Applied Ergonomics 58 (2017) 500–514

Contents lists available at ScienceDirect

Applied Ergonomics

journal homepage: www.elsevier.com/locate/apergo

A formal approach to discovering simultaneous additive masking between auditory medical alarms☆

Bassam Hasanain [a], Andrew D. Boyd [b], Judy Edworthy [c], Matthew L. Bolton [d, *]

[a] University of Illinois at Chicago, Department of Mechanical and Industrial Engineering, Chicago, IL, USA
[b] University of Illinois at Chicago, Department of Biomedical and Health Information Sciences, Chicago, IL, USA
[c] Plymouth University, School of Psychology, Cognition Institute, Plymouth, UK
[d] University at Buffalo, State University of New York, Department of Industrial and Systems Engineering, Buffalo, NY, USA

ARTICLE INFO

Article history:
Received 26 December 2015
Received in revised form
4 July 2016
Accepted 18 July 2016
Available online 29 August 2016

Keywords:
Medical alarms
Masking
Psychoacoustics
Formal methods
Model checking

ABSTRACT

The failure of humans to respond to auditory medical alarms has resulted in numerous patient injuries and deaths and is thus a major safety concern. A relatively understudied source of response failures has to do with simultaneous masking, a condition where concurrent sounds interact in ways that make one or more of them imperceptible due to physical limitations of human perception. This paper presents a method, which builds on a previous implementation, that uses a novel combination of psychophysical modeling and formal verification with model checking to detect masking in a modeled configuration of medical alarms. Specifically, the new method discussed here improves the original method by adding the ability to detect additive masking while concurrently improving method usability and scalability. This paper describes how these additions to our method were realized. It then demonstrates the scalability and detection improvements via three different case studies. Results and future research are discussed.

1. Introduction

Auditory medical alarms have many problems that can make them difficult to perceive and respond to (Edworthy, 2013; Boyd, 2010). The Pennsylvania Patient Safety Authority reports that there have been 194 documented problems with operators' responses to telemetry monitoring alerts from June 2004 to December 2008 resulting in at least 12 deaths (ECRI Institute & ISMP, 2009). A Sentinel Event Alert issued in 2013 reported 98 alarm-related incidents from 01/2009 to 06/2012: 80 resulted in patient death, 13 produced "permanent loss of function," and 5 extended the patient's hospital stay (The Joint Commission, 2013a). These types of problems occur because the sheer number of alarms sounding in modern medical environments often exceeds human perceptual and cognitive capabilities (Edworthy, 2013; Cvach, 2012; Way et al., 2014; The Joint Commission, 2013a; Lacherez et al., 2007). In statistics cited by the Joint Commission (The Joint Commission, 2013a), one patient can produce hundreds of alarms a day. This corresponds to thousands of alarms per day from a single unit and tens of thousands of alarms a day in a hospital. These problems have shown themselves to be difficult to solve and, as a result, medical alarms have consistently been identified as one of the most significant technological hazards to patient safety for more than a decade (ECRI Institute, 2014; Stead and Lin, 2009).

One of the understudied problems that occurs with simultaneously sounding alarms is simultaneous masking, a condition where multiple sounds interact in a way that prevents the human perceptual system from hearing one or more of them (Fastl and Zwicker, 2006). The Joint Commission's 2014 National Patient Safety Goal (NPSG) to "improve the safety of clinical alarm systems" claimed that "individual alarm signals are difficult to detect" (The Joint Commission, 2013b) and thus partially responsible for the patient safety problems associated with medical alarms (The Joint Commission, 2013a). As such, alarm audibility and distinguishability are largely acknowledged as problems that need to be overcome to address this NPSG (ECRI Institute, 2014; Vockley, 2014). Simultaneous masking is one of the factors that influences

☆ A subset of the results described in this manuscript will be presented at the 2016 Human Factors and Ergonomics Society (HFES) annual meeting (Bolton et al., 2016). This manuscript constitutes a substantial contribution beyond what is reported in the HFES paper. Specifically, we provide a significantly more detailed description of our method and its implementation, we present two additional case studies, and we report additional results of the shared case study. This manuscript also features a substantively extended discussion.

* Corresponding author.
E-mail address: mbolton@buffalo.edu (M.L. Bolton).

http://dx.doi.org/10.1016/j.apergo.2016.07.008
0003-6870/

whether or not an alarm is audible. It is difficult to assign specific injury or fatality numbers to alarm masking because it is an extremely challenging problem to identify and no standards exist for determining if masking plays a role in alarm non-response. However, we do have good evidence that masking is, in part, responsible for medical practitioners failing to respond to alarms. In an analysis of 26 operating room alarms and 23 intensive care unit alarms, Momtahan et al. (1993) found 25 pairs of alarms where one alarm was completely masked by the other. This analysis did not account for the effect of multiple, concurrently sounding alarms nor did it account for additive masking. Thus, there were likely more instances of masking than were detected. Further, Toor et al. (2008) found that low priority sounds could often mask higher priority alarms in an operating room environment. As such, many experts and researchers (Edworthy and Meredith, 1994; Meredith and Edworthy, 1995; Konkani et al., 2012; Edworthy and Hellier, 2006, 2005; Patterson et al., 1990; Patterson, 1982) have acknowledged that simultaneous masking is a problem that needs to be addressed.

Despite this, the vast majority of the work on alarm safety has focused on other areas (Edworthy, 2013). Auditory masking can be very difficult to detect experimentally because it may only occur with very specific interactions of multiple, concurrently sounding medical alarms. This problem is exacerbated by two conditions. First, most medical alarms are represented as melodies (patterns) of tonal sounds, which are particularly susceptible to simultaneous masking. Second, the number of alarms in modern medical environments (Thangavelu et al., 2014), given that the likelihood of masking increases with the frequency and number of concurrently sounding alarms (Humes and Jesteadt, 1989; Bosi and Goldberg, 2003).

To give analysts the ability to detect masking in configurations of tonal medical alarms without the need for experimentation, we have developed a computational method (Hasanain et al., 2014, 2015). The method uses a novel combination of psychoacoustic modeling and model checking. The psychoacoustics of simultaneous masking quantitatively relate sounds' physical characteristics (frequency/tone and volume) to the masking effect the sounds have on human perception using biologically-grounded mathematics (Bosi and Goldberg, 2003; Baumgarte et al., 1995; Schroeder et al., 1979; Ambikairajah et al., 1997; Brandenburg and Stoll, 1994; Brandenburg and Bosi, 1997). Model checking is an automated approach for conducting proofs (sometimes referred to as formal verification) (Clarke et al., 1999). Used together in our method, an analyst is able to model the sounding behavior of a configuration of tonal alarms and use model checking to determine if the psychoacoustics predict if the represented alarms can mask each other.

In the research presented here, we present an updated version of our method that improves its masking detection capabilities while simultaneously improving its scalability. Below we provide the necessary background to understand our previous and updated methods. We then state the objectives of our new work, present an updated version of the method, and present results that demonstrate its improved scalability and analysis capabilities.

2. Background

In the following, we cover the necessary background on model checking, the psychoacoustics of simultaneous masking, and the previous version of our method.

2.1. Model checking

Model checking comes from the field of formal methods. Formal methods are mathematical languages and techniques for the specification, modeling, and verification of systems (Wing, 1990). Specifications are formulated to rigorously describe desirable system properties, systems are modeled using mathematical languages, and verification mathematically proves whether or not the model satisfies the specification. Formal methods have been used successfully in a number of applications, especially in the analysis of computer hardware and software.

Model checking is an automated approach to formal verification (Clarke et al., 1999). A model describes a system as a set of variables and transitions between variable states, usually as a state machine or automaton. Specification properties, typically in a temporal logic (Emerson, 1990), assert ordinal, temporal relationships between system elements using system model variables. Verification processes exhaustively search through the system model to determine if these propositions hold. If they do, the model checker returns a confirmation indicating that it has proven the property is true. If there is a violation, an execution trace through the model, called a counterexample, is produced that shows exactly how the failure occurred. Model checking is particularly good at finding problems in systems with concurrency, where system elements can interact in ways unanticipated by designers (Grumberg and Veith, 2008). Model checking is normally used to evaluate discrete systems. However, hybrid modeling and analysis techniques can avoid this limitation (Dutertre and Sorea, 2004; Henzinger, 1996; Podelski and Wagner, 2006) by associating each discrete state in a model (like the sounding state of an alarm) with a value from a non-discrete continuum. For example, when using a timed automaton (Alur and Dill, 1994; Dutertre and Sorea, 2004), every discrete state in a formal model is assigned a real numbered time.

Researchers have used formal verification to successfully find and correct human factors issues in automated systems (see (Bolton et al., 2013) for a review). However, outside of our previous results (Hasanain et al., 2014, 2015), none of this work has explored how human perception and problems associated with it can be included in formal analyses.

2.2. The psychoacoustics of simultaneous masking

The psychoacoustics of simultaneous masking mathematically relate a sound's physical characteristics (its frequency/tone and volume) to the masking effect the sound has on human perception. The most successful of these are based on the expected excitation patterns of the human ear's basilar membrane (the physical structure largely responsible for allowing humans to distinguish between different sounds) (Bosi and Goldberg, 2003; Baumgarte et al., 1995; Schroeder et al., 1979; Ambikairajah et al., 1997; Brandenburg and Stoll, 1994; Brandenburg and Bosi, 1997). Conceptually, these models predict how a potentially masking sound (the *masker*) will stimulate the receptors on the basilar membrane based on its volume and relative frequency to the potentially masked sound (the *maskee*). This stimulation creates a higher volume threshold (in dB) that the *maskee* must exceed to be perceivable (Bosi and Goldberg, 2003).

