Formal Modeling and Analysis of Human Body Exposure to Extreme Heat in HI-Maude

Muhammad Fadlisyah[1], Peter Csaba Ölveczky[1], and Erika Ábrahám[2]

[1] University of Oslo, Norway
[2] RWTH Aachen University, Germany

Abstract. In this paper we use HI-Maude to model and analyze the human thermoregulatory system and the effect of extreme heat exposure on the human body. This work is motivated by the 2010 Sauna World Championships, which ended in a tragedy when the last two finalists were severely burnt in surprisingly short time (one of them died the next day). HI-Maude is a recent rewriting-logic-based formal modeling language and analysis tool for complex hybrid systems whose components influence each others' continuous dynamics. One distinguishing feature of HI-Maude is that the user only needs to describe the continuous dynamics of *single* components and interactions, instead of having to explicitly define the continuous dynamics of the entire system. HI-Maude analyses are based on numerical approximations of the system's continuous behaviors. Our detailed models of human thermoregulation and the sauna used in the world championships allow us to use HI-Maude to formally analyze how long the human body can survive when experiencing extreme conditions, as well as analyzing possible explanations for the still unsolved tragedy at the 2010 Sauna World Championships.

1 Introduction

Experimentation on humans might be the best way to understand the human body and mind, and can provide enormously important medical, scientific, and psychological knowledge for the good of society. However, experimentation on humans is typically quite costly, does (hopefully) not allow studying the body's reaction to extreme stress, is ethically fraught, and has a history with some very dark episodes. Although animals can sometimes replace humans in such experiments, they often function quite differently than humans, and experimentation on animals must take the growing animal rights sentiment into account.

Computer-based simulation and analysis can be a cheap and useful way to study the human body's reaction to stimuli – even fairly extreme stress that would be unethical to perform on a human. The human thermoregulatory system is an important part of the human body, as it tries to keep the person at a comfortable temperature even in difficult environments. Understanding the human thermoregulatory system is crucial to understand how our body will

F. Durán (Ed.): WRLA 2012, LNCS 7571, pp. 139–161, 2012.

respond to hostile and extreme environments, such as when firefighting, mining, deep-sea diving, traveling in space, or just practicing sports.[1]

Defining useful computer models of the human thermoregulatory system is a challenging task. We need to model as closely as possible the complex continuous dynamics of the various parts (skin, core, blood, etc.) as well as the complex continuous interactions between these parts, and between the body and its environment. We may also want to model behavioral, and therefore nondeterministic, aspects of the thermoregulatory system. Finally, we must be able to account for changing configurations, such as when a human jumps from an oppressively hot sauna into the cold snow; in this case, the thermal interactions change instantly, and the continuous dynamics of the entire system must be recomputed.

In this paper we use the rewriting-logic-based HI-Maude language and tool [8] to model the human thermoregulatory system according to established medical/physiological models and to analyze it under extreme conditions. In particular, our investigation is motivated by the 2010 Sauna World Championships, where both finalists collapsed after surprisingly short time. We therefore also present a fairly detailed model of the thermodynamics of a sauna, and use the HI-Maude tool to analyze how long people in different states of fitness can survive in different kinds of saunas and to analyze possible explanations for the tragedy in 2010 (the cause of which is still unknown).

HI-Maude is a recent extension of Real-Time Maude [13] to support the object-oriented formal modeling, simulation, and model checking of *hybrid systems* with combined discrete and continuous behaviors. The tool targets large and complex hybrid systems that typically have multiple physical entities that interact and influence each other's continuous behavior. For a thermal systems example, consider a cup of hot coffee which interacts with the surrounding room through different kinds of heat transfer, leading to a decrease in the coffee temperature and to a slight increase in the room temperature. One distinguishing feature of HI-Maude is the *modularity* and *compositionality* of the specification of the system's continuous dynamics. Non-compositional specification of the *whole* system is very hard, as it involves combining the ordinary differential equations (ODEs) that specify the dynamics of its components; it also requires redefining the system's continuous dynamics for each new configuration of interacting physical components. To achieve the desired modularity and compositionality, HI-Maude offers an object-oriented modeling methodology [5] that allows us to specify the continuous dynamics of *single physical entities* (such as the cup of coffee and the room) and of *single physical interactions* (such as thermal conduction and convection). Not only does this make it easier to specify the continuous dynamics of a system of interacting physical components, but it also means that the specification does not need to be redefined for each new configuration of physical entities. If we want to add a cup of coffee to the room, we just add a new coffee object and appropriate physical interaction objects to the state.

To analyze the system, whose continuous dynamics is usually defined by ordinary differential equations that are not analytically solvable, HI-Maude uses

[1] Heat stroke is the third leading cause of deaths among athletes in the U. S. [18].

adaptations of different numerical methods (the Euler method and Runge-Kutta methods of different order) to give fairly precise approximate solutions to coupled ordinary differential equations. These approximations are then used in HI-Maude simulation, reachability analysis, and linear temporal logic model checking.

In this paper we follow as much as possible established medical facts and models when defining our own model. The reference [11] gives an overview of some approaches to model the human thermoregulatory system. We choose the two-node Gagge model [9] as the basic model of the human body, since much scientific and engineering research on human thermoregulation is based on this model. For the formulas used to model the physiological aspects of the human thermoregulatory system, our main sources are [14,1]. We use [12] as the main source for some physics-related equations for modeling aspects of the interaction between the human body and its environment, and use [1,10] as main sources for modeling the behavioral aspect of human thermoregulatory system. Our main sources on how to model experimental subjects in different physiological conditions and degrees of preparedness for this competition are [16,2,15,14].

We have tried to model the sauna as closely as possible to the one used in the Sauna World Championships. Since we have not found any official description about the sauna used in that event, we rely on information gathered from stories, photographs, and videos available on the web, and from technical specifications of the equipment (e.g, the heater) and physical properties of the material used (e.g., the heat capacity of the rocks used for the heating). We use information from [4] for some physical properties of the environment.

In [8] we introduce the HI-Maude tool and illustrate its use on a fairly simple "proof-of-concept" model of a few aspects of the human thermoregulatory system. The model presented in this paper is completely different, and aims at being a fairly detailed model of the human thermoregulatory system and its interactions with equally faithfully modeled environments, including sophisticated models of the thermodynamics of different kinds of saunas. Just to mention a few differences: the model in [8] only considers the temperature of the body *core* (and hence only problems of hypothermia and hyperthermia), whereas this paper also takes into account the effect on the skin of exposure to heat (burn injuries) and the hydration state of the body (dehydration), which also needs a more refined model of the sweating rate; in [8], the blood vessels are modeled by three discrete states, whereas in this paper their dynamics is continuous; in [8], the dynamics of the heat flow by convection and radiation only considers the temperature difference between the body skin and the air temperature, whereas in this paper, the heat flow dynamics also considers the humidity factor of the surrounding environment, which also changes continuously because water poured periodically on the heating rocks during the sauna event will increase the humidity of the environment (and significantly increase the stress on the body, and is a crucial parameter that must be taken into account to seriously analyze the sauna accident); the model in [8] does not include heat exchange through respiration, whereas this paper includes the heat exchange through the air inhaled and exhaled during respiration; in [8] we only consider the physiological aspects of the human thermoregulatory system, whereas the model presented in this paper considers both physiological and behavioral aspects; the model in [8] considered the body to have the form of a cylinder when computing the area of the skin, whereas we now use the Dubois equation to approximate the size of the area of the body; and so on. As for the environment, in [8] the model is very simple and only considers that the room is filled with air (with a controlled heater to keep the maximum temperature), whereas in the current case study we model the sauna room with realistic parameters used in the event: our model of the sauna includes the heater, the heating rocks, and the periodic pouring of water onto the heating rocks, which increases heat to the skin and decreases the ability of the body to lose heat by evaporation. In this paper we also model three kinds of persons: a normal person, a person who has practiced in such conditions (like the contestants in the Sauna World Championships), and an unhealthy person. In short, the current model is incomparably more faithful and detailed and includes many more aspects needed to be able to analyze the sauna accident with a reasonable degree of accuracy.

