On Quantitative Software Quality Assurance Methodologies for Cardiac Pacemakers

Marta Kwiatkowska, Alexandru Mereacre, and Nicola Paoletti

Department of Computer Science, University of Oxford, UK

Abstract. Embedded software is at the heart of implantable medical devices such as cardiac pacemakers, and rigorous software design methodologies are needed to ensure their safety and reliability. This paper gives an overview of ongoing research aimed at providing software quality assurance methodologies for pacemakers. A model-based framework has been developed based on hybrid automata, which can be configured with a variety of heart and pacemaker models. The framework supports a range of quantitative verification techniques for the analysis of safety, reliability and energy usage of pacemakers. It also provides techniques for parametric analysis of personalised physiological properties that can be performed *in silico*, which can reduce the cost and discomfort of testing new designs on patients. We describe the framework, summarise the results obtained, and identify future research directions in this area.

Keywords: model-based design; quantitative verification; hybrid automata; heart modelling; cardiac pacemakers.

1 Introduction

The growing reliance on implantable medical devices controlled by embedded software calls for rigorous software design methodologies to ensure their safe operation and to avoid costly device recalls. We focus here on cardiac pacemakers, which are battery-powered devices implanted under a patient's skin that sense the electrical signals in the heart and regulate the heart rhythm. Of paramount importance here is the safety of the device's operation, but analysis of characteristics such as energy usage are also needed to improve the designs. An important observation is that evaluating the operation of the pacemaker must take into account the characteristics of the heart rhythm of the patient, and therefore personalisation of the methodology is desirable.

Several models for pacemakers have been proposed, to mention [21, 24, 10, 19, 25, 16, 17]. Since the basic function of the pacemaker is to maintain a normal heart rhythm of 60-100 beats per minute (BPM), the models need to capture real-time, in addition to being able to sense electrical signals, typically (non-linear) continuous flows. Therefore, natural models for the pacemaker are (deterministic) timed or hybrid automata, which are then composed with a heart model, typically a hybrid automaton, for the analysis. An important consideration in our work has been *stochasticity*, which manifests itself in several ways: sensor noise, modulated rate in response to activity level, as well as the randomness in the timing of the heart beats, which is specific to the patient and can switch between normal and diseased behaviours. We have thus concentrated our efforts on developing effective methodologies to provide software quality assurance for pacemakers in presence of stochasticity through *quantitative verification* techniques.

This paper reports on a comprehensive model-based framework to provide software quality assurance for cardiac pacemakers developed within the VERIWARE and VERIPACE projects and described in [4–6, 18]. The framework is based on hybrid input-output automata models, and can be instantiated with a number of heart models, including a model based on synthetic ECG that can be learnt from patient data and a physiologically-relevant heart model built as a network of cardiac cells. Models of pacemakers of differing functionalities can be plugged into the framework for analysis: we consider a basic pacemaker design inspired by [17], an advanced design that can handle pacemaker mediated tachycardia, as well as a rate-adaptive pacemaker. We implement the framework in Simulink and provide a broad range of analysis techniques, which are based on simulation, as well as approximate quantitative verification, for checking safety, reliability and detailed energy-usage. We also develop analysis methods for advanced physiological properties, including pacemaker mediated tachycardia correction and parametric analysis to support *in silico* testing of the rate-modulation functionality under different personalised scenarios, e.g., age of the patient and activity level. We demonstrate the usefulness of our methodology through a range of experiments. Finally, we summarise future research directions and challenges in this area.

2 Model-based Framework for the Verification of Pacemakers

Our framework for modelling and quantitative verification of pacemaker models is based on the formalism of *hybrid input-output automata* [20], and supports the composition of a heart model and a pacemaker model on which verification is performed. We consider a discrete-time simulation semantics, which enables a sound and straightforward encoding of the formal specification into MATLAB Simulink/Stateflow models. In the following, we recall the basic details of the framework that we introduced in [6].

Let $\mathcal{X} = \{x_1, \ldots, x_d\}$ be a set of variables in $\mathbb{R}$. An $\mathcal{X}$-valuation is a function $\eta : \mathcal{X} \to \mathbb{R}$ assigning to each variable $x \in \mathcal{X}$ a real value $\eta(x)$. Let $\mathcal{V}(\mathcal{X})$ denote the set of all valuations over $\mathcal{X}$. A *constraint* on $\mathcal{X}$, denoted by grd, is a conjunction of expressions of the form $x \bowtie c$ for variable $x \in \mathcal{X}$, comparison operator $\bowtie \in \{<, \leq, >, \geq\}$ and $c \in \mathbb{R}$. Let $\mathcal{B}(\mathcal{X})$ denote the set of constraints over $\mathcal{X}$. Let $\mathcal{Y}(\mathcal{X})$ denote the set of all real-valued functions over $2^{\mathcal{X}}$. We define $\mathcal{L}(\mathcal{X}) := \{x := u \mid x \in \mathcal{X} \wedge u \in \mathcal{X} \cup \{0\}\}$ to be the set of *update* assignments over the set of variables $\mathcal{X}$.

Definition 1 (Hybrid I/O Automaton). *A hybrid I/O automaton (HIOA)* $\mathcal{A} = (\mathcal{X}, Q, q_0, E_1, E_2, \mathrm{Inv}, \rightarrow, \mathrm{Diff})$ *consists of:*

- *a finite set of variables* $\mathcal{X}$*;*
- *a finite set of modes* Q*, with the initial mode* $q_0 \in Q$*;*
- *a finite set* E_1 *of* input *actions and a finite set* E_2 *of* output *actions with* $\mathcal{E} = E_1 \cup E_2$*;*
- *an invariant function* $\mathrm{Inv} : Q \rightarrow \mathcal{B}(\mathcal{X})$*;*
- *a transition relation* $\rightarrow \subseteq Q \times (\mathcal{E} \cup \{\varsigma\}) \times \mathcal{B}(\mathcal{X}) \times 2^{\mathcal{L}(\mathcal{X})} \times Q$*, where* ζ *is the internal action; and*
- *a derivative function* $\mathrm{Diff} : Q \times \mathcal{X} \rightarrow \mathcal{Y}(\mathcal{X})$ *that assigns a function to a variable* $x \in \mathcal{X}$*.*

We use a *network of HAs* for the composition of more than one HA. In order to obtain a deterministic network we impose some restrictions on HAs as follows:

- they must be *input enabled*, meaning that, for each mode and each input action, there is an edge labelled by the input action;
- the output actions have the highest priority, meaning that they are always *urgent*, i.e., if at any state the output action is enabled, the system must execute that action;
- the input actions are never enabled unless the corresponding output actions from the environment synchronise with them: once they can be synchronised, they are urgent;
- for each mode, there is a self-loop labelled by the internal action.

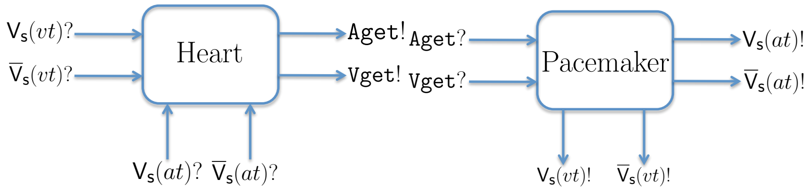
Definition 2 (Network of hybrid automata). *Let* m *be the number of HAs in the network. A state of the network is* $\big((q^{(1)}, \eta^{(1)}), \cdots, (q^{(m)}, \eta^{(m)})\big)$*. There is a transition*

$$\Big((q_i^{(1)}, \eta_i^{(1)}), \ldots, (q_i^{(m)}, \eta_i^{(m)})\Big) \rightarrow \Big((q_{i+1}^{(1)}, \eta_{i+1}^{(1)}), \ldots, (q_{i+1}^{(m)}, \eta_{i+1}^{(m)})\Big),$$

where

- *either, for each* $1 \leq k \leq m$*,* $(q_i^{(k)}, \eta_i^{(k)})$ *has a continuous evolution;*
- *or, for each* $1 \leq k \leq m$*,* $(q_i^{(k)}, \eta_i^{(k)})$ *has a discrete transition. If, for some* k*,* $(q_i^{(k)}, \eta_i^{(k)})$ *enables an output action* $a \in E_2^{(k)}$*, then all the other* $(q_i^{(k')}, \eta_i^{(k')})$ *must take a corresponding input action* $a \in E_1^{(k')}$ *(notice that this is guaranteed by input enabledness); otherwise, each state evolves by taking the internal action.*