The psychoacoustics used to describe masking represent frequency on the Bark scale (Zwicker and Feldtkeller, 1967). Specifically, the Bark scale maps a frequency (in Hz) to a location on the basilar membrane where the sound stimulates the receptors the strongest. Frequency to Bark conversion is calculated as

$$z_{sound} = 13 \cdot \arctan(0.00076 \cdot f_{sound}) + 3.5 \cdot \arctan\left((f_{sound}/7500)^2\right), \quad (1)$$

where f_{sound} is the sound's frequency in hz (Zwicker and Feldtkeller, 1967).

The masking threshold is then represented by a "masking curve" formulated as:

$$\text{curve}_{masker}(z_{maskee}) = \text{spread}_{masker}(\delta z) + v_{masker} - \Delta. \quad (2)$$

v_{masker} is the volume of the masker in dB. δz is defined as

$$\delta z = z_{maskee} - z_{masker}, \quad (3)$$

where z_{maskee} and z_{masker} are the frequency of the maskee and masker respectively on the Bark scale. spread_{masker} defines how the volume/magnitude of the masking threshold changes with δz. Δ is the minimum difference between a masker's and maskee's volume under which masking can occur.

There are multiple psychoacoustic spreading functions for different types of sounds (Bosi and Goldberg, 2003). Similarly, there can be different formulations of Δ depending on types of sounds. An example of a masking curve can be seen in Fig. 1.

These psychoacoustics can determine if a single sound can mask another sound and were the basis for previously published results (Hasanain et al., 2014, 2015). However, when there are multiple concurrent sounds, their combined masking threshold can be greater than the sum of each individual masker's effect. This additive masking (Humes and Jesteadt, 1989; Bosi and Goldberg, 2003) is modeled by combining the masking curve values of each potential masker on the power scale. Using the following equation to represent a volume (v in dB) on the power scale

$$\text{power}(v) = 10^{v/10}, \quad (4)$$

for a given potential maskee and N potential maskers, the aggregate masking threshold (in dB) is calculated as

$$\text{power}(\text{mthresh}_{maskee}) = \text{power}(\text{abs}_{maskee}) + \left(\sum_{n=1}^{N} \text{power}(\text{curve}_{masker_n}(z_{maskee}))^{\alpha}\right)^{1/\alpha}, \quad (5)$$

In this, α is a positive constant (Green, 1967) and abs_{maskee} is the absolute threshold of hearing (in dB) at the maskee's frequency (f_{maskee} in Hz) calculated as (Terhardt, 1979)

$$\text{abs}_{maskee} = 3.64 \cdot \left(\frac{f_{maskee}}{1000}\right)^{-0.8} - 6.5 \cdot e^{-0.6\left(\frac{f_{maskee}}{1000} - 3.3\right)^2} + 10^{-3} \cdot \left(\frac{f_{maskee}}{1000}\right)^4. \quad (6)$$

These psychoacoustics have shown themselves to be valid and useful for predicting masking for normal human hearing for decades (Bosi and Goldberg, 2003). They have been used to identify masking between recorded medical sounds (Toor et al., 2008). They have also served as the basis for lossy audio compression techniques like those used in the different versions of MPEG (Bosi and Goldberg, 2003).

2.3. Our original method

In our original method (Hasanain et al., 2014, 2015), an analyst was required to manually model a configuration of medical alarms based on a set architecture and code patterns. This allowed an analyst to describe each alarm as a state machine, where the frequency (in Hz) and volume (in Db) would change at specific times. Specifications could also be created using specification patterns for asserting the absence of partial and total masking. Model checking could then be used to determine if any given alarm could ever be masked by other alarms based on the psychoacoustics in Eq. (2). It is important to note that masking could only be detected between pairs of alarms, though multiple pairs of alarms could ultimately contribute to the total masking of a given alarm.

In this version of the method, we used the spreading function (for computations using Eq. (2)) from the MPEG2 audio codec (Schroeder et al., 1979). This is formulated as

$$\text{spread}_{\text{masker}}(\delta z) = 15.81 + 7.5 \cdot (\delta z + 0.474) - 17.5 \cdot \sqrt{1 + (\delta z + 0.474)^2}, \quad (7)$$

This spreading function was chosen specifically because of the expressiveness limitations of model checking. In particular, model checkers are unable to represent non-linear mathematical operations on model variables. Thus, non-linear psychoacoustics were implemented using lookup tables (Hasanain et al., 2015). Because this spreading function has only one independent variable (δz), it was computationally feasible to implement a lookup table for all of the possible values of δz at a resolution of 0.1 barks. We also used a Δ formulated as

$$\Delta = 14.5 + z_{masker} \quad (8)$$

because (Jayant et al., 1993) found this to be appropriate for tonal maskers.

While this version of the method proved itself to be useful (see Hasanain et al., 2014, 2015) it has four significant limitations. First, because it only considers masking between pairs or alarm sounds, it does not account for the additive effect of masking (Humes and Jesteadt, 1989; Bosi and Goldberg, 2003). Second, because Eq. (7) was predominantly used because of its computational convenience, there are psychoacoustics better suited to modeling the effect of tonal alarms. Third, the method scaled badly, resulting in analyses that would take prohibitively long to give useful results for complex applications (Hasanain et al., 2015). Fourth, the method required manual formal modeling and specification by analysts and provided no user support for interpreting analysis results.

3. Objectives

The work presented here shows how our method was re-implemented to address the above limitations of the original. To this end, we enable our method to account for the additivity of masking. We also update the psychoacoustics used to compute masking curves to better reflect the tonal nature of the masking sounds of alarms. We further create a computer program that enables the lookup tables to be optimized to improve scalability while using new psychoacoustics. Finally, this program was given features

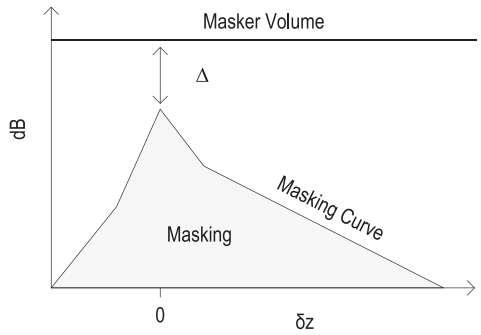

Fig. 1. An example masking curve. Peak masking occurs at $\delta z = 0$, where any sound with a volume Δ below the masker's will be masked. The masking effect decreases in accordance with the masking curve's spreading function as δz moves away from zero.

to simplify the model creation process and allow analysts to automatically visualize counterexamples to identify when and how masking can occur. Below we describe how these features were realized. To demonstrate the scalability improvements of the method and show that the new version has comparable detection capabilities to the original, we re-evaluate the application presented in (Hasanain et al., 2014, 2015) and compare the results. We also demonstrate the ability of the method to detect additive masking in another simple application. Finally, we show the ability of the method to detect additive masking in a realistic application by applying it to the alarms used in a telemetry monitoring system.

4. Method

The updated version of our method is shown in Fig. 2. To use it, an analyst first examines alarm documentation and describes the behavior of the alarms using a MS Excel spreadsheet, where each alarm is described as a sequence of tones (and pauses between tones) each with a defined frequency (Hz), volume (dB), and duration (s). Fig. 3 shows an example of how a single alarm would be described in our spreadsheet. When done modeling alarms, the analyst uses the computer program to automatically convert the described alarm configuration's behavior into a formal model.

4.1. Formal modeling architecture

The formal model used in the method has a set architecture (Fig. 4). The formal system model is made of a set of synchronously composed sub-models, each with a particular purpose. The clock sub-model uses a timed automaton (Alur and Dill, 1994; Dutertre and Sorea, 2004) to advance model time (*globalTime*) and communicate it to the other sub-models. Each alarm is represented as a sub-model that can start or stop sounding at appropriate times and adjust its state based on its current state and how long it has been sounding. Alarm state represents each of the distinct tones or pauses that occur over a complete sounding. For example, the alarm shown in Fig. 3 would have six states: one for when it is not sounding and one for each of the listed tones and pauses. A single masking computation sub-model uses the current state of each alarm, its associated "power alpha" (discussed subsequently), and the psychoacoustics of simultaneous masking to determine if any alarm is masked by the other sounding alarms. This sub-model also finds the minimum of the alarms' recommended next times (the *alarmNextTime* variables) to recommend a maximum amount (*maxNextTime*) to advance the clock.

4.2. The method's psychoacoustics

Model checkers cannot handle the nonlinear arithmetic of model variables (De Moura et al., 2004). Thus our method uses a pre-computed lookup table (functions) to represent nonlinear psychoacoustic computations. However, the size of your lookup tables can reduce the efficiency of your model. Thus, in our new method, we encapsulate all of the necessary non-linear mathematical operations into a single lookup table. This was optimized to ensure the minimum number of necessary entries for any given model. This was done to reduce verification time.

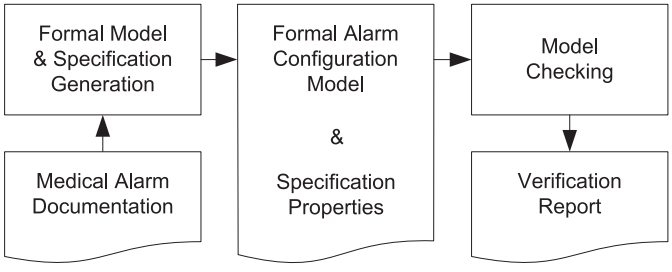

Fig. 2. A sequence diagram of our masking detection method.