We can only provide a sampler of our model in a short paper. The entire executable formal HI-Maude model, the analysis commands, a long report, and the HI-Maude tool itself, are available at `http://folk.uio.no/mohamf/HI-Maude`.

Section 2 briefly introduces the HI-Maude tool and the effort/flow-based modeling methodology upon which it is based. Section 3 gives an overview of the human thermoregulatory system. Section 4 presents some parts of our model of the human thermoregulatory system. Finally, Section 5 uses HI-Maude to analyze how long humans can stay safely in saunas and tries to understand what happened at the 2010 Sauna World Championships.

2 Modeling and Analysis of Hybrid Systems in HI-Maude

This section gives a brief overview of the HI-Maude tool and the modeling methodology in [5], upon which the tool is based, which adapts the *effort/flow* method [17] to model a physical system as a network of *physical entities* and *physical interactions* between the entities.

2.1 Effort/Flow Modeling of Interacting Hybrid Systems

In effort/flow modeling of a physical system, a *physical entity* is described by a real-valued *effort* value, a set of *attribute* values, and the entity's *continuous dynamics* (see Fig. 1, top left). The effort variable represents a dynamic physical quantity, such as temperature, that evolves over time as given by the continuous dynamics in the form of an ordinary differential equation (ODE), where its time derivative is a function of both the entity's attribute values *and* the flows of connected interactions (i.e., the time derivative $\dot{e}$ of the effort e can be described by an equation of the form $\dot{e} = f(\sum \mathit{flows}, \mathit{atts})$).

A *two-sided interaction* between two physical entities is described by a real-valued *flow*, a set of *attribute* values, and a *continuous dynamics*. The flow value

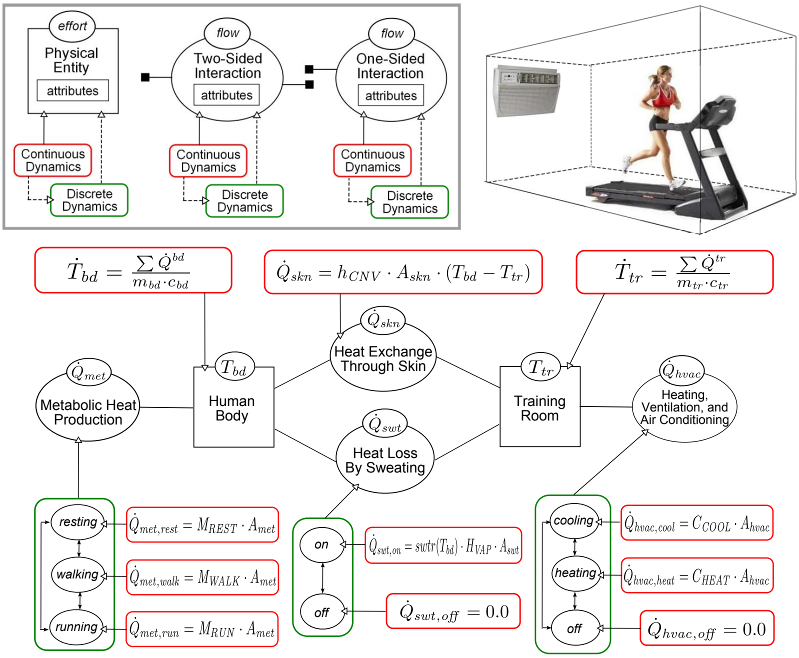

Fig. 1. A simple effort/flow model of the thermodynamics of a person in a gym

describes the dynamic interaction between two entities, whose evolution over time is specified by the continuous dynamics as an equation with the flow variable on the left-hand side and an expression referring to the interaction's attributes and the efforts of the connected entities on the right-hand side (i.e., $\mathit{flow} = g(\mathit{effort}_1, \mathit{effort}_2, \mathit{atts})$). A *one-sided interaction* represents an interaction of a physical entity with its environment. The system may also exhibit discrete dynamics, for representing, e.g., the changes of physical states of the system components, explicit control behaviors, communication, etc.

Figure 1 illustrates our modeling methodology on a simplified *thermal* system consisting of a woman working out at a gym. There are two *physical entities* of interest for thermal reasoning: the human body and the training room, both with the temperature T as their effort variable. The body produces heat, and the heat production increases as the exercise gets harder. The body releases heat to the room through the skin (e.g., by convection), and through sweating (heat is released from the body as the sweat evaporates). These heat transfers are represented as *two-sided interactions* where the flow variable ($\dot{Q}$) denotes the heat flow rate of the interactions. The *one-sided interaction* is used to model the heat production inside the human body through metabolism, and also to model the system which handles heating and cooling of the gym. Beside continuous dynamics, there are some physical phenomena which are suitably modeled as discrete dynamics, e.g., the changes of activity during the training (from running, to walking, to resting), activation and deactivation of sweating, and so on.

2.2 The HI-Maude Tool

The HI-Maude tool [8] supports the object-oriented effort/flow modeling and approximation-based formal analysis of interacting hybrid systems. Since we target complex systems and therefore do not restrict to *linear* ODEs for describing the continuous dynamics of a component/interaction, the continuous dynamics of a system is in general not analytically solvable. HI-Maude therefore uses numerical techniques to *approximate* the continuous behaviors by advancing time in small discrete time increments, and approximating the values of the continuous variables at each "visited" point in time. We have adapted the *Euler* [5], the *Runge-Kutta 2nd order* (RK2), and the *Runge-Kutta 4th order* (RK4) methods [7] to the effort/flow framework. Once the dynamics of the single physical components has been defined, HI-Maude

1. automatically defines the continuous dynamics of the entire systems, and
2. provides the usual Real-Time Maude formal analysis commands, but where the desired built-in approximation algorithm and the desired time increments used by the approximations are additional parameters of the commands.

Modeling. Since HI-Maude is an extension of Maude [3], a *membership equational logic* [3] theory (Σ, E), with Σ a signature[2] and E a set of *conditional equations* and *memberships*, specifies the system's state space as an algebraic data type. The system's instantaneous transitions are specified by a set R of (possibly conditional) *labeled instantaneous rewrite rules* `crl [`l`] :` t `=>` t' `if` *cond*, where l is a *label*, t and t' are two Σ-terms, and the condition *cond* is a conjunction of equations, memberships, and rewrites. We refer to [3] for more details on the syntax of Maude.

A declaration `class` C `|` att_1 `:` s_1`, ...,` att_n `:` s_n declares a *class* C with attributes att_1 to att_n of sorts s_1 to s_n. An *object* of class C is represented as a term `<` $O : C \mid att_1 : val_1, ..., att_n : val_n$ `>` of sort `Object`, where O, of sort `Oid`, is the object's *identifier*, and where val_1 to val_n are the current values of the attributes att_1 to att_n. The state is a term of sort `Configuration`, and has the structure of a *multiset* of objects and messages, with multiset union denoted by a juxtaposition operator that is declared associative and commutative, so that rewriting is *multiset rewriting* supported in Maude.

