3-16-2016

Automated Closed-Loop Model Checking of Implantable Pacemakers using Abstraction Trees

Zhihao Jiang
University of Pennsylvania, zhihaoj@seas.upenn.edu

Houssam Abbas
University of Pennsylvania, habbas@seas.upenn.edu

Pieter Mosterman
Mathworks, Pieter.Mosterman@mathworks.com

Rahul Mangharam
University of Pennsylvania, rahulm@seas.upenn.edu

Automated Closed-Loop Model Checking of Implantable Pacemakers using Abstraction Trees

Abstract
Autonomous medical devices such as implantable cardiac pacemakers are capable of diagnosing the patient condition and delivering therapy without human intervention. Their ability to autonomously affect the physiological state of the patient makes them safety-critical. Sufficient evidence for the safety and efficacy of the device software, which makes these autonomous decisions, should be provided before these devices can be released on the market. Formal methods like model checking can provide safety evidence that the devices can safely operate under a large variety of physiological conditions. The challenge is to develop physiological models that are general enough to cover the large variability of human physiology, and also expressive enough to provide physiological contexts to counter-examples returned by the model checker. In this paper, the authors develop a set of physiological abstraction rules that introduce physiological constraints to heart models. By applying these abstraction rules to a initial set of heart models, an abstraction tree is created. The root model covers all possible inputs to a pacemaker and derived models cover inputs from different heart conditions. If a counter-example is returned by the model checker, the abstraction tree is traversed so that the most concrete counter-example(s) with physiological contexts can be returned to the domain experts for validity check. The abstraction tree framework replaces the manual abstraction and refinement framework, which reduced the amount of domain knowledge required to perform closed-loop model checking. It encourages the use of model checking during the development of autonomous medical devices, and identifies safety risks earlier in the design process.

Disciplines
Computer Engineering | Electrical and Computer Engineering

Comments
@inproceedings{ Jiang2015AbstractionTree, title={Automated Closed-Loop Model Checking of Implantable Pacemakers using Abstraction Trees}, author={Zhihao Jiang and Houssam Abbas and Pieter J Mosterman and Rahul Mangharam},
journal={Medical Cyber Physical Systems Workshop 2016},
year={2016} }

Automated Closed-Loop Model Checking of Implantable Pacemakers using Abstraction Trees

Zhihao Jiang[1], Houssam Abbas[1], Pieter J. Mosterman[2], and Rahul Mangharam[1]

[1]Department of Electrical & System Engineering, University of Pennsylvania

[2]School of Computer Science, McGill University, Canada and MathWorks, USA

ABSTRACT

Autonomous medical devices such as implantable cardiac pacemakers are capable of diagnosing the patient condition and delivering therapy without human intervention. Their ability to autonomously affect the physiological state of the patient makes them safety-critical. Sufficient evidence for the safety and efficacy of the device software, which makes these autonomous decisions, should be provided before these devices can be released on the market. Formal methods like model checking can provide safety evidence that the devices can safely operate under a large variety of physiological conditions. The challenge is to develop physiological models that are general enough to cover the large variability of human physiology, and also expressive enough to provide physiological contexts to counter-examples returned by the model checker. In this paper, the authors develop a set of physiological abstraction rules that introduce physiological constraints to heart models. By applying these abstraction rules to a initial set of heart models, an abstraction tree is created. The root model covers all possible inputs to a pacemaker and derived models cover inputs from different heart conditions. If a counter-example is returned by the model checker, the abstraction tree is traversed so that the most concrete counter-example(s) with physiological contexts can be returned to the domain experts for validity check. The abstraction tree framework replaces the manual abstraction and refinement framework, which reduced the amount of domain knowledge required to perform closed-loop model checking. It encourages the use of model checking during the development of autonomous medical devices, and identifies safety risks earlier in the design process. [1]

1. INTRODUCTION

[1]This work was supported in part by the TerraSwarm Research Center, one of six centers supported by the STARnet phase of the Focus Center Research Program (FCRP) a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

DOI: 10.1145/1235

Medical devices are developed for *diagnostic* or *therapeutic* applications to infer and treat specific health issues. For instance, an X-ray machine takes images of dense tissue in the body, which is useful for diagnostic purposes. An infusion pump delivers a pre-programmed amount of fluid into the patient for treatment. There is an emerging category of medical devices that have both the diagnostic and therapeutic functionality which are capable of operating autonomously without intervention from medical professionals. We refer to these devices as *Autonomous Medical Devices*. For instance, an implantable cardiac pacemaker monitors the electrical activities of the heart with two leads and delivers electrical pacing to manage the heart rate (Fig. 1).

1.1 Challenges for Safe Autonomous Medical Devices

Developing safe and effective autonomous medical devices has the following challenges:

1. **Physiological Complexity:** Human physiology is complex and only partially understood. For instance, the functionality of the heart can be interpreted from *multiple perspectives*: from its electrical activity, mechanical contractions of the heart muscles, and dynamics of blood flow. The physiology of the heart can also be analyzed with *multiple scales*: from the molecular level to cellular level all the way to the organ and system level. It is impossible to encode all these contexts into the device, hence inappropriate diagnosis and therapy are observed due to the lack of physiological contexts [16, 5].

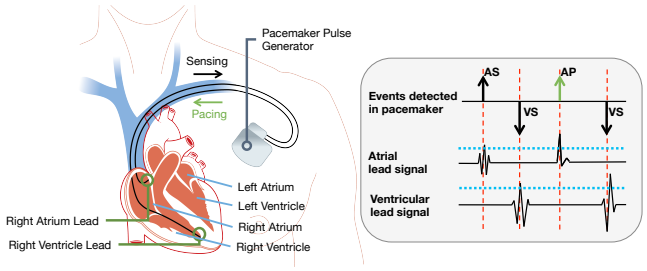

Figure 1: A dual chamber pacemaker has one lead in the right atrium and the other in the right ventricle. It thresholds local electrical activities into sensed events (AS,VS) and delivers electrical pacing (AP,VP) when the heart rate is low.