Name	Freq (Hz)	Vol (dB)	Time (s)	…
AnAlarm	523	72	0.1	
	0	0	0.1	
	698	72	0.1	
	0	0	0.1	
	784	72	0.1	

Fig. 3. An example of an alarm as it would be described in a spreadsheet in our implementation of the method. The presented alarm has three tones separated by two pauses, where the order of tones and pauses is specified from top to bottom. The … is used to indicate that additional alarms would be described similarly to the right of the presented alarm.

The "power alpha" value discussed above and in Fig. 4 plays an important role in this optimization and allows for the detection of masking using a model checker. By transforming the maskee's volume and the masking effect of maskers into "power alpha" values using lookup tables, masking can be detected using only linear arithmetic operations. Fig. 5 explains the formulation and rational for the "power alpha" transformation.

Our method uses the relationship from Eq. (14) (Fig. 5) as the basis for its optimization. Specifically, the computer program pre-computes each alarm's "power alpha" values (using Eq. (12)) for each of the alarm's states. In the formal model, the alarm's state and "power alpha" value are communicated to the masking computation sub-model (see Fig. 4). The masking computation sub-model uses the lookup table (pre-computed by the computer program to implement Eq. (13) and optimized to minimize the number of entries) to obtain the "power alpha" value associated with each potential maskee-masker pair of alarms, based on each alarm's respective state. For each sounding alarm, the masking computation sub-model adds up each of the "power alpha" values with the given alarm as the maskee and compares it with the "power alpha" value from that alarm's associated alarm sub-model to determine if masking is occurring (see Eq. (14)). Thus, the use of the "power alpha" values allows our method to implement additive masking detection formally. We use α=0.33, which Lutfi (1983) found best captured the "over adding" of the masking effects of tones. However, the computer program allows for different analyst specified α

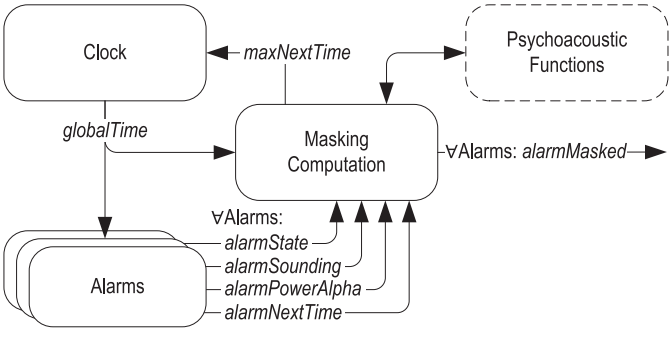

Fig. 4. The architecture for formally modeling a configuration of medical alarms in our method.

We know from Eq. (5) that a set of maskers will mask a maskee if

$$\text{power}\,(v_{maskee}) \leq \text{power}\,(\text{absolutethreshold}_{maskee}) + \left(\sum_{n=1}^{N} \text{power}\,(\text{curve}_{masker_n}(z_{maskee}))^{\alpha}\right)^{1/\alpha}. \tag{9}$$

Using basic algebraic operations, we know that

$$\text{power}\,(v_{maskee}) - \text{power}\,(\text{absolutethreshold}_{maskee}) \leq \left(\sum_{n=1}^{N} \text{power}\,(\text{curve}_{masker_n}(z_{maskee}))^{\alpha}\right)^{1/\alpha} \tag{10}$$

and thus that

$$(\text{power}\,(v_{maskee}) - \text{power}\,(\text{absolutethreshold}_{maskee}))^{\alpha} \leq \sum_{n=1}^{N} \text{power}\,(\text{curve}_{masker_n}(z_{maskee}))^{\alpha}. \tag{11}$$

If we let

$$\text{poweralpha}_{maskee} = (\text{power}\,(v_{maskee}) - \text{power}\,(\text{absolutethreshold}_{maskee}))^{\alpha} \tag{12}$$

and

$$\text{maskingpoweralpha}_{masker}(maskee) = \text{power}\,(\text{curve}_{masker}(z_{maskee}))^{\alpha}, \tag{13}$$

then we know that the maskee will be masked by the set of N maskers if

$$\text{poweralpha}_{maskee} \leq \sum_{n=1}^{N} \text{maskerpoweralpha}_{masker_n}(maskee). \tag{14}$$

Fig. 5. Explanation of "power alpha" and how it can be used to determine if masking is occurring.

values.

The "power alpha" computation in Eq. (13) relies on masking curve values. Because these values are pre-computed in our new method, we were able to be more selective in what spreading function Eq. (2) we use. Specifically, we can now use the more computationally complicated spreading function of

$$\text{spread}_{masker}(\delta z) = \begin{cases} -17\cdot\delta z + 0.15\cdot v_{masker} \\ \qquad \cdot(\delta z - 1)\cdot\theta(\delta z - 1) & \text{for } \delta z \geq 0 \\ -(6 + 0.4\cdot v_{masker})\cdot\left|\delta z\right| & \\ -(11 + 0.4\cdot v_{masker}\cdot(|\delta z| - 1)) & \\ \quad \cdot\theta(|\delta z| - 1) & \text{otherwise} \end{cases} \tag{15}$$

where $\theta(x)=1$ for $x\geq 0$ and $\theta(x)=0$ otherwise. This particular spreading function was chosen because it is the most appropriate for modeling the masking effects of tones on other tones (Brandenburg and Stoll, 1994). We also updated the way that Δ (from Eq. (2)) was computed. In the new version

$$\Delta = 6.025 + 0.275\cdot z_{masker}\ \text{dB}. \tag{16}$$

This new formulation was used for several reasons. First, it has been shown to be appropriate for tones (Ambikairajah et al., 1997). It was also used in the MPEG audio codec (Bosi and Goldberg, 2003), thus it has a well-established validity. Further, it will always be smaller than the Δ used in the original method (Eq. (8)). This means that it will increase the chances that our method will detect masking. Given that missing the detection of masking has significantly worse consequences than a false alarm, this was a preferable value of Δ for our purposes.

4.3. Formal model and specification generation

When the computer program generates the formal model and specifications, it creates a formal model in the input language of the symbolic analysis laboratory's (SAL's) infinite bounded model checker (De Moura et al., 2004) (see Fig. 6).

The model has eight parts. First, there are type definitions. These represent variable types that are used by other elements in the modeling architecture for representing real-valued time (which cannot be negative), "power alpha" values, and alarm state. Note that alarm state assumes there are `M` alarms, where each alarm will have `N` states and `N` can be different for each alarm. This is followed by constant definitions. The model contains only one constant, `bigMax`, which represents an arbitrarily large maximum on how much time can advance in any given modeled step. The constant definitions are followed by function definitions. This represents the lookup table used for computing the "power alpha" values of masking curves. See Section 4.3.3 for a deeper discussion of how this is computed.

The clock sub-model, which is responsible for maintaining and advancing time, is next. It is described in Section 4.3.1. A series of sub-models representing the behavior of each alarm in the configuration come next. Each of these represents the behavior of a given alarm. Section 4.3.2 describes how each alarm is modeled. The masking computation sub-model follows and is responsible for determining if masking is occurring at any given clock-indicated time. This is described further in Section 4.3.3. All of the sub-models are ultimately synchronously composed into the complete system model.

Finally, specification properties are used to assert the absence of masking in a model. These are discussed in Section 4.3.4.

4.3.1. The clock sub-model

The clock sub-model (Fig. 7) is unchanged from the previous version of the method (Hasanain et al., 2015). The clock is responsible for communicating the current time (`globalTime`) to the other sub-models. It is also responsible for advancing the clock. The `globalTime` is initially set to 0. For every following step in the model, `globalTime` is advanced to a new time that is always greater than the current `globalTime` and less than or equal to the `maxNextTime`, an input from the masking computation sub-model.

4.3.2. The alarm sub-models

The behavior of each alarm is described in separate sub-models, where each alarm model follows the same implementation pattern (Fig. 8). Each alarm has a constant value representing the length of its sounding cycle in seconds (`alarmXCycleTime`) which is set to a value [`TCycle`] derived by the description of the alarm behavior used in the generation process. Each alarm also has a variable representing its start time (`alarmXStartTime`). This is initially 0.

The alarm model is responsible for setting the start time and computing the amount of time the alarm has been sounding. Our model assumes that an alarm will sound for a single cycle and then stop (it can restart at any later time).

Thus, at any given `globalTime`, an alarm that is not sounding can begin sounding in the next state by setting the start time to

```
alarmConfiguration : CONTEXT =
BEGIN

  %Type definitions
  TIME       : TYPE = {X : REAL | X >= 0}; % in s
  POWERALPHA : TYPE = {X : REAL | X >= 1};
  ALARMSTATE : TYPE
    = {Alarm1_0, Alarm1_1, ..., Alarm1_N1,
       Alarm2_0, Alarm2_1, ..., Alarm2_N2,
       ...
       AlarmM_0, AlarmM_1, ..., AlarmM_NM};

  %Constants
  bigMax     : TIME = 60;

  %Function definitions
  ...

  %Clock sub-model
  ...

  %Alarm sub-models (alarm1 - alarmM)
  ...

  %Masking computation sub-model
  ...

  %Composition of the full system model
  system : MODULE = clock
                    || alarm1
                    || ...
                    || alarmM
                    || maskingComputation;

  %Specification properties
  ...

END
```

Fig. 6. An overview of the implementation of the formal modeling architecture (Fig. 4) as generated by our computer program. This implementation is written using the notation of SAL (see de Moura et al., 2003). Note that in this listing (and all subsequent listings), code highlighting is used to improve readability. SAL language reserved words (including built-in basic types) are blue; declared types are dark blue; constants are green; functions are orange (these appear in subsequent listing); comments start with a % and are gray; and everything else is black. Ellipses " …" are used to indicate the omission of content that is either detailed in subsequent listings or indicates an incremental series of like components or operations (e.g. the synchronous compositions of the alarm sub-models: `alarm1 || … || alarmM`). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the `globalTime` in the next state (see the code under `TRANSITION` in Fig. 8.). A start time greater than zero indicates that the alarm is sounding (`alarmXSounding = alarmXStartTime > 0`),

```
clock : MODULE =
BEGIN

  INPUT  maxNextTime : TIME
  OUTPUT globalTime  : TIME

  INITIALIZATION
    globalTime = 0;

  TRANSITION
    globalTime' IN {X: TIME | (X > globalTime)
                    AND (X <= maxNextTime)};

END;
```

Fig. 7. SAL code for representing the clock in the formal model.

information used by the masking computation sub-model. If sounding, the alarm computes how long it has been doing so as the difference between the global time and the alarm's start time (`alarmXTimeInCycle = globalTime - alarmXStartTime`). If the alarm is sounding and has not been sounding for longer than its cycle time in the next state's global time, the alarm keeps its current start time in the next state. If the alarm has been sounding for its full cycle time at the next global time, the alarm ceases to sound (sets the start time to zero) in the next state. Note that this behavior is the same as described in our original version of the method (Hasanain et al., 2015).

