System-Level Analysis and Design of Safety-Critical Cyber Physical Systems

Abdel-Latif Alshalalfah

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Electrical and Computer Engineering) at

Concordia University

Montréal, Québec, Canada

February 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Abdel-Latif Alshalalfah**

Entitled: **System-Level Analysis and Design of Safety-Critical**

                  Cyber Physical Systems

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Electrical and Computer Engineering)

complies with the regulations of this University and meets the accepted standards with respect to

originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Ahmed Soliman

_____ External Examiner
Dr. Malek Mouhoub

_____ Examiner
Dr. Mohammad Mannan

_____ Examiner
Dr. Wahab Hamou-Lhadj

_____ Examiner
Dr. Hassan Rivaz

_____ Supervisor
Dr. Otmane Ait Mohamed

Approved by   _____
                Yousef R. Shayan, Chair
                Department of Electrical and Computer Engineering

February 2, 2023   _____
                          Mourad Debbabi, Dean
                          Faculty of Engineering and Computer Science

Abstract

System-Level Analysis and Design of Safety-Critical
Cyber Physical Systems

Abdel-Latif Alshalalfah, Ph.D.

Concordia University, 2023

The reduction in size and cost of hardware together with the accelerating innovation and advancement in sensor and computational technologies have opened the door for cyber physical systems into all types of applications. While most early systems involved varying degrees of human involvement, the various success stories are encouraging designers to develop cyber physical systems for autonomous control.

The trustworthiness of a cyber-physical system is essential for it to be qualified for utilization in most real-life deployments. This is especially critical for systems that deal with precious human lives. which can be engaged directly as in biomedical systems or indirectly as in automotive systems. Although use-cases for biomedical and automotive systems are considered, the proposed generalized framework can be used to analyze the safety of various cyber-physical systems.

These safety-critical systems can be investigated using both experimental testing and model-based verification. Accurate models have the potential to permit investigating the system behavior under abnormal scenarios. Also, appropriate modeling can speed-up the development process by evaluating candidate designs at an early stage of the design cycle.

Model-based verification can be conducted using the less-exhaustive simulation testing or the resources-greedy model checking. As a trade-off, statistical model checking bears a feasible approach where statistical guarantees can be examined with a specific level of confidence. This research addresses the problem of utilizing accurate system-level models to analyze and design safety-critical cyber-physical systems.

The behavioral descriptions of cyber physical systems are modelled by constructing equivalent formal models. These system-level models are used to conduct statistical model checking to verify properties written using metric interval temporal logic and to provide statistical guarantees on the system safety. This approach is applied on biomedical and automotive systems to verify their safety with consideration for some distortions resulting from unintentional or intentional sources. The proposed verification approach enlightens the development process by providing feedback that can help elect the designs. Moreover, new robust and safe control techniques are proposed to enhance the safety of a closed-loop glucose controller system. Also, a systematic approach is proposed for safety analysis of cyber physical systems. This approach processes systems described using SysML diagrams and applies a new proposed automatic algorithm to construct equivalent formal models. This research work is a step towards bridging the gap between system-level models and formal models so that analysis can be conducted efficiently to enhance the safety and robustness of cyber-physical systems.

Dedication

I dedicate this work to my beloved parents, wife, kids, brothers and sisters.

Acknowledgments

First and foremost, I praise my God and thank him for giving me the energy to complete this thesis.

Also, I would like to thank my supervisor, Dr. Otmane Ait Mohamed. He was always available to offer guidance, support, and understanding. I would like to thank the people who supported me throughout the PhD journey. I would like to thank my family for giving me the physical and emotional assistance during the tough times. I would like to thank alumni members of HVG who gave me guidance and advice in my research work, especially: Dr. Ghaith Bany Hamad, Dr. Samir Ouchani, Dr. Ayman Atallah, and other members who always offered to help. Last but not least, I would like to thank all the members of Hardware Verification Group for making a healthy, warm, and exciting lab environment.

Contributions of Authors

Article I: *Towards Safe and Robust Closed-Loop Artificial Pancreas Using Improved PID-Based Control Strategies*

- **Abdel-Latif Alshalalfah**: Conceptualization, methodology, literature review, modeling, extensive experimentation, data interpretation, validation, writing, review and editing.

- **Ghaith Bany Hamad (coauthor)**: Conceptualization, methodology, validation, review and editing.

- **Otmane Ait Mohamed (supervisor)**: Conceptualization, methodology, validation, review and editing.

Article II: *A Framework for Modeling and Analyzing Cyber-Physical Systems using Statistical Model Checking*

- **Abdel-Latif Alshalalfah**: Conceptualization, methodology, literature review, modeling, implementation, results generation and interpretation, correctness proof, validation, writing, review and editing.

- **Otmane Ait Mohamed (supervisor)**: Conceptualization, methodology, correctness proof, review and editing.

- **Samir Ouchani (coauthor)**: conceptualization, methodology, correctness proof, review and editing.

Acronyms

AP Artificial Pancreas

ASIL Automotive Safety Integrity Levels

AWPID Adaptive Weighted Proportional Integral Derivative

BIS Bispectral Index

CEB Coordinated Emergency Braking

CEBP Coordinated Emergency Braking Protocol

CGM Continuous Glucose Monitor

CPS Cyber-Physical Systems

CVGA Control-Variability Grid Analysis

EAC Enhanced Activity Calculus

ECG ElectroCardioGram

EEG ElectroEncephaloGram

FDA Food and Drug Administration

FL Fuzzy Logic

HA Hybrid Automata

HOL Higher Order Logic

IAT	Inter-Arrival Time
IIR	Insulin Infusion Rate
ISO	International Organization for Standardization
LAPID	Look-Ahead Proporional Integral Derivative
LAPID-REC	Look-Ahead PID with Retrospective estimation Error Correction
LBM	Lean Body Mass
MB	Multi Basal
MITL	Metric Interval Temporal Logic
MPC	Model Predictive Control
ODE	Ordinary Differential Equations
ODESCD	ODE SysML Constraint Diagram
PD	Proportional Derivative
PDR	Packet Delivery Ratio
PER	Packet Error Rate
PID	Proporional Integral Derivative
PK-PD	PharmacoKinetic-PharmacoDynamic
PTA	Priced Timed Automata
QC-SOSMC	Quasi-Continuous Second-Order Sliding-Mode Controller
SEooC	Safety Element out of Context
SMC	Statistical Model Checking
SMT	Satisfiability Modulo Theory

SysML   System Modeling Language

TA   Timed Automata

TDMA   Time Division Multiple Access

T1D   Type 1 Diabetes

V2V   Vehicle to Vehicle

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Whether human-operated or autonomous, the main goal of developing embedded systems is to enhance the quality of life for people. Evolving from embedded computing and distributed systems, the term Cyber Physical Systems (CPS) has been used since 2006 to describe computing systems that interact with control or management objects (Skorobogatjko, Romanovs, & Kunicina, 2014).

In CPS, embedded cyber subsystems monitor and excite physical processes, while at the same time communicate with central computational subsystems through networking subsystems via feedback loops as shown in Fig. 1.1. These CPS can compose various subsystems of different characteristics. This diversity reflects on the complexity of CPS analysis. Moreover, it reflects on the level of exposure to the different sources of threat.

The accelerating advances in technology, the reduced cost of hardware, and the progress in control techniques have widen the potentials offered by CPS. This has boosted the use of CPS in various applications even the ones involving human lives either directly as in biomedical systems or indirectly as in automotive systems. For such safety-critical systems, the CPS has to be proven to
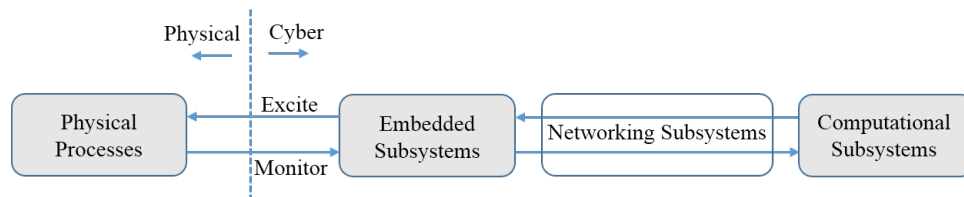
Figure 1.1: Overview of CPS Subsystems

preserve safety under all possible scenarios.

Therefore, any design of a safety-critical system cannot be implemented in real-life deployments unless it has gained proper certifications by specialized governing entities such as Food and Drug Administration (FDA) or International Organization for Standardization (ISO). These entities form special committees of experts to define the requirements to approve a specific type of systems. Hence, an application that is filed to acquire certification from a licensing entity has to provide solid proofs to demonstrate its compliance with all the safety requirements. These proofs should involve both experimental as well as model-based analyses. Still, model-based approach can speed-up the process. For example, in 2008 FDA accepted a diabetes simulator to replace pre-clinical animal testing (Man et al., 2014).

As the risk level increases, the requirements tend towards being more stringent. For example, four Automotive Safety Integrity Levels (ASILs) are defined in ISO 26262 (International Organization for Standardization, 2018) to specify the minimum requirements that can achieve a tolerable risk level. A level is determined for a specific application depending on the severity, controllability, and probability of exposure to hazards. From ASIL A to ASIL D, ASIL D imposes the most stringent safety metrics in terms of failure in time, single point fault metric, and latent fault metric.

Many aspects of CPS safety requirements are not yet well defined and are under continuous discussion among the scientific, social, and governance communities (Lee & Hess, 2020). In particular, when technology providers bring new CPS ideas and designs, the discussion starts among the different players about the feasibility, safety, sustainability, and interoperability. Initial regulations are usually more conservative with only partial automation allowed or additional safety margins (Claybrook & Kildare, 2018; Fagnant & Kockelman, 2015). After that, the deployments are observed with feedback from the field to add amendments that address new emerging safety issues and to exploit new chances to make the most use of technology for the good of humanity (Kokubugata, Kawashima, Fukui, & Kamata, 2022).

The development process of CPS can be boosted by utilizing model-based analysis and design. The essence of this approach is to employ accurate models that describe the system behavior and make experiments to reveal the different aspects of the CPS. Not only does that save time and cost

of analysis, but also permits testing scenarios that might involve hazards if implemented in reality.

Constructing valid models is the main pillar in model-based design and analysis. Not only is the system under development modelled, but the surrounding systems and environments are also modelled. These models are constructed from mathematical models that are either derived from calculus or developed empirically. These behavioral models are converted into the modeling language adopted by the analysis tool in use. Analysis is then conducted to verify the operation of the CPS in real-life scenarios.

One type of model-based analysis is when the micro-behaviors of the components are substituted with abstract behaviors to focus on the system behavior as a whole. This is referred to as system-level analysis where less detailed systems are modeled . System-level analysis provides a utility to verify CPS safety at an early stage of the design cycle. In this approach, specific assumptions of the CPS elements are modeled instead of modeling the hardware components themselves. This allows specifying component requirements, and proposing improvements before having the real manufactured components in hand. Additional to allowing vendor interoperability in CPS components, system-level analysis is essential to cope well with industrial standards such as Safety Element out of Context (SEooC) defined in ISO 26262 (International Organization for Standardization, 2018).

1.1   Motivation and Problem Statement

Regardless of their sources, CPS failure to preserve safety can retard or even prohibit using them for safety-critical systems. These safety violations can be due to intrinsic characteristics of the CPS or due to environmental effects that might result from natural unintentional disturbances or from intentional security threats. Therefore, extensive verification is required to guarantee safety before deployment.

Lab and field experimental testing is essential because it provides the most accurate results versus the estimates from model-based analysis. But experimental testing usually requires prototypes which are only available in advanced phases of the design process. Also, experimentation might imply re-configuring hardware for each scenario which requires more time to cover more scenarios.

Moreover, any detected design problems will incur additional time and costs of development which would make it impractical to exclude system-level analysis.

System-level analysis can be conducted using simulation testing. This is a simple and affordable approach to replicate CPS behavior at an early stage of the design cycle. This approach shows the response of the system for the input test vector, but it provides no guarantees on the coverage of the state space. On the other side formal techniques conduct rigorous analysis that covers the whole state space, but they suffer from high computational requirements that limit their usability to simpler systems.

Another requirement for system-level analysis to succeed is to utilize accurate models. A model is considered accurate if and only if it behaves exactly similar to the modeled CPS component. Model accuracy is essential for a trustworthy CPS verification. Safety properties and specifications that the CPS needs to meet for it to be considered acceptable are also defined and assessed.

Using low-cost hardware components in the design of CPS can help enable affordable solutions for personal applications within budget. For these cheap types of equipment, the hardware components that operate CPS are susceptible to various faults. These faults can arise from intrinsic or extrinsic factors. The CPS to be deployed in a safety-critical scenario needs to mitigate such faults. Therefore, when verifying a safety-critical CPS, it is vital to assess the risks that result from potential faults. This risk assessment can help developers to modify CPS designs so that they can better handle faults.

Safety guarantees are pre-requisites for the deployment of CPS in real-life applications. These guarantees are required to be proven both analytically and in field-proven experiments. The addressed problem in this research is to propose a systematic framework that can efficiently analyze the safety of CPS, guide the design choices, and help develop more safe systems.