We assume in our framework that both the heart model and the pacemaker model are specified as hybrid input-output automata. To allow user-specified models, we define fixed component interfaces for the heart and pacemaker models, as shown in Figure 1. The heart and the pacemaker communicate via input and output actions which are marked by ? and ! respectively. The pacemaker communicates with the heart through four output actions, $\mathsf{V}_{\mathsf{s}}(at)$!, $\overline{\mathsf{V}}_{\mathsf{s}}(at)$!, $\mathsf{V}_{\mathsf{s}}(vt)$! and $\overline{\mathsf{V}}_{\mathsf{s}}(vt)$!. The actions $\mathsf{V}_{\mathsf{s}}(at)$! and $\overline{\mathsf{V}}_{\mathsf{s}}(at)$! denote the beginning and the end of the *atrial* stimulus, respectively, while $\mathsf{V}_{\mathsf{s}}(vt)$! and $\overline{\mathsf{V}}_{\mathsf{s}}(vt)$! denote the beginning and end of the *ventricle* stimulus. The heart communicates with the pacemaker using two output actions `Aget`! and `Vget`!.


$\mathsf{V_s}(vt)$?
$\overline{\mathsf{V}}_{\mathsf{s}}(vt)$?
Heart
Aget!
Vget!
$\mathsf{V_s}(at)$? $\overline{\mathsf{V}}_{\mathsf{s}}(at)$?
Aget?
Vget?
Pacemaker
$\mathsf{V_s}(at)$!
$\overline{\mathsf{V}}_{\mathsf{s}}(at)$!
$\mathsf{V_s}(vt)$! $\overline{\mathsf{V}}_{\mathsf{s}}(vt)$!

(a) Heart model (b) Pacemaker model

Fig. 1. Interfaces for the heart and pacemaker models.

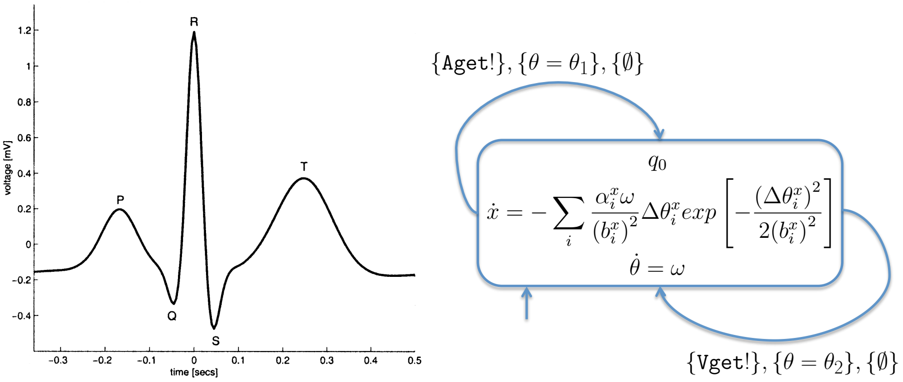
3 Heart Modelling

In this section we present two heart models, the ECG and the cardiac cell network model, and we show how they can be connected to the pacemaker and integrated within the overall verification framework. Each heart model has its own advantages and disadvantages. For instance, the ECG heart model can be easily adapted to a given patient, whereas the cardiac cell heart model is more physiologically relevant. By providing a common interface to the pacemaker, we can effectively evaluate and compare the behaviour of multiple heart models in a modular fashion.

3.1 The ECG heart model

This heart model is based on synthetic ECG rhythms developed by Clifford *et al.* [8]. An ECG is a signal recorded from the surface of the human chest, which describes the activity of the heart. The ECG signal is an approximation of the electrical activity inside the human heart. An example ECG is given in Figure 2(a).

Typically, an ECG signal describes a cardiac cycle composed of three main waves, P, QRS and T. The P wave denotes the *atrial depolarisation*. The QRS wave reflects the rapid *depolarisation of the right and left ventricles*. The T wave denotes the *repolarisation of the ventricles*. In Figure 2(b) we present the hybrid automaton for the ECG heart model. It is based on a system on nonlinear ODE with two variables $x(t)$ (the value of the ECG signal at time t) and θ. Here θ_1 represents the beginning of the P wave; θ_2 represents the beginning of the Q wave; α_i^x and b_i^x, respectively, are the amplitude and width of the Gaussian functions used to model the ECG; $\theta \in [-\pi, \pi]$ is the *cardiac phase*; $\Delta\theta_i^x = (\theta - \theta_i^x) \bmod 2\pi$; and $\omega = \frac{2\pi h}{60\sqrt{h_{av}}}$ is the *angular velocity*, where h is the *instantaneous (beat-to-beat) heart rate* in BPM and h_{av} is the mean of the last n heart rates (typically with $n = 6$) normalized by 60 BPM. To use the ECG heart model one has to define the instantaneous (beat-to-beat) heart rate function $h(t)$ ($t \in \mathbb{R}_{\geq 0}$), which specifies the distance between two consecutive R-events (highest peak in Figure 2(a)). Technically, it is equivalent to the so called *RR-series* $\chi(n)$, $n \in \{1, \ldots, N\}$, where N denotes the length of the series.


R
P
T
Q
S
voltage [mV]
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
time [secs]
{Aget!}, $\{\theta = \theta_1\}$, $\{\emptyset\}$
q_0
$\dot{x} = -\sum_i \frac{\alpha_i^x \omega}{(b_i^x)^2} \Delta\theta_i^x exp\left[-\frac{(\Delta\theta_i^x)^2}{2(b_i^x)^2}\right]$
$\dot{\theta} = \omega$
{Vget!}, $\{\theta = \theta_2\}$, $\{\emptyset\}$

(a) Example electrocardiogram [23]

(b) ECG hybrid automaton

Fig. 2. ECG heart model.

The value of $\chi(n)$ denotes the time between two consecutive heart beats. More details on the construction of the function $h(t)$ can be found in [23].

3.2 The cardiac cell heart model

This heart model is based on modelling the *electrical conduction system (ECS)* of the heart (see [6]). The ECS is a network of nerves whose role is to propagate the *action potential (AP)* through the heart tissue. We abstract the conduction system as a network of cardiac cells, a model that is both physiologically meaningful and computationally tractable (in [6] we model a network of 33 cells). The ECS of the heart consists of conduction pathways with different *conduction delays*. Cells are connected by pathways. The delays of the pathways depend on the physiology of the tissue considered, and can be tuned to reproduce various tissue diseases.

Our model consists of the SA node, whose role is to generate sequences of AP signals which are propagated through the ECS of the heart, and 32 cells that share similar properties.

The cell model in Figure 3, taken from [26], consists of four modes, each associated with an AP phase: *resting and final repolarisation* (q_0), *stimulated* (q_1), *upstroke* (q_2), and *plateau and early repolarisation* (q_3). The cell model is characterised by two timed periods: effective refractory period (ERP) is the time period where the cell cannot be stimulated and relative refractory period (RRP) is the time period where a secondary excitation event is possible.

The variables of the model are: the membrane voltage v, which controls mode switches; i_{st}, which is the stimulus current; and a restitution-related variable v_n, used to modify the next ERP phase upon a new round of excitation. Specifically, this is achieved through the function $f(\lambda) = 1 + 13\sqrt[6]{\lambda}$ (mode q_3), where $\lambda = \frac{v_n}{V_R}$ and V_R is a model-specific constant called *repolarisation voltage* [26].