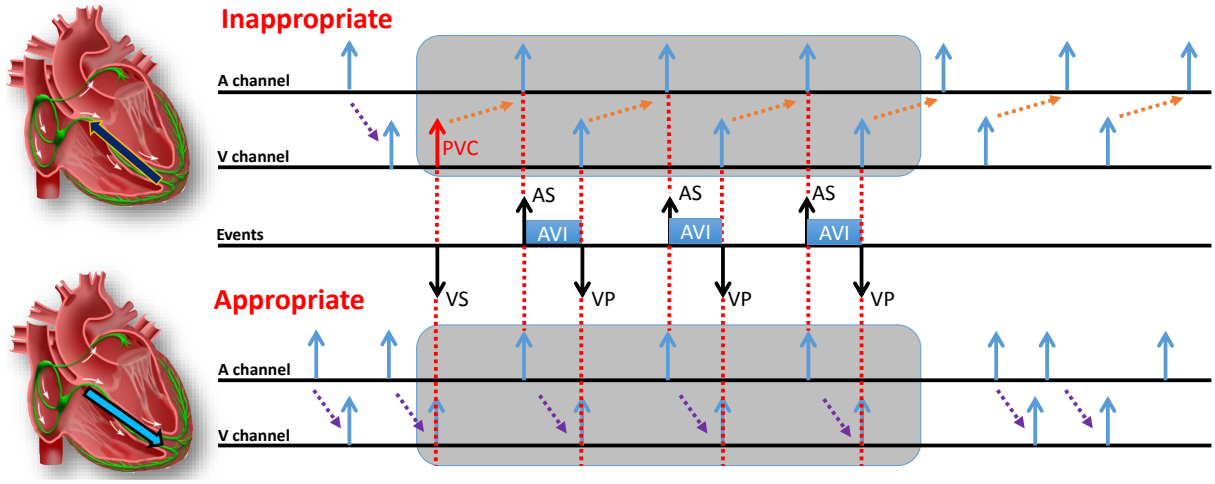

Figure 2: Two event traces of a dual chamber pacemaker. In the "appropriate" case, heart rate is controlled by the natural pacemaker of the heart in the right atrium. Electrical signals are traveling from the atria to the ventricles (indicated by the purple arrows). The pacemaker paces the ventricle (VP) after each sensed atrial contraction (AS) to maintain appropriate timing (AVI) between atrial and ventricular contraction. In the "inappropriate" case, a Premature Ventricular Contraction (PVC) triggers retrograde conduction from the ventricles to the atria (indicated by the orange arrows), triggering a sensed event (AS) in the pacemaker. The pacemaker paces the ventricle with the same AVI delay (VP). The pace triggers another retrograde conduction and the feedback loop continues without interruption. The two cases may look exactly the same to a pacemaker, as highlighted by the gray region, causing ambiguity. In the appropriate case the heart rate is changing according to physiological needs, while in the inappropriate case the heart is fixed to a fast rate.

2. **Physiological Variability:** Physiological conditions and parameters demonstrate different levels of variability both within the individual at different times, levels of exertion and under the influence of medication and also across individuals. For instance, a segment of the population may have additional conduction pathways within their heart, which makes them vulnerable to certain heart diseases. Consequently, autonomous medical devices should be able to safely operate under a large variety of physiological conditions. This is difficult to guarantee, as the device designer must consider all possible physiological conditions (including rare conditions) during the design of the device.

3. **Limited Observability:** Autonomous medical devices normally rely on minimally invasive measurement of the physiological parameters in order to allow the patients to live their normal life. For example, implantable pacemakers and defibrillators commonly have just two leads and therefore two points of observation for the whole heart. The limited observability inevitably leads to ambiguities as different physiological conditions can map to the same input sequence to the device, resulting in inaccurate diagnosis.

   Consider the example in Fig. 2. The middle trace shows the **events** detected by the pacemaker: (a) Ventricular Sense (VS) indicates the pacemaker has detected a depolarization of the ventricles; (b) Atrial Sense (AS) indicates the pacemaker has detected a depolarization of the atria; (c) Ventricular Pacing (VP) indicates the pacemaker has paced the heart in response to a perceived missing beat. This event trace can be generated in two different ways, one of which is healthy and the other unhealthy. The lower trace shows a healthy situation in which the electrical depolarizations originate from the right atrium (detected as an AS by the pacemaker) but do not make it to the ventricles. The pacemaker paces the ventricles (indicated as VPs) after a programmed delay to maintain heart rate. The heart rate in this situation is the normal one (e.g. 60bpm) as set by the heart's own pacemaker in the right atrium. The upper trace shows an unhealthy situation: a Premature Ventricular Contraction (PVC) is transmitted to the atria via ventricle-to-atrium conduction and triggers an atrial contraction (AS). Even though the heart was operating within the normal heart rate and *does not require pacing*, the pacemaker paces the ventricles after a programmed delay, which again triggers ventricle-to-atrium conduction. This AS-VP pattern persists at a rate set by the implanted pacemaker itself and is higher than the normal heart rate. This condition is known as Endless Loop Tachycardia (ELT). As can be seen, the pacemaker cannot distinguish these two situations, causing inappropriate therapy.

The capability of cardiac pacemakers to autonomously affect the physiological state of the patient makes it safety-critical, and sufficient evidence for the software's safety and efficacy should be provided before these devices can be released on the market. In particular, they are classified as a High Risk Device by the Food and Drug Administration in the U.S. Such autonomous medical devices increasingly rely on software, and device function and clinical performance can be affected by seemingly minor changes to software. The current form of closed-loop validation of medical devices is clinical trials, in which the devices are evaluated on a small group of patients before they can be released to the market. Clinical trials are costly to perform and revising device design at this late stage is very expensive.

1.2 Evaluating the Device Design with Formal Methods

Formal methods such as model checking explore the entire state space of a model for property violations. Model checking can be used to evaluate the device design early in the development process. The safety and efficacy of autonomous medical devices have to be evaluated within their physiological context. We identify three challenges for developing physiological models for closed-loop model checking of autonomous medical devices:

1. **Model Interpretability:** How much detail should the physiological models have in order to unambiguously describe a physiological behavior? In particular, if the model checker returns an execution trace as counter-example, how much detail should the physiological model have so that the execution traces can be interpreted by medical domain experts?