The dynamic behavior of concurrent object systems is axiomatized by specifying each of its transition patterns by a rewrite rule. For example, the rule

```
rl [l] :  < O : C | a1 : 0, a2 : y, a3 : w, a4 : z >  =>
          < O : C | a1 : T, a2 : y, a3 : y + w, a4 : z >
```

defines a parametrized family of transitions which can be applied whenever the attribute `a1` of an object `O` of class `C` has the value `0`, with the effect of altering the attributes `a1` and `a3` of the object. "Irrelevant" attributes (such as `a4`, and the *right-hand side occurrence* of `a2`) need not be mentioned in a rule (or equation).

---

[2] i.e., Σ is a set of declarations of *sorts*, *subsorts*, and *function symbols*.

A *subclass* inherits all the attributes and rules of its superclasses.

HI-Maude provides built-in classes for specifying physical entities and interactions. Concrete physical entities and interactions must then be defined as object instances of user-defined subclasses of these built-in classes. For modeling physical entities, the class `PhysicalEntity` is used:

```
class PhysicalEntity | effort : Float .
```

Sometimes we need additional continuous variables whose dynamics are time-derivative functions. The tool therefore provides the classes `PhysicalEntityAC`*k*, where k denotes the number of additional continuous variables:[3]

```
class PhysicalEntityACk | contvar1 : Float , ... , contvark : Float .
subclass PhysicalEntityAC1 ... PhysicalEntityACn < PhysicalEntity .
```

The attributes `contvar`*i* denote the additional continuous variables.

Objects of the classes `TwoSidedInteraction` and `OneSidedInteraction` are used to model two-sided and one-sided physical interactions, respectively:

```
class PhysicalInteraction | flow : Float, contdyntype : ContDynType .
class TwoSidedInteraction | entity1 : Oid, entity2 : Oid .
class OneSidedInteraction | entity : Oid .
subclass TwoSidedInteraction OneSidedInteraction < PhysicalInteraction .
```

The `contdyntype` attribute denotes the type of continuous dynamics specified for the interaction. The `entity1` and `entity2` attributes denote the two physical entities involved in the two-sided interaction. The `entity` attribute of the class `OneSidedInteraction` denotes the entity interacting with the environment.

HI-Maude requires the user to define the continuous dynamics by defining the function `effortDyn` for each physical entity:

```
op effortDyn : Object Float -> Float .
```

The first argument of `effortDyn` is the entity object itself; the second argument is the sum of the values of the flows to/from the entity, and is provided by the tool. `effortDyn(`$object, \sum \dot{Q}$`)` therefore defines the time derivative of the effort variable of the object. That is, if $\dot{e} = f(\sum \mathit{flows}, \mathit{atts})$, then we define `effortDyn(< O : C |` *atts* `>, X) =` f`(X,`*atts*`)`.

For physical entities with additional continuous variable(s), the functions `contvar`*i*`Dyn` define the continuous dynamics of those variables:

```
ops contvar1Dyn ... contvarnDyn : Object Configuration -> Float .
```

The first argument of the function is the entity object itself; the second argument is the entire multiset of objects in the system.

The function `flowDyn` defines the continuous dynamics of the physical interactions. To define the continuous dynamics of, respectively, two-sided interactions and one-sided interactions, the following formats are used:

[3] The tool currently provides the entity class with two additional continuous variables.

```
op flowDyn : Object Float Float -> Float .
op flowDyn : Object Float -> Float .
```

The first argument of the function is the interaction object itself. The second (and third) arguments are the effort variable values of the interacting physical entity/entities. Sometimes attribute values from objects that are not directly related to the interaction must be used to define the continuous dynamics of an interaction, in which case the following function is used:

```
op flowDyn : Object Configuration -> Float .
```

The second argument is the entire multiset of objects in the system.

Discrete transitions are modeled as rewrite rules. To ensure that such a rule is applied in a timely manner, HI-Maude provides the function

```
op timeCanAdvance : Configuration -> Bool .
```

so that if the user does *not* want time to advance when an object is in a certain state, (s)he must define `timeCanAdvance` to be `false` for those object states.

Formal Analysis. HI-Maude extends Real-Time Maude's analysis commands by allowing the user to select: (i) the numerical approximation technique used to approximate the continuous behaviors, (ii) the time increment used in the approximation, and (iii) discrete-switch-detection-based adaptive time increments. If we use a fixed time increment in the approximations, we may "miss" the time when the (approximated) effort value is such that a discrete event should take place (e.g., the body should go to state *hyperthermia* when its temperature reaches $38.9°C$). We can therefore also use *adaptive* time increments in connection with the Euler method to stop time advance exactly when a given continuous attribute has a desired value.

HI-Maude's hybrid *rewrite* command is used to simulate one behavior of the system from an initial state *initState* up to a certain duration *timeLimit*:

```
(hrew initState in time ~ timeLimit using numMethod stepsize stepSize
         discreteswitch dswitchType .)
```

$\sim$ is either '`<=`' or '`<`'; *numMethod* $\in \{$`euler`, `rk2`, `rk4`$\}$ is the numerical method used to approximate the continuous behaviors; *stepSize* is the time increment used in the approximation of the continuous behaviors; and (if *numMethod* is `euler`) *dswitchType* is `accurate` if adaptive step size should be used to stop time exactly when a discrete event must take place, and is `nonaccurate` otherwise.

HI-Maude's search command searches for up to *n* states that are matched by a search pattern with a substitution that satisfies an (optional) condition and that can be reached from an initial state in a given time interval:

```
(hsearch [n] initState =>* searchPattern [such that cond] in time ~ timeLimit
              using numMethod stepsize stepSize discreteswitch dswitchType .)
```

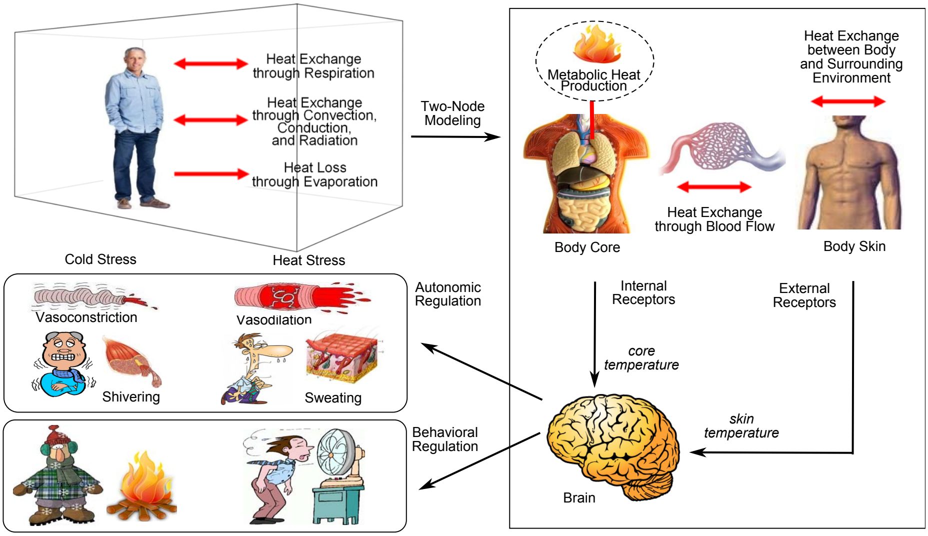

Fig. 2. The human thermoregulatory system

where $\sim \in \{$<, <=, >, >=$\}$, and *cond* is a condition on the variables in the search pattern. The following command finds the shortest time needed to reach a state:

```
(hfind earliest init =>* pattern [such that cond] using numMethod stepsize sSize
        discreteswitch dswitchType .)
```

Finally, HI-Maude's model checker extends Real-Time Maude's explicit-state time-bounded linear temporal logic model checker in the same way. The time-bounded hybrid model checking command is written with syntax

```
(hmc initState |=t formula in time ~ timeLimit using numMethod stepsize sSize
       discreteswitch dswitchType .)
```

3 The Human Thermoregulatory System

The human body needs to maintain a body temperature of around $37°C$ to function normally. The metabolic heat production within the body is the only internal factor affecting body temperature of a healthy person. The environment surrounding the body affects the body temperature by heat loss or gain through physical processes such as radiation, evaporation, convection, and conduction. *Hyperthermia* and *hypothermia* occur, respectively, when the body temperature increases, resp. decreases, significantly beyond normal.