In the original method (Hasanain et al., 2015), an alarm would update its volume, frequency, and next time based on the amount of time the alarm had been sounding. In the new method, the alarm model updates its state in response to the alarm's sounding time. Based on this state, the alarm sub-model computes its other output values: the "power alpha" (`alarmXPowerAlpha`) associated with that state (a value pre-computed by the generating computer program) and the alarm next time (`alarmXNextTime`; the time that the next state change will occur).

4.3.3. The masking computation sub-model

At every time assumed by the clock (`globalTime`), the masking computation model (Fig. 10) does two things. First, it uses the state and information of all of the alarms to determine if each alarm is being masked, where a Boolean variable associated with each alarm (`alarm1Masked – alarmNMasked`) is computed to be true or false if masking is or is not occurring respectively. The values of these variables are determined using the pre-computed "power alpha" lookup table (a function) generated by the computer program (see Fig. 9). Specifically, for each possible maskee, the `thresholdPowerAlpha` function values are computed for each other alarm being treated as a masker and summed together. This sum is then compared to the given maskee's "power alpha" value. If the sum is greater than or equal to this value, then the Boolean variable is true. Otherwise it is false.

The second responsibility of the masking computation sub-model is to calculate the maximum next time (`maxNextTime`). This is the maximum amount of time the clock can advance to in the next step. It is calculated by finding the minimum of all of the alarm next times from each of the alarm sub-models.

4.3.4. Specification properties

To model check whether or not masking is present in a model, specifications must assert its absence. Our computer program also generates specification properties using minor variations of the patterns identified in our previous work (Hasanain et al., 2015). For each alarm, two properties are created (Fig. 11). The first (`alarmXPartialMasking`) is used to detect if any masking of a given alarm can occur. This uses linear temporal logic (LTL) to assert that, for all paths through the model (G), there should never be a situation where the alarm is making noise (`alarmXPowerAlpha > 1`) and the alarm is masked (`alarmXMasked`). The second specification (`alarmXTotalMasking`) is meant to check that an alarm is never completely masked: masked for its entire sounding cycle. Using LTL, this states that for all (G) paths through the model, it should never be true that the alarm goes from not sounding to sounding and masked in the next (X) state such that, from then on, the alarm is sounding and masked until (U) it is no longer sounding.

4.3.5. Running the model checker

With a completed alarm configuration model, model checking can be performed. The model our computer program creates is meant to be run with the infinite bounded model checker of SAL (De Moura et al., 2004). The model is run as on a Linux command

```
alarmX : MODULE =
  BEGIN

  INPUT  globalTime        : TIME

  LOCAL  alarmXStartTime   : TIME
  LOCAL  alarmXCycleTime   : TIME
  LOCAL  alarmXTimeInCycle : TIME

  OUTPUT alarmXState       : ALARMSTATE
  OUTPUT alarmXSounding    : BOOLEAN
  OUTPUT alarmXPowerAlpha  : POWERALPHA
  OUTPUT alarmXNextTime    : TIME

  INITIALIZATION
    alarmXStartTime   =  0;

  DEFINITION
    alarmXCycleTime   =  alarmXTCycle;
    alarmXTimeInCycle =  globalTime - alarmXStartTime;
    alarmXSounding    =  alarmStartTime > 0;
    alarmXState       =  IF    alarmXStartTime > 0 AND (alarmXTimeInCycle < alarmXTime1) THEN alarmX_1
                         ELSIF alarm1StartTime > 0 AND (alarmXTimeInCycle < alarmXTime2) THEN alarmX_2
                         ...
                         ELSE  alarmX_0 ENDIF;
    alarmXPowerAlpha  =  IF    alarmXState = alarmX_1  THEN alarmXPAlpha1
                         ELSIF alarmXState = alarmX_2  THEN alarmXPAlpha2
                         ...
                         ELSE 1 ENDIF;
    alarmXNextTime    =  IF    alarmXState = alarmX_1  THEN alarmStartTime + alarmXTime1
                         ELSIF alarmXState = alarmX_2  THEN alarmStartTime + alarmXTime2
                         ...
                         ELSE BigMax ENDIF;

  TRANSITION
    alarmStartTime'  IN {X: TIME | ((alarmStartTime = 0) AND ((X = globalTime') OR (X = 0)))
                         OR ((alarmXStartTime > 0)
                             AND (globalTime' < (alarmXStartTime + alarmXCycleTime))
                             AND (X = alarmXStartTime))
                         OR ((alarmXStartTime > 0)
                             AND (globalTime' >= (alarmXStartTime + alarmXCycleTime))
                             AND (X = 0))};
END;
```

Fig. 8. Generic SAL code for representing alarm behavior. `alarmX` is the name of the generic alarm that would be replaced with an actual alarm name in the generated model code. Note that red names represent alarm-dependent values that are inserted into the model by the computer program. `alarmXTCycle` represents the alarm's cycle time in seconds. `alarmXTime1` … represent the times at which each corresponding alarm state ends. For example, the alarm is in state `alarmX_1` if `alarmXTimeInCycle < alarmXTime1`. `alarmXPAlpha1` … represent the "power alpha" values associated with each state. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

line as follows:

```
sal-inf-bmc model specification -d D
```

where `sal-inf-bmc` is the command for invoking the model checker, `model` is the name of the SAL model being evaluated, `specification` is the name of the specification to check, and `-d D` sets the search depth to positive integer value `D`. In our analyses, depths should be set high enough to account for all of the possible transition states an analyst wants to consider. To ensure that all of the possible alarm interactions contained in a configuration are accounted for, this should be set to the sum of the total number of alarm states of each alarm. Such a value ensures that every alarm can fully transition through a entire cycle independently of all the

```
thresholdPowerAlpha(maskerState : ALARMSTATE, maskeeState : ALARMSTATE): POWERALPHA
  = IF    maskerState = Alarm1_1  AND maskeeState = Alarm2_1 THEN powerAlpha1_1_2_1
    ELSIF maskerState = Alarm1_1  AND maskeeState = Alarm2_2 THEN powerAlpha1_1_2_2
    ...
    ELSE 1 ENDIF;
```

Fig. 9. Lookup table implementation of Eq. (13) for use in the determination of whether masking (including additive masking) is occurring. Note that this uses the masking curve formulation from Eq. (2) with the spreading function from Eq. (15) and the Δ from Eq. (16). The number of entries in this table is computed by the computer program based on the number of possible pairs between potential maskee and masker alarms. For example, `powerAlpha1_1_2_1` represents the pre-computed "power alpha" value associated with masker Alarm1 in state `Alarm1_1` and maskee Alarm2 in state `Alarm2_1`.

```
maskingComputation : MODULE =
BEGIN

  INPUT  globalTime       : TIME
  OUTPUT maxNextTime      : TIME

  INPUT  alarm1State      : ALARMSTATE
  INPUT  alarm1PowerAlpha : POWERALPHA
  INPUT  alarm1NextTime   : TIME
  INPUT  alarm1Sounding   : BOOLEAN
  OUTPUT alarm1Masked     : BOOLEAN
  ...

  DEFINITION
    alarm1Masked = alarm1Sounding AND (thresholdPowerAlpha(alarm2State, alarm1State)
                                       ...
                                       + thresholdPowerAlpha(alarmMState, alarm1State)) >= alarm1PowerAlpha;
    ...

    maxNextTime     IN  {x: TIME | (x = alarm1NextTime OR ... OR x = alarmMNextTime)
                                   AND (x <= alarm1NextTime AND ... AND x <= alarmMNextTime)};

END;
```

Fig. 10. Generic SAL code for the masking computation sub-model.

other alarms. However, it is important to note that smaller search depth can be used. Because increasing the search depth will likely increase verification time, an analyst may wish to save time by using smaller depth. Any counterexamples returned from shorter searchers will still constitute valid results. However, a failure to find a counterexample at a depth below the suggested one may not genuinely indicate the absence of masking.

4.4. Counterexample visualization

In the original method, a model checker produced counterexample needed to be manually interpreted by analysts. We found that a variation of a vertical bar graph could be used to effectively show how alarms in a counterexample sounded in relation to each other and indicate when masking was occurring (Hasanain et al., 2015). Thus, in our new version of the method, we have added the ability to automatically create these graphs from a counterexample input.

Created graphs list the names of each alarm in a configuration along the vertical axis. Time is shown on the horizontal axis. Bars are plotted in the chart to show when the tones of each alarm are sounding. Smaller black lines are overlaid on the bars to show when a particular tone (or part of a tone) is masked. Examples of these plots can be seen later in Figs. 12–14.