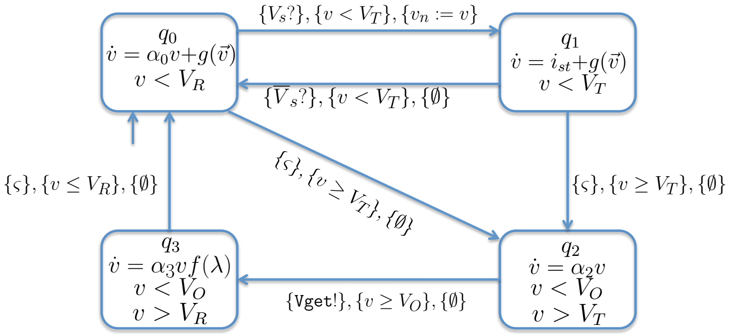

$\{V_s?\}, \{v < V_T\}, \{v_n := v\}$
q_0
$\dot{v} = \alpha_0 v + g(\vec{v})$
$v < V_R$
q_1
$\dot{v} = i_{st} + g(\vec{v})$
$v < V_T$
$\{\overline{V}_s?\}, \{v < V_T\}, \{\emptyset\}$
$\{\varsigma\}, \{v \leq V_R\}, \{\emptyset\}$
$\{\varsigma\}, \{v \geq V_T\}, \{\emptyset\}$
$\{\varsigma\}, \{v \geq V_T\}, \{\emptyset\}$
q_3
$\dot{v} = \alpha_3 v f(\lambda)$
$v < V_O$
$v > V_R$
q_2
$\dot{v} = \alpha_2 v$
$v < V_O$
$v > V_T$
$\{\texttt{Vget!}\}, \{v \geq V_O\}, \{\emptyset\}$

Fig. 3. Hybrid automaton for a ventricular cardiac cell.

We denote with $\boldsymbol{v} = [v_1 \dots v_N]^T$ the vector of the membrane voltages of a network with N cells. We define a function $g_k(\boldsymbol{v})$ to express the voltage contribution to a cell k from the neighbouring cells, as follows:

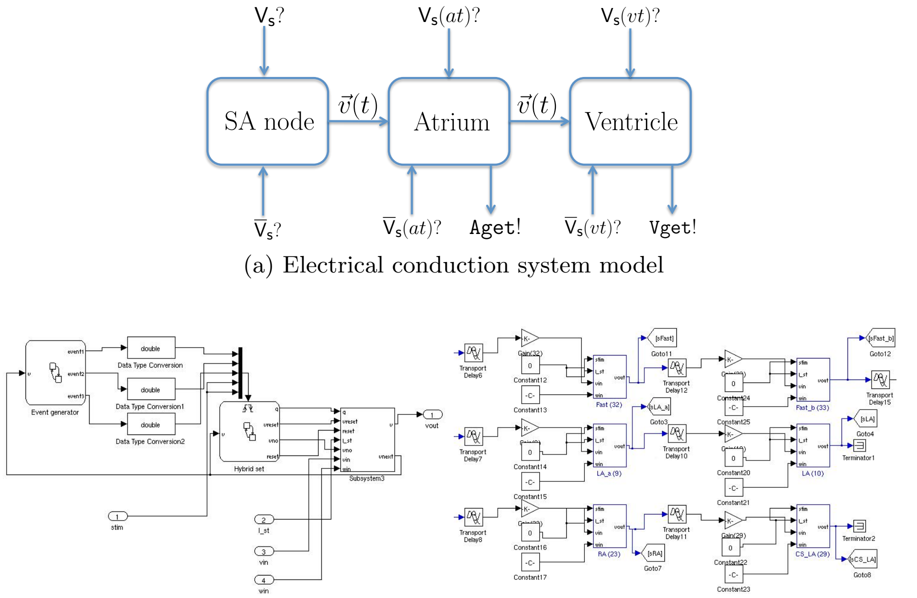
$$g_k(\boldsymbol{v}) = \sum_{i=1, i \neq k}^{N} v_i(t - \delta_{ki}) \cdot a_{ki} - v_k \cdot d_k, \tag{1}$$

where a_{ki} is the gain applied to the potential v_i from cell i, δ_{ki} is the time it takes for the potential to reach cell k, and d_k is the distance coefficient. These coefficients depend on the conduction system, and in particular on the conduction delays.

In Figure 4(a) we depict three blocks representing the connection of cells in the ECS. This component provides a template suitable for potentially including any kind of multi-cellular model of the cardiac tissue, and defines the interface with the pacemaker model.

Every cell in the atrium and the ventricle blocks can be stimulated by the pacemaker using the input actions $\mathsf{V}_\mathsf{s}(at)?$, $\overline{\mathsf{V}}_\mathsf{s}(at)?$ and $\mathsf{V}_\mathsf{s}(vt)?$, $\overline{\mathsf{V}}_\mathsf{s}(vt)?$, respectively. The output actions `Aget!` and `Vget!` notify the pacemaker that the AP in the atrium and the ventricle (where the pacemaker leads are inserted) have reached a given threshold. The function $\boldsymbol{v}(t)$ is the output voltage from a given cell, which is the endpoint of the source block.

Figure 4(b) shows the Simulink implementation of a cardiac cell, which is given by three main blocks: *Event generator*, *Hybrid set* and *Subsystem*. The *Event generator* block is responsible for generating the input events to the cell. The *Hybrid set* implements the cell hybrid automaton model (see Fig. 3). The *Subsystem* block performs the integration procedure to compute the voltage level of the cell. In Figure 4(c), a simplified network of six cells is depicted. Each cell block is composed from the three sub-blocks shown in Figure 4(b) and connected to other cells through delay and gain components.



(a) Electrical conduction system model

(b) Cell block

(c) Cell connection

Fig. 4. Cardiac cell model

3.3 Switching between Different Heart Behaviours

The introduced heart models can exhibit only a single heart behaviour, such as normal, bradycardia or tachycardia, which is determined by the frequency of the RR-series (and sets the firing rate of the SA node in the cardiac cell model).

However, a real human heart exhibits several spontaneous changes of heart rhythms. In [6], we reproduce such dynamics by modelling the *probabilistic transition* between three modes, imposing a Normal (N), Bradycardia (B) and Tachycardia (T) rhythm, respectively, according to a prescribed RR-series for each mode. We also assume an initial distribution $\alpha \in \text{Distr}(\{N, B, T\})$ and transition probabilities $\mathbf{P}_i \in \text{Distr}(\{N, B, T\})$ for $i \in \{1, 2, 3\}$. We want to remark that both the initial distribution and the transition probabilities between behaviours can be learned from patient data, which enables the parametrization of personalized heart models.

4 Pacemaker Modelling

In this section we provide the specification of two pacemaker models in our framework. We consider the model by Jiang et al. [17], hereafter called the *basic pacemaker*, which is specified as a network of Timed Automata (TA); and an extension of the basic pacemaker presented in [6, 18], which we call the *enhanced pacemaker*, with advanced features such as sensing noise, energy consumption, and the ability to adapt the pacing rate depending on the physical activity of the patient.

4.1 Basic pacemaker model

The pacemaker is implanted under the chest skin and sends impulses to the heart at specific time intervals. The role of the basic pacemaker is to keep the heart rhythm at a given rate. It has two leads: one for the atrium and one for the ventricle. Each lead has the ability to sense or deliver an electrical signal.

The basic pacemaker model consists of five core TA components, named according to their specific function: LRI, AVI, URI, PVARP and VRP. The *lower rate interval (LRI)* component (Fig. 5(a)) has the function of keeping the heart rate above a given minimum value. The *atrio-ventricular interval (AVI)* component (Fig. 5(c)) is designed to maintain the synchronisation between the atrial and the ventricular events. An event is when the pacemaker senses or generates an action. The AVI component also defines the longest interval between an atrial event and a ventricular event. The *post ventricular atrial refractory period (PVARP)* component (Fig. 5(b)) notifies all other components that an atrial event has occurred. The *upper rate interval (URI)* component (Fig. 5(d)) sets a lower bound on the times between consecutive ventricular events. Finally, the *ventricular refractory period (VRP)* component (Fig. 5(d)) filters noise and early events that may cause undesired behaviour.