1.2   Thesis Contributions

The proposed work in this research aims at allowing more credible CPS deployments in safety-critical applications. This is achieved by proposing a feasible framework for system-level analysis and design of CPS. By excluding the unsafe designs before implementation, the proposed approach

can be used to compare designs and advise changes at an early stage of the design cycle. The main contributions of this thesis are listed as following :

- Proposing an approach to model and analyze a closed-loop artificial pancreas system using statistical model checking. The system components are modeled using priced timed automata in UPPAAL-SMC tool. The tool is used to parallel-compose the components and analyze them in a security scenario that involves a replay attack to verify statistical safety properties that are expressed using metric interval temporal logic. This work is explained in 3.2.1. and published in [Cf1].

- Proposing an approach to model and analyze a closed-loop anesthesia control system under fault using statistical model checking. This work is explained in 3.2.2 and published in [Cf3].

- Proposing an approach to model a coordinated vehicular emergency braking system using priced timed automata. Statistical model checking is applied to verify the system safety under a scenario that involves degraded wireless connectivity. The usability of a message retransmission scheme is also evaluated. This work is explained in 3.3.1 and published in [Cf4].

- Utilizing system-level analysis to propose safety-oriented control approaches for enhancing closed-loop artificial pancreas. The first proposed technique published in [Cf2] upgrades the functionality of PID controller by appending an adaptive weight to help bring early response for meal intake. The other approach published in [Jr1] achieves better improvements in terms of satisfying safety properties by utilizing a look-ahead PID controller. These works are explained in Chapter 4.

- A systematic approach to model and analyze CPS for statistical model checking is proposed. This approach processes CPS models described using SysML diagrams and automatically constructs equivalent priced timed automata models that are then analyzed using UPPAAL-SMC tool. This work is explained in Chapter 5 and submitted to [Jr2].

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- **Chapter 2 - Literature Review:** This chapter presents previous works on verifying hybrid systems and introduces background about the tool used in this research.

- **Chapter 3 - System Level Formal Modeling and Analysis of CPS Components and Dynamics:** This chapter introduces the proposed approach to model CPS using PTA components and to conduct SMC analysis using UPPAAL-SMC tool. The proposed approach is utilized to conduct fault analysis and security analysis on real-life CPS.

- **Chapter 4 - Towards Safe and Robust Closed-Loop Artificial Pancreas Using Improved PID-Based Control Strategies:** This chapter introduces the proposed approaches to enhance the safety of closed-loop glucose controllers by utilising an adaptive weight, or by using a look-ahead estimation with backward error correction.

- **Chapter 5 - A Framework for Modeling and Analyzing Cyber-Physical Systems using Statistical Model Checking:** This chapter introduces the proposed systematic framework for modeling a CPS specified by SysML diagrams using PTA components. An automatic conversion algorithm is proposed and demonstrated on a real CPS.

Chapter 2

Literature Review and Background

With the growing demand for CPS applications, several research works have investigated the verification and safety analysis problems related generally to CPS. Based on the surveyed initiatives, two main categories are identified : *Simulation* based approaches and *Formal verification*.

2.1 Simulation based approaches

Even before the advent of modern computer systems, the term *Simulation* is known as the process of designing a model of a real system to conduct experiments (Shannon, 1975). These experiments aim at understanding the system's behavior or evaluating a strategy associated with the system. Simulation software tools have flourished with the advent and availability of low-cost computational systems.

Liu, Kockelman, Boesch, and Ciari (Liu et al., 2017) have used the open-source toolkit MATSim (W Axhausen, Horni, & Nagel, 2016) to investigate large-scale transportation patterns for shared autonomous vehicles. In their work, agent-based modelling is applied to estimate mode choices between human-driven vehicles, shared autonomous vehicles, and public transit. Following a cost function that takes into account, the out-of-pocket, the trip time, and the waiting time, each driver chooses one of the three options of travel mode. The analysis is done for different fare levels, demographic settings, and shared autonomous vehicles availability to give implications on sustainability.

In (Lakshmanan, Yan, Baek, & Alghodhaifi, 2019), an assessment of the safety of leader-follower configurations for autonomous radar semi-trucks is made based on different environmental conditions. The simulation model is developed with the commercial platforms AmeSim, PreScan, and Matlab-Simulink to study the effect of environmental conditions on safety margins in semi-truck convoy platooning. The autonomy in their simulated vehicles is enabled by adopting sensors for radar, global positioning systems, and short-range inter-vehicle communication.

Instead of fully autonomous vehicles, the work in (Arnaout & Arnaout, 2014) addressed semi-autonomous vehicles implementing adaptive cruise control coexisting with regular vehicles and trucks. The vehicles enter the four-lane highway with a user-predefined arrival rate in the microscopic Java-based F.A.S.T. traffic simulator. Their findings show that a high penetration of semi-autonomous vehicles can increase traffic performance, especially under high traffic conditions.

Connected and autonomous vehicles and their impact on road safety are discussed in (Papadoulis, Quddus, & Imprialou, 2019). Initially, the simulation software VISSIM is utilized to study a test-bed that mimics a three-lane motorway with traffic statistics measured from a real one in England. A lateral and longitudinal control algorithm is then tested for its ability to reduce traffic conflicts at different market penetration rates.

From a healthcare perspective, a falsification approach is presented in (Cameron, Fainekos, Maahs, & Sankaranarayanan, 2015; Sankaranarayanan et al., 2017) to simulate and verify the artificial pancreas controller in a simulation environment. The S-Taliro tool which applies falsification simulations terminates with either finding a safety violation or failure to find one, without the explicit guarantee that such one does not exist. Instead, the tool uses robustness metric to predict the distance between simulation outcomes and safety margins.

2.2   Formal based approaches

Unlike the numerical simulation approaches which mimic the behavior of real systems, formal methods apply analytical reasoning to derive mathematically-proven properties that characterize the system behavior. These characteristics are not always attainable, but when achieved they provide guaranteed outcomes which is an asset that helps verify safety-critical systems.

In (Kekatos, Forets, & Frehse, 2017b), piece-wise affine hybrid automata was used to analyze the wind turbine dynamics in SpaceEx verification platform (Frehse et al., 2011). Even though Kekatos et al. reduced some blocks for better scalability, the resulting model contained around 16 million locations, which would hinder the ability to analyze more elaborate systems. However, classical hybrid automata (HA) tools and methodologies suffer from this limitation (Schupp et al., 2015).

The deductive theorem prover KeY (Beckert, Hähnle, & Schmitt, 2007) combined with the algebraic computer system Mathematica are utilized to implement the hybrid system verification tool KeYmaera (Platzer & Quesel, 2008). In this tool, sequent calculus with axiomizing hybrid systems transition behavior are implemented to conduct symbolic computations of hybrid systems dynamics. Higher Order Logic (HOL) is used in Isabelle to investigate the reachability of continuous systems by using overapproximations with explicit error bound guarantees (Immler, 2015). Although these theorem provers provide a concrete and computationally-feasible approach to verify hybrid systems, they are interactive tools where the user is required to guide the tool towards a proof.

The problem of formally analyzing a swarm of robots is handled by Schupp, Leofante, Behr, Ábrahám, and Taccella (Schupp et al., 2022). The cooperative decentralized robots are modeled as a hybrid system and investigated by *flowpipe* analysis where the sets of reachable states are iteratively over-approximated (Frehse, 2015). Although the work in (Schupp et al., 2022) deals with a simple model of distributed synchronization, it still causes some scalability challenges that are partially encountered by compositional analysis and optimized transition emulation.

Using a combination of simulations and formal analysis, (Pajic et al., 2012) examines patient-controlled analgesia's safety. So, to analyze the resulting CPS, its detailed behavior is modeled in Simulink. Then, to qualify the CPS for model checking, the continuous dynamics are abstracted away from the system model and then replaced by simple timing constraints with the target to be analyzed in UPPAAL model checker (Behrmann, David, & Larsen, 2004). Additionally, UPPAAL is also used in (Jiang, Pajic, Moarref, Alur, & Mangharam, 2012) to verify control algorithms in a dual chamber implantable pacemaker. Meanwhile, a timed automata representation of the heart and the pacemaker are used to specify the ability of the algorithms to avoid unsafe regions of the state space.

Formal approaches such as model checking (Clarke Jr, Grumberg, Kroening, Peled, & Veith, 2018) explore the system's state space exhaustively to verify properties on the reachable states and find bugs. Unfortunately, the reachability problem of hybrid systems is generally undecidable (Alur et al., 1995; Asarin, Maler, & Pnueli, 1995; Henzinger, Kopke, Puri, & Varaiya, 1995; Mysore & Pnueli, 2005; Vladimerou, Prabhakar, Viswanathan, & Dullerud, 2008) except for special categories of decidable hybrid systems such as Timed Automata (TA) (Alur & Dill, 1994), rectangular automata (Henzinger et al., 1995), semi-algebraic STORMED (Vladimerou et al., 2008) and o-minimal (Lafferriere, Pappas, & Sastry, 2000). Also, model checkers suffer from the infamous state space explosion problem which restricts their applicability (Koopman & Wagner, 2016).

2.3   Statistical model checking based approach

SMC consists of observing a number of simulation runs or system executions and using statistical methods to reason about formal properties (Legay et al., 2019). Different tools exist that implement SMC algorithms such as PRISM (Kwiatkowska, Norman, & Parker, 2011), UPPAAL (A. David, Larsen, Legay, Mikučionis, & Poulsen, 2015; A. David, Larsen, Legay, Mikučionis, & Wang, 2011), BIP (Mediouni et al., 2018), and Ymer (H. L. S. Younes, 2004).

After some preliminary works such as the hypothesis testing of modal properties in process algebra (Larsen & Skou, 1991), initial results for SMC had witnessed progress since 2002 (H. L. Younes & Simmons, 2002) with the corresponding term introduced for the first time in 2004 (Sen, Viswanathan, & Agha, 2004). Reasoning about reachability problems with SMC algorithms provides mainly guarantees on the probability error bound. Depending on the type of reachability expression being dealt with, the error bound can be calculated by utilizing the appropriate classical mathematics such as Monte Carlo with Chernoff-Hoeffding error bounds (Hérault, Lassaigne, Magniette, & Peyronnet, 2004; Okamoto, 1959) or hypothesis testing using Wald's sequential analysis (Wald, 2004).

Different tools exist that implement SMC algorithms such as PRISM (Kwiatkowska et al., 2011), UPPAAL-SMC (A. David et al., 2015, 2011), BIP (Mediouni et al., 2018), and Ymer (H. L. S. Younes, 2004). Since their inception, SMC tools have been utilized to study many discrete-time and continuous-time systems. To list a few: airplane cabin communication system

(Basu, Bensalem, Bozga, Delahaye, & Legay, 2012), distributed sensor network (Lekidis, Bourgos, Djoko-Djoko, Bozga, & Bensalem, 2015), energy-aware house heating (A. David, Du, Larsen, Mikučionis, & Skou, 2012), biological mechanisms of the genetic oscillator (A. David, Larsen, et al., 2012), and real-time streaming protocol (Ouchani, Jarraya, Mohamed, & Debbabi, 2012).

2.4   Model Construction

In order to analyze the system, it is necessary to first convert the specifications into the modeling language used by the analysis tool. Furthermore, an adequate level of expertise is required to model the system properly when done manually. Furthermore, formal modeling languages tend to be more error-prone due to their low readability. Therefore, the need arises to facilitate the process of constructing formal models by automatically translating high-level models that incur better readability.

In (Kekatos et al., 2017b), the system modeled in Simulink is translated into SpaceEx modeling language in four steps. After the Simulink model is modified to comply with the verification standards, the tool SL2SX (Minopoli & Frehse, 2016) is employed to handle the main translation step and construct a SpaceEx model. Afterwards, compositional syntactic hybridization (Kekatos, Forets, & Frehse, 2017a) and validation are conducted to achieve a model ready to be analyzed.

An approach to transform Simulink models into UPPAAL-SMC is proposed in (Filipovikj et al., 2016). The work is employed on two automotive use cases for brake-by-wire and an adjustable speed limiter. The Simulink models are first reduced by the flattening procedure. Then, each block is replaced by an equivalent timed automaton composed of three locations: start, offset, and operate. Still, their approach does not implement complex real-valued blocks in UPPAAL-SMC but addresses them in Simulink instead.

Instead of commercial modeling tools, System Modeling Language (SysML) (Specification, 2007) can be used to specify CPS. SysML is the defacto standard modeling language for systems engineering with rich semantics and expressive power sufficient to describe system structures and behaviors at various levels of abstraction (Holt & Perry, 2008). Ouchani, Mohamed, and Debbabi

([Ouchani et al., 2014](#)) constructed probabilistic automata by converting SysML models. The resulting models were incurred to analyze security properties of the real-time streaming protocol using the probabilistic model checker PRISM ([Kwiatkowska et al., 2011](#)).