5. Case studies

Below we present three different case studies that illustrate the power of the presented method. First, we evaluate the case study originally presented by Hasanain et al. (2015) to show both that the new formulation of the method can detect the masking of the original, but also that it does so more quickly. Second, we apply the method to a simple case study that demonstrates the ability of the method to detect additive masking. Third, we apply the method to a realistic application based on the GE CARESCAPE™Monitor B850 (GE Healthcare, 2010), a telemetry monitoring system.

In all of the reported case studies, formal verifications were performed using SAL's infinite bounded model checker on a Linux workstation with a 3.3 GHz Intel Xeon processor and 64 GB of RAM.

5.1. Case study 1: the original application

In the case study originally presented in (Hasanain et al., 2015), there were three alarms (Table 1). All three of these had a cycle featuring two tones separated by a pause. The frequencies, pause lengths, and volumes were all consistent with those commonly found in medical alarms (Momtahan et al., 1993; IEC 60601-1-8, 2003-08-14).

When evaluated with the original method (Hasanain et al., 2015), analyses were conducted on four different models. There was one model for each possible pair of alarms from Table 1 and one with all three. Each of these models were also modeled and evaluated with our new method to ensure the same analysis results were achieved and to compare verification times. A comparison of the analysis results with both methods can be seen in Table 2. These show that each specification property produced the same outcome when verified. Further, an examination of the counterexamples with our visualizer revealed that they both discovered the same masking conditions. Alarm 2 can be partially masked by Alarm 1 when Alarm 2's second tone overlaps with Alarm 1's first tone. Alarm 2 is partially masked by Alarm 3 when Alarm 2's first tone overlaps with Alarm 3's second tone. With specific timing such that Alarm 2's first tone completely overlaps with Alarm 3's second tone and Alarm 2's second tone completely overlaps with Alarm 1's first tone, Alarm 2 is completely masked. An illustration of this condition found by both versions of the method can be seen in Fig. 12.

The results in Table 2 also demonstrate the scalability

```
alarmXPartialMasking : THEOREM system
  |- G(NOT(alarmXPowerAlpha > 1 AND alarmXMasked));
alarmXTotalMasking   : THEOREM system
  |- G(NOT((NOT alarmXSounding)
       AND X(alarmXSounding AND alarmXMasked
             AND U(alarmXSounding AND alarmXMasked,
                   NOT alarmXSounding))));
```

Fig. 11. Specification property patterns for a given alarm (`alarmX`). `alarmXPartialMasking` asserts the absence of any masking for a given alarm. `alarmXTotalMasking` asserts that a given alarm will never be masked over its entire sounding cycle.

Table 1
Case study 1 alarm Configuration.

Name	Freq. (Hz)	Vol. (dB)	Time (s)
Alarm 1	261	80	0.250
	0	0	0.100
	370	80	0.250
Alarm 2	277	60	0.150
	0	0	0.050
	277	60	0.150
Alarm 3	524	85	0.200
	0	0	0.075
	294	85	0.200

Note. A given alarm's tones are listed vertically (from top to bottom) in the order that they sound in the alarm's cycle. A pause is indicated by a frequency and/or volume of 0.

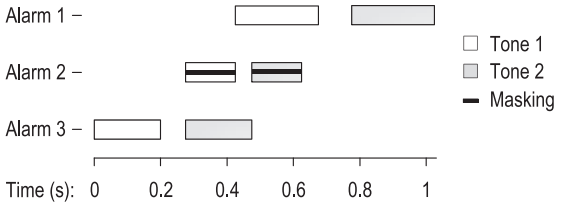

Fig. 12. Method-created counterexample visualization of how Alarm 2 can be completely masked by alarms 1 and 3 for the configuration described in Table 1. Specifically, the first tone of Alarm 2 is masked by the second tone of Alarm 3 and the second tone of Alarm 2 is masked by the first tone of Alarm 1.

improvements of our new method. Specifically, the new method was able to perform all of the analyses substantially faster than the original method, with reductions in verification times from 69.96% to 99.92%.

This case study illustrates the performance improvements that were achieved with the new version of the system. It also shows that comparable detection capabilities exist in this new version. However, this case study does not illustrate the additive masking detection capabilities of the method because, as the result in Fig. 12 show, masking does not require multiple maskers to overlap to produce masking. Additive masking detection is demonstrated in the next case study.

5.2. Case study 2: additive masking detection

The second case study evaluated the alarms shown in Table 3. These particular alarms were chosen because, with the exception of minor timing differences, these alarms are consistent with reserved alarm sounds from the international medical alarm standard (IEC 60601-1-8, 2003-08-14).

Because we were particularly interested in seeing if our method could detect additive masking, we used these alarms to construct four different models, one for each possible pair of alarms and one with all three alarms. By using our method to evaluate all four of these models we were able to determine if our method could find additive masking. Specifically, if we found masking that occurred due to the overlapping of two or more masking alarms with a maskee, where masking did not occur when each potential masker alone overlapped the maskee, then our method could find additive masking conditions.

We checked the specification properties for each alarm (for both partial and total masking) in each model created as part of our method. For models containing two alarms, verification search depths were set to 12. The model with all three alarms used a search depth of 18 for all verifications. Verification results are shown in Table 4.

These results reveal that the only masking that occurs happens when all three alarms are in the model. The results also show that only partial masking occurs for Alarms B and C. When the counterexamples associated with these verification failures were visualized (Fig. 13) they showed that masking only occurred as a result of additive masking. Specifically, partial masking of Alarm B's third tone occurs when it sounds concurrently with Alarm A's first tone and Alarm C's second tone. Alarm C's second tone can be partially masked if it sounds concurrently with Alarm A's first tone and Alarm B's third tone. Since no masking occurs in the models that only contain two alarms, this indicates that the observed masking is additive.

It is important to note that the partial masking shown in the two plots of Fig. 13 both occur in the same condition: the concurrent sounding of Alarm A's first tone, Alarm B's third tone, and Alarm C's second tone. This would mean that, when actually heard by a human, that either of the potentially masked tones (Alarm B's third or Alarm C's second) would be unhearable. It is a potential limitation of the method that it cannot identify which of these would actually be masked. Despite this limitation, the presented analyses do demonstrate the ability of the method to detect additive masking.

Table 2
Case study 1 verification results.

Model alarms	Alarm	Masking spec.	Original output		New outputs		Decrease
			Time (s)	Result	Time (s)	Result	
1 & 2	1	Partial	87.26	✓	0.15	✓	99.83%
		Total	63.76	✓	0.11	✓	99.83%
	2	Partial	99.31	×	0.47	×	99.53%
		Total	56.11	✓	0.24	✓	99.57%
1 & 3	1	Partial	67.05	✓	0.11	✓	99.84%
		Total	32.88	✓	0.12	✓	99.64%
	3	Partial	66.37	✓	0.16	✓	99.76%
		Total	127.57	✓	0.1	✓	99.92%
2 & 3	2	Partial	180.56	×	1.24	×	99.31%
		Total	95.82	✓	0.17	✓	99.82%
	3	Partial	85.49	✓	0.15	✓	99.82%
		Total	69.52	✓	0.09	✓	99.87%
1, 2, & 3	1	Partial	392.76	✓	6.65	✓	98.31%
		Total	320.62	✓	1.58	✓	99.51%
	2	Partial	1281.26	×	89.97	×	92.98%
		Total	492.76	×	148.29	×	69.91%
	3	Partial	297.92	✓	3.74	✓	98.74%
		Total	1205.43	✓	1.46	✓	99.88%

Note. ✓ indicates a verification confirmation and × indicates a verification failure with a counterexample.

Table 3
Case study 2 alarm Configuration.

Name	Freq. (Hz)	Vol. (dB)	Time (s)
Alarm A	261	84	0.1
	0	0	0.1
	329	84	0.1
	0	0	0.1
	392	84	0.1
Alarm B	261	84	0.1
	0	0	0.1
	329	84	0.1
	0	0	0.1
	293	84	0.1
Alarm C	523	84	0.1
	0	0	0.1
	293	84	0.1
	0	0	0.1
	392	84	0.1

However, even though the included alarms are realistic, this case study is still artificial. The ability of the method to detect masking in a more realistic context is explored in the third case study.

5.3. Case study 3: the GE CARESCAPE™Telemetry monitor

To evaluate a realistic application, we used our updated method to analyze the alarms in the GE CARESCAPE™Monitor B850 (GE Healthcare, 2010), a telemetry monitoring system compatible with the international medical alarm standard (IEC 60601-1-8, 2003-08-14). The GE monitor had the alarms shown in Table 5. There were four high-priority alarms that each played the same ten-tone alarm melodies (with the same timings), one medium-priority alarm with a three-tone melody, and one low-priority alarm with one tone. Our analyses assumed that any of these alarms could sound simultaneously.

We modeled these alarms in our new method and evaluated each alarm to see if they were ever partially or totally masked. Because of the complexity of the model, we anticipated scalability problems. Thus we attempted to minimize the search depth used in the verifications. Specifically, all properties were verified iteratively starting with the minimum depth capable of detecting masking. If no masking was found, the search depth was increased by one for each verification until masking was discovered or the verification took a prohibitively long amount of time. For partial masking properties, this meant search depths started at 2 and increased from there. For total masking properties, search depths started at the total number of states in the associated alarm and were iteratively increased up to 21. Note that search depths greater than 21 were not considered because of the amount of time required for the analyses. Further, because a depth of 21 would encapsulate what was likely to be the worst possible masking condition for the three high-priority alarms (when they all sounded at the same time as each other due to them all having the same tones), this was seen as sufficient.