Physiological and *behavioral* thermoregulation respond to changing environments in an attempt to ensure human survival and comfort. The primary control center of physiological thermoregulation is located in a part of the brain called the hypothalamus. The hypothalamus enables mechanisms to support heat loss from the body when the body temperature is increasing above normal levels that include: increasing the diameter of blood vessels to let more blood flow underneath the skin (*vasodilation*), which promotes heat loss by radiation, convection, and conduction; and increasing sweat production, which promotes heat loss by evaporation. When the body temperature is decreasing, the hypothalamus enables the following mechanisms to reduce heat loss and increase heat production: decreasing the diameter of blood vessels to let less blood flow underneath the skin (*vasoconstriction*), and stimulating the skeletal muscles to cause shivering, which increases heat production by the body. *Behavioral* response to heat or cold stress include taking off clothes or switch on a fan when the temperature is felt to be too hot, and putting on more clothes or moving closer to the fireplace when it feels too cold. Behavioral thermoregulation is related to a part of the brain called the cerebral cortex.

The Two-Node Modeling Approach. In the two-node Gagge model [9] the human body is considered as consisting of two concentric layers where the inner layer is the central core, and the outer layer is the skin shell. Heat exchange between the body and the environment takes place continuously at the skin surface. Heat generated inside the body is transfered to the skin surface through blood flow. From the skin, heat is transferred to the environment by convection, conduction, radiation, and sweat evaporation. Heat in excess of that which can be dissipated is stored in the tissue, resulting in a rise of body temperature. As mentioned below, we extend this basic Gagge model to also take heat exchange between the body and the environment through respiration into account.

4 Modeling the Human Thermoregulatory System

Using the two-node modeling approach, we model the body core, the body skin, and the surroundings as *thermal entities*, and the heat flow among these entities as *thermal interactions*, as shown in Fig 3. Heat flows between the core and the skin through blood vessels, and between the body and the environment through respiration. Heat flows between the skin and the environment through convection, radiation, and evaporation. The heat production inside the body through metabolic processes and the heat production by muscles through shivering are represented as one-sided thermal interactions.

4.1 Thermal Entities of the Human Body

The change of temperature of a thermal entity with mass m and specific heat capacity c is given by $\dot{T} = \frac{\sum \dot{Q}}{m \cdot c}$, where $\sum \dot{Q}$ is the amount of heat transferred per time unit. We therefore model a thermal entity by extending the built-in class `PhysicalEntity` with the entity's heat capacity and mass:

```
class ThermalEntity | mass : Float, heatCap : Float .
subclass ThermalEntity < PhysicalEntity .
```

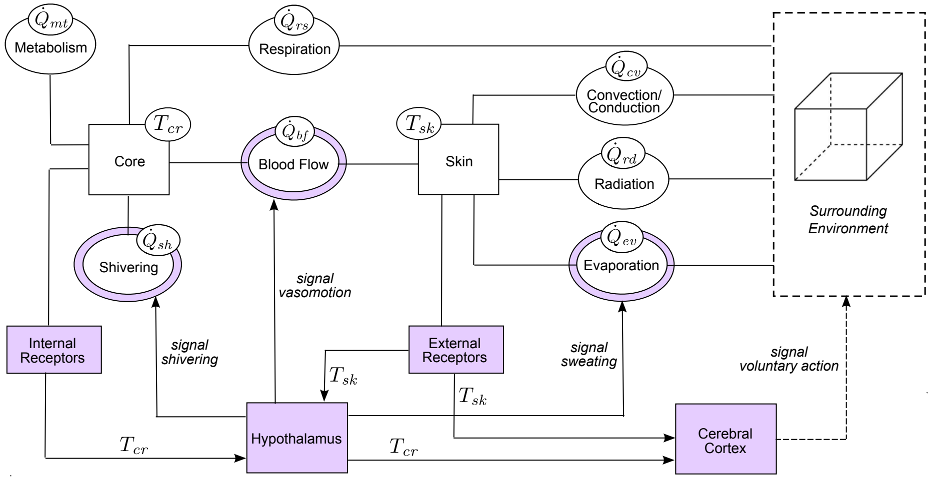

Fig. 3. Effort/flow model of the human thermoregulatory system

The continuous dynamics of the effort variable of the entity is defined as[4]

```
eq effortDyn(< TE : ThermalEntity | mass : MASS, heatCap : HC >, SF) = SF / (MASS * HC) .
```

Since we want to add another continuous variable (the amount of water in a person) to the body core, we define the body core as a subclass of both `ThermalEntity` and `PhysicalEntityAC1`, where the new continuous attribute `contvar1` denotes the amount of water in the person. The body core component is defined by extending these classes with the entity's core state, body water state, the initial amount of water in the body, and some factor values for sweating, blood flow, and respiration:

```
class CoreHumanBody | coreState : CoreState,  bWaterState : BWaterState,
                      bWaterInit : Float, sweatRateFactor : Float,
                      bloodFlowRateFactor : Float, respiRateFactor : Float  .
subclass CoreHumanBody < ThermalEntity PhysicalEntityAC1 .
```

We define the temperature-related and body-water-related states of the core:

```
sort CoreState .
ops normal mildHyperthermia modHyperthermia sevHyperthermia mildHypothermia
    modHypothermia sevHypothermia death : -> CoreState [ctor] .
sort BWaterState .
ops normal modDehydration sevDehydration death : -> BWaterState [ctor] .
```

The continuous dynamics of the body water of the component is defined as

[4] In this paper we follow the Maude convention that variables are written with (only) capital letters, and do not show the variable declarations.

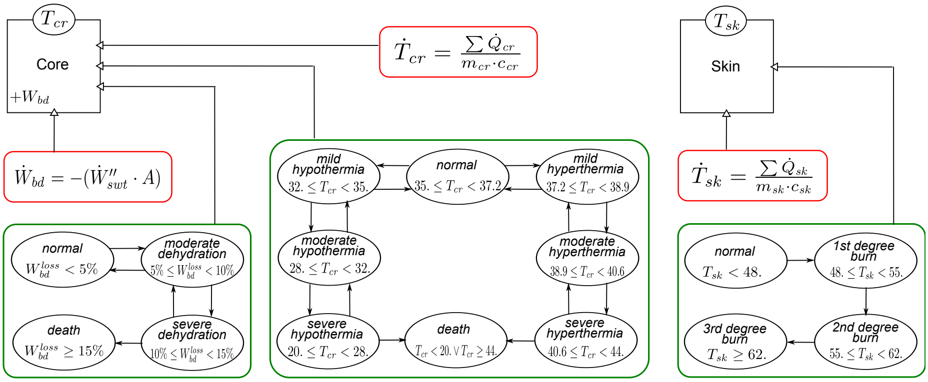

Fig. 4. The discrete and continuous dynamics of the core and the skin

```
eq contVar1Dyn(< CORE : CoreHumanBody | >,
               < BLF : BloodFlow | entity1 : CORE, entity2 : SKIN >
               < SKIN : SkinHumanBody | >
               < EVAP : Evaporation | entity1 : SKIN, sweatRate : SWTR, area : A >
               REST) = SWTR * -1.0 * A .
```