Compared to the studied initiatives, the main objective of this thesis is to propose an effective approach for analyzing cyber-physical systems. Statistical model checking is adopted to avoid the feasibility limitations of classical formal methods while still providing statistical guarantees about the state space coverage. While existing works on SMC deal with partial models of the system, the objective is to analyze the CPS as a whole comprising continuous-time dynamics, discrete-time dynamics, and abnormal behaviors. Additionally, a generalized framework is proposed to process CPS specified by SysML diagrams. A new systematic procedure is developed to construct models for the analysis tool.

2.5 UPPAAL-SMC

UPPAAL is a toolbox jointly developed by Uppsala University and Aalborg University for "verification of real-time systems represented by (a network of) timed automata extended with integer variables, structured data types, and channel synchronization" ([A. David et al., 2015](#)). UPPAAL-SMC adopts statistical model checking as an alternative to support more expressive modeling power and to avoid exhaustive exploration of the model state space. In UPPAAL-SMC, non-deterministic choices of transitions and time delays are replaced by probabilistic choices and probabilistic distributions, respectively. Also, the modeling formalism is extended with support for arbitrary clock rates (prices) mixed with double precision floating point type to subsidize arithmetic expressions.

UPPAAL-SMC queries are written using Metric Interval Temporal Logic (MITL). The tool runs the system model under consideration and encodes each run as a Bernoulli variable that takes its value from the result of the corresponding atomic proposition. It processes the Bernoulli trials using one of two statistical approaches:

- For quantitative analysis to estimate the probability of a property, it will utilize the classical Monte Carlo simulation. The implemented algorithm ([Hérault et al., 2004](#)) computes the number of runs needed to produce an approximation interval $[p - \epsilon, p + \epsilon]$ for the property

with confidence $1 - \alpha$. Binomial analysis (Clopper & Pearson, 1934) is conducted as the analysis goes on until the required confidence interval is achieved.

- For qualitative analysis about the property probability of satisfaction or to compare probabilities, hypothesis testing approach is utilized. Where the null hypothesis H_0 resembles when the probability of the atomic proposition satisfaction is above $\theta + \delta_0$ and the alternative hypothesis H_1 resembles when the probability is below $\theta - \delta_1$. In this case, the region of probabilities in between defines an indifference region. The strength parameters α,β are the error probabilities for false positives and false negatives, respectively. This problem is solved using Wald's sequential analysis (Wald, 2004) to give statistical guarantees on the property under consideration.

Chapter 3

System Level Formal Modeling and Analysis of CPS Systems

In this chapter, the approach to construct models of CPS using UPPAAL-SMC modeling language is proposed. The use of this approach to verify CPS safety and security is demonstrated through examples on medical and automotive CPS. The flow diagram shown in Fig. 3.1 summarizes the proposed methodology. The approach starts by identifying the CPS functionality and safety properties. The structure of the system is defined and behavioral models of the different components are imported from the literature. These behavioral models are used to construct system-level PTA components. The PTA blocks are parallel composed and verified for safety using SMC. This analysis helps evaluate the safety risks and guide the development of more safe designs. In this chapter, more elaboration will be presented on this proposed approach.

3.1 Modeling System Components

3.1.1 Modeling Physical (Continuous-Time) Dynamics

In this modeling step, credible models from the literature are borrowed to describe the system dynamics of physical processes. Formal models are proposed to describe the system level behaviors. The dynamics of the physical processes are described using systems of Ordinary differential

14
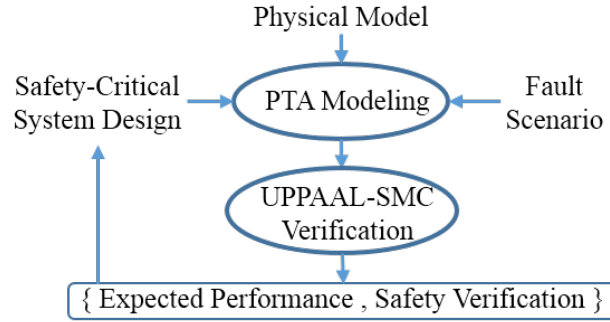
Figure 3.1: Methodology of the Proposed Approach

equations. These equations characterize dynamics of the physical quantities in real-life. These equations can be found using exact derivations from state of the art science as in motion laws, approximated empirically from datasets as in biomedical systems, or estimated from manufacturers' experimentation.

To formally model a physical process, a PTA is constructed with the corresponding ODEs embedded in the *Invariants* field of the PTA's main location. Each physical quantity in the ODE is defined in the PTA as a clock variable. An *Invariant* is added so that the evolution rate of the clock variable is specified by the mathematical expression of the differential of the corresponding physical quantity. For example, the PTA model of the glucose and insulin processes is shown in Fig. 3.2. This PTA models the physiological dynamics for the glucose and insulin throughout the body compartments as described by the system of ODEs in Appendix B. This PTA starts at the initial location *loc0* which is an urgent location [1]. The PTA proceeds to the main location $loc1$ after calling the function $init1()$ which initializes the parameters of the PTA. As the time progress, the $Invariants$ constraint the clock evolution rates of the physical quantities in consistence with the corresponding ODEs. The inputs and outputs of a PTA are defined as variables that are shared with other PTAs. For example, when placing the PTA shown in Fig. 3.2 in a closed-loop glucose control system the glucose concentrations $\{G, G_s\}$ and the exogenous insulin injection IIR are specified as the output and input, respectively.

---

[1] a transient location where no time progress occurs

15

loc0

init1()

loc1
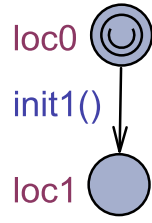
I_sc1'==-(kd+k_a1)*I_sc1+IIR &&
I_sc2'==kd*I_sc1-k_a2*I_sc2 &&
X1'==-p_2u*X1+p_2u/VI*I_p-p_2u*Ib &&
Gs'==-1/Ts*Gs+1/Ts*G &&
I_1'==-ki*I_1+ki/VI*I_p &&
I_d'==ki*I_1-ki*I_d &&
I_l'==-(m1+m6*m1/(1-m6))*I_l+m2*I_p &&
I_p'==k_a1*I_sc1+k_a2*I_sc2+m1*I_l+(-m2-m4)*I_p &&
G'==-k_p3/VG*I_d-(k_p2+k1)*G+k2/VG*Gt+
    (rag+k_p1-F_cns)/VG-k_e1*max(0,G-k_e2/VG) &&
Gt'==k1*VG*G-k2*Gt-(V_m0+V_mx*X1)*Gt/(K_m0+K_mx*X1+Gt)

Figure 3.2: Proposed PTA Model for Glucose-Insulin Dynamics

3.1.2 Modeling Cyber (Discrete-Time) Components

Unlike the physical components, cyber components are characterized by having discrete-time behaviors. This section introduces the proposed PTA modeling for three commonly existing components in any realistic CPS:

- Sensor: In its simplest form, a sensor is a sampling unit for a specific physical quantity. The sensor PTA shown in Fig. 3.3 periodically samples the physical variable phy_var into the the measurement variable $meas\_var$. The clock variable t is constrained by the invariant $\{t <= T_p\}$ and the guard $\{t >= T_p\}$ so that a measurement is taken periodically with period of T_p. The channel signal $ch!$ notifies the other PTAs about the new measurement taken so that those PTAs can use the new measurement to update their response.

- Controller: A simple controller might look as the one shown in Fig. 3.4. This controller would initialize its parameters with the function $init_c()$ and it will wait for new measurements. Once the channel $ch?$ is activated by the sensor to indicate a new measurement value

16
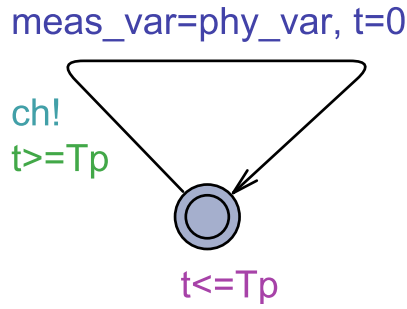
Figure 3.3: Proposed PTA Model for a Sensor
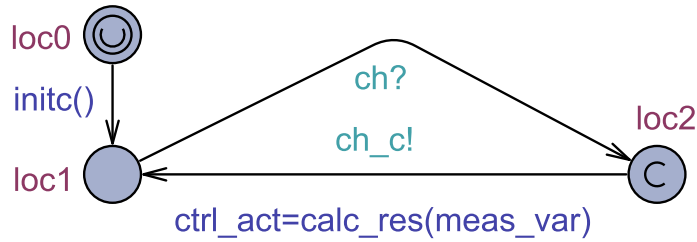
Figure 3.4: Proposed PTA Model for a Controller

$meas\_var$, the controller will take it to calculate a new control action $ctrl\_act$ using the function $calc\_res()$. It will announce that a new action is calculated using the channel ch_c!. The committed location (state) is an intermediate location that align the synchronization between different PTAs to ensure that the values of the variables evolve in the correct order.

- Actuator: Contrary to the sensor which gets values of physical quantities without modifying the physical PTA, an actuator receives commands from a cyber component and modifies the physical quantities accordingly. A simple actuator is modelled using the PTA shown in Fig. 3.5. This PTA waits to get an announcement on the channel ch_c about the existence of a new control value $ctrl\_act$ after which it will use this value to update the physical quantity variable phy_act.
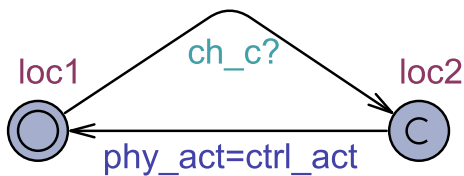
Figure 3.5: Proposed PTA Model for an Actuator

The exact modeling of a component's PTA depends on the prototype of the CPS under consideration. Likewise, it depends on the CPS attribute being examined. For example, the modeling can be modified to account for wireless transmission, sensor faults, or security attacks, etc. This will become clearer after presenting the contributions made to analyze specific CPS systems.

The PTAs of the various continuous-time and discrete-time components of the CPS are instantiated and parallel-composed to form a network of PTAs. This network is the model for the whole system. The resultant model is then analyzed for safety and performance using UPPAAL-SMC verification tool (A. David et al., 2015) where properties specified in a stochastic temporal logic are analyzed using numerical and symbolic methods (Agha & Palmskog, 2018).

3.2   Proposed Modeling and Analysis of Biomedical CPS

3.2.1   Closed-Loop Glucose Controller Security

Pancreatic beta-cells in Type 1 Diabetes (T1D) suffer from destruction by an autoimmune response. This results in a shortage of blood insulin which is needed to regulate the glucose levels. If not appropriately treated, high levels of glucose (hyperglycemia) can cause serious and permanent damage to some organs e.g., kidney failure (Van Belle, Coppieters, & Von Herrath, 2011). Contrarily, injecting an excessive amount of insulin may result in a dramatic drop in blood glucose level (hypoglycemia) which can be fatal. Artificial Pancreas (AP) system provides an automatic method to regulate the glucose level. The Continuous Glucose Monitor (CGM) provides the glucose measurements to the controller which decides the suitable amount of insulin to be injected into the patient's body through an insulin pump in a closed-loop manner.

In addition to the legacy control algorithms such as on-off controller and Multi-Basal (MB) controller, Proportional-Integrative-Differential (PID) controller was proposed by (G. Steil, Rebrin, & Mastrototaro, 2006) to emulate the 3-phase response of normal beta cells in healthy individuals. PID was adopted in the first commercial hybrid closed-loop AP system which was launched by Medtronic and approved by FDA in 2016 (Weaver & Hirsch, 2018). Since the integral term in PID controllers can cause the over-administration of insulin and hence postprandial hypoglycemia, most PID controller implementations try to avoid this problem by ignoring the integral term to get
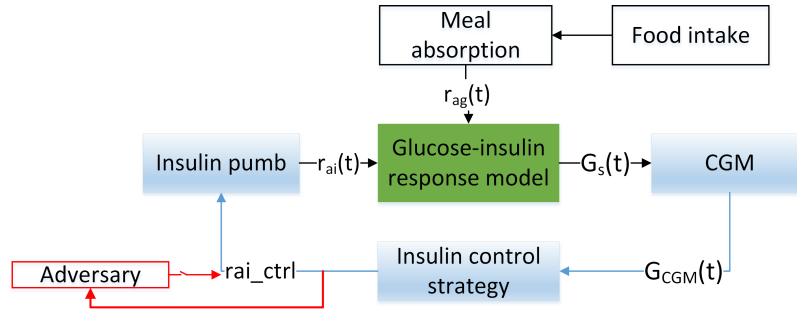
Figure 3.6: Overview of the Closed Loop Glucose System

the Proportional-Derivative (PD) controllers (Bequette, 2005). The controller can be located in close proximity to the patient or in a remote location at the central computer. In both cases, these systems are exposed to different types of passive and active attacks. In this work, the impact of the replay attack, which requires less capability from the adversary to conduct attacks, is investigated. The analysis is conducted on the physiological models of five patients selected randomly from a publicly available dataset. PTA modeling of an adversary on the channel between the controller and the insulin pump is also introduced. In this model, the adversary's goal is to increase the blood glucose level as much as possible using the replay attack. The impact of the attack on the glucose level under both control algorithms is evaluated.