Table 4
Case study 2 verification results.

Model alarms	Alarm	Masking spec.	Verification output	
			Time (s)	Outcome
A & B	Alarm A	Partial	4.48	✓
		Total	1.09	✓
	Alarm B	Partial	3.42	✓
		Total	2.34	✓
A & C	Alarm A	Partial	4.31	✓
		Total	0.99	✓
	Alarm C	Partial	2.89	✓
		Total	1.85	✓
B & C	Alarm B	Partial	3.96	✓
		Total	1.40	✓
	Alarm C	Partial	3.60	✓
		Total	1.48	✓
A, B & C	Alarm A	Partial	189.20	✓
		Total	13.56	✓
	Alarm B	Partial	670.23	×
		Total	9.83	✓
	Alarm C	Partial	815.29	×
		Total	16.94	✓

Note. ✓ indicates a verification confirmation and × indicates a verification failure with a counterexample.

A summary of our results can be seen in Table 6. These show that masking is possible between the alarms of the GE CARESCAPE. Specifically, partial masking was observed for all of the alarms. Two of the alarms can be totally masked: CPU-C1 and SystemLow (Fig. 14(b) and (h) respectively). This is concerning because it means that these alarms may not be heard or responded to by an observer. Because SystemLow is a low-priority alarm, one could argue that it being masked by higher priority alarms is not of that much concern. However, CPU-C1 is a high-priority alarm. The length of this alarm's cycle is 8.2 s. This means that if the alarm is masked, someone will not respond to it for at least that long. In a safety critical medical environment, this is an extremely long time. As such, the ability of CPU-C1 to be completely masked represents a significant patient safety problem with this device.

6. Conclusion and discussion

The work presented here has introduced a novel extension of our original method that accounts for its shortcomings. By changing the way that masking detection is performed in our computational model, we are now able to properly account for the additive effect of multiple maskers as well as use a more appropriate spreading function. This means that our masking prediction is more accurate than in the previous version and can thus detect masking conditions that it could not previously. In implementing our new approach to masking detection, we have also improved the scalability of our method. This improves the usefulness and applicability of the method. By providing a computer program that can generate formal models from alarm configuration descriptions and visualize counterexamples, we have improved the usability of the method.

In addition to these contributions, we have provided three different case studies that demonstrate different capabilities of the method. The first case study illustrates the significant scalability improvements that were achieved without any loss of detection capabilities. The second one shows that the method is indeed capable of detecting additive masking in a simple alarm configuration. Finally, the last case study showed how the method could be used to find masking conditions in a real application. As such, the method clearly has utility and, if used by medical device

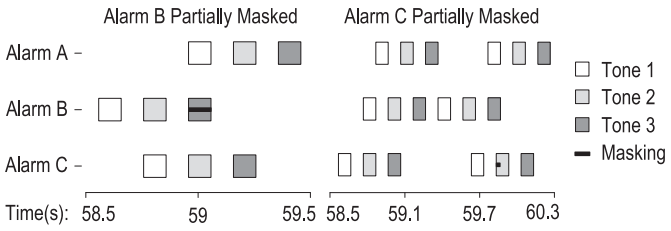

Fig. 13. Method-created counterexample visualization of how Alarms B and C can be partially masked by additive masking caused by the other two alarms in the configuration.

Table 5
Alarms from case study 3, the GE CARESCAPE telemetry monitoring system.

Name	Freq. (Hz)	Vol. (dB)	Time (s)	Name	Freq. (Hz)	Vol. (dB)	Time (s)	Name	Freq. (Hz)	Vol. (dB)	Time (s)
CPU-C1	523	72	0.1	D15K	523	81	0.1	D19KT	523	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	698	72	0.1		698	81	0.1		698	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	784	72	0.1		784	81	0.1		784	82	0.1
	0	0	0.3		0	0	0.3		0	0	0.3
	880	72	0.1		880	81	0.1		880	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	988	72	0.1		988	81	0.1		988	82	0.1
	0	0	1		0	0	1		0	0	1
	523	72	0.1		523	81	0.1		523	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	698	72	0.1		698	81	0.1		698	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	784	72	0.1		784	81	0.1		784	82	0.1
	0	0	0.3		0	0	0.3		0	0	0.3
	880	72	0.1		880	81	0.1		880	82	0.1
	0	0	0.1		0	0	0.1		0	0	0.1
	988	72	0.1		988	81	0.1		988	82	0.1
	0	0	5		0	0	5		0	0	5
SystemHigh	523	84	0.1	SystemMedium	523	83	0.2	SystemLow	523	79	0.2
	0	0	0.1		0	0	0.2				
	698	84	0.1		784	83	0.2				
	0	0	0.1		0	0	0.2				
	784	84	0.1		988	83	0.2				
	0	0	0.3		0	0	19				
	880	84	0.1								
	0	0	0.1								
	988	84	0.1								
	0	0	1								
	523	84	0.1								
	0	0	0.1								
	698	84	0.1								
	0	0	0.1								
	784	84	0.1								
	0	0	0.3								
	880	84	0.1								
	0	0	0.1								
	988	84	0.1								
	0	0	5								

Note. CPU-C1, D15K, D19KT, and SystemHigh are high-priority alarms. SystemMedium is a medium-priority alarm. SystemLow is a low priority alarm.

engineers and/or hospitals to evaluate and design alarm configurations, could significantly increase the chance that medical alarms are perceivable. This could have a profound impact on patient safety.

Despite these advances, there are still ways that the method could be improved and applied. These should be addressed in future efforts.

Table 6
Case study 3 verification results.

Alarm	Masking spec.	Search depth	Time (s)	Outcome
CPU-C1	Partial	2	145.70	×
	Total	21	60,967.05	×
D15K	Partial	2	135.21	×
	Total	21	145,870.50	✓
D19KT	Partial	2	135.21	×
	Total	21	148,252.81	✓
SystemHigh	Partial	2	139.02	×
	Total	21	395,441.48	✓
SystemMedium	Partial	2	104.24	×
	Total	21	203,702.73	✓
SystemLow	Partial	2	81.24	×
	Total	4	216.66	×

Note. Because a full search depth of $21 \times 4 + 7 + 2 = 93$ was not used in these analyses, it is possible (but unlikely) that a verified property might be untrue at higher search depths. Thus, verified properties are only guaranteed to be true at the presented search depth.

6.1. Scalability

Although our new version of the method significantly improved the method's scalability, the results shown for the third case study still indicate that the method does not scale well. The approach for iteratively increasing the search depth should allow analysts to mitigate some scalability concerns. However, scalability will definitely limit what systems the method can be applied to and when it will be appropriate. For example, the time required to run an analysis on a complex application would likely take a prohibitively long time for use in a dynamic environment like a hospital. There may be additional ways to improve the scalability of the method. Compositional verification (Cobleigh et al., 2003) is a process where large models can be verified using smaller independently verified, model components. As such, it may be possible to use compositional verification to check the interactions of specific tones across multiple analyses and use these analyses to draw conclusions about more complex configurations that use the analyzed tones. Many alarms, like those in the GE CARESCAPE, can have repeated patterns both within and between alarms. Thus, it may be possible to exploit symmetry in the models to further improve the scalability of the method (Emerson and Sistla, 1996). Future efforts should investigate how

