wedge \Box W$. The order selected for system S was $N = 6$ and the sets Z and W where specified as follows:

$$Z = \{x \in X \,|\, a(x_{[2,\ldots,s(x)-1]}) \in [0.33, 0.66]\}$$

$$W = \{x \in X \,|, \sum_{i=1}^{s(x)} 2^{s(x)-i} \leq 235\}$$

where $s(x)$ denotes the dimension of $x \in X$; *i.e.* we require that the number represented by the binary encoding of the last 6 decisions never exceeds 235 and its average eventually enters the range $[0.33, 0.66]$ (and remains there).

The resulting reward structure can be seen in Figure 1, in which the dotted line represents f_1 and the dashed one f_0, and the values $f_1(-1)$ is denoted by an asterisk in the vertical axis, and $f_0(-1)$ as a square in the vertical axis.

Next, we generated another reward structure by forcing as unique input $u = 1$ at all states where the controller S_c offered both inputs, *i.e.* we force $Q_{e1} = Q_1 \cup Q_{0,1}$. In general, doing this leads to controllers that are not implementable as a reward function, however in this case it resulted in the reward function shown in Figure 2.

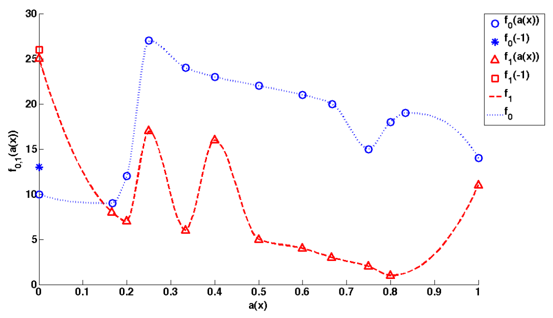

Fig. 1. Reward structure for $\Diamond\Box Z \wedge \Box W$.

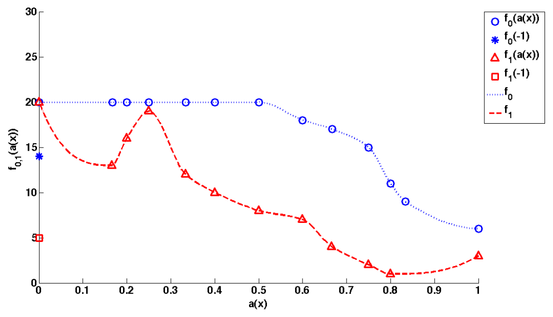

Fig. 2. Reward structure for $\Diamond\Box Z \wedge \Box W$ with $Q_{e1} = Q_1 \cup Q_{0,1}$.

In both figures, the circles and triangles denote the values of the rewards that the human can actually obtain, that is those in the finite set $U \times A$, while the lines connecting them were constructed by interpolation to provide figures more similar to those in the literature studying the WSLS model.

VI. Conclusions and Future Work

We have proposed an approach to the design of rewards for humans taking decisions in a two choice sequential task. The reward functions generated guarantee, for a deterministic model of human decision, that the sequence of choices taken by the human subject satisfy a prescribed specification. As a side result, the same procedure can be employed to verify a reward function against a specification for the human behavior. We note that, while in the current paper we only concentrated on the simple problems of safety and reachability, it should be possible to devise similar solutions for more general specifications given in some temporal logic, like *e.g.* LTL, by appropriately enlarging the order N of the system in which the decision model is embedded.

We are aware of the great limitations that assuming a deterministic model for human decision making pose from a practical perspective. Thus, a natural step for future work is to extend the proposed approach to non-deterministic, possibly stochastic, models. One can start by enriching the human decision making model with probabilities and apply the presented approach to verify properties when a reward structure is given. However, a similar extension to reward function synthesis does not seem to have a simple answer, and will require a more detailed investigation. In this same direction, experimenting with human subjects following the proposed approach is an interesting venue, not just to validate the proposal, but also to guide how to enrich the model with the right uncertainties to make the approach truly practical. Regardless of this, one can still consider the approach interesting in certain applications in which one might want to provide very simple instructions to a human operator, *e.g.* apply the WSLS strategy, to control a given device.

Finally, our next goal is to study the interaction of this kind of human decision making modeling with physical processes, to devise integrated approaches to the design of *cyborg-physical systems*: systems in which a human and a machine collaborate to control a physical process.

References

[1] M. Rahimi and W. Karwowski, *Human-robot interaction*. London: Taylor-Francis, 2004.

[2] B. Peerdeman et al., “Myoelectric forearm prostheses: state of the art from a user-centered perspective,” *Journal of Rehabilitation Research and Development*, vol. 48, pp. 719–738, 2011.

[3] Edited by J. Baillieul, N. E. Leonard, and K. A. Morgansen, “Interaction dynamics: The interface of humans and smart machines,” *Proceedings of the IEEE*, vol. 100, 2012, special issue.

[4] A. Stewart, M. Cao, A. Nedic, D. Tomlin, and N. E. Leonard, “Towards human-robot teams: Model-based analysis of human decisin making in two-alternative choice tasks with social feedback,” *Proceedings of the IEEE*, vol. 100, pp. 751–775, 2012.

[5] J. Li, S. M. McClure, and P. R. Montague, “A computational role for dopamine delivery in human decision-making,” *Journal of Cognative Neuroscience*, vol. 10, pp. 623–630, 1998.

[6] P. R. Montague and G. S. Berns, “Neural economics and the biological substrates of valuation,” *Neuron*, vol. 36, pp. 265–284, 2002.

[7] M. Cao, A. Stewart, and N. E. Leonard, “Integrating human and robot decision-making dynamics with feedback: Models and convergence analysis,” in *Proc. of the 47th IEEE Conference on Decision and Control*, 2008, pp. 1127–1132.

[8] ——, “Convergence in human decision-making dynamics,” *Systems and Control Letters*, vol. 59, pp. 87–97, 2010.

[9] C. Woodruff, L. Vu, K. A. Morgansena, and D. Tomlin, “Deterministic modeling and evaluation of decision-making dynamics in sequential two-alternative forced choice tasks,” *Proceedings of the IEEE*, vol. 100, pp. 734–750, 2012.

[10] H. Robbins, “Some aspects of the sequential design of experiments,” *Bulletin of American Mathematical Society*, vol. 58, pp. 527–535, 1952.

[11] M. Nowak and K. Sigmund, “A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner’s Dilemma game,” *Nature*, vol. 364, pp. 56–58, 1993.

[12] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2009.

[13] L. Vu and K. Morgansen, “Modeling and analysis of dynamic decision making in sequential two-choice tasks,” in *Proc. of the 47th IEEE Conference on Decision and Control*, 2008, pp. 527–535.

[14] C. Woodruff, L. Vu, K. Morgansen, and D. Tomlin, “Deterministic modeling and evaluation of decision-making dynamics in sequential two-alternative forced choice tasks,” *Proceedings of the IEEE*, vol. 100, no. 3, pp. 734 –750, march 2012.

[15] C. Baier and J. Katoen, *Principles of model checking*. The MIT Press, 2008.

[16] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, ser. Mit Electrical Engineering and Computer Science Series. MIT Press, 1990.

[17] M. Mazo Jr., A. Davitian, and P. Tabuada, “Pessoa: A tool for embedded controller synthesis.” in *CAV*, ser. Lecture Notes in Computer Science, T. Touili, B. Cook, and P. Jackson, Eds., vol. 6174. Springer, 2010, pp. 566–569.