In this modeling of the closed-loop AP, the modeled subsystems are classified into three categories: physiological processes, control system, and attack model. As shown in Fig. 3.6, the physiological processes are modeled using two PTA components: the glucose-insulin response model and the glucose ingestion model. The control system is composed of three main components: the CGM sensor, the insulin pump, and the controller that makes decisions about the insulin injection rate based on a preconfigured control algorithm. Lastly, a replay attack considered in this system is modeled by an adversary who is monitoring the insulin control channel.

The physiological processes of meal absorption (Appendix A) and glucose-insulin dynamics (Appendix B) are based on an FDA-approved model proposed in (Dalla Man, Camilleri, & Cobelli, 2006; Dalla Man, Rizza, & Cobelli, 2007; Man et al., 2014). The PTA for the glucose-insulin dynamics is similar to the one shown previously in Fig. 3.2. The inputs to this model are glucose rate of appearance ra_g from the meal absorption model and the insulin infusion rate IIR from the

rag=f*k_abs*Q_gut/BW
clk_inc?
Q_{sto1} : Solid stomach glucose
Q_{sto2} : Triturated stomach glucose
Q_{gut} : Gut stomach glucose
r_{ag} : Glucose rate of appearance

init_meals()

Q_sto1'==-k_gri*Q_sto1+meal_intake(t) &&
Q_sto2'==-k_empt(Q_sto1+Q_sto2,t)*Q_sto2+k_gri*Q_sto1 &&
Q_gut'==-k_abs*Q_gut+k_empt(Q_sto1+Q_sto2,t)*Q_sto2
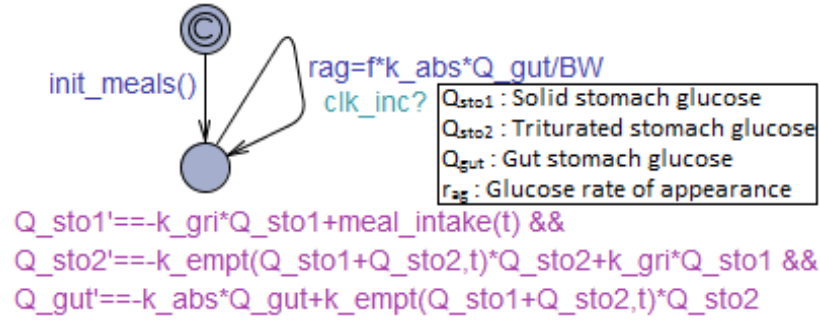
Figure 3.7: Proposed PTA Model for Meal Absorption

actuator (insulin pump), and the output is the subcutaneous glucose concentration G_s monitored by the sensor (continuous glucose monitor). The meal absorption is modeled by the PTA shown in Fig. 3.7. The function *init_meals* initializes the meals setting during the change from initial state to operational state at time $t = 0$. The variable ra_g is updated in the loop-back edge triggered by the global clock increment signal.

The CGM sensor periodically acquires measurements of G_s and passes the measurement value in the variable G_{CGM} to the controller. The controller is notified about new measurements using the broadcast channel nsr. The controller uses the measurements with a pre-configured algorithm to specify the suitable amount of insulin injection. It sends these values using the variable $ra_{i_{ctrl}}$ to the insulin pump together with notification on the channel nic. When receiving the *nic* signal, the insulin pump uses the values of r_{ai_ctrl} to update the value of the physical variable quantity ra_i in the glucose-insulin dynamics PTA.

In this work, two control algorithms are modeled: PD controller and MB controller. The PTAs that model both controllers are shown in Fig. 3.8 and Fig. 3.9, respectively. In the implementation, the PD controller is tuned for each patient by the *Ziegler and Nichols* tuning procedure (Ziegler, Nichols, et al., 1942) as described in (Haugen, 2010). On the other hand, the implemented MB controller switches between five fixed rates of insulin infusion (suggested by *Strategy II* in (Chen, Dutta, & Sankaranarayanan, 2017)) following the latest glucose measurement.

The glucose controller usually communicates with the CGM sensor and insulin pump over wireless connections. This wireless communication is necessary for the patients' convenience but exposes the system to a new class of passive and active attacks. In this work, an adversary model is
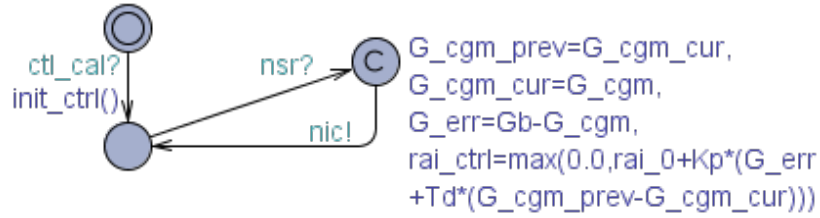
Figure 3.8: Proposed PTA Model for the PD AP Controller
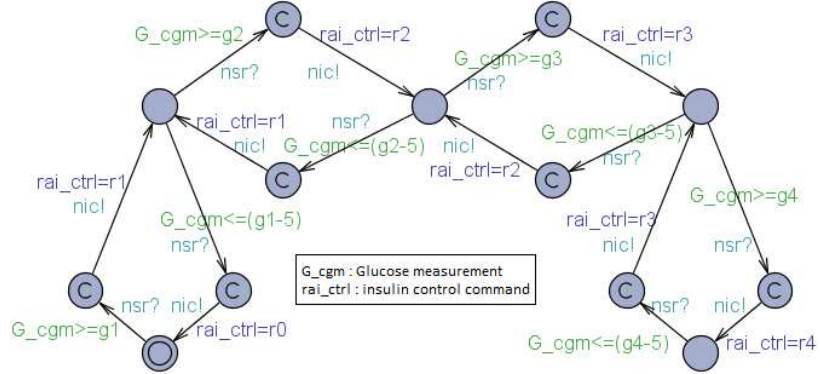
Figure 3.9: Proposed PTA Model for the Multi-Basal AP Controller

proposed where the adversary cannot generate customized packets but it can resend previously sent packets to conduct a replay attack scenario. In the modeling, the adversary is assumed to be capable of reading and analyzing the payload of transmitted packets but it is unable to create legitimate custom packets. This scenario is plausible if, for example, the packets are transmitted unencrypted but the integrity of the packets is secured. The adversary targets the insulin control channel using replay attack for the goal of causing the patient's blood glucose to elevate to large values and hence endanger the patient.

As can be seen in Fig. 3.10, the adversary starts in the listening state and responds to *nic* signal that is originated from the legitimate controller to buffer the value of insulin rate update command in its local variable *pkt* which is continuously updated to store the command with minimal value of insulin rate. Following any command from the controller, the adversary sends a false command with this buffered value. In particular, the adversary model modifies the shared variable rai_{ctrl} with the buffered value and signifies the insulin pump about this change by signalling the broadcast channel *nic*. This behavior is demonstrated in Fig. 3.11 which shows a segment of the insulin rate update commands for one of the patients.

21
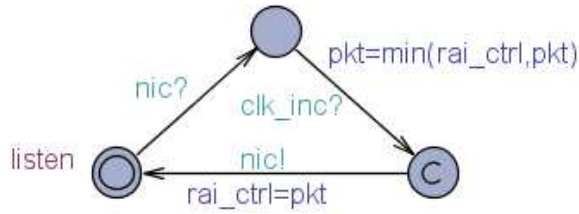
Figure 3.10: Proposed PTA Model for the Adversary
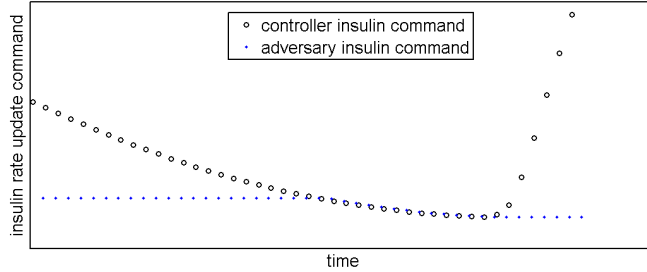
Figure 3.11: Insulin Rate Update Commands under Replay Attack

The model of the AP system is simulated and formally analyzed on a *2GHz Intel Core 2Duo CPU* that operates Microsoft Windows 7 Professional. Adversary model is included to analyze the impact of a replay attack on the system, and can be removed to analyze the system safety under the attack-free scenario. Similarly, PD or MB controller can be included in the system in order to evaluate the behavior of a specific controller. The insulin rates and glucose values of MB controller are used similar to the suggested *Strategy II* in (Chen et al., 2017). For PD controller, the parameters for each patient are tuned by the *Ziegler and Nichols* tuning procedure (Ziegler et al., 1942) as described in (Haugen, 2010).

In this analysis, the behavior of the AP system is evaluated for five patients from a publicly available dataset (Man et al., 2014). For each of the patients under test, the system is analyzed for a full day and night scenario of 24 hours. The test starts at 7:00 AM, the patient is assumed to get three meals at 8:00 AM, 1:00 PM, and 7:00 PM. Each meal contains 70 grams of carbohydrates.

Two safety properties are analyzed for the high and low blood glucose levels written in Metric Interval Temporal Logic (MITL) queries:

- **Pr[t≤1440] (<> G>300) ≤ 0.0101**: which means: *"the probability of reaching hyperglycemia state of blood glucose level greater than 300 (mg/dL) at any time of the day should*
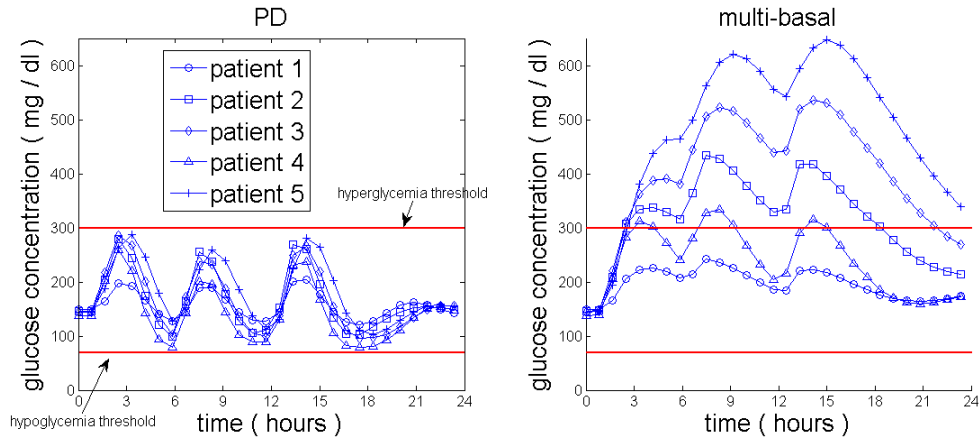
Figure 3.12: Attack-Free Simulation Results for 24 Hours

not exceed 0.0101"

- **Pr[t≤1440] (<> G<70) ≤ 0.0101**: which means: *"the probability of reaching hypoglycemia state of blood glucose level less than 70 (mg/dL) at any time of the day should not exceed 0.0101"*

The simulation results for the attack-free scenario are shown in Fig. 3.12, the PD controller successfully regulated the blood glucose levels for the five patients to satisfy both safety properties at all times. Contrarily, the MB controller only succeeded to satisfy the hyperglycemia safety property for patient 1 but all other patients exceeded the hyperglycemia threshold of 300 (*mg/dL*).

For the analysis under replay-attack scenario, the adversary model is included in the system by composing the PTA describing the attack with the system behaviour. As shown in Fig. 3.13, the PD control satisfies the hyperglycemia safety property for only two patients while the other three patients slightly exceed the hyperglycemia threshold for varying amounts of time. For MB control, none of the patients satisfies the safety properties.

Reachability analysis is conducted for both controllers in both security scenarios. The results showed that the hypoglycemia safety property was satisfied for all patients, control strategies, and attack scenarios. On the other hand, the hyperglycemia safety property provided results that depends on the patient, the control strategy used, and the attack scenario proposed. For each setting, these results are consistent with the simulation results in Fig. 3.12 and Fig. 3.13. The average runtime is 443 seconds per run for 0.95 confidence level which is a reasonable time duration for this type of

23
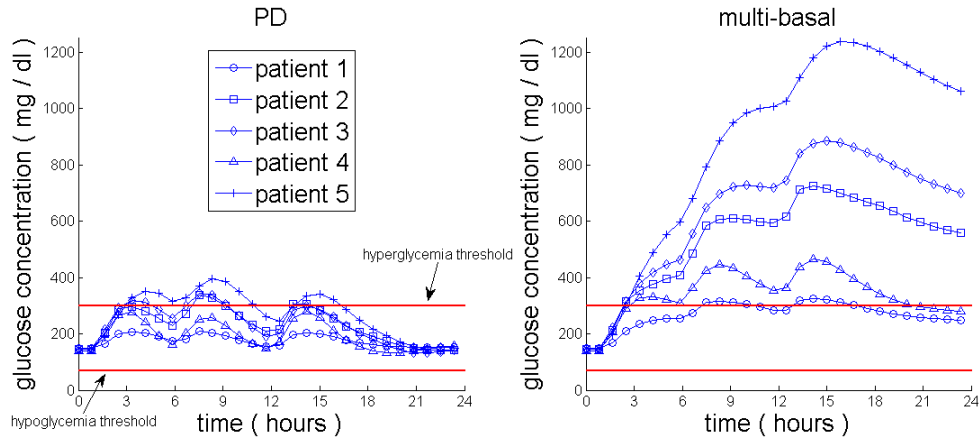
Figure 3.13: Simulation Results for 24 Hours under Replay Attack Condition

verification.