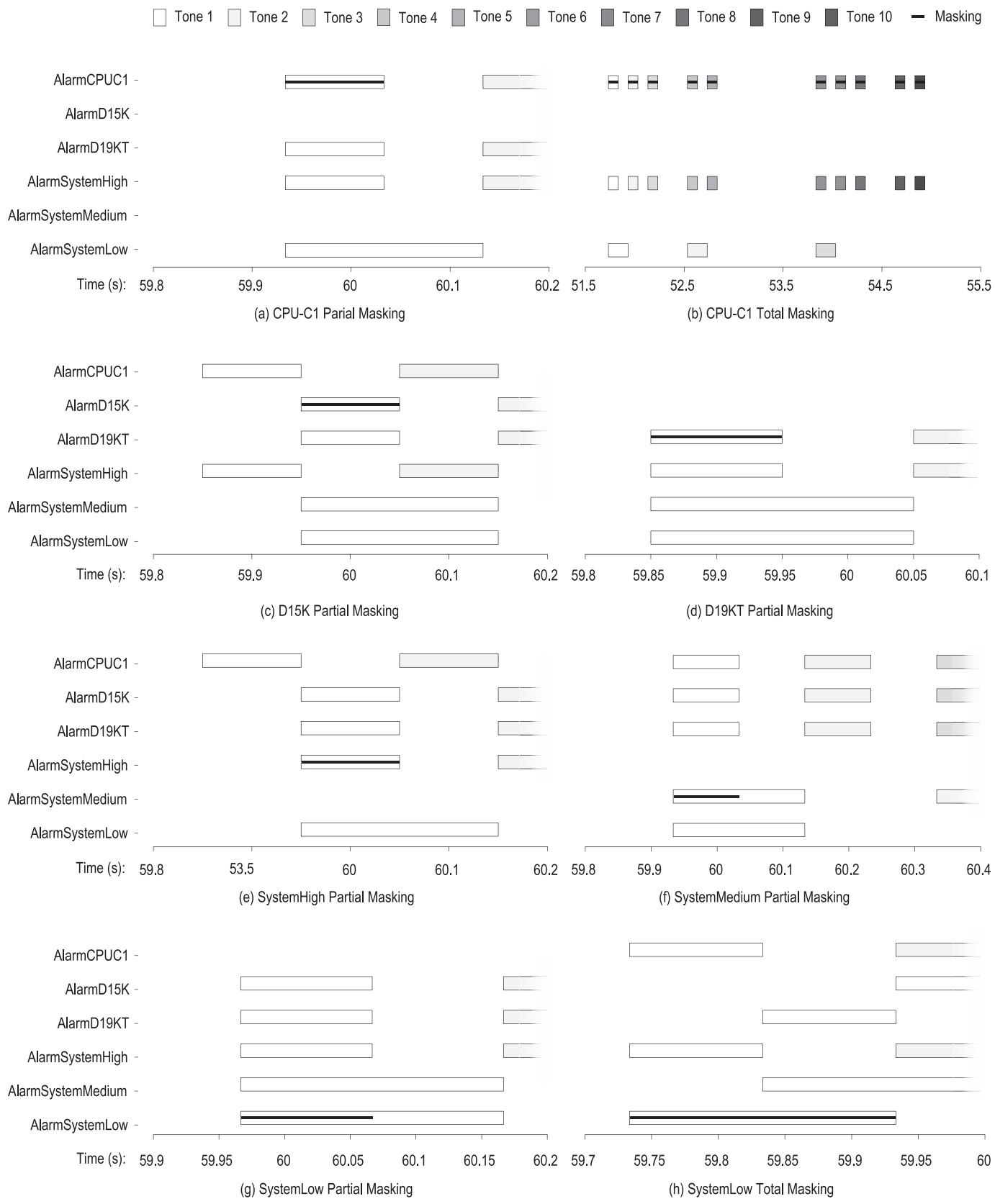

Fig. 14. Method-created counterexample visualizations of how the alarms in the GE CARESCAPE can mask each other. Note that only masking found in the respective analyses are shown in the graphs. The x-axis of the plots was reduced to only show the period of masking and not the full sounding cycles of the alarms.

these and other approaches could be used to improve our method's scalability.

Even if scalability persists as a limitation, this does not preclude the usefulness of the method. Specifically, the method could still be used for simple alarm configurations. Alternatively, system designers should have enough time to evaluate designed systems with other common alarm sounds (like those found in the international standard; (IEC 60601-1-8, 2003-08-14)) to ensure they avoid masking. Further, analyses could target alarm standards so that, although analyses may take a long time, their results could influence standard development and thus impact the safety of medical alarms across the industry without the method needing to be used by designers or hospitals. This is further explored in the next section. Thus, while future efforts may improve the scalability of the method, the method should still have utility.

6.2. The international medical alarm standard

The analyses presented under case studies 2 and 3 have interesting implications for future analyses. Specifically, the GE CARESCAPE alarms (case study 3) are in conformance with the IEC 60601-1-8 international medical alarm standard (IEC 60601-1-8, 2003-08-14). The alarms in case study 2 are nearly identical to reserved alarms sounds from IEC 60601-1-8 (there are only minor and likely inconsequential timing differences). Because masking was detected in both of these case studies, this would indicate that there are masking problems for alarms designed to adhere to the standard. This is potentially very dangerous. The method presented here offers the capabilities to systematically evaluate the alarm requirements and reserved sounds found in IEC 60601-1-8 and potentially explore solutions to discovered problems. This should be a priority of future work.

6.3. More complex alarm behavior and sounds

The tonal alarms we evaluated are realistic in that they were based on existing alarm systems and sounds from the IEC 60601-1-8 standard. However, there are other types of alarms beyond the tonal sounds currently supported by the method.

Features of IEC 60601-1-8 are not currently supported by the method. Specifically IEC 60601-1-8 alarms can have sub-frequencies, additional simultaneous frequencies that make each tonal sound more complicated (IEC 60601-1-8, 2003-08-14). Given that the method currently supports additive masking, accounting for these sub-harmonics should be an easy extension of the method. This should be the subject of future work. It is important to note that the inclusion of sub-frequencies could make the problem of alarm identification worse by adding additional masking potential. This could make it more likely that the primary harmonics of the alarm would be masked and thus potentially make it more difficult to identify. Future work should consider this in future investigations.

Additionally, there are a number of different spreading functions for representing the masking properties of different types of sounds (tonal vs noisy) (Bosi and Goldberg, 2003). Thus future extensions of the method should be able to readily account for different types of sounds.

Finally, the current formulation of our method works with discrete transitions in alarm state. However, alarm sounds can have many dynamic elements related to the frequencies and volumes of different components of a sound. In fact, IEC 60601-1-8 and expert design recommendations (Patterson, 1982; Patterson et al., 1990) allow for the use of dynamic changes and rhythms in alarms to facilitate identification. Accommodating these dynamics naturalistically would require a significant change in the way the method models alarms. However, heuristics or abstractions could be created to allow dynamics to be modeled with discrete steps within a given model or between models. Future work should investigate how such abstractions could be used with our method to allow these types of alarm features to be accounted for. Even if this is not successful, automated theorem proving techniques exists that can account for a wider variety of input model behaviors (Loveland, 2014). However, using these requires much more skilled analyst interaction and significantly more analysis time. Future efforts should investigate whether automated theorem proving is viable for this application.

6.4. Additional masking detection

Our method is capable of detecting simultaneous masking. However, there is also a phenomenon called temporal masking (Fastl and Zwicker, 2006). Temporal masking describes a situation where non-concurrent sounds, but ones that sounds in close temporal proximity, are masked. Such a phenomenon could increase the instances of masking in a given configuration. Psychoacoustics exist for accounting for these factors, however they are not readily adaptable to formal modeling. Thus, future work could investigate how to include these in our method using either extended formal modeling techniques or through clever exploitation of other analysis approaches with formal verification. Further, background and transient noises in health care environments can mask alarms or exacerbate other masking conditions (Block, 1992). Laroche et al. (1991) and Zheng et al. (2003) have developed a tool capable of evaluating the perceivability of alarms and alerts in noisy environments. Future work should explore how such detection capabilities could be included in our method.

6.5. Deeper analysis support

As illustrated in the result for case study 2, our method can detect masking conditions where it is unclear which of two or more alarms would actually be masked. Future work should investigate how to disambiguate such analysis results. Additionally, it is the nature of the model checking analyses that they only ever find one instance of a problem (a specification violation). This means that there could be more masking conditions in any given configuration than those initially found by the method. Ideally, our method would be able to find all of the conditions in a configuration that produce masking. Such capabilities should be investigated in the future. Finally, while our method can find masking problems, it is not clear how the method should be used to find solutions to those problems. Future work should focus on extending the method to support the exploration of design solutions that will avoid masking.

6.6. Experimental validation

The psychoacoustics used in our method have been well validated over the years and have served as the basis for the MPEG family of lossy audio codecs (Bosi and Goldberg, 2003). Thus, we expect our method to give accurate masking predictions. However, experiments with actual human subjects in realistic listening environments would allow us to validate our method's findings. This should be the subject of future work.

6.7. Other application domains

The work presented here and in previous analyses (Hasanain et al., 2014, 2015) have focused on medical alarms. However, the perceivability of alarms can play a critical role in the safety of aviation (Bliss, 2003), automobiles (Bliss and Acton, 2003), and

industrial systems (Rothenberg, 2009). Thus, future work should investigate how our method could be used in these and other safety critical domains.

6.8. Other alarm problems

There are many problems facing medical alarms beyond simultaneous masking (Edworthy, 2013). Because our work constitutes the first attempt to address alarm problems formally, there may be many future opportunities for extending our work to explore other alarm issues. In particular, there is good evidence suggesting that human mental workload can contribute to alarm mistrust, fatigue, and inattentional deafness (Dehais et al., 2014; Bliss and Dunn, 2000; Edworthy, 2013; Cvach, 2012; Way et al., 2014). Formal methods could help researchers discover when these conditions could occur. Such an analysis will need to integrate formal approaches for modeling alarm perception (Hasanain et al., 2014, 2015), workload (Moore et al., 2014; Houser et al., 2015), and task behavior (Bolton et al., 2011, 2012; Bolton and Bass, 2013) to be successful. This should be explored in future work.

References

ECRI Institute & ISMP, 2009. Connecting remote cardiac monitoring issues with care areas. Pa. Patient Saf. Auth. 6, 79–83. http://patientsafetyauthority.org/ADVISORIES/AdvisoryLibrary/2009/Sep6(3)/Pages/79.aspx.

Alur, R., Dill, D.L., 1994. A theory of timed automata. Theor. Comput. Sci. 126, 183–235.

Ambikairajah, E., Davis, A., Wong, W., 1997. Auditory masking and MPEG-1 audio compression. Electron. Commun. Eng. J. 9, 165–175.

Baumgarte, F., Ferekidis, C., Fuchs, H., 1995. A nonlinear psychoacoustic model applied to ISO/MPEG layer 3 coder. In: Proceedings of the Audio Engineering Society Convention. New York: Audio Engineering Society.

Bliss, J.P., 2003. Investigation of alarm-related accidents and incidents in aviation. Int. J. Aviat. Psychol. 13, 249–268.

Bliss, J.P., Acton, S.A., 2003. Alarm mistrust in automobiles: how collision alarm reliability affects driving. Appl. Ergon. 34, 499–509.

Bliss, J.P., Dunn, M.C., 2000. Behavioural implications of alarm mistrust as a function of task workload. Ergonomics 43, 1283–1300.

Block, F.E., 1992. Evaluation of users' abilities to recognize musical alarm tones. J. Clin. Monit. Comput. 8, 285–290.

Bolton, M.L., Bass, E.J., 2013. Generating erroneous human behavior from strategic knowledge in task models and evaluating its impact on system safety with model checking. IEEE Trans. Syst. Man Cybern. Syst. 43, 1314–1327.

Bolton, M.L., Siminiceanu, R.I., Bass, E.J., 2011. A systematic approach to model checking human-automation interaction using task-analytic models. IEEE Trans. Syst. Man, Cybern. Part A 41, 961–976.