A thermal interaction between two thermal entities is a two-sided interaction, with an additional attribute that denotes the area of the flow. Similarly, the source of heat flow to a thermal entity is a one-sided interaction:

```
class ThermalInteraction | area : Float .
subclass ThermalInteraction < TwoSidedInteraction .
class ThermalFlowSource | area : Float .
subclass ThermalFlowSource < OneSidedInteraction .
```

The skin component is defined by extending the class `ThermalEntity` with an attribute for different degrees of burn injuries:

```
class SkinHumanBody | skinState : SkinState .
subclass SkinHumanBody < ThermalEntity .
sort SkinState .
ops normal firstDBurn secondDBurn thirdDBurn : -> SkinState [ctor] .
```

Changes in the body condition caused by temperature changes are represented as discrete events. For example, the core experiences *severe hyperthermia* if the core temperature exceeds $40.6°C$; this causes the sweating process to stop:

```
crl [modhyperthermia-to-sevhyperthermia] :
   < CORE : CoreHumanBody | effort : TEMP, coreState : modHyperthermia >
   < BLF : BloodFlow | entity1 : CORE, entity2 : SKIN >
   < SKIN : SkinHumanBody | >
   < EVAP : Evaporation | entity1 : SKIN >
   =>
   < CORE : CoreHumanBody | coreState : sevHyperthermia >     < BLF : BloodFlow | >
   < SKIN : SkinHumanBody | >     < EVAP : Evaporation | state : off >
   if TEMP > 40.6 .
```

To ensure that these rules are applied in a timely manner, we use the built-in `timeCanAdvance` function to define, for each core state, when time can advance *without* a rule having to be taken. For example:

```
eq timeCanAdvance(< CORE : CoreHumanBody | effort : TEMP, coreState : sevHyperthermia >)
 = TEMP > 40.6 and TEMP <= 44.0 .
```

Changes in the volume of water in the body may cause discrete changes in the body core (see Fig. 4). For example, the body water state changes to *severe dehydration* if the body has lost more than 10% of its initial amount of water:

```
crl [moddehydration-to-sevhydration] :
    < CORE : CoreHumanBody | contVar1 : BWATERCUR, bWaterInit : BWATERINIT >
   =>
    < CORE : CoreHumanBody | bWaterState : sevDehydration >
   if bWaterLoss(BWATERCUR, BWATERINIT) >= 0.1 .
```

Change in skin temperature may change the state of the skin, e.g., to *second degree burn* if the skin temperature exceeds $55.0°C$; this causes the evaporation to stop (the skin experiences *third degree burn* if its temperature exceeds $62.0°C$):

```
crl [1st-degree-burn-to-2nd-degree] :
    < SKIN : SkinHumanBody | effort : TEMP, skinState : firstDBurn >
    < SWEAT : Evaporation | entity1 : SKIN >
   =>
    < SKIN : SkinHumanBody | skinState : secondDBurn >
    < SWEAT : Evaporation | state : off >     if TEMP >= 55.0 .
```

4.2 Thermal Interactions of the Human Body

Due to lack of space, we only explain the modeling of one of the thermal interactions in the system, and refer to [6] for an overview of the other interactions.

The heat flow rate between the core and the skin through the blood vessels per square meter of body area can be computed using the equation $\dot{Q}_{bf} = (K + \dot{m}_{bl} \cdot c_{bl})(T_{cr} - T_{sk})$, where K is the thermal conductivity between core and skin, $\dot{m}_{bl}$ is the blood flow rate and c_{bl} is the specific heat capacity of blood [14]. The model of heat exchange through blood flow is shown in Fig 5. Two discrete states represent whether the blood flow is active or not, since blood stops flowing when a person is dead (since the heart cannot pump blood anymore). Increasing or decreasing blood flow through vasodilation or vasoconstriction is managed by the hypothalamus which determines the value of the blood flow rate.

The blood flow component is defined by extending the `ThermalInteraction` class with attributes for the blood flow rate, the blood thermal conductivity, and the blood heat capacity:

```
class BloodFlow | state : onOffStatusType, conduct : Float, heatCap : Float,
                  bloodFlowRate : Float .
subclass BloodFlow < ThermalInteraction .
```

The continuous dynamics of the flow variable is defined in the usual way:

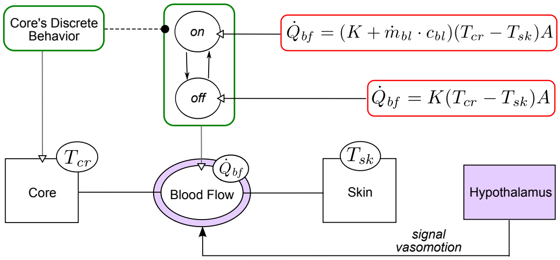

Fig. 5. The dynamics of heat exchange through the blood flow

```
eq flowDyn(< BF : BloodFlow | state : on, conduct : COND, heatCap : HC,
                            bloodFlowRate : BFR, area : A >,
           TEMPCR, TEMPSK) = (COND + HC * BFR) * (TEMPCR - TEMPSK) * A .

eq flowDyn(< BF : BloodFlow | state : off, conduct : COND, area : A >, TEMPCR, TEMPSK)
 = COND * (TEMPCR - TEMPSK) * A .
```

The following rule defines the behavior of the blood flow component when it receives a signal (message) from the hypothalamus containing the value of the desired blood flow rate:

```
rl [vaso] :
   signalVaso(BF, HYPOTHAL, BFR)
   < BF : BloodFlow | entity1 : CORE >
   < CORE : CoreHumanBody | coreState : CRST, bWaterState : BWST,
                            bloodFlowRateFactor : BFRF >
 =>
   < BF : BloodFlow | bloodFlowRate : if CRST =/= dead and BWST =/= death
                                      then BFRF * BFR else 0.0 fi >
   < CORE : CoreHumanBody | >   fromActuator(HYPOTHAL, BF) .
```

4.3 The Controllers

To model the regulatory process in the human body, we have defined a control system with *controllers*, *sensors*, and *actuators*. A sensor is connected to a component in a physical system to monitor the value of some variable and to periodically send the value to one or more controllers. A controller receives information from one or more sensors, and performs the controlling actions by sending messages/signals to one or more actuators. We again refer to the longer report [6] for details about this control system infrastructure and the cortex.