2. **Behavior Coverage:** What approach must we use for physiological models to cover the large variability of human physiology? How can these models cover rare physiological cases and those that are unknown to us?

3. **Model Ambiguity:** As seen in the ELT example, multiple physiological conditions can map to the same event trace due to the limited observability of the device. How can we eliminate only the healthy execution from the model so that it would not cause a false-positive?

1.3 The need for Domain Expertise with CEGAR

In previous work [10], we performed closed-loop model checking of an implantable pacemaker design in order to tackle the 3 challenges mentioned above.

First, based on clinical electrophysiology, we developed a timed-automata [1] heart structure which models the electrical conduction and propagation. The model structure provides sufficient complexity to represent a variety of heart conditions, and the executions of the model output signals which are interpretable by medical domain experts.

Next, over-approximation and non-determinism are used to create a *Random Heart Model (RHM)* that is able to produce all possible input sequences to the pacemaker. Properties satisfied under the RHM are satisfied under all heart conditions. However, the RHM also contains behaviors that are physiologically invalid which should not be considered as safety violations. For instance, in the ELT case, with the RHM the healthy execution is not distinguishable from ELT, which can potentially be returned as counter-example, causing a false-positive, or *spurious counter-example*.

In order to eliminate behaviors which are not physiologically relevant, a Counter-Example Guided Abstraction Refinement (CEGAR) framework [3] was adopted. The closed-loop system including the RHM and the pacemaker model is first verified against physiological properties in model checker UPPAAL. If the property is satisfied, the pacemaker model satisfies the property under all heart conditions. If a property is violated, an abstract counter-example is returned by the model checker. The counter-example is manually examined for potential physiological invalid behaviors, and

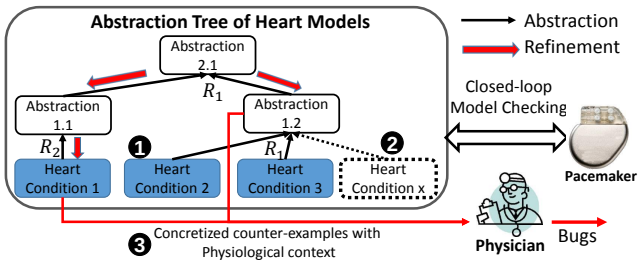

Figure 3: Automated Closed-loop Model Checking with Abstraction Tree

all possible physiological conditions that can produce the counter-example to avoid false-positives.

If ambiguities or invalid behaviors exist, the user must create a more refined heart model so that these false-positive behaviors are eliminated in the new model. The property is then verified again with the refined model and the pacemaker model, till there is no counter-example and the system is considered safe, or the counter-example is both physiologically valid and unambiguously representing a safety violation.

For instance, in the ELT example in Fig. 2, the input-output sequence is returned as counter-example from the RHM. The distinction between the healthy execution and ELT is the direction of electrical conduction. By adding a conduction direction into the heart model, the ambiguity is resolved.

Note also that in this process, domain expertise is required during 1) heart modeling, 2) manual model refinement, and 3) checking the physiological validity of counter-examples returned by the model checker. While domain expertise will always be required when building a model of any phenomenon, it is desirable to minimize the need for it in the verification process itself as this opens up opportunities for future automation.

Thus the previous work left the following questions open:

- How can a variety of cardiac conditions be modeled in an efficient manner and yet not require manual refinement and analysis of counter-examples?

- How can heart models be built in such a way that the subsequent model checking procedure does not require domain expertise?

1.4 Automated Model Checking with Abstraction Tree

The contributions of this effort are to automate the manual counter-example analysis and model refinement approach. The details of the method are provided in the following sections while a brief overview is presented here.

We start out with a set of heart models covering different conditions such as atrial flutter, bradycardia, etc. However, having multiple models is not sufficient to cover all relevant behaviors that might be produced by a heart, not to mention possible combinations of several conditions. Thus, we develop abstraction rules that automatically add new behavior to a given model. The key point is that the rules are designed such that the new behavior is mostly physiologically valid, that is, it might actually be produced by a human heart. Thus a counter-example found only in an

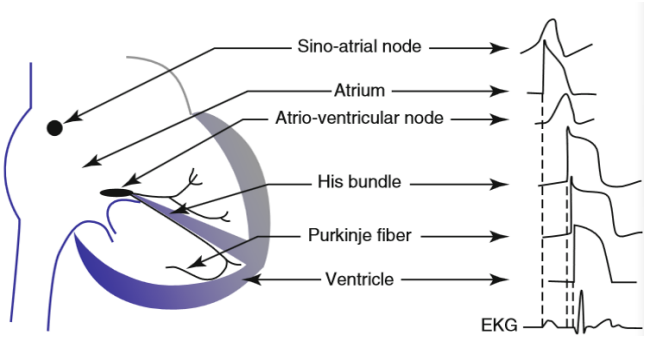

Figure 4: Electrical signal generated by the sino-atrial (SA) node which propagates throughout the heart. Note the different signal morphology and conduction delays between different locations of the heart [6]

abstracted model (but not in the more concrete one) is not automatically considered to be "spurious" the way it is in traditional abstraction/refinement frameworks. By applying these rules to the various initial models and in different sequences, a tree of models is obtained, each of which are related by abstraction/refinement relations. This is referred to as an Abstraction Tree (see Fig. 7 for an illustration).

Model checking proceeds from the root of the tree (which is the most abstract model) to its leaves. If the pacemaker model is found to satisfy the physiological property, model checking stops. If a counter-example is found, verification proceeds to the model's children and so on, till it reaches the leaves or a child-model is found to satisfy the property. See Fig. 3. This constitutes a form of automatic refinement, but over a tree of models. The most concrete model still evidence a counter-example is then analyzed to determine whether the counter-example is physiologically valid or not.

In this procedure, domain expertise is needed only when building the initial set of models and physiological abstraction rules. Constructing the more abstract and behavior-rich models, model checking, and refinement are automated.