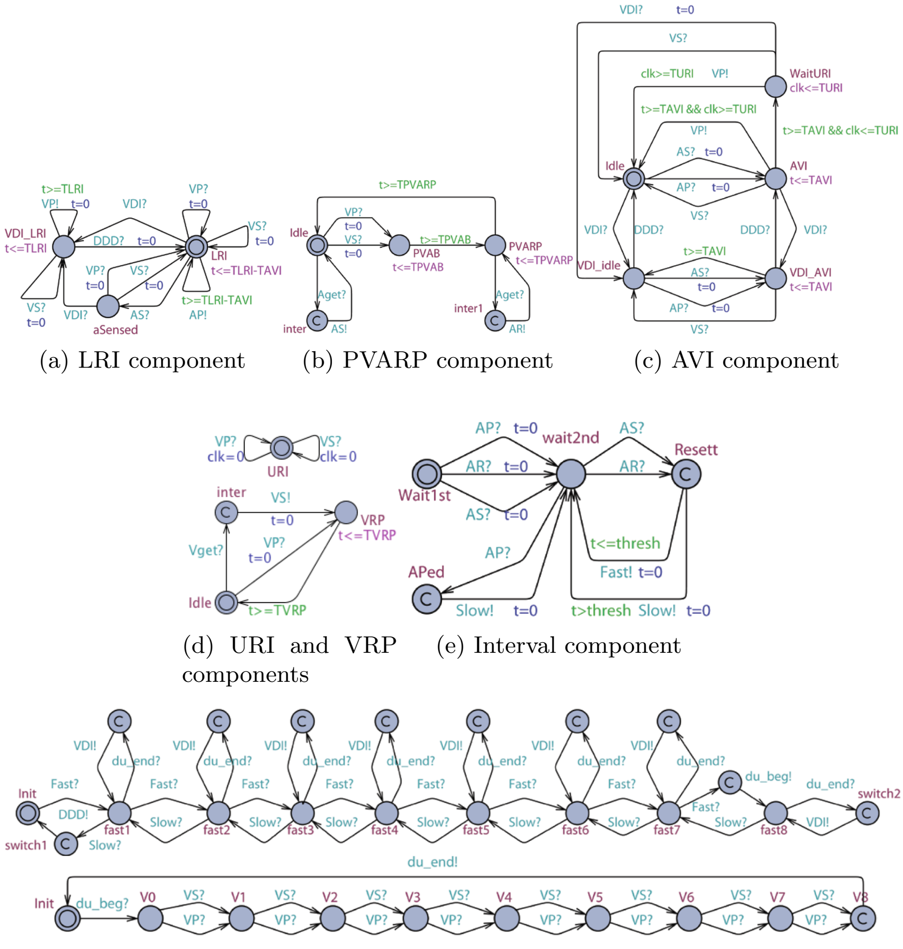
Three additional components, *Interval*, *Counter* and *Duration* (Figure 5(e) and (f)), are included in the basic pacemaker to detect and correct *pacemaker mediated tachycardia (PMT)*, an event occurring when the pacemaker increases the heart rate inappropriately. Such components switch the functioning modes of the pacemaker from DDD (pacing and sensing of the atrium and ventricle) to VDI (pacing and sensing only the ventricle). More details will be given in Section 5.1, and can be found in [17, 6].

There are four actions in the pacemaker model that serves as the interface with a generic heart model: the input actions Aget? and Vget? notify the pacemaker when there is an AP from the atrium or from the ventricle, respectively (see also Sect. 3.2), and likewise for the output actions AP! and VP! are responsible for pacing the atrium and the ventricle.

4.2 Enhanced pacemaker model

In this section, we extend the functionalities of the basic pacemaker model by considering noise, energy consumption and rate modulation through physiological sensors.

Pacing Noise. One of the important design issues of pacemakers is the need to tolerate noise. For instance, when the pacemaker tries to deliver a beat, the beat might get lost due to noise on the channel. The basic pacemaker is constructed under the simplified assumption that it can pace the heart perfectly. Here we



(a) LRI component (b) PVARP component (c) AVI component

(d) URI and VRP components (e) Interval component

(f) Counter and Duration component

Fig. 5. Timed automata of the five core pacemaker components (a,b,c,d), and of Interval, Counter and Duration components for PMT analysis (e,f). Locations labelled with **C** indicate committed locations that do not allow time to elapse.

consider a more realistic scenario, modelling the so called "failure-to-capture", a kind of sensing noise due to insufficient contact between the lead and the myocardium, or due to lead fracture [11]. In particular, we add to the fixed stimulus current i_{st} (cf. Figure 3) a *normally distributed noise* with mean μ and variance σ^2 each time the pacemaker wants to pace the cell.

In this way, if the noise added to the channel is too high, a "missing stimulus" is generated, i.e. the stimulus from the pacemaker will *not* be high enough to stimulate the cell.

Energy. Pacemaker's life time is limited and is crucially dependent on the battery embedded into the devices. When the battery depletes, the pacemaker needs to be re-implanted, and hence the analysis of energy usage and, ultimately, the design of more energy-efficient devices are indispensable.

In our framework, we consider the so called *Kinetic Battery model (KiBaM)* [22] to describe the dynamics of energy consumption. The model consists of the following system of ODEs:

$$\frac{dy_1(t)}{dt} = -\iota(t) + k\left(\frac{y_2(t)}{1-c} - \frac{y_1(t)}{c}\right), \quad \frac{dy_2(t)}{dt} = -k\left(\frac{y_2(t)}{1-c} - \frac{y_1(t)}{c}\right). \tag{2}$$

The battery charge is distributed in two wells: the *available-charge* $y_1(t)$ and the *bound-charge* $y_2(t)$. The current applied to the battery at time t is described by the function $\iota(t)$. When the value of $\iota(t)$ is zero the battery enters the recovery mode, where the energy from the bound-charge well flows to the available-charge well. This mode allows a nearly discharged battery to recover in a period of zero or low current by increasing its available-charge. When the current $\iota(t)$ is not zero, both charges $y_1(t)$ and $y_2(t)$ decay over time. The battery is considered to be empty when there is no charge in the available-charge well, i.e., $y_1(t) = 0$. For details on the composition between the KiBaM and the pacemaker model see [6].

Rate adaptive pacemaker. Physiological sensors are an essential component of the so-called *rate adaptive (RA) pacemaker*, where the pacing rate is adjusted according to the levels of activity (physical, mental or emotional) detected in the patient. RA pacemakers represent the only choice for individuals with chronotropic incompetence, that is, when the heart rate cannot naturally adapt to increasing demand (e.g. AV block). A number of different pacing methods and sensors have been developed so far [2]. However, they require extensive testing on cardiac patients especially to assess the device under varying levels of physical exercise. Our model-based framework provides an effective test-bed for these kinds of devices, where different (and possibly multiple) sensors can be integrated into available pacemaker models, and formal verification enables the automated design and debugging of rate modulation protocols in order to ensure safe behaviour of the heart under the different stress levels which the patient can undergo.

In [18], we develop a HIOA model of a *VVIR pacemaker* (sensing and pacing of the ventricle, and with rate modulation) based on a *QT interval (QTI) sensor*, a highly specific metabolic sensor that exploits the fact that physical activity shortens the QT interval (see Fig. 2a), and thus requires an increased heart rate. We implement the QT sensor through a runtime ECG detection algorithm that allows to simulate and validate the model with patient ECG data.

The *RA component* (Fig. 6) is connected to the components of the VVI pacemaker and is responsible for changing the pacing rate (TLRI) according to the signals from the QT sensor, which outputs an action $\mathsf{TE!}$ whenever a T wave is detected. To this aim, we established a relationship between QTI lengths and TLRI by means of a non-linear regression analysis performed over ECG data from the PhysioNet database [1], and described by the following equation

$$\mathsf{RR}(\mathsf{QT}) = -\frac{\log\left((a - \mathsf{QT})/b\right)}{k} \tag{3}$$

where a, b and k are the estimated regression parameters; QT is the QTI length; and RR is the RR interval length which is used to update TLRI.