The *hypothalamus* component is defined by extending the controller class with attributes for temperature set points for the core and the skin. The hypothalamus component is triggered when it receives the core and skin temperature values. If the hypothalamus is not inactive, it uses these values to compute the appropriate messages/signals to send to the sweating, shivering, and blood flow components, and to send the core temperature value to the cortex.

```
class Hypothalamus | setPointCore : Float, setPointSkin : Float .
subclass Hypothalamus < Controller .

rl [hypo-manages-involuntary-thermoreg] :
   tempValCore(HYPOTHAL, RECEPTCR, TEMPCR)
   tempValSkin(HYPOTHAL, RECEPTSK, TEMPSK)
   < HYPOTHAL : Hypothalamus | status : waiting, setPointCore : SPCR,
                               setPointSkin : SPSK >
   < SWEAT : EvapSkinSweating | controller : HYPOTHAL >
   < SHIVER : Shivering | controller : HYPOTHAL >
   < BF : BloodFlow | controller : HYPOTHAL >
   < CORTEX : Cortex | dataProvider : HYPOTHAL >
   =>
   < HYPOTHAL : Hypothalamus | status : running, sensors : RECEPTCR ; RECEPTSK,
                               actuators : BF ; SWEAT ; SHIVER >
   < SWEAT : EvapSkinSweating | >      < SHIVER : Shivering | >
   < BF : BloodFlow | >                < CORTEX : Cortex | >
   signalVaso(BF, HYPOTHAL, bloodFlowRate(TEMPCR, TEMPSK, SPCR, SPSK))
   makeSignalSweating(SWEAT, HYPOTHAL, TEMPCR, TEMPSK, SPCR, SPSK)
   makeSignalShivering(SHIVER, HYPOTHAL, TEMPCR, TEMPSK, SPCR, SPSK)
   tempValCoreFromHypothalamus(CORTEX, HYPOTHAL, TEMPCR) .
```

Some of the "messages" above are functions which generate the appropriate message. For example, when the body temperature is considered too hot, a signal containing the `on` value and the sweat rate value is sent to the sweating component. If the body temperature is considered fine, only the `off` value is sent. The treatment is similar for messages to the shivering component:

```
op makeSignalSweating : Oid Oid Float Float Float Float -> Msg .
op makeSignalShivering : Oid Oid Float Float Float Float -> Msg .

msg signalVaso : Oid Oid Float -> Msg .
msgs signalSweating signalShivering : Oid Oid OnOffStatusType Float -> Msg .
msg hypothalamusInactive : Oid Oid -> Msg .

ceq makeSignalSweating(SWEAT, HYPOTHAL, TEMPCR, TEMPSK, SPCR, SPSK) =
    signalSweating(SWEAT, HYPOTHAL, if SWTR =/= 0.0 then on else off fi, SWTR)
    if SWTR := sweatRate(TEMPCR, TEMPSK, SPCR, SPSK) .

ceq makeSignalShivering(SHIVER, HYPOTHAL, TEMPCR, TEMPSK, SPCR, SPSK) =
    signalShivering(SHIVER, HYPOTHAL, if SHVR =/= 0.0 then on else off fi, SHVR)
    if SHVR := shiverRate(TEMPCR, TEMPSK, SPCR, SPSK) .
```

The function `bloodFlowRate`, governing vasodilation and vasoconstriction, is defined according to the equation for the blood flow rate for two-node models as $\dot{m}_{bl} = \frac{1}{3600}[6.3 + \frac{200 \cdot WSIG_{cr}}{1+0.5 \cdot CSIG_{sk}}]$, where $WSIG_{cr}$ and $CSIG_{sk}$ are the effector controlling signals for, respectively, vasodilation and vasoconstriction [14].

The equation for the shivering heat production rate for the two-node model is $\dot{Q}''_{sh} = 19.4 \cdot CSIG_{sk} \cdot CSIG_{cr}$ (in W/m^2; we multiply by 10^{-3} since we use kiloWatts):

```
op shiverRate : Float Float Float Float -> Float .
eq shiverRate(TEMPCR, TEMPSK, SPCR, SPSK) =
   19.4 * cSigSK(TEMPSK, SPSK) * cSigCR(TEMPCR, SPCR) * 0.001 .
```

Fig. 6. The Sauna World Championships in Heinola, Finland

The function computing the sweat rate is based on [14] where it is defined as $\dot{m}_{sw} = 4.7 \cdot 10^{-5} \cdot WSIG_{body} \cdot exp(\frac{WSIG_{sk}}{10.7})$:

```
op sweatRate : Float Float Float Float -> Float .
eq sweatRate(TEMPCR, TEMPSK, SPCR, SPSK) =
   4.7 * 0.00001 * wSigBody(TEMPCR, TEMPSK, SPCR, SPSK) * exp(wSigSK(TEMPSK, SPSK)).
```

5 Extreme Exposure: The Sauna World Championships

The Sauna World Championships were an annual event held in Heinola, Finland. The winner is the contestant who can stay the longest in an oppressively hot sauna. Before the torturing game starts, the sauna is pre-heated to $110°C$ (warmer than the boiling point for blood). To make the conditions even worse, every thirty seconds half a liter of water is poured onto the hot sauna rocks which are the heat source of the sauna. The intense vapor from this water increases the humidity of the sauna, which makes it more difficult for the participants' sweat to evaporate. The world record of 18 minutes and 15 seconds was set in the 2008 championships. The championships in 2010 ended in a tragedy. The two last finalists collapsed with severe burn injuries after about six minutes. One of them died a day later, and the other, a five-time champion, survived after two months in coma, with serious damage to his skin, lungs and kidneys. They were both conscious but were unable to get out of the sauna on their own. The cause of this tragedy is still under investigation. The temperature in the sauna was similar to those in previous years and the times of the competitors were about the same, until this terrible final round.

This section shows how we can use HI-Maude and our model to formally analyze the ability of an unhealthy, a normal, and a trained human body to survive extreme conditions similar to those in the sauna world championships. We also

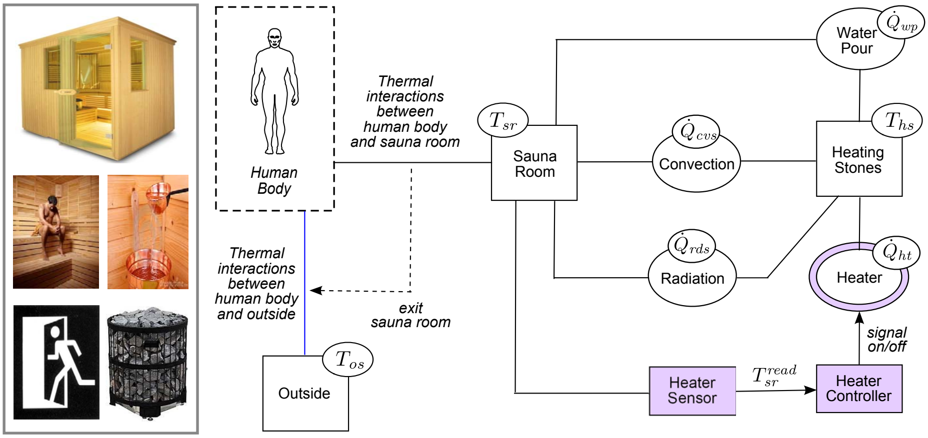

Fig. 7. The sauna room environment

try to find out what may have happened that fateful day in 2010. In particular, Section 5.1 gives an overview of our model of the thermodynamics of the sauna. We do not have any official information about the sauna environment used in the world championships, but after researching literature, product descriptions, etc., we have obtained information that we use to define the parameter values (see [6]). Section 5.2 then presents some snippets of our HI-Maude analyses.

5.1 Modeling the Sauna

The sauna is modeled as a room which uses special rocks to provide heat to the room, as shown in Fig. 7. The rocks are heated by a heater, which is connected to a control system that manages the temperature of the room. Some amount of water is periodically poured on the rocks.

Thermal Entities. The thermal entity for the sauna room is modeled in a different way than the other thermal entity components. For example, there are two attributes for the mass: for the dry air and the water vapor, since we have to model the effect of pouring water on the rocks.

```
class SaunaRoom | massDryAir : Float, spHeatAir : Float, massWaterVap : Float,
                  spHeatWater : Float, relHumid : Float, timer : Int, period : Int,
                  vol : Float, gcDryAir : Float, gcWaterVap : Float .
subclass SaunaRoom < PhysicalEntity .
```

The attributes `massDryAir` and `massWaterVap` specify the mass of dry air and of water vapor, respectively; `spHeatAir` and `spHeatWater` specify the specific heat of air and water, respectively; `relHumid` keeps the value of relative humidity of the room; `timer` is used for triggering periodical events; `vol` denotes the volume of the sauna room; and gcDryAir and gcWaterVap are the gas constants of dry air and water vapor, respectively.