The abstraction tree approach achieves the following improvements over the manual CEGAR framework in [10]:

- **Property specification:** In previous work [10], fairly constrained properties have to be specified that described specific ways in which an unhealthy condition might develop. This was done to avoid a large number of spurious or irrelevant counter-examples. In the abstraction tree approach, this is no longer a concern since ambiguities can be resolved by automatically refining the counter-examples. Therefore the properties can be more general, which increases the probability to identify unknown safety violations.

- **Counter-example Analysis**: Counter-examples returned from an abstract model are difficult to analyze because of the lack of physiological context. In this case, the abstraction tree provides the most concrete counter-example as guidance for checking validity and possible refinements, which is a huge improvement over the manual process in [10].

The paper is organized as follows: Section 2 introduces a dual chamber pacemaker design and the physiological properties that should be satisfied. Section 3 introduces the timed-automata heart model structure that can be used to

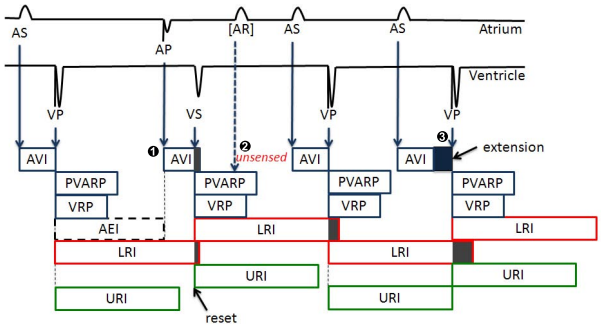

Figure 5: Basic timers for a dual chamber pacemaker. AS: Atrial Sense, VS: Ventricular Sense, AP: Atrial Pacing, VP: Ventricular Pacing.

model different heart conditions. Section 4 discusses the construction of an abstraction tree by applying a set of physiological rules on an initial set of heart models. Section 5 is a case study in which the abstraction tree of heart models is used for automated closed-loop model checking of an implantable pacemaker design.

2. IMPLANTABLE PACEMAKER

This section first introduces a physiological basis for implantable pacemakers. Next, a dual chamber pacemaker design is provided that is to be validated with model checking.

2.1 Electrophysiology Basics

A healthy heart generates periodic electrical impulses with specialized tissue called SA node to control heart rates according to physiological needs. These impulses propagate through the heart, triggering coordinated muscle contractions, and pump blood to the rest of the body. Fig. 4 shows how electrical impulse generated by the SA node propagates throughout the heart and triggers electrical activities at different locations of the heart. The underlying pattern and timing of these impulses determine the heart's rhythm and are the key to proper heart functions. Derangements in this rhythm are referred to as *arrhythmia*, which impair the heart's ability to pump blood and compromise the patients' health. Arrhythmias are categorized into so-called Tachycardia and Bradycardia. Tachycardia features undesirable fast heart rate which results in inefficient blood pumping. Bradycardia features slow heart rate which results in insufficient blood supply. Different heart conditions can be distinguished by the *timing* of the electrical conduction and the *topology* of the electrical conduction system of the heart, which are researched in clinical setting referred to as *Electrophysiology (EP)* [11].

2.2 Model of a Dual Chamber Pacemaker

A dual chamber pacemaker has two leads, one anchored to right atrium and the other to the right ventricle. The leads are able to sense electrical activities of local tissue, which is referred to as intra-cadiac electrogram (EGM). When the voltage exceed a pre-programmed threshold, the pacemaker will register a sensed event (AS, VS). Heart conditions are inferred from the temporal relationship between sensed events, and the pacemaker can deliver electrical pacing (AP, VP) through the leads to trigger heart contraction if the heart rate is low.

A dual chamber pacemaker has several basic timers that are shown in Fig. 5:

Atrial Escape Interval (AEI) defines the maximum interval between the most recent ventricular event (VS,VP) to an atrial event (AS,AP). If no AS happened before the AEI timer expires, atrial pacing (AP) is delivered to the heart (Marker 1 in Fig. 5).

Atrio-Ventricular Interval (AVI) defines the maximum interval between the most recent atrial event (AS,AP) to a ventricular event (VS,VP). If no VS happened before the AVI timer expires, and the time since the most recent ventricular event (VS,VP) is no less than URI, ventricular pacing (VP) is delivered to the heart (Marker 3 in Fig. 5).

Post-Ventricular Atrial Refractory Period (PVARP) and Ventricular Refractory Period (VRP) define the minimum period that an AS or VS can happen since the most recent ventricular event (VS,VP).

The pacemaker operates according to the timing among discrete events, therefore can be intuitively modeled as timed automata [1]. The detailed UPPAAL timed automata implementation of the pacemaker design can be found in [10].

2.3 Physiological Requirements

A dual chamber pacemaker is designed to increase the heart rate during bradycardia, but it should also not increase the heart rate inappropriately. Inappropriate increase of heart rate by the pacemaker is referred to as *Pacemaker Mediated Tachycardia (PMT)*. Previous work [10] used model checking to identify two known PMT conditions. However, in order to avoid ambiguities in the counter-examples, the properties for the two PMTs were specified very specifically, which abandoned the advantage of model checking to find unknown safety violations.

With the abstraction tree approach, the ambiguities can potentially be resolved since the abstraction tree considers all known heart conditions. Therefore the property for PMTs can be specified more generally. In this paper, we specify a property such that

Property 1: The interval between ventricular events (Vget, VP) should not be shorter than TURI for 30 consecutive beats

which means that the ventricular rate should not be faster than the upper rate interval (TURI) for too long, either intrinsically or because of pacemaker interaction. Counter-example because of intrinsic fast ventricular rates can be removed from results after analysis from the abstraction tree.

A UPPAAL monitor M_{con} for the property is shown in Fig. 6, and the TCTL property is specified as:

$$A[]\ not\ M_{con}.err$$

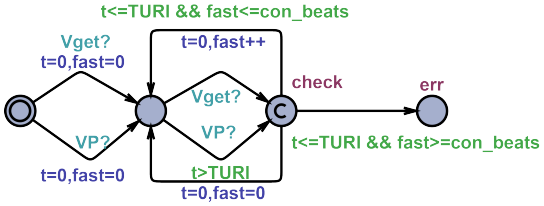

Figure 6: UPPAAL monitor for Property 1

3. HEART MODELING FOR CLOSED-LOOP MODEL-CHECKING OF IMPLANTABLE PACEMAKER

A pacemaker design should be able to satisfy safety properties under all possible heart conditions. In closed-loop model checking of implantable pacemaker, behaviors of these heart conditions must be covered by models of the heart.