From the initial state q_0, the RA component waits for a ventricle sense or pace event (Vget or VP, resp.) to start timers t_{VP} and t_{QT}. t_{VP} defines the refractory window of size T_R where the RA component disables the pacemaker inputs, while t_{QT} models the duration of the QT interval and is terminated by the synchronization with a $\mathsf{TE!}$ signals. If the obtained t_{QT} falls within an admissible interval $[T^l, T^u]$, the corresponding adapted value for TLRI is calculated through function f_{QT}, which applies the above regression law over the mean of the last four detections. This averaging mechanism ensures prompt response to fast changing QTIs and, at the same time, allows us to mitigate the effects of wrongly sensed intervals.

The ECG detection algorithm implemented in the QT sensor component relies on a signal processing algorithm based on [27, 12], and is thoroughly explained in [18]. Note that the behaviour of the QT sensor is inherently stochastic, since it processes and filters ECG signals that are subject to random noise. However, other sources of uncertainty can be incorporated, like random under- and over-sensing.

5 Pacemaker Verification

In this section, we report some experimental results obtained in the evaluation of our framework with the basic and the enhanced pacemaker models. All the following experiments have been performed with the cardiac cell model. First, we show how the pacemaker corrects bradycardia when the probability of deviating from the normal behaviour is varied, and how cases of pacemaker mediated tachycardia are solved by mode switching. Second, we evaluate the behaviour of our model under different levels of sensing noise; we analyse the energy consumption of the pacemaker and its dependence on the pacing rate; and we conduct experiments for the rate-modulation property, considering multiple inputs from the QT sensor and physical exercise curves.

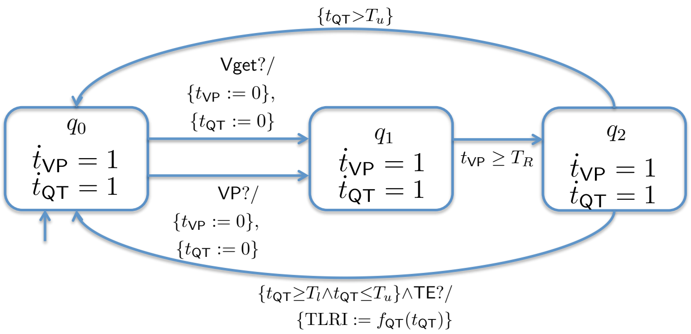

$\{t_{\mathsf{QT}}>T_u\}$
Vget?/
$\{t_{\mathsf{VP}} := 0\}$,
$\{t_{\mathsf{QT}} := 0\}$
q_0
$\dot{t}_{\mathsf{VP}} = 1$
$\dot{t}_{\mathsf{QT}} = 1$
q_1
$\dot{t}_{\mathsf{VP}} = 1$
$\dot{t}_{\mathsf{QT}} = 1$
$t_{\mathsf{VP}} \geq T_R$
q_2
$\dot{t}_{\mathsf{VP}} = 1$
$\dot{t}_{\mathsf{QT}} = 1$
VP?/
$\{t_{\mathsf{VP}} := 0\}$,
$\{t_{\mathsf{QT}} := 0\}$
$\{t_{\mathsf{QT}}\geq T_l \wedge t_{\mathsf{QT}}\leq T_u\}\wedge\mathsf{TE}?/$
$\{\mathrm{TLRI} := f_{\mathsf{QT}}(t_{\mathsf{QT}})\}$

Fig. 6. Hybrid automaton of the rate adaptive component

5.1 Verification of the basic pacemaker model

Probabilistic Switching. We conduct experiments considering the probabilistic transitions between different heart behaviours, as explained in Sect. 3.3.

In this analysis, we obtain a relationship between the probability to generate bradycardia and the number of pacemaker beats to the ventricle, shown in Figure 7. We range the probability from 0.05 to 0.95 and run 40 experiments, each representing 8 minutes of heart beat. We clearly observe that, by increasing the probability of a bradycardia behaviour, the pacemaker delivers more beats to the ventricle. This gives evidence for the ability of our pacemaker to correct random bradycardia episodes.

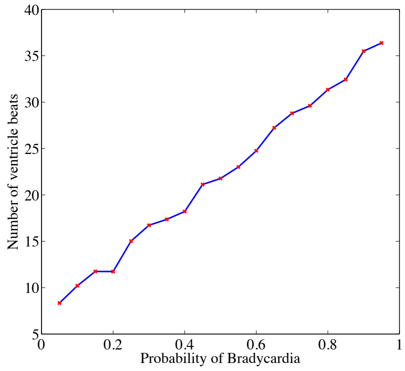


Fig. 7. Paced ventricular beats at varying probabilities of Bradycardia behaviour.

Pacemaker Mediated Tachycardia. In human hearts, the atrium can beat faster than the ventricle, at ratio 2:1 or 3:1. The resulting heart beat can still be regular due to a special cell called the AV node, which has a blocking period longer than the other cells. The AV node connects the ECS of the atrium to the ECS of the ventricle. The pacemaker tries to maintain a 1:1 AV conduction through the AVI component. Thus, in the event of PMT, the pacemaker increases the beats in the ventricle inappropriately. In order to avoid this behaviour we need to switch the pacemaker from the DDD mode to the VDI mode when the PMT event is detected. After PMT is successfully corrected and a normal heart beat is re-established, the pacemaker can switch back to the DDD mode.

In Figure 8 we show an experiment where a tachycardia episode in the ventricle due to PMT (red curve), is corrected by a mode switch from DDD to VDI at time 13. As a result, the number of ventricle beats decreases and the regular heart rhythm is recovered (blue curve).

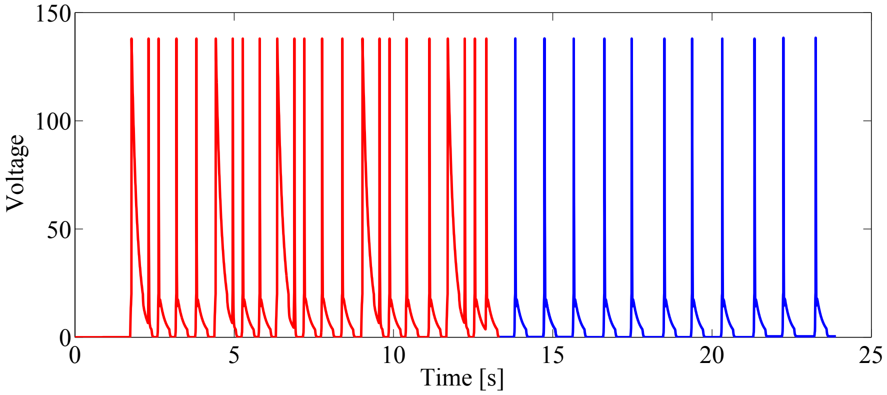


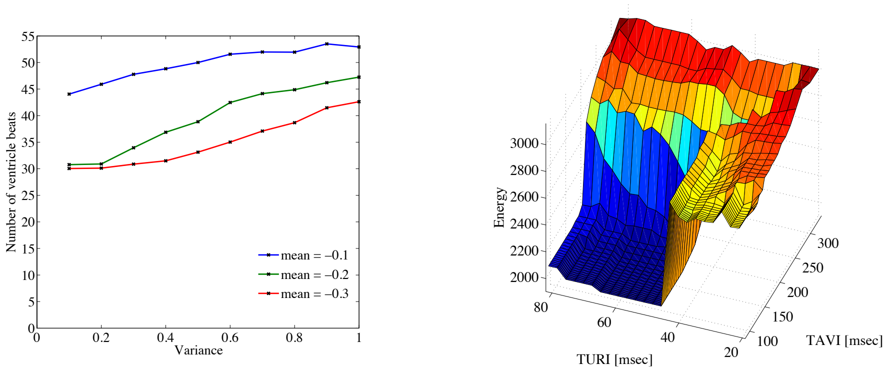
Fig. 8. AP in the ventricle during a PMT episode. The red curve shows a tachycardia frequency, corrected through mode switch at time 13 (blue curve).