The results show that the PD control is performing better than MB control. In the attack-free scenario, the PD control satisfies the safety properties for all patients. Under replay-attack, some PD-controlled patients violates hyperglycemia safety properties. However, these violations are smaller compared to the MB control case.

3.2.2 Closed-Loop Anesthesia Controller Safety Under Sensor Faults

Anesthesia control is a common practice in medical treatment in which a specific amount of drug is applied to sedate the patient. During surgery, the anesthesiologist selects the suitable amounts and times of individualized drug delivery to maintain general anesthesia. The reliability and stability of anesthesia control are dependent on the anesthesiologist's expertise as the one responsible for observation-based intervention. Closed-loop anesthesia control aims at automating this process to achieve finer control that is more accurate. An automatic controller monitors the status of physiological measures to quantify the sedation level and apply the suitable drug dose based on a pre-configured closed-loop technique.

Besides being convenient, closed-loop control provides a dedicated tracking system that continuously monitors the status of the patient without human interaction. The only human intervention takes place when initializing the configuration and parameter tuning. The selection of tuning parameters is based on well-known pharmacokinetic-pharmacodynamic (PK-PD) models that estimate the

24

prospective behavior of the human body as a function of its criteria such as weight, age, gender, and height (Sahinovic, Struys, & Absalom, 2018).

On the downside, automatic control is usually sensitive to disturbances in parameters estimation resulting from unaccounted inter-patient variations. Also, the reliability of the system can be affected by device faults. In this work, a network of PTAs is proposed to model the closed-loop anesthesia control system using two control techniques: Proportional-Integral-Derivative (PID) controller and Quasi-Continuous Second-Order Sliding-Mode Controller (QC-SOSMC).

As a result of its favorable drug effect profile, propofol has been widely adopted as a hypnotic intravenous drug in medical treatment (Sahinovic et al., 2018). When a propofol dose is administered in the body, its distribution and clearance throughout the body over time is described by the three-compartment PK model (Sahinovic et al., 2018). In 1987, Gepts et al. (Gepts, Camu, Cockshott, & Douglas, 1987) conducted an experiment on a group of adults to estimate the constant PK parameters.

Various experiments have been conducted since then to better estimate the PK model parameters. These experiments revealed the relationships between the PK parameters with bodyweight (Marsh, White, Morton, & Kenny, 1991), height, Lean Body Mass (LBM) and gender (Schnider et al., 1998). Each of these models is tuned for a sub-population which reduces their ability to expect behaviors on patients with different characteristics.

To overcome this limitation, recent work from Eleveld et al. aggregated propofol datasets from multiple previous studies and analyzed them to develop a single general PK model (D. J. Eleveld, Proost, Cortinez, Absalom, & Struys, 2014) and PD model (D. Eleveld, Colin, Absalom, & Struys, 2018). Retrospective evaluations on performance have shown that this general model behaves as well as or even better than models developed for specific populations (Sahinovic et al., 2018). Also, this model used random residual errors to describe unaccounted-for inter-patient variations.

One of the most common techniques to monitor the sedation level is the Bispectral Index (BIS) which analyzes the electroencephalogram (EEG) waveforms. BIS provides a non-invasive technique to track the level of hypnosis under surgery. To maintain general anesthesia, BIS values between 40 and 60 are recommended (Johansen, 2006). In spite of being accurate, BIS monitors can fail to

25
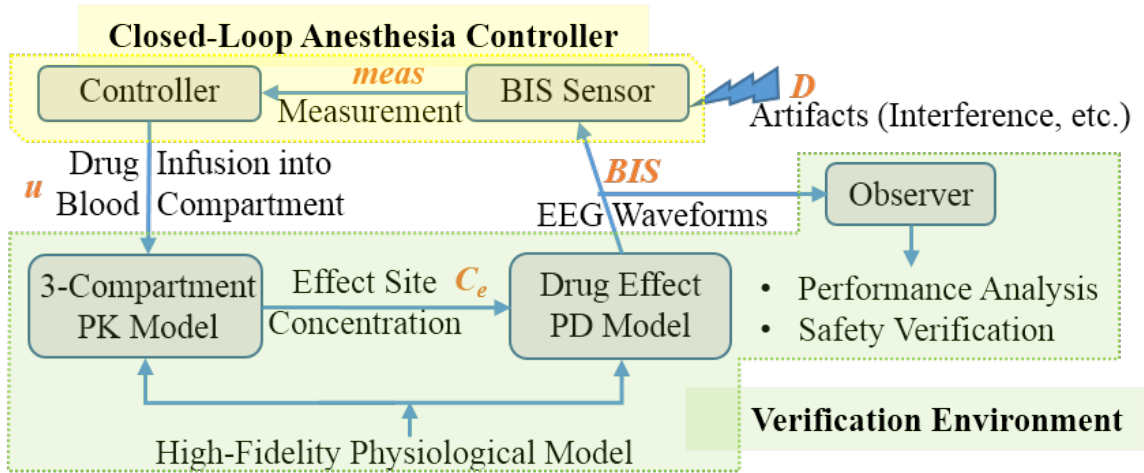
Figure 3.14: Proposed Modeling

provide correct measurements when affected by nearby artifacts such as pacer-induced (Gallagher, 1999) and electrocardiogram (ECG) (Myles & Cairo, 2004).

For the automated closed-loop anesthesia control systems to gain trust, their control techniques are required to demonstrate robustness against device faults and randomly-distributed inter-patient PK-PD variability. In general, evaluating the impact of faults on these systems can lead to situations that would affect the health conditions of the experiment population. Therefore, the accurate *in-silico* analysis provides a safe alternative approach to evaluate fault scenarios.

The overview of the system is shown in Fig. 3.14. The PK-PD model is implemented based on the recently proposed high-fidelity physiological model (D. Eleveld et al., 2018; D. J. Eleveld et al., 2014). The propofol kinetics as it moves between plasma, fast-equilibrating, and slow-equilibrating compartments are governed by the 3-compartment model where $\{C_1, C_2, C_3\}(\mu g/mL)$ are the concentrations of propofol in these compartments, respectively. Moreover, a theoretical effect-site compartment with drug concentration $C_e(\mu g/mL)$ is attached to the central compartment C_1 to model the diffusion of drug towards effect site. The PK dynamics of propofol in these compartments are described by a system of Ordinary Differential Equations (ODE) where each k_{ij} is the drug transfer from compartment i to compartment j. The BIS value is derived from C_e following the PD model in (D. Eleveld et al., 2018). The adopted model in this work accounts for inter-patient variations and residual errors.

In this work, two control techniques are modeled and analyzed: the PID controller, and the QC-SOSMC controller. When conducting PID control (Ziegler et al., 1942), the controller input to the system is governed by the following equation:

$$u(t) = u_0 + k_c \left(e(t) + \frac{1}{\tau_I} \int e(t) dt + \tau_D \frac{de(t)}{dt} \right) \tag{1}$$

where $e(t)$ is the difference between the measurement and the target value and the parameters $(u_0, k_c, \tau_I, \tau_D)$ are calculated using *Ziegler and Nichols* tuning procedure (Ziegler et al., 1942).

Sliding mode controllers are known for their insensitivity to parameter variations. In this work, a QC-SOSMC (Hernández et al., 2013) controller is implemented which outputs the following control signal:

$$u(t) = -\alpha \frac{\sigma'(t) + \beta |\sigma(t)|^{1/2} sign(\sigma(t))}{|\sigma'(t)| + \beta |\sigma(t)|^{1/2}} \tag{2}$$

where (α, β) are controller tuning parameters, $\sigma(t)$ is the sliding variable calculated as the difference between target value and the measured value and $\sigma'(t)$ is its derivative.

The overview of the proposed modeling is shown in Fig. 3.14. The actual BIS value determined from the physiological model is measured by a sensor that can be affected by temporal external artifacts with random arrival rate. The sensor delivers the measurements to the controller which conducts calculations using a pre-configured technique to decide about the amount of propofol injection into the plazma and hence provide closed-loop control.

The proposed PTA model of the physiological models is shown in Fig. 3.15 where the model parameters are first initialized in the function $init\_params()$. The contents of the location's $Invariants$ are specified to apply the constraints of the ODE as described previously. The ODE variables are the propofol concentrations in the three compartments and the effect-site compartment. For each time step, the BIS value is updated by the function $calc\_BIS()$.

The PTA model of the *Sensor* shown in Fig. 3.16 measures BIS and is affected by temporal faults that occur randomly with exponential arrival that is characterized by Inter-Arrival Time (IAT) of IAT_F. When a fault happens, it is assumed to last for $F_D(min)$ which represents the fault duration before diminishing. When a new measurement is conducted, the sensor notifies the controller
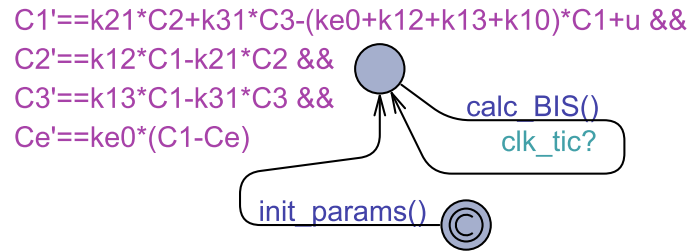
Figure 3.15: PTA of the Propofol PK-PD Model
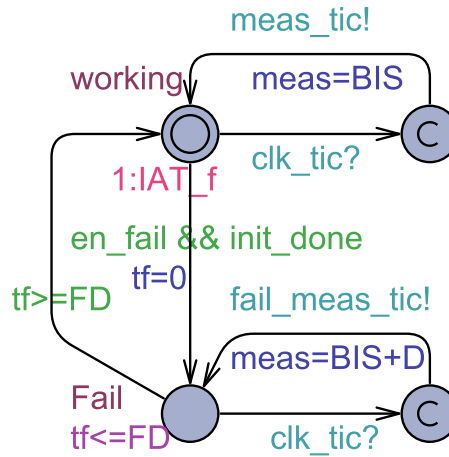
Figure 3.16: Proposed PTA Model of the Sensor

using the broadcast channel $meas\_tic$. Otherwise, it uses the broadcast channel $fail\_meas\_tic$ to announce measurement with artifact where D is the uniformly distributed additive distortion due to fault.

Each of the two modeled controllers has a PTA that looks like the one shown in Fig. 3.17 with little differences. When the controller receives a notification through the broadcast channel $meas\_tic$, it updates the control variable using the function $update\_ctrl()$. If instead the controller is notified about a measurement with an artifact, the controller will either ignore the measurement or take it depending on probability weights that represent the controller's hardware ability to detect the existence of distortion. The function $update\_ctrl()$ calculates the control variable using either (1) for PID or (2) for SOSMC and updates $u(t)$ which is the intravenous propofol injection rate.

In the PTA shown in Fig. 3.18, the observer waits for the BIS to settle inside the target range limited by the strictly-bounded upper and lower range limits (R_L, R_U) while the safety range is limited by the wider range bounded by (R_{LC}, R_{UC}). When any of these ranges is violated, the
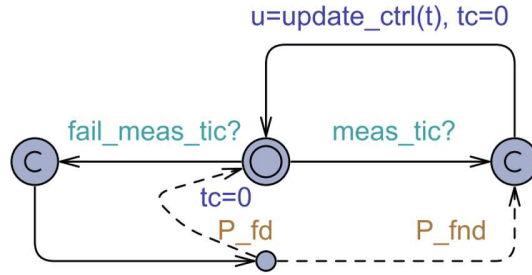
Figure 3.17: PTA of the Controller
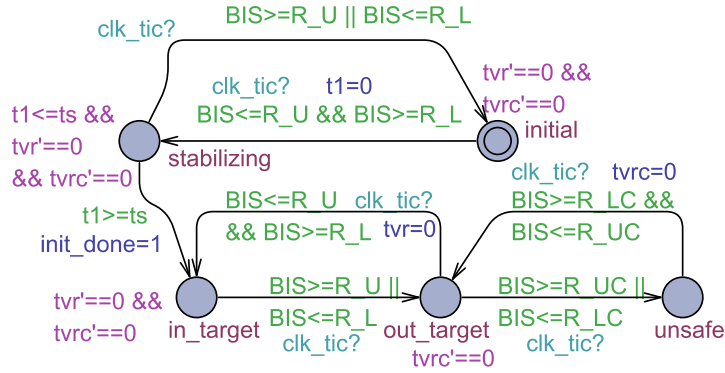
Figure 3.18: Proposed PTA Model of the Observer

observer uses the time variables tvr and $tvrc$ to track the durations of incidents outside the target range and safety range, respectively. The safety of each controller was evaluated using the following Metric Interval Temporal Logic (MITL) query:

$$Pr[t <= test\_time](<> tvrc > 5) <= 0.0101 \qquad (3)$$

which verifies the existential probability of a consecutive period outside the safety zone for longer than 5 minutes.