The continuous dynamics of the effort variable of the sauna is defined by

```
ceq effortDyn(< ROOM : SaunaRoom | massDryAir : MASSDA, spHeatAir : SHDA,
                                   massWaterVap : MASSWV, spHeatWater : SHWV >, SF)
  = SF / ((MASSDA + MASSWV) * specHeatHumid(SHDA, SHWV, HUMSPEC))
    if HUMSPEC := humidRatio2(MASSDA, MASSWV) .
```

This equation is the basic equation for thermal entities, taking into account that the mass of the room is a combination of the mass of the dry air and of the water vapor. The function specHeatHumid computes the specific heat of a mix of dry air and water vapor, and humidRatio computes the humidity ratio of the mix.

The *sauna rocks* component is defined as a simple thermal entity:

```
class SaunaRocks .      subclass SaunaRocks < ThermalEntity .
```

Heat transfer from the rocks occurs through convection and radiation. We use the basic forms of these heat transfers:

```
class ConvectionBasic | convectCoeff : Float .
class RadiationBasic | emmissiv : Float .
subclass ConvectionBasic RadiationBasic < ThermalInteraction .

eq flowDyn(< CONV : ConvectionBasic | convectCoeff : COEFF, area : A >, TEMP1, TEMP2)
 = COEFF * (TEMP1 - TEMP2) * A .

eq flowDyn(< RAD : RadiationBasic | emmissiv : EMMI, area : A >, TEMP1, TEMP2)
 = EMMI * stefBoltzConst * A * ((TEMP1 ^ 4.0) - (TEMP2 ^ 4.0)) .
```

Pouring Water. We make some simplifying assumptions when considering the thermal effect of pouring water on the rocks: the effect of pouring water is a heat loss for the heating rocks and a heat gain for the sauna; all the water is vaporized; and the vaporization is instantaneous. The *water pouring* component is defined as a thermal interaction between the rocks and the sauna room:

```
class WaterPouringHeatLoss | mass : Float, temp : Float, heatCap : Float,
                             heatEvap : Float, timer : Int, period : Int .
subclass WaterPouringHeatLoss < PhysicalInteraction .
```

The attribute mass specifies the amount of water poured; temp defines the temperature of the poured water; heatCap and heatEvap represent the heat capacity and the latent heat evaporation of the water, respectively; and timer and the period are used to trigger the water pouring event periodically.

The flow variable of the water pouring component represents the rate of heat flow from the rocks to the sauna room. We separate between the case when the water is poured, and when nothing happens between two water pouring events:

```
eq flowDyn(< WPHL : WaterPouringHeatLoss | mass : MASS, temp : TEMP, heatCap : HC,
                                            heatEvap : HEVAP, timer : TMR >,
            EFF1, EFF2)
 = if TMR =/= 0 then 0.0
   else MASS * HC * (100.0 - TEMP) + MASS * HEVAP * factorPourWater(EFF1, TEMP) fi .
```

When the water is poured, assuming that the water is vaporized at once, the heat transfered from the rocks to the room is the sum of the heat needed to increase the water to the boiling point and the heat needed to change the water from the liquid to vapor.

The water is poured periodically. When the associated timer expires (i.e., becomes 0), water is poured on the rocks, and the vapor content of the sauna room increases by the amount of water poured (half a litre):

```
rl [add-watervap-to-sauna] :
   < ROOM : SaunaRoom | timer : 0, massWaterVap : MASSWV, period : PER >
  =>
   < ROOM : SaunaRoom | timer : PER, massWaterVap : MASSWV + 0.5 > .
```

The function `timeCanAdvance` must be used to ensure that the rule is applied at the moment when the timer expires:

```
eq timeCanAdvance(< ROOM : SaunaRoom | timer : TMR >) = TMR > 0 .
```

The other effect of pouring water on the heating rocks is an increase in the relative humidity of the sauna room. The function `computeAfterEF` is used to update the value of relative humidity, and also the timer (which should be synchronized with the timer of the pouring water component):

```
ceq computeAfterEF(< ROOM : SaunaRoom | timer : TMR, effort : TEMPAM, vol : VOL,
                                        massWaterVap : MASSWV, gcWaterVap : GCWV,
                                        massDryAir : MASSDA, gcDryAir : GCDA > REST)
  = < ROOM : SaunaRoom | timer : TMR - 1,
                         relHumid : relativeHumidity(PRESTOT, PRESSAT, HUMSPEC) >
    computeAfterEF(REST)
    if TMR > 0 /\ PRESWV := gasPressure(MASSWV, GCWV, TEMPAM, VOL)
               /\ PRESDA := gasPressure(MASSDA, GCDA, TEMPAM, VOL)
               /\ PRESTOT := PRESWV + PRESDA
               /\ PRESSAT := waterVaporPressure(TEMPAM)
               /\ HUMSPEC := humidRatio(MASSDA, MASSWV) .
```

The Sauna Heating System. The heating system considered here is an automatic control system which manages to keep the temperature of the sauna room at a specified value. The room temperature read by a sensor determines the activation and deactivation of the heater. The specification is fairly standard (in the context of this paper!) and is not further explained.

5.2 HI-Maude Analysis

We are now ready to use HI-Maude to analyze how long people in different states of fitness for the event can endure in different kinds of saunas, as well as to analyze some of our own hypotheses for what may have happened at the 2010 championships. As described in [6], we model the persons according to known medical facts/assumptions and models. We use physical parameter values for the sauna heating system based on real values found in product descriptions of

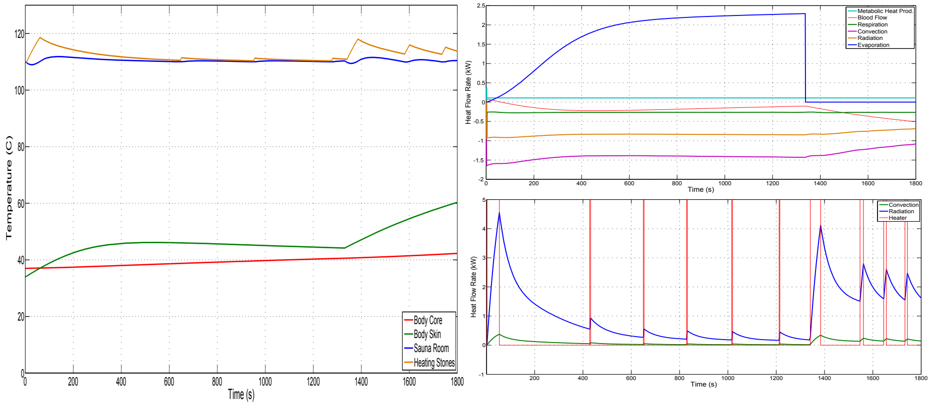

Fig. 8. Simulation results for the dry sauna: temperatures of the human body and the sauna environment thermal entities (left), and heat flow rates of the human body (top right) and of the sauna (bottom right) thermal interactions

commercial sauna heating systems (see [6]). The experiments were performed on a computer with an Intel Pentium 4 CPU 3.00 GHz and 3 GB of RAM. All analyses are carried out using Euler's numerical method with fixed time increment size one.