3.1 Related Works

Electrical properties of the heart have been studied and models have been developed at different levels. Cellular models ([15]) have been developed and by combining these cellular models with the structural models, the electrical activities of the whole heart are studied, especially the mechanism of different arrhythmia ([17], [4], [13]). Intrinsic heart rate variability has been modeled to synthesize optimal control of pacemaker pacing. ([2]) Abstraction of the electrical cellular model has also been attempted by [7] to reduce model complexity without sacrificing accuracy. The electrical properties and the mechanical properties of the heart are closely coupled. Models combining both of these aspects are also developed to study the effects of different arrhythmia on cardiac outputs ([17], [14]).

The aforementioned models were developed for understanding the physiological behaviors. In the application of closed-loop model checking of implantable pacemakers, the heart models should not only provide behavior coverage in terms of device inputs, but also be able to provide physiological context to the execution. The physiological context will not only resolve ambiguities caused by abstract counter-examples, but also provide interpretability for domain experts.

3.2 Electro-Physiology Heart Model Structure

The pacemaker under study only has two leads into the heart and only uses onset of electrical activities for decision making. The limited observability can be exploited and superfluous detail ignored during modeling. In previous work [10] a timed automata heart model structure has been developed that models the generation and conduction of electrical events throughout the heart. With *non-determinism*, the heart models can cover behaviors of large varieties of heart conditions. By doing so the heart models sacrifice accuracy for coverage, and may include behaviors that are physiologically invalid. Fortunately, these invalid behaviors are only relevant when they are returned as counter-examples to certain property. The physicians are the final judges for the validity of the counter-examples.

At cellular level, heart tissue can be activated by an external voltage. Certain tissues also has the capability to self-activate, which contributes to natural heart beats. Once activated (Marker 1 in Fig. 7), the voltage outside the tissue changes over time, which is referred to as *Action Potential* (Fig. 7(a)). The action potential can be divided into two functional timing periods: The *Effective Refractory Period (ERP)*, during which the tissue cannot be triggered by another activation; and the *Rest* period, during which the tissue can be activated and at the end of which the tissue will self-activate. The timing behaviors of the action potential are modeled as *node automaton* (Fig. 7(b)).

A **node automaton** initializes with the Rest state. From the Rest state, the node can either self-activate or be acti-

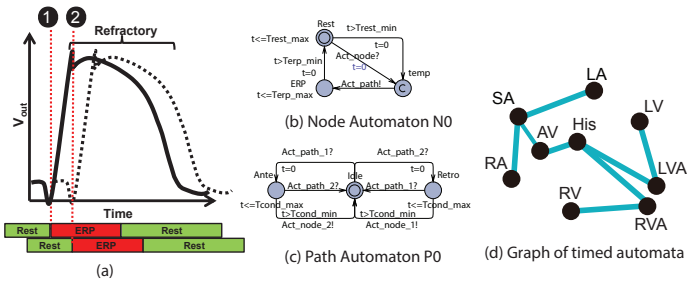

Figure 7: Node and Path Automata which models the timing properties of the heart tissue. A network of node and path automata models the generation and conduction of electrical activities of a heart

vated by external activations (indicated by Act_node). Upon activation the node transitions to the ERP state and activates all the paths connecting to the node (indicated by Act_path). In the ERP state the node does not respond to external activations. At the end of the ERP state the node transition to the Rest state. The duration a node automaton can stay in the Rest state is in the range [$Trest\_min$, $Trest\_max$], and the duration it can stay in ERP is in the range [$Terp\_min$, $Terp\_max$]. For heart tissue without the capability to self-activate, the parameters $Trest\_min$ and $Trest\_max$ are set to ∞. $Trest$ and $Terp$ are referred to as *parameters* of the node automaton.

The voltage change of the heart tissue will activate the tissue nearby with certain delay (Marker 2 in Fig. 7). This timing delay between heart tissue is modeled using *path automata* A_e (Fig. 7(c)). The initial state of a **path automaton** is Idle, which corresponds to no conduction. A path has two conduction directions, forward and backward. These are represented by the states Ante and Retro, named after their standard physiological terms Antegrade and Retrograde. If Act_path event is received from one of the nodes (1 or 2) connected to the path, the transition to Ante or Retro will occur in the path automaton. At the end of Ante and Retro the path will transition to Idle and send Act_node signal to the node automaton connected to the other end of the path (2 or 1). The spatial and temporal properties of a given human heart condition can be modeled by a network of node and path automata with different parameters (i.e., Fig. 7(d)).

To interact with the pacemaker, the Act_path events in node SA and RVA correspond to the inputs to the atrial and ventricular channel, respectively. The atrial and ventricular pacing (AP and VP) from the pacemaker will trigger Act_node events in node SA and RVA.

4. ABSTRACTION TREE OF HEART MODELS

The ideal heart model for closed-loop model checking of an implantable pacemaker not only covers all possible inputs to the pacemaker, but also has physiological explanations to all known heart conditions. However, no **single** heart model can satisfy both requirements. Therefore, a set of heart models must be employed where the different abstraction levels of the models strike a balance between coverage and expressiveness. More importantly, the heart models should have rigorous relationships among each other to provide formal guarantees.

In this section we present the abstraction tree framework that maintains formal *Timed Simulation* relationships between heart models and enables automated closed-loop model checking of implantable pacemaker.

4.1 Initial Set of Heart Models

The heart model structure described in the previous section is based on clinical electrophysiology, with state variables and parameters directly corresponding to physiological parameters. Therefore, domain experts from clinical electrophysiology can construct models of different heart conditions with their domain expertise and literature.

An example set of initial heart models is shown in Fig. 8. The different topologies of node and path automata represent the mechanism of different heart diseases. These heart models represent the current knowledge for heart condition variability, thus the set is inherintly incomplete, meaning there is no guarantee for 100% safety even if a property is satisfied in all of these models. These models are mostly used for providing physiological contexts for counter-examples returned by the model checker. Domain experts can always expand the set with knowledge of new heart conditions.