5.2 Verification of the enhanced pacemaker model

Noise. Here we address the occurrence of random “failure to capture” events, generated by the presence of random noise on the pacemaker leads (illustrated in Section 4.2). In the following experiments, two parameters are considered: the mean μ and the variance σ^2 of the normally distributed noise. Figure 9(a) shows the number of ventricular beats for different values of μ (red line with $\mu = -0.3$, green line with $\mu = -0.2$ and blue line with $\mu = -0.1$). We choose a negative μ in order to simulate the undersensing effect. In each experiment with fixed mean μ, we make the variance range from 0.1 to 1 with step of 0.1.

The results demonstrate that, when Gaussian noise with small mean (indicating a high degree of undersensing) is added to the stimulus, the number of beats in the ventricle decreases, since more beats induced by the pacemaker will be lost. On the other hand, increasing the variance of the normal distribution will yield a higher number of beats. Indeed, higher variance to the noise, when centred at negative mean, produces better chances of picking positive samples from the normal distribution. This, in turn, implies a better chance for the stimulus to be high enough to stimulate the cell.

Energy. In this analysis, we are interested in the energy consumption of the pacemaker when setting the SA node to induce bradycardia, thus forcing the device to deliver paced beats. Figure 9(b) shows the results obtained by varying two parameters, TAVI and TURI, which are the default programmable parameters used by technicians to ensure a heart beat between 60 and 100 BPM. We make TAVI range in the interval [70 − 300] ms with 10 ms increment, and the value of TURI in [50 − 175] BPM with 5 BPM increment. Fig. 9(b) evidences a steep increase in energy consumption when TURI< 50 or TAVI> 200. This behaviour is caused by the fact that we are forcing the pacemaker to wait less between two consecutive ventricular events. Therefore, the pacemaker will initiate most of the ventricular beats before the occurrence of a natural beat, thus leading to a more prominent depletion of battery charge.



(a) Number of ventricle beats with random undersensing at different variances.

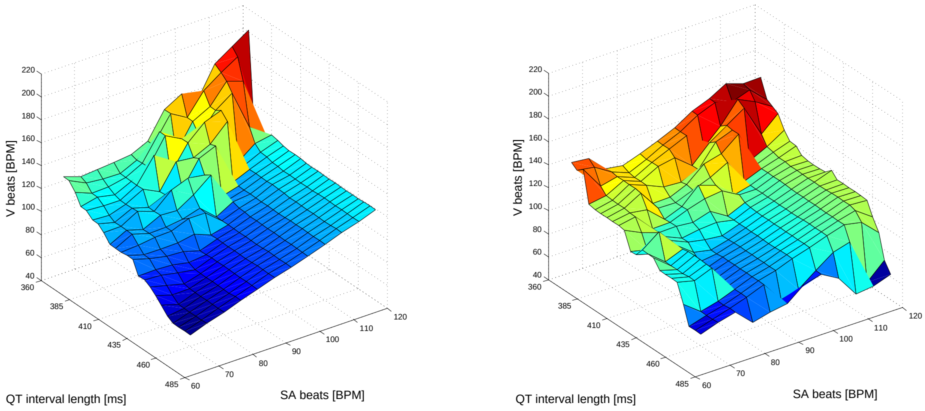
(b) Battery charge in 1 min period under Bradycardia, at varying TAVI and TURI.

Fig. 9. Sensing noise (a) and energy consumption (b) experiments in the enhanced pacemaker verification.

Parametric Analysis and Sensor Induced Tachycardia. We perform an exhaustive parameter exploration for evaluating the behaviour of the rate adaptive pacemaker model over a wide spectrum of firing rates of the sinus node (SA node) and QTI lengths. The SA frequency models the ideal heart rate demand and expresses the levels of stress and activity; the QTI lengths detected by the QT sensor are used to update the pacing rate as illustrated in Sect. 4.2. For evident vital reasons, the application on real devices and patients of this kind of quantitative analyses can involve only a limited range of safe parameter settings and feasible activity levels, and is therefore insufficient for assessing the effects of sensors faults and of extreme SA rates.

Instead, with our formal framework, we can distinguish the parameter regions under which the pacemaker correctly operates from those where phenomena of *sensor-induced tachycardia (SIT)* occur, i.e. when sensors malfunctioning (in our case, wrongly detected short QTIs) lead to inappropriately fast pacing rate. Figure 10 compares the number of ventricular beats in healthy conditions (a) and in presence of AV block (b), over 552 different combinations of QTI lengths and SA firing rates.

Such analysis provides evidence of a diagonal threshold of ideal QTI lengths and SA rates, below which we observe a SIT phenomenon, characterized by a ventricular rate constantly higher than the SA rate, which is amplified as the QTI decreases. This faulty behaviour is slightly less evident in the AV block scenario, because of the number of beats lost by the defective AV node. On the other hand, if for each SA rate appropriate QTIs are considered (above the ideal threshold), we observe a regular pattern in the number of ventricular beats. With a healthy AV node, they increase linearly in the number of SA beats, thus reproducing a correct conduction system. In the case of AV block, the frequency in the ventricle grows linearly before reaching a final plateau, indicating the inability to deliver high frequencies.



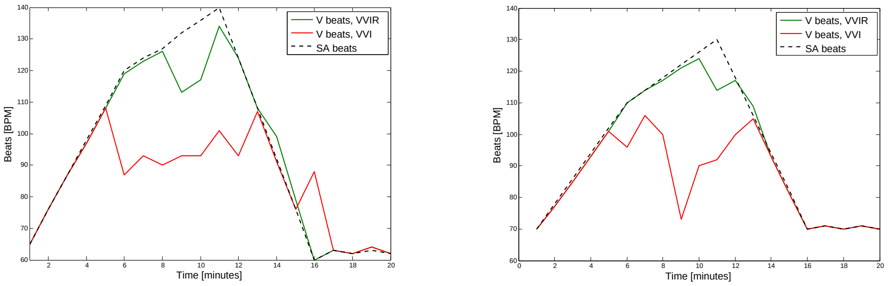
(a) V beats, Normal AV node

(b) V beats, AV block

Fig. 10. Number of ventricular beats (z-axis) over multiple QTIs (x-axis) and SA node firing frequencies (y-axis).

Modulation during Physical Activity. We validate our VVIR model by comparing it to its fixed-rate counterpart (VVI) over typical exercise curves of a young (Fig. 11(a)) and old (Fig. 11(b)) individual. Heart rate during physical exercise is characterized by four stages: neural slope (initial fast increase); metabolic slope (slower increase); decay (fast decrease during recovery); and resting. Since old subjects cannot generally provide the same exercise intensity as young individuals, their activity curves are characterized by a lower maximum heart rate.

Results during a 20 minutes exercise demonstrate that our VVIR implementation successfully manages to modulate the pacing rate according to the intensity of physical activity in both classes of patients. Minor rate discrepancies occur in the most intense phases; these are, however, negligible if compared to the behaviour of the fixed rate pacemaker (unable to provide an appropriate rate at SA rates higher than 110 BPM). Moreover, no SIT events are detected during exercise, regardless the intensity of physical activity.



(a) Young patient

(b) Old patient

Fig. 11. Rate modulation during exercise in young (a) and old (b) patients. The SA rate (black dashed line) gives the metabolic demand following typical activity curves. The number of ventricular beats is compared between the VVIR (green curve) and the VVI (red curve) pacemakers.

6 Future Directions

In the previous section we described the verification of the pacemaker model together with two heart models: the ECG and the cardiac cell network. In future, we plan to use more advanced heart models to capture the physiological characteristics of the heart. Also, we plan to synthesise crucial timing parameters of the pacemaker model such that it satisfies a given specification.