Bolton, M.L., Bass, E.J., Siminiceanu, R.I., 2012. Generating phenotypical erroneous human behavior to evaluate human-automation interaction using model checking. Int. J. Human-Computer Stud. 70, 888–906.

Bolton, M.L., Bass, E.J., Siminiceanu, R.I., 2013. Using formal verification to evaluate human-automation interaction in safety critical systems, a review. IEEE Trans. Syst. Man Cybern. Syst. 43, 488–503.

Bolton, M.L., Hasanain, B., Boyde, A.D., Edworthy, J., 2016. Using model checking to detect masking in IEC 60601-1-8-compliant alarm configurations. In: Proceedings of the 2016 International Annual Meeting of the Human Factors and Ergonomics Society. HFES, Santa Monica, p. 5 (in press).

Bosi, M., Goldberg, R.E., 2003. Introduction to Digital Audio Coding and Standards. Springer, New York.

Boyd, A.D., 2010. Centralized Telemetry Monitoring Center Human Factors Report. Technical Report. University of Illinois at Chicago.

Brandenburg, K., Bosi, M., 1997. Overview of MPEG audio: current and future standards for low bit-rate audio coding. J. Audio Eng. Soc. 45, 4–21.

Brandenburg, K., Stoll, G., 1994. ISO/MPEG-1 audio: a generic standard for coding of high-quality digital audio. J. Audio Eng. Soc. 42, 780–792.

Clarke, E.M., Grumberg, O., Peled, D.A., 1999. Model Checking. MIT Press, Cambridge.

Cobleigh, J., Giannakopoulou, D., Păsăreanu, C., 2003. Learning assumptions for compositional verification. Tools Algorithms Constr. Analysis Syst. 331–346.

Cvach, M., 2012. Monitor alarm fatigue: an integrative review. Biomed. Instrum. Technol. 46, 268–277.

de Moura, L., Owre, S., Shankar, N., 2003. The SAL Language Manual. Technical Report. CSL-01–01 Computer Science Laboratory, SRI International Menlo Park.

De Moura, L., Owre, S., Rueß, H., Rushby, J., Shankar, N., Sorea, M., Tiwari, A., 2004. SAL 2. In: Proceedings of the 16th International Conference on Computer Aided Verification. Springer, pp. 496–500.

Dehais, F., Causse, M., Vachon, F., Régis, N., Menant, E., Tremblay, S., 2014. Failure to detect critical auditory alerts in the cockpit evidence for inattentional deafness. Hum. Factors 56, 631–644.

Dutertre, B., Sorea, M., 2004. Timed Systems in SAL. Technical Report. NASA/CR-2002–211858, SRI International.

ECRI Institute, 2014. Top 10 Health Technology Hazards for 2015. Health Devices, November. http://www.ecri.org/2015hazards.

Edworthy, J., 2013. Medical audible alarms: a review. J. Am. Med. Inf. Assoc. 20, 584–589.

Edworthy, J., Hellier, E., 2005. Fewer but better auditory alarms will improve patient safety. Qual. Saf. Health Care 14, 212–215.

Edworthy, J., Hellier, E., 2006. Alarms and human behaviour: implications for medical alarms. Br. J. Anaesth. 97, 12–17.

Edworthy, J., Meredith, C.S., 1994. Cognitive psychology and the design of alarm sounds. Med. Eng. Phys. 16, 445–449.

Emerson, E.A., 1990. Temporal and modal logic. In: van Leeuwen, J., Meyer, A.R., Nivat, M., Paterson, M., Perrin, D. (Eds.), Handbook of Theoretical Computer Science. MIT Press, Cambridge, pp. 995–1072 (Chapter 16).

Emerson, E.A., Sistla, A.P., 1996. Symmetry and model checking. Formal methods Syst. Des. 9, 105–131.

Fastl, H., Zwicker, E., 2006. Psychoacoustics: Facts and Models, vol. 22. Springer.

Green, D.M., 1967. Additivity of masking. J. Acoust. Soc. Am. 41, 1517–1525.

Grumberg, O., Veith, H., 2008. 25 Years of Model Checking: History, Achievements, Perspectives. Springer, Berlin.

Hasanain, B., Boyd, A., Bolton, M.L., 2014. An approach to model checking the perceptual interactions of medical alarms. In: Proceedings of the 2014 International Annual Meeting of the Human Factors and Ergonomics Society. HFES, Santa Monica, pp. 822–826.

Hasanain, B., Boyd, A., Bolton, M., 2015. Using model checking to detect simultaneous masking in medical alarms. IEEE Trans. Human-Machine Syst. 46 (2), 174–185. http://dx.doi.org/10.1109/THMS.2014.2379661.

Healthcare, G.E., 2010. CARESCAPE™Monitor B850 Technical Specifications Supplement. Technical Report 2040386–084D General Electric Company.

Henzinger, T.A., 1996. The theory of hybrid automata. In: Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science. IEEE Computer Society, Washington, pp. 278–292.

Houser, A., Ma, L.M., Feigh, K., Bolton, M.L., 2015. A formal approach to modeling and analyzing human taskload in simulated air traffic scenarios. In: Proceedings of the IEEE International Conference on Complex Systems Engineering. IEEE, Piscataway (6 pages).

Humes, L.E., Jesteadt, W., 1989. Models of the additivity of masking. J. Acoust. Soc. Am. 85, 1285–1294.

IEC 60601-1-8 (2003-08-14). Medical Electrical Equipment - Part 1-8. Geneva: International Electrotechnical Commission.

Jayant, N., Johnston, J., Safranek, R., 1993. Signal compression based on models of human perception. Proc. IEEE 81, 1385–1422.

Konkani, A., Oakley, B., Bauld, T.J., 2012. Reducing hospital noise: a review of medical device alarm management. Biomed. Instrum. Technol. 46, 478–487.

Lacherez, P., Seah, E., Sanderson, P., 2007. Overlapping melodic alarms are almost indiscriminable. Hum. Factors 49, 637–645.

Laroche, C., Quoc, H.T., Hétu, R., McDuff, S., 1991. 'Detectsound': a computerized model for predicting the detectability of warning signals in noisy workplaces. Appl. Acoust. 32, 193–214.

Loveland, D.W., 2014. Automated Theorem Proving: a Logical Basis. Elsevier, New York.

Lutfi, R.A., 1983. Additivity of simultaneous masking. J. Acoust. Soc. Am. 73, 262–267.

Meredith, C., Edworthy, J., 1995. Are there too many alarms in the intensive care unit? an overview of the problems. J. Adv. Nurs. 21, 15–20.

Momtahan, K., Hetu, R., Tansley, B., 1993. Audibility and identification of auditory alarms in the operating room and intensive care unit. Ergonomics 36, 1159–1176.

Moore, J., Ivie, R., Gledhill, T., Mercer, E., Goodrich, M., 2014. Modeling human workload in unmanned aerial systems. In: 2014 AAAI Spring Symposium Series. AAAI, Palo Alto.

Patterson, R.D., 1982. Guidelines for Auditory Warning Systems on Civil Aircraft. Civil Aviation Authority.

Patterson, R.D., Mayfield, T.F., Patterson, R.D., Mayfield, T.F., 1990. Auditory warning sounds in the work environment. Philosophical Trans. R. Soc. Lond. B, Biol. Sci. 327, 485–492.

Podelski, A., Wagner, S., 2006. Model checking of hybrid systems: from reachability towards stability. In: Hybrid Systems: Computation and Control. Springer, pp. 507–521.

Rothenberg, D.H., 2009. Alarm Management for Process Control: a Best-practice Guide for Design, Implementation, and Use of Industrial Alarm Systems. Momentum Press, New York.

Schroeder, M.R., Atal, B.S., Hall, J., 1979. Optimizing digital speech coders by exploiting masking properties of the human ear. J. Acoust. Soc. Am. 66, 1647–1652.

Stead, W.W., Lin, H.S. (Eds.), 2009. Computational Technology for Effective Health Care: Immediate Steps and Strategic Directions. National Academies Press, Atlanta.

Terhardt, E., 1979. Calculating virtual pitch. Hear. Res. 1, 155–182.

Thangavelu, S.D., Ifeachor, E., Edworthy, J., Yunus, J., Chinna, K., 2014. Challenges and recommendation of clinical alarm system in intensive care units from user perspective. In: 2014 IEEE Region 10 Symposium. IEEE, Piscataway, pp. 366–369.

The Joint Commission, 2013a. Medical device alarm safety in hospitals. Sentin. Even. Alert 50.

The Joint Commission, 2013b. Npsg.06.01.01: improve the safety of clinical alarm systems. Jt. Comm. Perspect. 33.
Toor, O., Ryan, T., Richard, M., 2008. Auditory masking potential of common operating room sounds: a psychoacoustic analysis. In: Anesthesiology. American Society of Anesthesiologists, Park Ridge, p. A1207 volume 109.
Vockley, M., 2014. Clinical Alarm Management Compendium. AAMI Foundation, Arlington.
Way, R.B., Beer, S.A., Wilson, S.J., 2014. Whats that noise? Bedside monitoring in the emergency department. Int. Emerg. Nurs. 22, 197–201.
Wing, J.M., 1990. A specifier's introduction to formal methods. Computer 23 (8), 10–22, 24.
Zheng, Y., Giguère, C., Laroche, C., Sabourin, C., 2003. Detectsound version 2: a software tool for adjusting the level and spectrum of acoustic warning signals. Can. Acoust. 31, 76–77.
Zwicker, E., Feldtkeller, R., 1967. Das Ohr als Nachrichtenempfnger. Hirzel Verlag, Stuttgart.