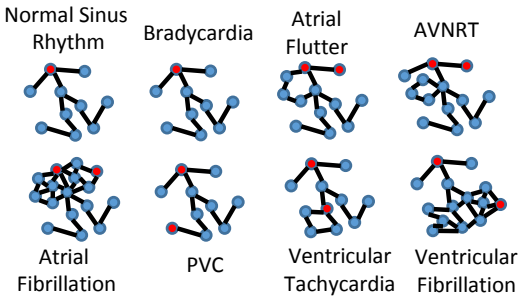

Figure 8: Examples of the initial set of heart models. The models are different in node and path topology and/or timing parameters.

4.2 Physiological Abstraction Rules

The initial set of heart models only represents a subset of all possible conditions. There always exists conditions that beyond our knowledge or that are combinations of known conditions. By using *over-approximation*, heart models can be created that cover the observable behaviors of the initial set and that introduce behaviors that were not captured in the initial set. Inevitably, some of the introduced behaviors will be physiologically invalid. This problem can be alleviated by carefully designing the abstraction rules so that behaviors introduced are mostly physiologically valid. The physiologically invalid behaviors can be eliminated during a validity check in the abstraction tree.

Physiological abstraction rules are developed to cover observable behaviors of heart models. Applying one abstraction rule to heart model(s) $H_1, H_i \ldots H_n, n \geq 1$ yields an abstract heart model H' such that all observable behaviors of H_i are covered by H'. For each heart model H_i, H' is a *timed simulation* of H_i. To illustrate, a subset of abstraction rules is described intuitively. The complete set of abstraction rules and the proofs of timed simulation relationship can be found in the tech report [8].

4.2.1 Rule R1: Convert Reentry Circuits to Activation Nodes

Within the conduction network of the heart, there can be multiple pathways between two locations, forming conduction loops. If the timing parameters of the tissue along the loop satisfy certain properties, there can be scenarios in which a depolarization wave circling along the circuit. The circuits are referred to as *Reentry Circuits*. Since the time interval for an activation wave to circle a reentry circuit is usually less than the intrinsic heart cycle length, the heart rate will be "'hijacked"' by the reentry circuit once the cycling is triggered, causing tachycardia. Reentry is the most common mechanism for tachycardia, which can be captured by our heart models that are used in [9].

The effect of reentry tachycardia is that activation signals coming out of the circuit with a given cycle length equal the sum of conduction delays along the circuit. It is therefore reasonable to model a reentry circuit as a self-activation node with the self-activation range equal to the sum of conduction delays.

Applicable Condition: The rule only affects the topology of the model, and therefore can be applied without preliminaries.

Output model: The "essential structure" of a heart model is the shortest paths (in terms of conduction delay) connecting self-activation nodes and/or sensing nodes. First detect all circles in the input graph. For each circle with nodes $N_i, i \in [1 \dots n]$ and paths $P_j, j \in [1 \dots m]$, remove all "non-essential" nodes and paths, create a node automaton N_s and connect to the nearest sensing node with a path automaton P_s.

Effect on parameters: For the new node automaton N_s, the minimum of the Trest parameter is set to the minimum of the sum of the conduction delays within the reentry circuit, and the maximum is set to infinity

Effect on behaviors: The new model captures the behavior of the original model when the reentry circuit is active and inactive. Additionally, the new model captures the behaviors of other heart conditions in which the rate of the reentry circuit is lower.

Fig. 9 shows an example in which a circle is replaced by a self-activation node.

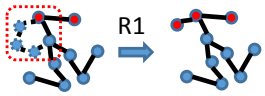

Figure 9: Rule 1: Remove reentry circuits from the model

4.2.2 Rule R2: Remove Irrelevant Structures

After the circles within the topology are removed, the topology of the heart model is in the form of a tree. Since the "non-essential" structures do not affect the activation signals from and/or to the sensing nodes, all the "non-essential" structures can be removed.

Applicable Conditions: The rule can only be applied after Rule 1 has been applied.

Output model: Trimmed topology with only the essential structure remaining.

Effects on parameters: There are no effects on parameters of the node and path automata.

Effects on behaviors: Applying this rule does not affect the observable behaviors of the model.

Fig. 11 shows an example in which non-essential structures are removed.

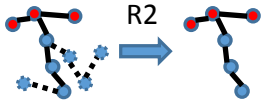

Figure 11: Rule 2: Remove non-essential structures

4.2.3 Rule R4: Merge Parameter Ranges

Timing periods of heart tissue, such as Rest and ERP, are modeled as locations in the node and path automata. The minimum and maximum time an automaton can remain in a location is governed by the parameters in the guards and invariants. By merging and expanding these periods, new behaviors are introduced where a heart model may remain longer in Rest, activate or self-activate a node faster, andsoforth.

Applicable Conditions: This rule applies to heart models with the same node and path topology but possibly with different parameters.

Output Model: The abstract model has the same topology as the original models.

Effects on parameters: The parameter ranges in the new model are a super-set of the parameter ranges in the old models.

Effects on behaviors: The abstract model captures all behaviors of the original models. In addition, heart conditions with parameters outside of the ranges of the original models are covered.

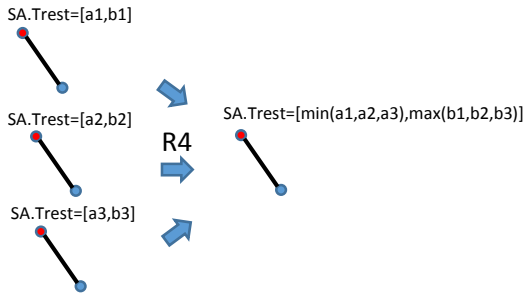

Figure 12: Rule 4: Merging parameter ranges

4.3 Abstraction Tree

By applying the abstraction rules to the initial set of heart models, an abstraction tree is created. Fig. 10 shows an example of an abstraction tree with the root model capturing all possible input sequences to the pacemaker. Self-activating nodes are marked as red and the Trest parameters are specified next to them. Note that this abstraction tree is not unique. With a different initial set of heart models and/or different rule application orders the abstraction tree can be very different. The abstraction tree can also be extended at any time if new heart conditions are specified. The following section demonstrates the use of this abstraction tree during the closed-loop model checking of the pacemaker design.