To analyze the safety of the system, both control techniques are evaluated on the reference parameters in (D. Eleveld et al., 2018) under error-free scenario and temporal sensor fault incidents with fault duration F_D set to 5 minutes. The faults are assumed to generate additive noise uniformly distributed in the range $[-20, 20]$ and the controller's hardware is assumed to detect the existence of artifact with detection probability of 75%. The parameters of the PID controller were tuned (Ziegler et al., 1942) for the nominal values of the patient parameters and the SOSMC parameters
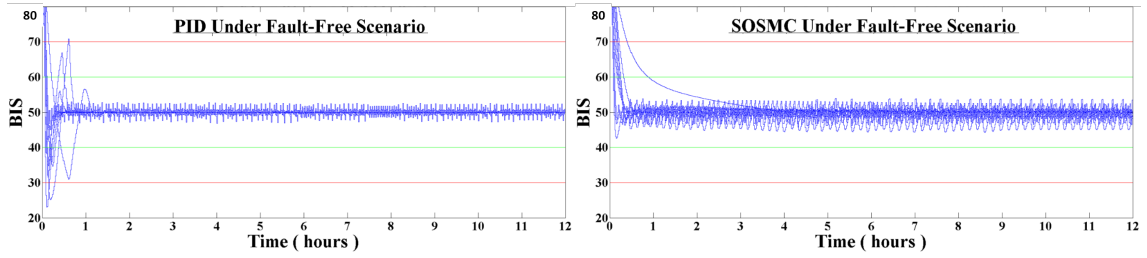
Figure 3.19: Simulation Results Under an Error-Free Scenario
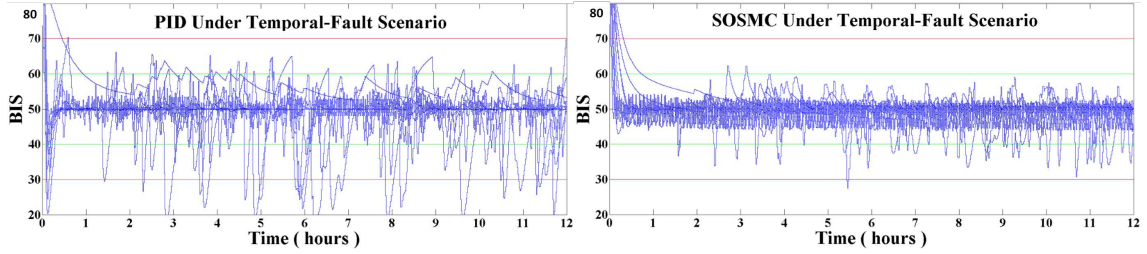
Figure 3.20: Simulation Results Under a Temporal Sensor Fault Scenario (fault rate = 1/30)

were set to $\{\alpha = 3, \beta = 9\}$. Both controllers were constrained for propofol infusion rates capped at $10(\mu g/mL/min)$.

The simulation results for both controllers under error-free scenario and under temporal-fault scenario are shown in Fig. 3.19 and Fig. 3.20, respectively. It can be seen that PID controller is susceptible to oscillations and is adversely affected by sensor fault while SOSMC controller is more robust against faults. The green lines and the red lines in Fig. 3.19 and Fig. 3.20 are the limits for the target range and the safety range respectively.

For a test time of 24 hours, the safety property defined in (3) was found to be satisfied for both controllers under an error-free scenario with confidence 95%. Then the safety property was checked for both controllers under temporal sensor faults with fault duration $F_D = \{1, 3, 5, 7\}(minutes)$ for different values of IAT_F. Table 3.1 summarizes the minimum tolerable fault inter-arrival time after which each controller starts to violate the safety property in (3). The table shows that both controllers suffer from the fault with notable advantage for SOSMC over PID.

Figure 3.21 shows the expected maximum duration outside the target range before recovery for both controllers under different fault rates (Inf is equivalent to error-free). For each rate, 500 random simulation runs were conducted on a *1064MHz Intel(R) Xeon(R) CPU E7- 8870* that operates

Table 3.1: Minimum Tolerable Fault's Inter-Arrival Time

fault duration ($minutes$)	1	3	5	7
min tolerable fault's IAT ($hour$) (**PID**)	14	16	20	70
min tolerable fault's IAT ($hour$) (**SOSMC**)	1	4	9	15
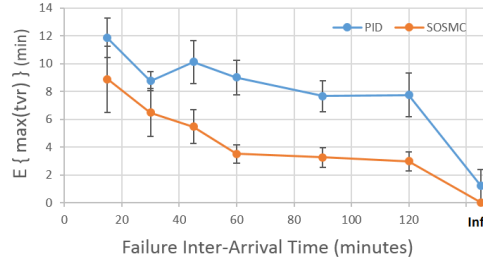
Figure 3.21: Expected Maximum Duration Outside Target Range Before Recovery

Scientific Linux 7.7 (Nitrogen) with average runtime of 226 seconds per run. The results show that SOSMC outperforms PID for closed-loop anesthesia control under temporal fault.

3.3 Proposed Modeling and Analysis of Automotive CPS

3.3.1 Coordinated Vehicular Emergency Braking System Safety Under Degraded Wireless Connectivity

The gradual increase in the number of autonomous vehicles driving in the streets paves the way for utilizing coordination among vehicles. This coordination promises to maximize the benefits offered in terms of reducing fatalities, saving fuel consumption, and increasing traffic throughput.

Besides hardware and software faults, automated systems respond to incidents using pre-configured algorithms that might fail to adapt to new unaccounted-for situations. Also, the full potential of benefits of automated systems can only be harvested when these systems are pushed to operate on the limits which requires more stringent safety specifications for these systems. For example, reducing inter-vehicle distances between vehicles in a platoon results in less aerodynamic drag and consequently in significant fuel saving (Liang, Mårtensson, & Johansson, 2015). However, it is crucial for these platoons to conduct efficient coordination to avoid rear-end collisions. Therefore, safety properties describing the ability of all the vehicles to reach a collision-free full stop in case of emergency

31

is essential for the design of these systems.

In this work, new modeling and formal analysis are proposed to analyze the safety of a vehicular Coordinated Emergency Braking (CEB) system. The safety of the system is investigated under and an error-free scenario and under a scenario that involves degraded wireless connectivity. A retransmission scheme is also analyzed which resulted in safety improvement especially under severe levels of wireless degradation.

Uncooperative stand-alone automatic vehicular systems such as adaptive cruise control are required to sustain larger values of inter-vehicle distances to achieve safety and avoid collisions (Rajamani, 2011). By incorporating inter-vehicle communication, vehicles can share information about their parameters and surroundings to allow more efficient platooning that would involve shorter inter-vehicle distances and hence better fuel saving (Rajamani, Tan, Law, & Zhang, 2000).

Several systems have been proposed in the literature for collision avoidance. The possibility to avoid rear-end collisions by accelerating is studied in (Zheng, Nakano, Yamabe, & Suda, 2013). In (Zheng et al., 2014), during a coordinated braking maneuver the last vehicle of the platoon is instructed to decelerate at the highest rate and the rate is gradually decreased for the preceding vehicles. In (Murthy & Masrur, 2016), all the vehicles of the platoon adjust their maximum deceleration to the limitation of the vehicle with the weakest deceleration rate.

Various studies have investigated the safety of emergency braking systems in vehicular platoons. A secondary brake system is proposed in (Aki et al., 2014) to operate when the main brake system fails. A synchronized braking approach is proposed in (Hasan, Balador, Girs, & Uhlemann, 2019) where the whole platoon brakes simultaneously. Learning based testing approach is adopted in (Bergenhem, Meinke, & Ström, 2018) to investigate the safety of Coordinated Emergency Braking Protocol (CEBP). Unlike these test-based simulation studies, we are proposing a new PTA-based modeling and SMC-based analysis approach for verifying the safety of CEBP under parameter uncertainties as well as degraded wireless connectivity.

The analyzed target system is composed of N cooperative vehicles where a section of the platoon is shown in Fig. 3.22. v_i, a_i are the velocity and acceleration of the ith vehicle, x_i is the inter-vehicle distance between the ith vehicle and the preceding $(i + 1)^{th}$ vehicle. The first-order
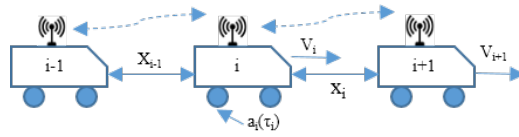
Figure 3.22: Target System Architecture

inertial delay between the desired and the actual accelerations is represented by the time constant τ_i (Li, Deng, Zheng, & Peng, 2015).

The vehicles in the platoon use Vehicle-to-Vehicle (V2V) communication to coordinate their cruising speed and inter-vehicle time headway T_{hw} distances. Also, they use it to agree on the adopted procedure for dealing with anticipated circumstances especially the emergency procedure. Although cooperative cruising control systems can maintain the velocity and T_{hw} around a nominal value, they suffer from transient response that can result in temporary compression waves that can distort these nominal values prior to initiating the emergency braking procedure.

For the vehicles to travel safely with smaller time headway values, they have to preserve connectivity in order to act in a timely manner to any emergency. Although it is intended to support vehicular networks, IEEE 802.11p adopts a multiple channel access technique that can lead to unbounded delays especially in a congested environment. So it is not uncommon to find Time Division Multiple Access (TDMA)-based implementations for vehicular applications (Bilstrup, Uhlemann, Ström, & Bilstrup, 2009). When using this approach, a time slot is reserved for each member of the network. Also, retransmission slots can be assigned to nodes with low Packet Delivery Ratios (PDR) in order to equalize their connectivity (Böhm, Jonsson, Kunert, & Vinel, 2014).

In the CEBP (Bergenhem et al., 2018), when the leading vehicle decides to initiate an emergency braking procedure it sends an emergency request to the rear vehicle. The request travels in a multi-hop manner until it reaches the last vehicle which brakes instantaneously and sends an acknowledgement that travels in the opposite direction. Each of the vehicles in front will either receive the acknowledgement and start braking or wait for a time-out before braking anyway and notifying the other vehicles with a message.

Formal modeling and analysis of this CEBP is investigated under normal connectivity conditions and under degraded wireless connectivity on a single node. This can happen for example when that
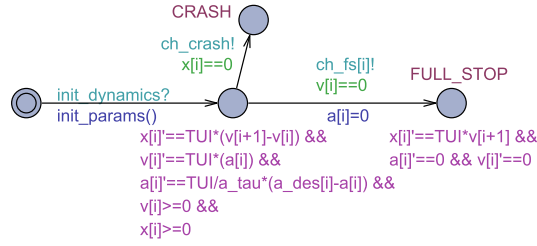
Figure 3.23: Vehicle PTA Modeling

node is affected by a passenger's wireless device that is interfering with the vehicular network. The ability of a retransmission scheme (Böhm et al., 2014) to mitigate this situation is also investigated.

The vehicle dynamics are modeled using the PTA shown in Fig. 3.23. Starting from the initial location, the parameters are initialized in the function $init\_params()$. The differential variables constrained in the invariant field of the main location are based on the laws of motion. If either the velocity or the distance from the preceding vehicle drops to zero then the PTA moves to the respective location and sends a broadcast channel to announce for the other PTAs.

The wireless channel environment is modeled using the PTA shown in Fig. 3.24. This PTA assigns the slots for the nodes using the broadcast channel vector ch_turn and forwards the messages to the next hop with a probabilistic choice for successful packet delivery following either the fault probabilities for the degraded-wireless node or the normal probability weights for the other nodes. Also, vehicle controllers similar to the one shown in Fig. 3.25 are used to model how wireless messaging and acceleration control are managed.

These are the PTAs which construct the main system model. All the modeled PTAs representing the whole system are instantiated and parallel-composed to construct the overall system-level model. This model can be customized by selecting the respective options that direct the system to operate in a specific mode, such as error-free or degraded wireless scenarios.

In this work, SMC-based analysis is used to evaluate the safety of a CEBP protocol on two platoons with different number of vehicles under normal operating conditions and under wireless connectivity degradation affecting a single vehicle with varying levels of Packet Error Rate (PER). Also, the impact of imposing a retransmission scheme is evaluated.