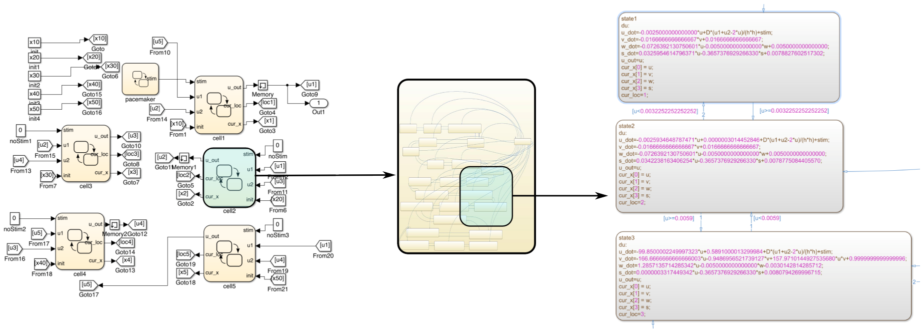
6.1 The minimal ventricular cardiac cell heart model

As an alternative to the previous heart models we propose to use the minimal ventricular (MV) model of Bueno-Orovio et al. [3]. Unlike the previous two models, the MV model can reproduce realistic and important AP phenomena, e.g. alternans [14], and yet is computationally more efficient than some of the other models in the literature. Using the techniques from Grosu et al. [13], we can abstract the MV model into a network of hybrid automata (see Figure 12) that fits our developed framework for pacemaker verification. For details see [15].

The MV model describes the flow of currents through a cell. The model is defined by four nonlinear PDEs representing the transmembrane potential $x_1(\boldsymbol{d}, t)$, the fast channel gate $x_2(\boldsymbol{d}, t)$, and two slow channel gates, $x_3(\boldsymbol{d}, t)$ and $x_4(\boldsymbol{d}, t)$. All of the four variables are time and position $\boldsymbol{d} := (d_x, d_y, d_z) \in \mathbb{R}^3$ dependent. For one dimensional tissue, i.e., $\boldsymbol{d} := d_x$, the evolution of transmembrane potential is given by:

$$\frac{\partial x_1(d_x, t)}{\partial t} = D\frac{\partial^2 x_1(d_x, t)}{\partial d_x^2} + e(x_1, t) - (J_{\mathrm{fi}} + J_{\mathrm{so}} + J_{\mathrm{si}}), \tag{4}$$

where $D \in \mathbb{R}$ is the diffusion coefficient, $e(\boldsymbol{d}, t)$ is the external stimulus applied to the cell, J_{fi} is the fast inward current, J_{si} is the slow inward current and J_{so} is the slow outward current. The currents J_{fi}, J_{so} and J_{si} are described by Heaviside function. For more details see [3]. To define the propagation of the action potential on a cardiac ring of length L, we set the boundary conditions to: $x_i(0, t) = x_i(L, t)$ for all $i \in \{0, \ldots, 4\}$ and $t \in \mathbb{R}$.



```
state1
du:
u_dot=-0.0025000000000000*u+D*(u1+u2-2*u)/(h*h)+stim;
v_dot=-0.0166666666666667*v+0.0166666666666667;
w_dot=-0.0726392130750601*u-0.0050000000000000*w+0.0050000000000000;
s_dot=0.0325954614796371*u-0.3657376929266330*s+0.0078827602517302;
u_out=u;
cur_x[0] = u;
cur_x[1] = v;
cur_x[2] = w;
cur_x[3] = s;
cur_loc=1;
```

```
state2
du:
u_dot=-0.0025934648787471*u+0.0000003014452846+D*(u1+u2-2*u)/(h*h)+stim;
v_dot=-0.0166666666666667*v+0.0166666666666667;
w_dot=-0.0726392130750601*u-0.0050000000000000*w+0.0050000000000000;
s_dot=0.0342238163406254*u-0.3657376929266330*s+0.0078775084405570;
u_out=u;
cur_x[0] = u;
cur_x[1] = v;
cur_x[2] = w;
cur_x[3] = s;
cur_loc=2;
```

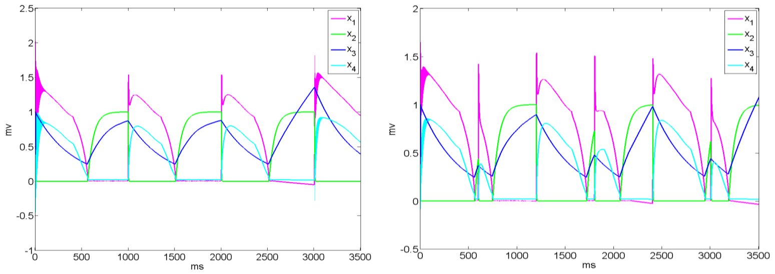
```
state3
du:
u_dot=-99.8500002249997323*u+0.5891000013299984+D*(u1+u2-2*u)/(h*h)+stim;
v_dot=-166.6666666666666003*u-0.9486956521739127*v+157.9710144927535680*u*v+0.9999999999999996;
w_dot=1.2857135714285342*u-0.0050000000000000*w-0.0030142814285712;
s_dot=0.0000003317449342*u-0.3657376929266330*s+0.0080794269996715;
u_out=u;
cur_x[0] = u;
cur_x[1] = v;
cur_x[2] = w;
cur_x[3] = s;
cur_loc=3;
```

Fig. 12. Left: top-level Simulink/Stateflow model for a ring of five cardiac cells; the *Pacemaker* block stimulates one cell. Center: Stateflow model of a single cardiac cell. Right: dynamics and guards in 3 locations of a single cell.

HA approximation. One alternative to solving these highly nonlinear PDEs is to discretize space and hybridize the dynamics. The result is the HA model. Following the approach of [13], we first hybridize the dynamics and obtain a HA with 29 locations. The basic idea is to approximate the Heaviside function from J_{fi}, J_{so} and J_{si} with a sequence of ramp functions. Each location of the resulting HA contains a multi-affine ODE such as:

$$\begin{aligned}
&\dot{x_1} = -0.935x_1 + 12.70x_2 - 8.0193x_1x_2 + 0.529x_3x_4 + 0.87 + st\\
&\dot{x_2} = -0.689x_2;\ \dot{x_3} = -0.0025x_3;\ \dot{x_4} = 0.0293x_1 - 0.0625x_4 + 0.0142,
\end{aligned}$$

where *st* is the time-varying stimulus input. The 29 locations represent the final HA model of a single cardiac cell. By discretising the spatial location we obtain a network of cells that can be connected in a ring or in a tree depending on the



(a)

(b)

Fig. 13. Reach set projected on x_{11} (AP) for stimulation period of 1000 msec (a) and 600 msec (b) with x-axis for time and y-axis for voltage.

physiological characteristics of the heart that we would like to model. In Figure 12 we depict a Simulink/Stateflow implementation of 5 cardiac cells connected in a ring; in Figure 13 we depict the voltage level of a cardiac cell for a set of initial conditions.

In [15] we have developed techniques on how to compute the over-approximation of the reach set, i.e., the voltage level of the cardiac cell at a given time moment, for a network of cardiac cells given by the MV model. As a future direction we plan to connect the minimal ventricular cardiac cell heart model to the pacemaker model, and investigate more advanced specifications such as linear duration properties [7].

6.2 Automated Synthesis of Pacemaker Software

Pacemaker devices have a limited number of programmable parameters and there is considerable agreement among manufacturers on the appropriate values to set according to the considered heart condition, implying that the same pacemaker settings are used for large classes of cardiac patients exhibiting the same disease.

We believe that model synthesis methods can significantly advance the automated design of *highly personalized* pacemaker devices where, instead of choosing among a limited number of condition-specific settings, parameters are automatically derived according to patient's clinical history, and continuously adapted to reflect the real-time monitoring of her/his conditions.

Given that patient-specific models of the heart can be constructed from the detailed electro-physiological data obtainable with current diagnostic means, and given a (formal) property describing the desired behaviour of the heart, synthesis techniques would provide pacemaker models that are correct-by-design, so that, when composed with the heart model, the required behaviour is ensured without the burden of formally verifying the property against all the possible combinations of parameters.