5. CASE STUDY: CLOSED-LOOP MODEL CHECKING OF IMPLANTABLE PACEMAKER

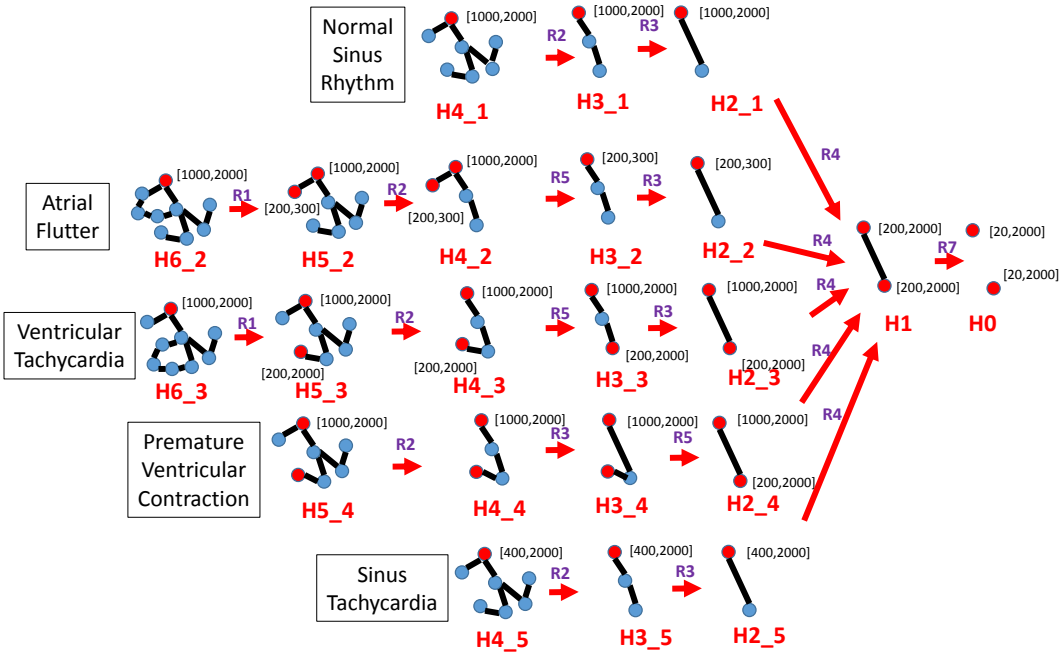

Figure 10: One example of abstraction tree of heart models

The model checker UPPAAL[12] was used to check Property 1 on the pacemaker model using the abstraction tree of heart models. The property is violated in $H0||PM$, thus thus the abstraction tree is followed to select pair $H1$ with the pacemaker model and Property 1 is verified again. The process continues till either the leaves of the tree are reached of the property is satisfied. The result is shown in Fig. 13, which demonstrates 5 different scenarios that can happen when using the abstraction tree. The shaded area marks the heart models with counter-examples.

Case 1: Property 1 is violated in $H2\_1||PM$ but is satisfied in its children $H3\_1||PM$. Careful examination of the counterexample finds it to be spurious and so it is successfully eliminated by model refinement.

Case 2: CE_{af} is returned by $H3\_2||PM$ and corresponds to intrinsic atrial tachycardia with fast atrial rate, which is a sub-optimal but non-lethal condition. The AV node of the heart will block a subset of the electrical events and maintain a normal ventricular rate. However, despite the filters in the pacemaker, the pacemaker still paces the ventricle for every 3 atrial activations, which extends fast atrial rate to more dangerous fast ventricular rate. The condition is referred to as Atrial Tachycardia Response in which the ventricular rate is increased inappropriately, thus requires revision of the pacemaker design.

Case 3: CE_{vt} is returned by $H3\_3||PM$ and corresponds to intrinsic ventricular tachycardia with fast ventricular rate. The counter-example is physiologically valid but the fast ventricular rate is due to the heart itself and is beyond pacemaker functionality. Therefore this scenario demands no revision of the pacemaker design.

Case 4: CE_{st} is returned by $H3\_5||PM$ and corresponds to sinus tachycardia, for instance, when the patient is exercising. The pacemaker improved the open-loop heart condition by pacing the ventricles AVI after each atrial event, which is a correct operation of the pacemaker despite the requirement violation.

Case 5: CE_{pvc} is returned by $H4\_4||PM$ and has a very similar input-output relationship to CE_n. However, the activations of the atrial node are triggered by retrograde conduction from ventricular paces (marker cond). The atrial activations trigger another ventricular pace after AVI, which will trigger another retrograde conduction. In this case the heart rate is inappropriately high, which corresponds to a dangerous closed-loop behavior referred to as *Endless Loop Tachycardia*.

From the above result, it can be seen that abstraction tree is able to 1) refine a heart model to eliminate spurious counter-examples as in Case 1; 2) provide (multiple) physiological explanations to a counter-example as in Case 2-5; and 3) resolve ambiguities caused by abstraction as in Case 4 and 5.

6. CONCLUSION

This paper demonstrated the use of an abstraction tree to automate the abstraction and refinement of heart models during closed-loop model checking of implantable pacemakers. Similar approaches can potentially be used for other autonomous medical devices to promote the use of model checking and formal methods during the development and certification of medical devices.

References

[1] R. Alur and D. L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126:183–235, 1994.

[2] P. Bogdan, S. Jain, and R. Marculescu. Pacemaker control of heart rate variability: A cyber physical system perspective. *ACM Transactions on Embedded Computing Systems*, 12(1s):50:1–50:22, 2013.

[3] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counter Example-Guided Abstraction Refinement for Symbolic Model Checking. *J. ACM*, 50(5):752–794, 2003.