It is assumed that prior to the emergency braking, the vehicles sustain specific values of velocity
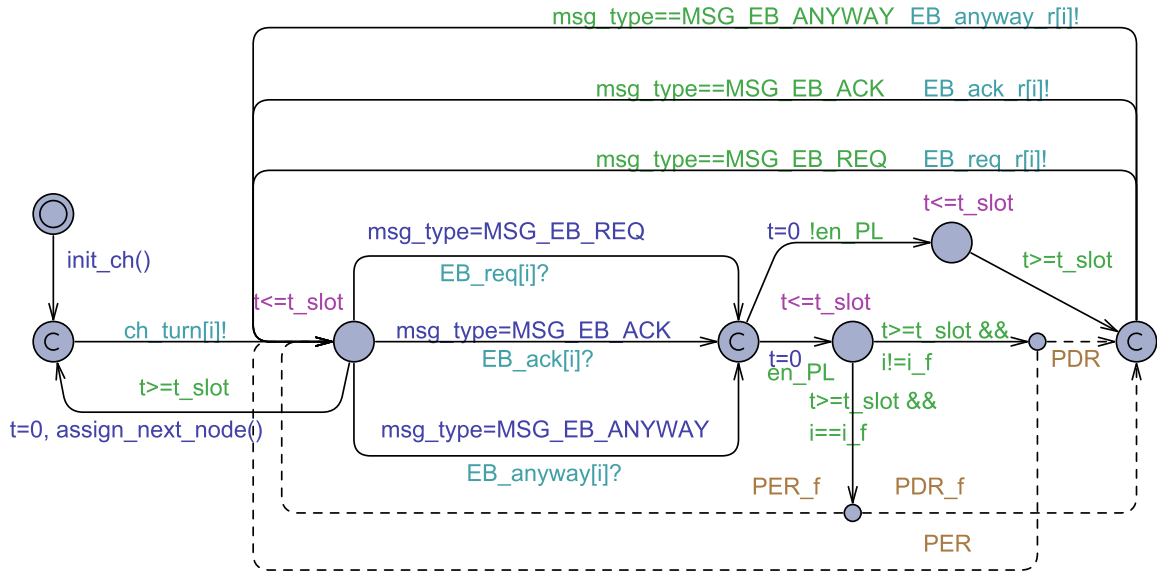
Figure 3.24: PTA Modeling of the Wireless Channel Environment
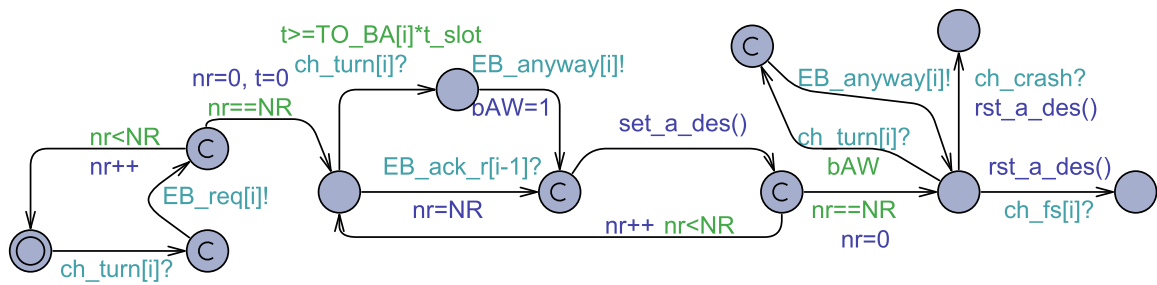
Figure 3.25: Vehicle Control PTA Modeling (Leading Vehicle)

35
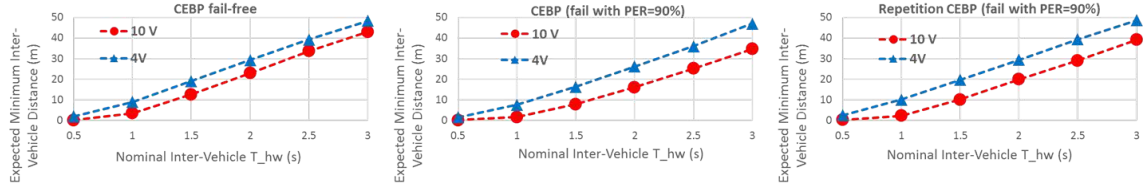
Figure 3.26: Minumum Inter-Vehicle Distance During an Emergency Break under normal conditions (left) and under a wireless-degraded node with PER=90% using CEBP (middle) and CEBP with retransmission scheme (right)

and time headway distance. While different values for T_{hw} are used, the speed nominal value is set to $20\,(m/s)$ and nodes can deviate from these nominal values with an upper bound $\pm 15\%$ to account for disturbances such as the ones caused by cruising transient response. It is also assumed that the vehicles have agreed in advance on the maximum braking deceleration of $4\,(m/s^2)$ to accommodate the less-capable vehicles. The acceleration time constant for all vehicles is assumed to be $0.1\,(s)$. For the wireless communication parameters, TDMA-based channel access technique is assumed where each time slot is equal to $10\,(ms)$. The PER under normal conditions is 4% between two consecutive vehicles. For the node with degraded wireless connectivity, different values of PER are analyzed. When retransmission scheme is used, this node will be assigned extra time slots in order for it to equalize it's equivalent PER.

For different values of preset headway times T_{hw}, the expected minimum inter-vehicle distance during the braking procedure is estimated following 500 random simulations per data-point. The results are shown in Fig. 3.26 where it can be noted that the fault scenario reduces the expected minimum inter-vehicle distance and its effect is more notable for the larger platoon with 10 vehicles. Also it can be noted that the retransmission scheme tends to reduce the effect of the fault especially for the larger platoon.

The analysis also investigated formal properties written using Metric Interval Temporal Logic (MITL) queries. For the emergency braking system, the following safety property is checked:

- **Pr[t$\leq T_m$] ($<>$ NO_CRASH** $\&\&$ **ALL_FS)** $\geq$ **0.99**

Which checks whether with a 95% of confidence the probability of reaching to a state of no crash and all the vehicles are fully stopped within time bound T_m is above 0.99.
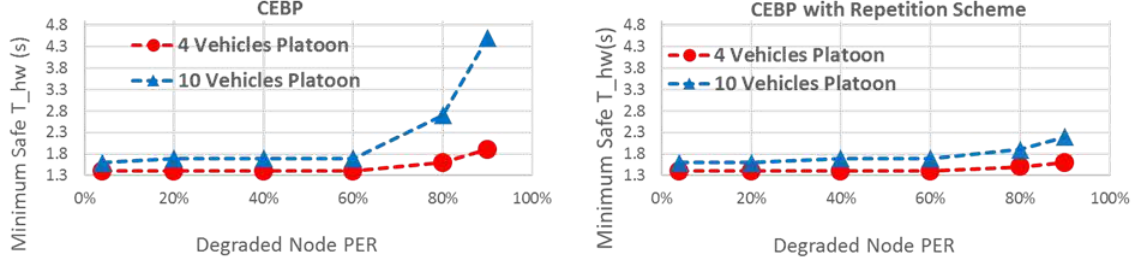
Figure 3.27: Minumum Safe Inter-Vehicle Headway Time for CEBP (Left) and CEBP with a Re-transmission Scheme (Right)

In order to evaluate the system's capability to satisfy the safety property defined above, the nominal time headway T_{hw} was changed at increments of $100\ (ms)$ until the minimum acceptable value that can satisfy the safety property is found. The process was repeated for different values of PER using both CEBP and CEBP with retransmission scheme.

The resulting minimum safe T_{hw} values are shown in Fig. 3.27. Under the evaluated scenario, the system can satisfy the safety property under normal connectivity with as low as $1.4\ (s)$ and $1.6\ (s)$ for platoons of 4 and 10 vehicles, respectively. When PER values on the faulty node increase gradually, the required T_{hw} to satisfy safety increases by $500\ (ms)$ and $200\ (ms)$ for CEBP and CEBP with retransmission scheme, respectively. For a larger platoon of 10 vehicles, CEBP follows the same behavior as in the smaller platoon case until PER value of 60%. After this PER value, the required T_{hw} dramatically increases up to 3 times the error-free nominal value. On the other hand, CEBP with retransmission scheme has shown better mitigation for large PER values.

3.4   Summary

In this chapter, a new approach of formal modeling for CPS is proposed. In this approach, a PTA is constructed to model the behavior of each of the physical and cyber components of the system as well as the abnormal behaviors under consideration. The parallel-composition of all the PTAs form the behavioral model of the whole system. This model is then simulated and formally analyzed using UPPAAL-SMC tool to verify safety properties.

This proposed approach is used to analyze medical and automotive CPS behaviors under faults and security attack scenarios. This early analysis can provide valuable feedback that can give insights into the ability of CPS to meet safety requirements under realistic conditions, and hence guide the design of more robust and safe systems.

From the experiments conducted, a few characteristics can be noted to differ between the biomedical models and the automotive models:

- The time units: The biomedical dynamics tend to change over time of minutes or slower, which is attributed to the nature of the underlying biochemical reactions and diffusion which happen gradually over time. Automotive dynamics involve faster changes in general especially when modeling real time motions.

- Number of instances: When processing the biomedical systems, the internal dynamics of one subject are processed one subject per experiment. On the contrary, automotive subjects are analyzed in the existence of other subjects whether connected or standalone, so multiple instances of the dynamics are processed per experiment.

- Complexity: The dynamics of a biomedical subject involve various parameters that affect the behavior and hence tend to be more complicated than an automotive subject.

However, it should be said that the above differences might not be true in all cases. For example, the dynamics of heat dissipation or battery processes in an automotive subject are slower in nature than the mechanical motion kinetics, and hence require slower time units. Similarly, instead of one large model in biomedical systems, some dynamics on the cellular level might impose the processing of a large number of entities with simpler dynamics per instance.

Chapter 4

Article I: Towards Safe and Robust Closed-Loop Artificial Pancreas Using Improved PID-Based Control Strategies

Authors: Abdel-Latif Alshalalfah, Ghaith Bany Hamad, Otmane Ait Mohamed

Abstract: Artificial pancreas enhances the life experience for diabetic patients by allowing them to live normally with their glucose levels controlled automatically with minimal or no intervention. For closed-loop glucose controllers to be approved for clinical practice, they have to prove safety under all potential scenarios. One of the biggest challenges of closed-loop glucose control is to handle the distortion caused by meal intake. This challenge becomes more problematic when taking into account the imperfections and limitations of glucose sensors. In this paper, we propose new Proportional-Integral-Derivative (PID)-based control strategies for robust glucose control under varying meal conditions. The proposed approaches aim at counteracting the challenges imposed by the large delays incurred in glucose sensing and insulin action. Statistical model checking was utilized to analyze the performance figures and safety properties as compared with existing closed-loop techniques. The results have shown that one of the proposed approaches provide substantial enhancements towards safe and robust glucose control especially under sensor noise. Where, under

a typical relative meal size between 75 and $125 (g/100Kg)$, the proposed approach can satisfy hypo-glycemia safety property for 90% of the patients compared to lower than 50% of the patients for the other investigated techniques. These enhancements can be achieved without additional personalized tuning beyond the standard PID control.

4.1   Introduction

For a healthy individual, the regulation of blood glucose level is achieved by endogenous insulin that is secreted by beta cells in the pancreas. This insulin acts in the blood to excite the storage of blood glucose into fat cells and hence reduces the blood glucose. In Type-1 Diabetic (T1D) patients, beta cells are affected by autoimmune destruction which prevents insulin production (Todd, 2010). For those patients, exogenous insulin is essential to regulate the blood glucose levels.

Artificial Pancreas (AP) systems provide mechanisms to automate the delivery of insulin and allow T1D patients to live normal life. In AP systems, insulin pumps are configured to infuse con-trolled amounts of insulin. Insulin pumps can be programmed to operate in an open-loop, hybrid closed-loop, or closed-loop scheme. The first two approaches involve varying amounts of human intervention to modify the insulin injection rate or to announce the amounts of carbohydrates in-gestion. However, closed-loop AP systems are only configured during the initialization and then operate autonomously.

Food and Drug Administration (FDA) has approved the first commercial hybrid AP device de-veloped by Medtronic in 2016 (Weaver & Hirsch, 2018). Since then, other device manufacturers have worked towards approval for their AP systems such as Tandem (Lal, Ekhlaspour, Hood, & Buckingham, 2019). However, these systems offer automated insulin delivery but not a fully closed-loop. Instead, they provide hybrid closed-loop control where the user is expected to announce meals and take additional insulin boluses with meals.

For closed-loop AP systems to be approved by authorization entities, they have to demonstrate sufficient reliability and safety. A key criterion for the safety of an AP system is the ability to decide the suitable amounts of insulin. If less-than-required amounts are administered, the blood glucose levels might increase resulting in hyperglycemia. Frequent and long periods of hyperglycemia can

result in permanent damages to some organs such as the kidney. On the other side, excessive insulin injection might result in a dramatic blood glucose drop (hypoglycemia) which can cause coma and even death.

In this paper, two Proportional-Integral-Derivative (PID)-based control strategies are proposed for closed-loop AP: an Adaptive Weighted PID (AWPID) controller, and a Look-Ahead PID with Retrospective estimation Error Correction (LAPID-REC). In AWPID, the proportional gain of the PID controller is weighted based on the short-term history of glucose measurements. The weights are selected to account for the expected physical interactions. In the LAPID-REC approach, the current measurement used in classical PID is replaced by prospective estimates of future measurements to calculate the control action with retrospective estimation error correction. This LAPID-REC approach counteracts the long delays incurred in glucose measurement and insulin action. Moreover, retrospective correction helps rectify the variations of glucose measurements, whether caused by physiological dynamics or sensor imperfections.