Moreover, synthesis methods for implantable pacemakers need to cope with a range of *uncontrollable parameters*, which, unlike the controllable timings of a pacemaker, cannot be adjusted to our needs. Think, for example, about the timing at which ventricle beats are fired, or any other physiological parameter of the heart. Hence, the purpose is to find a positive solution to an *optimal synthesis problem*, which consists in finding optimal values for the controllable parameters such that the composed heart-pacemaker model meets the required (healthy) behaviour specification, regardless of the uncontrollable parameters. As usual, the notion of optimality underlies a quantitative objective function we want to maximize or minimize (e.g. energy consumption).

In [9], an initial approach to the optimal synthesis of pacemaker devices is proposed, based on symbolic constraint reasoning, and with application to networks of timed I/O automata. We are currently working to extend the approach towards richer and more complex models, featuring hybrid and probabilistic dynamics.

7 Conclusion

In this paper, we presented a model-based framework for the formal analysis of implantable pacemakers, which supports the plug-in and the integration of different heart and pacemaker models by means of a small set of pre-defined interfaces and modelling templates. In the composed heart-pacemaker model, stochasticity comes into play in several ways, such as in the probabilistic behaviour of the heart or in the occurrence of random sensor faults. The framework enables the analysis of a broad range of electro-physiological and device-related properties, computed through simulation or quantitative verification, thus providing safety guarantees of the pacemaker in presence of multiple sources of uncertainty. Tool support is of crucial importance, and we, indeed, provide a sound implementation of the formal framework in MATLAB Simulink/Stateflow.

We evaluated our framework over two heart models, the ECG and the cardiac cell model; and over an enhanced pacemaker design, built by modularly adding advanced functionalities, including energy, sensor noise and rate-adaptation on top of a basic model inspired by [17]. We reported here only some of the experimental results obtained in previous work [4–6, 18], showing how quantitative verification can provide practical guidance for safer and more efficient designs of pacemaker devices, and, ultimately, give insight into the defective dynamics of heart diseases.

Current research efforts are directed towards the analysis of more advanced and physiologically accurate heart models, and to the synthesis of pacemaker parameters. As future work, we aim to formulate and implement novel synthesis methods, able to automatically derive not just pacemaker parameters, but also complete specifications of pacing algorithms and protocols, which are optimal under safety and cost-effectiveness, and account for the stochastic dynamics of the heart and sensors.

Acknowledgments. This work is supported by the ERC AdG VERIWARE, ERC PoC VERIPACE and the Institute for the Future of Computing, Oxford Martin School.

References

1. PhysioNet. `http://http://www.physionet.org/physiobank/`.
2. S. S. Barold, R. X. Stroobandt, and A. F. Sinnaeve. *Cardiac pacemakers and resynchronization step by step: An illustrated guide*. John Wiley & Sons, 2010.
3. A. Bueno-Orovio, E. M. Cherry, and F. H. Fenton. Minimal model for human ventricular action potentials in tissue. *Journal of Theoretical Biology*, 253(3):544 – 560, 2008.
4. T. Chen, M. Diciolla, M. Kwiatkowska, and A. Mereacre. Quantitative verification of implantable cardiac pacemakers. In *Real-Time Systems Symposium (RTSS), 2012 IEEE 33rd*, pages 263–272. IEEE, 2012.
5. T. Chen, M. Diciolla, M. Kwiatkowska, and A. Mereacre. A simulink hybrid heart model for quantitative verification of cardiac pacemakers. In *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control (HSCC 2013)*, pages 131–136, 2013.
6. T. Chen, M. Diciolla, M. Kwiatkowska, and A. Mereacre. Quantitative verification of implantable cardiac pacemakers over hybrid heart models. *Information and Computation*, In press, 2014.
7. T. Chen, M. Diciolla, M. Z. Kwiatkowska, and A. Mereacre. Verification of linear duration properties over continuous-time markov chains. *ACM Trans. Comput. Log.*, 14(4):33, 2013.
8. G. Clifford, S. Nemati, and R. Sameni. An Artificial Vector Model for Generating Abnormal Electrocardiographic Rhythms. *Physiological Measurements*, 31(5):595–609, May 2010.
9. M. Diciolla. *Quantitative Verication of Real-Time Properties with Application to Medical Devices*. PhD thesis, Department of Computer Science, University of Oxford, 2014.
10. A. O. Gomes and M. V. M. Oliveira. Formal specification of a cardiac pacing system. In *FM 2009: Formal Methods*, pages 692–707. Springer, 2009.
11. S. Greenhut, J. Jenkins, and R. MacDonald. A stochastic network model of the interaction between cardiac rhythm and artificial pacemaker. *Biomedical Engineering, IEEE Transactions on*, 40(9):845–858, 1993.
12. F. Gritzali, G. Frangakis, and G. Papakonstantinou. Detection of the P and T waves in an ECG. *Computers and Biomedical Research*, 22(1):83–91, 1989.
13. R. Grosu, G. Batt, F. H. Fenton, J. Glimm, C. Le Guernic, S. A. Smolka, and E. Bartocci. From cardiac cells to genetic regulatory networks. In *Computer Aided Verification*, pages 396–411. Springer, 2011.
14. M. R. Guevara, G. Ward, A. Shrier, and L. Glass. Electrical alternans and period-doubling bifurcations. *Computers in Cardiology*, pages 167–170, 1984.
15. Z. Huang, C. Fan, A. Mereacre, S. Mitra, and M. Kwiatkowska. Invariant verification of nonlinear hybrid automata networks of cardiac cells. In *Computer Aided Verification*. Springer, 2014.
16. Z. Jiang, M. Pajic, A. Connolly, S. Dixit, and R. Mangharam. Real-time heart model for implantable cardiac device validation and verification. In *ECRTS*, pages 239–248, 2010.
17. Z. Jiang, M. Pajic, S. Moarref, R. Alur, and R. Mangharam. Modeling and verification of a dual chamber implantable pacemaker. In C. Flanagan and B. König, editors, *TACAS*, volume 7214 of *Lecture Notes in Computer Science*, pages 188–203. Springer, 2012.

18. M. Kwiatkowska, H. Lea-Banks, A. Mereacre, and N. Paoletti. Formal modelling and validation of rate-adaptive pacemakers. In *IEEE International Conference on Healthcare Informatics 2014 (ICHI 2014), to appear*, 2014.
19. J. Lian, H. Krätschmer, D. Müssig, and L. Stotts. Open source modeling of heart rhythm and cardiac pacing. *Open Pacing Electrophysiol Ther J*, 3:4, 2010.
20. N. Lynch, R. Segala, F. Vaandrager, and H. B. Weinberg. *Hybrid I/O automata.* Springer, 1996.
21. H. D. Macedo, P. G. Larsen, and J. Fitzgerald. Incremental Development of a Distributed Real-Time Model of a Cardiac Pacing System Using VDM. In *Formal Methods*, page 181, 2008.
22. J. F. Manwell and J. G. McGowan. Lead acid battery storage model for hybrid energy systems. *Solar Energy*, 50(5):399 – 405, 1993.
23. P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith. A dynamical model for generating synthetic electrocardiogram signals. *Biomedical Engineering, IEEE Transactions on*, 50(3):289–294, 2003.
24. D. Méry and N. K. Singh. Pacemaker's Functional Behaviors in Event-B. Rapport de recherche, MOSEL - INRIA Lorraine - LORIA, 2009.
25. L. A. Tuan, M. C. Zheng, and Q. T. Tho. Modeling and verification of safety critical systems: A case study on pacemaker. In *Secure Software Integration and Reliability Improvement (SSIRI), 2010 Fourth International Conference on*, pages 23–32. IEEE, 2010.
26. P. Ye, E. Entcheva, R. Grosu, and S. A. Smolka. Efficient modeling of excitable cells using hybrid automata. In *Proc. of CMSB*, volume 5, pages 216–227, 2005.
27. Y. C. Yeh and W. J. Wang. QRS complexes detection for ECG signal: The Difference Operation Method. *Computer methods and programs in biomedicine*, 91(3):245–254, 2008.