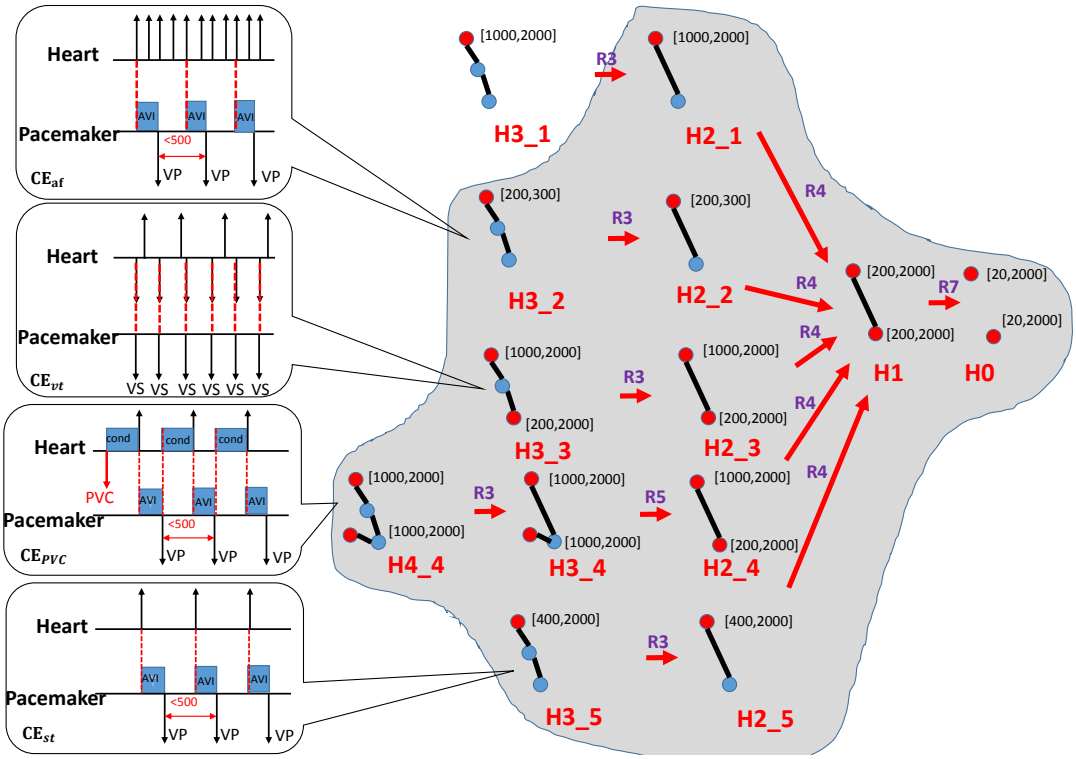

Figure 13: Four different physiological conditions in which Property 1 is violated. In CE_{af} the pacemaker extends a fast atrial rate to a dangerously fast ventricular rate; in CE_{vt} the ventricular rate is intrinsically fast; in CE_{st} the pacemaker appropriately maintained A-V conduction delay; in CE_{pvc} the pacemaker inappropriately increased the ventricular rate, causing Endless Loop Tachycardia

[4] R. Grosu, G. Batt, F. H. Fenton, J. Glimm, C. Le Guernic, S. Smolka, and E. Bartocci. From cardiac cells to genetic regulatory networks. In *Computer Aided Verification*, volume 6806 of *Lecture Notes in Computer Science*, pages 396–411. Springer Berlin Heidelberg, 2011.

[5] R. G. Hauser and B. J. Maron. "lessons from the failure and recall of an implantable cardioverter-defibrillator". *"American Heart Association, Circulation"*, pages 2040–2042, 2005.

[6] Ihor Gussak and Charles Antzelevitch and Arthur A.M. Wilde and Brian D. Powell and Michael J. Ackerman and Win-Kuang Shen. *Electrical Diseases of the Heart: Volume 1: Basic Foundations and Primary Electrical Diseases*. Springer, 2013.

[7] M. A. Islam, A. Murthy, A. Girard, S. A. Smolka, and R. Grosu. Compositionality results for cardiac cell dynamics. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, (HSCC '14), pages 243–252, 2014.

[8] Z. Jiang, H. Abbas, P. Mosterman, and R. Mangharam. Abstraction-Tree For Closed-loop Model Checking of Medical Devices. *Tech Report: http://repository.upenn.edu/mlab_papers/73*, 2015.

[9] Z. Jiang, A. Connolly, and R. Mangharam. Using the Virtual Heart Model to Validate the Mode-Switch Pacemaker Operation. *IEEE Engineering in Medicine and Biology Society*, pages 6690 –6693, 2010.

[10] Z. Jiang, M. Pajic, R. Alur, and R. Mangharam. Closed-loop verification of medical devices with model abstraction and refinement. *International Journal on Software Tools for Technology Transfer*, pages 1–23, 2013.

[11] M. Josephson. *Clinical Cardiac Electrophysiology*. Lippincot Williams and Wilkins, 2008.

[12] K. Larsen, P. Pettersson, and W. Yi. Uppaal in a Nutshell. *International Journal on Software Tools for Technology Transfer (STTT)*, 1997.

[13] A. Murthy, E. Bartocci, F. H. Fenton, J. Glimm, R. A. Gray, E. M. Cherry, S. A. Smolka, and R. Grosu. Curvature analysis of cardiac excitation wavefronts. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(2):323–336, 2013.

[14] S. Rossi, R. Ruiz-Baier, L. F. Pavarino, and A. Quarteroni. Active strain and activation models in cardiac electromechanics. *Proceedings in Applied Mathematics and Mechanics (PAMM)*, 11(1):119–120, 2011.

[15] F. B. Sachse, A. P. Moreno, and J. A. Abildskov. Electrophysiological modeling of fibroblasts and their interaction with myocytes. *Annals of Biomedical Engineering*, 36(1):41–56, 2008.

[16] K. Sandler, L. Ohrstrom, and R. McVay. Killed by Code: Software Transparency in Implantable Medical Devices. *Software Freedom Law Center*, 2010.

[17] N. A. Trayanova and P. M. Boyle. Advances in modeling ventricular arrhythmias: from mechanisms to the clinic. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 6(2):209–224, 2014.