We model three kinds of saunas:

– a dry sauna, where no water is poured on the heating rocks;
– a moderate wet sauna, where half a liter of water is poured every 5 minutes;
– an extreme wet sauna, where half a liter of water is poured every 30 seconds.

To analyze what happens to our virtual experimental subject after 30 minutes in the sauna, we can use the HI-Maude simulation command

```
(hrew cs1 in time <= 1800 using euler stepsize 1.0 discreteswitch nonaccurate .)
```

where `cs1` is an initial state consisting of all appropriate physical entity objects and physical interaction objects with their respective initial values (see again [6]). Figure 8 shows the simulation results for the dry sauna for the average person up to 30 minutes. In the beginning the skin can handle the heat from the sauna room well, while the core temperature increases slowly. However at some point between minute 20 and 23, the skin temperature increases drastically. As we see in the graph on the right, at that point the sweating stops, possibly due to severe hyperthermia or second degree skin burn. At the same point, the heat flow from the blood vessels transfers heat from the skin to the core at an increasing rate. We next use the hybrid find earliest command to find out when a person encounters severe hyperthermia:

```
(hfind earliest cs1 =>*
   {REST:Configuration   < personCore : CoreHumanBody | coreState : CRST >}
      such that (CRST == sevHyperthermia)
         using euler stepsize 1.0 discreteswitch nonaccurate .)
```

The following table shows the results of the analysis command above for different sauna environments and different persons, for severe hyperthermia, severe dehydration, and second degree skin burn:

Person	Body condition	Sauna		
		dry	moderate wet	extreme wet
Normal	sev.hyperthermia	1177 s CPU: 497756ms	1104 s CPU: 472856ms	719 s CPU: 316259ms
	sev.dehydration	no result CPU: –	no result CPU: –	no result CPU: –
	skin 2nd burn	1366 CPU: 651181ms	1258 s CPU: 606713ms	770 s CPU: 394679ms
Trained	sev.hyperthermia	2436 s CPU: 1404685ms	2160 s CPU: 1221302ms	1266 s CPU: 712206ms
	sev.dehydration	no result CPU: –	no result CPU: –	no result CPU: –
	skin 2nd burn	2792 s CPU: 1753605ms	2457 s CPU: 1790829ms	1392 s CPU: 821230ms
Unhealthy	sev.hyperthermia	780 s CPU: 330182ms	728 s CPU: 316502ms	466 s CPU: 298229ms
	sev.dehydration	no result CPU: –	no result CPU: –	no result CPU: –
	skin 2nd burn	784 s CPU: 335146ms	597 s CPU: 250891ms	340 s CPU: 159359ms

Our analyses show that even the average person should endure 12 minutes in the wet sauna before the onset of major injuries (and that all persons die from hyperthermia before becoming severely dehydrated). We next propose and analyze some possible explanations for the still unsolved tragedy that could cause major injuries to a five-time world champion in around 6 minutes:

– The initial sauna room temperature is, as expected, $110°C$. But the temperature of the heating rocks is $250°C$. We understand from the manual of the heating system of the product used in Heinola that the temperature sensor only monitors the air temperature of sauna room and not the heating rocks.
– The temperature sensor is wrong and the temperature is higher. It turns out that even at $150°C$, they should be fine for more than 10 minutes. A temperature of around $210°C$ is needed to explain the outcome.
– The humidity of the sauna is extremely high from the start. We start with 39 liters of water vapor instead of the expected 10 liters.

The results of our HI-Maude analyses of these hypothesis are:

Person	Body condition	Possible problems		
		Heating rocks $250°C$	Room air $210°C$	Very high humidity
Trained	sev.hyperthermia	349 s	328 s	785 s
	sev.dehydration	no result	no result	no result
	skin 2nd burn	414 s	393 s	311 s

Additional analyses are given in [6], including an analysis of whether a person with high tolerance level will be badly burnt or dehydrated before thinking about exiting the sauna.

6 Concluding Remarks

We have presented a detailed and realistic formal model, with all assumptions based on scientific/medical knowledge, of the human thermoregulatory system and its interactions with different kinds of environments, including oppressive saunas, and have simulated and further formally analyzed the reaction of different kinds of people to various saunas. The results of our analyses are decently close to what we know about how long both professionals and amateurs (sports writers, rock stars, etc.) can endure in the sauna.

We have also used HI-Maude to analyze some possible explanations for the still unresolved tragedy at the 2010 Sauna World Championships.

Acknowledgments. We gratefully acknowledge financial support for this work by The Research Council of Norway through the Rhytm project and by The Research Council of Norway and the German Academic Exchange Service through the DAAD ppp project HySmart.

References

1. ASHRAE Handbook: Fundamentals; SI Edition. American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. (2005)
2. Bloomfield, L.A.: How things work: the physics of everyday life. Wiley, Hoboken (2006)
3. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.: All About Maude - A High-Performance Logical Framework. LNCS, vol. 4350. Springer, Heidelberg (2007)
4. The Engineering ToolBox, `http://www.engineeringtoolbox.com/`
5. Fadlisyah, M., Ábrahám, E., Lepri, D., Ölveczky, P.C.: A rewriting-logic-based technique for modeling thermal systems. In: Proc. RTRTS 2010. Electronic Proceedings in Theoretical Computer Science, vol. 36 (2010)
6. Fadlisyah, M., Ölveczky, P.C., Ábrahám, E.: Formal modeling and analysis of extreme heat exposure to the human body in HI-Maude: A case study inspired by the tragedy at the 2010 Sauna World Championships. Tech. rep., Dept. of Informatics, Univ. of Oslo (2011), `http://heim.ifi.uio.no/mohamf/HI-Maude/techreport-casestudy.pdf`
7. Fadlisyah, M., Ölveczky, P.C., Ábrahám, E.: Formal modeling and analysis of hybrid systems in rewriting logic using higher order numerical methods and discrete-event detection. In: Proc. CSSE 2011. IEEE (2011)
8. Fadlisyah, M., Ölveczky, P.C., Ábrahám, E.: Object-Oriented Formal Modeling and Analysis of Interacting Hybrid Systems in HI-Maude. In: Barthe, G., Pardo, A., Schneider, G. (eds.) SEFM 2011. LNCS, vol. 7041, pp. 415–430. Springer, Heidelberg (2011)

9. Gagge, A., Stolwijk, J., Nishi, Y.: An effective temperature scale based on a simple model of human physiological regulatory response. ASHRAE Trans. 77(1), 247–262 (1971)
10. Hensel, H.: Thermoreception and temperature regulation. Monogr. Physiol. Soc. 38 (1981)
11. Hwang, C.L., Konz, S.A.: Engineering models of the human thermoregulatory system-a review. IEEE Transactions on Biomedical Engineering BME-24(4), 309–325 (1977)
12. McPherson, M.: Subsurface Ventilation and Environmental Engineering. Chapman & Hall (1993)
13. Ölveczky, P.C., Meseguer, J.: The Real-Time Maude Tool. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 332–336. Springer, Heidelberg (2008)
14. Parsons, K.: Human Thermal Environments: The effects of hot, moderate, and cold environments on human health, comfort and performance. Taylor & Francis, London (2003)
15. Plantadosi, C.: The Biology of Human Survival: Life and Death in Extreme Environments. Oxford University Press (2003)
16. Tipton, C.: ACSM's Advanced Exercise Physiology. Lippincott Williams & Wilkins (2006)
17. Wellstead, P.E.: Introduction to physical system modelling. Academic Press (1979)
18. Winkeljohn, M.: Heat injuries forcing new technologies (2010), `http://sports.espn.go.com/ncaa/recruiting/football/news/story?id=4372652` (accessed February 2, 2012)