Both control strategies were evaluated on a set of virtual patients from an FDA-approved model (Man et al., 2014) to evaluate their performance and safety (Alshalalfah, Bany Hamad, & Ait Mohamed, 2019). All components of the system were modeled using a Network of Priced Timed Automata (NPTA). Then, the safety properties of each method were investigated using Statistical Model Checking (SMC). The results have shown that AWPID enhances the safety and performance of the standard PID control. Furthermore, LAPID-REC was found to demonstrate superior performance against existing techniques for closed-loop AP systems. This is especially observable under sensor noise, where, under a typical relative meal size between 75 and 125 $(g/100Kg)$, the proposed LAPID-REC approach can satisfy hypoglycemia safety property for 90% of the patients compared to lower than 50% of the patients for the other investigated techniques. These performance gains are achieved without demanding any further personalized tuning beyond the standard PID controller. The rest of the paper is organized as follows: Section II discusses the previous work, section III discusses the system architecture, section IV describes the proposed PID-based control strategies, section V discusses the experimentation and results, and section VI concludes the paper.

4.2 Related Work

The pioneering prototype of glucose feedback control released in 1964 (AH, 1964) inspired various research groups to develop realizations of instruments for closed-loop glucose systems. Later in the 1970s, the Biostator was launched as the first commercial blood glucose controller device as an outcome of the work described in (Pfeiffer, Thum, & Clemens, 1974). These early systems which used intravenous routes for sensing and infusion were bulky in general and were more suited to hospitalized patients.

On the positive side, those early contributions demonstrated the feasibility and motivated more developments in the technology towards realizing more practical AP systems. Later in the 1980s, more modalities were developed for glucose control such as the wearable (Shichiri, Yamasaki, Kawamori, Hakui, & Abe, 1982) and implantable (LeBlanc, Chauvet, Lombrail, & Robert, 1986) AP systems. These two modalities are mostly adopted in recent works.

Wearable AP systems use a subcutaneous glucose sensor and an insulin pump to regulate blood glucose levels. For implantable systems, they are instead implanted in the intraperitoneal space. This space is closer to the major vasculature and hence provides faster insulin absorption and glucose diffusion (Huyett, Dassau, Zisser, & Doyle, 2018). These shorter delays can improve blood glucose control especially after meals (Dassau et al., 2017). Nonetheless, implanting these devices incurs increased invasiveness as it requires surgery for placement, and requires the pump reservoir to be periodically refilled with insulin in the hospital (Renard, 2008).

The minimally-invasive subcutaneous route for glucose measurement and insulin delivery made it the favorite choice in most clinically-tested AP devices (Huyett et al., 2018). Unfortunately, subcutaneous glucose sensors suffer from discrepancies that generate inaccuracies, which originate from physiological kinetics, sensor calibration errors, sensitivity variations, as well as zero-mean random measurement noise (Facchinetti et al., 2013).

Various approaches have been proposed in the literature to filter and denoise glucose measurements such as median (Poitout et al., 1993), Kalman (Facchinetti, Sparacino, & Cobelli, 2009), and autoregressive moving average graph filter (Isufi, Loukas, Simonetto, & Leus, 2016). These filters are required to operate online and hence they have to automatically tune their parameters. More

discussion about this topic can be found in (Facchinetti, Sparacino, & Cobelli, 2020).

Starting from legacy on-off and multi-basal controllers towards the more advanced controllers, all AP systems strive to achieve reliable glycemic control and to eliminate or minimize hyper/hypo-glycemia events using various algorithms, supplementary tools, and devices. Several control techniques have been proposed and tested in the literature for glucose control. The most popular approaches are: Model Predictive Control (MPC), PID, and Fuzzy Logic (FL) control.

MPC for AP was proposed and tested in various $in\text{-}silico$ (Magni et al., 2007), clinical (Hovorka et al., 2011), and outpatient trials (Kovatchev et al., 2014). MPC depends on a model that describes the dynamics of the process being controlled. During operation, the controller predicts the status of the physical plant to generate actions that would minimize a specific cost function by solving a constrained optimization problem. Although this approach demonstrated promising results in terms of glycemic control, individualized parameter tuning is invasive and expensive (Messori, Incremona, Cobelli, & Magni, 2018). Moreover, running an online optimization problem on an embedded device leads to more computational costs. These challenges have to be resolved for MPC to be realized in commercial systems.

A pilot study to evaluate a FL control for closed-loop glucose system is presented in (Mauseth et al., 2013). In their system, the controller is composed of two components: an FL dosing component and a dosing personalization component. In the FL dosing component, the current glucose level, rate of change, and acceleration are used to form a 3-tuple. This 3-tuple is used to select a dosing rule from a matrix that is codified with the clinical expertise of collaborating physicians. These dosing rules are then defuzzified using the standard Mamdani process (Mamdani, 1974) to calculate the updated insulin dose. This dose is scaled by the dosing personalization component which combines the patient's total daily dose and a personalization factor. This personalization factor is tuned once per patient and assigned to a constant from a set of 11 possible values.

PID is a standard control technique that is among the most widely used in industrial applications. It is an attractive approach for glucose control due to its few parameters and simple structure (G. M. Steil, 2013). In fact, standard PID control strategy combined with an insulin-on-board estimate (Hu & Li, 2015) has been adopted in commercial glucose controllers such as Medtronic $670G$

([Lal et al., 2019](#)). This system which was FDA-approved in 2016 is the first commercial hybrid closed-loop AP. Yet it does not operate as a fully closed-loop system; as it requires the user to announce the intake of carbohydrates in their meals.

In a fully closed-loop AP system, the user input is not available which requires the control approach to face the challenge of regulating glucose levels during prandial and postprandial times. In particular, if the controller injects a lower amount of insulin, it might fail to catch up with the speed of the glucose spike. If instead, insulin is aggressively injected, this will result in postprandial hypoglycemia. This is attributed to slow onset and the delayed peak of insulin response ([Slattery, Amiel, & Choudhary, 2018](#)).

Some works have proposed using an implantable AP system as a means to mitigate the long time delays of glucose-insulin loop ([Huyett, Dassau, Zisser, & Doyle III, 2015](#)). Others have proposed dual hormone systems to overcome the postprandial hypoglycemia ([El-Khatib, Russell, Nathan, Sutherlin, & Damiano, 2010](#)). These systems administer the delivery of both insulin and glucagon to balance the postprandial response. However, this approach did not eliminate the hypoglycemia problem ([Peters & Haidar, 2018](#)). New advances in the production of fast-acting insulin analogs promise to overcome some of the challenges; as they require a shorter delay to be absorbed from the subcutaneous tissue into the plasma ([Schiavon, Dalla Man, & Cobelli, 2017](#)). However, more clinical evidence is required before using fast-insulin pumps in clinical practice ([Evans et al., 2019](#)).

In this research, we are trying to address some of the challenges in closed-loop glucose control by developing an enhanced control strategy. Our analysis assumes a typical insulin device, *i.e.*, a subcutaneous single-hormone pump injecting regular insulin.

4.3 System Architecture

4.3.1 Artificial Pancreas

AP system is composed of three functional components: a Continuous Glucose Monitor (CGM), a controller, and an insulin pump. An overview of the AP system is shown in Fig. [4.1](#). The CGM continuously measures the glucose concentration in the subcutaneous tissue G_s. The controller uses
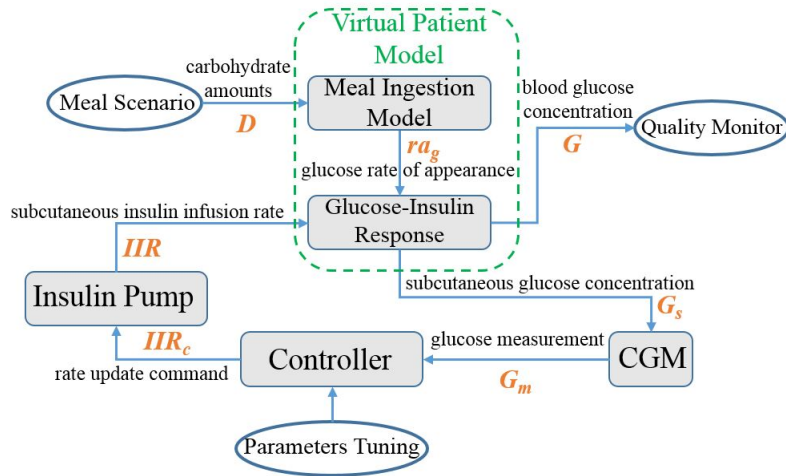
Figure 4.1: AP Overview

these measurements and analyzes them using a pre-configured control strategy to decide on the suitable amounts of insulin injection. The insulin pump receives control commands to modify its Insulin Infusion Rate (IIR) and conducts actuation injecting insulin in the interstitial space of the subcutaneous tissues.

4.3.2 Virtual Patient Model

Multiple models have been proposed in the literature to describe the physiological behaviors of glucose and insulin dynamics. Among those models, the model proposed by the Universities of Virginia and Padova (UVA/Padova) (Dalla Man et al., 2006, 2007; Man et al., 2014) is the only one approved by FDA to replace animal testing. This model provides publicly accessible parameters for a group of virtual patients and is commonly used to conduct $in\text{-}silico$ analysis before clinical trials. In this paper, this model and its parameters are used to evaluate AP systems.

In this model, the physiological behaviors describing the ingestion and absorption of meal carbohydrates (see Appendix A), as well as the interactions of glucose-insulin dynamics (see Appendix B) are represented by systems of Ordinary Differential Equations (ODEs) with different parameters for each patient. When a meal is consumed, the digestive system processes the carbohydrates which are then absorbed into the blood to result in elevation of glucose levels with the rate of appearance ra_g (A.5). Another input that affects the physiological dynamics is insulin which is infused by the

insulin pump at a rate of IIR. This exogenous insulin is absorbed into the body to regulate the blood glucose level G. The changes in blood glucose concentration will gradually affect the subcutaneous level G_s which is measured by the CGM to generate measurements G_m. These measurements are used by the control strategy to make decisions about the control actions.

4.3.3 Standard PID Control

The following equations describe the control method when implementing glucose PID control:

$$IIR_c(t) = IIR_0 + K_P\, e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt} \tag{4}$$

$$e(t) = G_t - G_m(t) \tag{5}$$

where IIR_0 is the bias, G_t is the target glucose concentration, $e(t)$ is the difference (error) between the current measurement and the target value, and K_x parameters are constants. The parameters $\{IIR_0, K_P, K_I, K_D\}$ are tuned using the *Ziegler and Nichols* open-loop tuning procedure (Ziegler et al., 1942) as described in (Haugen, 2010).

The control output IIR_c is constrained to non-negative values since the insulin pump cannot infuse negative rates. Also, due to the delayed insulin response, it is common for glucose controllers to ignore or restrict the integral term to avoid the over-administration of insulin which can result in postprandial hypoglycemia (Bequette, 2005).

4.4 Proposed Improved PID-Based Control Strategies

The goal of a glucose controller is to infuse the correct amounts of insulin promptly into the patient body to regulate the blood glucose levels. In case of inappropriate amounts of insulin or incorrect time of injection, it could induce hyper/hypo glycemic events that could result in health complications. In the following, the two proposed PID-based algorithms are presented and described.
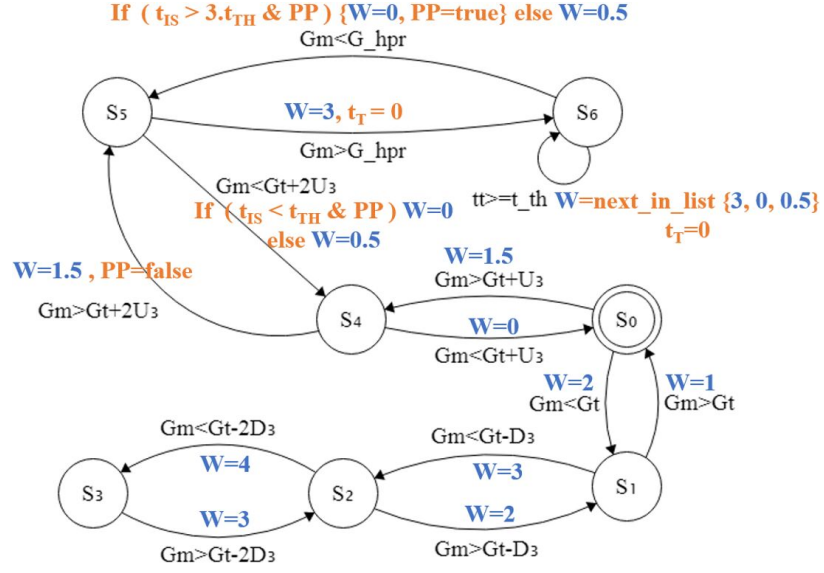
Figure 4.2: FSM for Adaptive Weight Calculation $\{t_{IS}$: time in state, t_{TH}: time threshold, G_{hpr}: hyperglycemia threshold, U_3: one third of the difference between G_t and G_{hpr}, D_3: one third of the difference between G_t and hypoglycemia threshold, PP: a flag to signify postprandial behavior$\}$

4.4.1 Proposed Adaptive Weighted PID

Although standard PID strategy provides effective glycemic control under smaller meal amounts, the control performance degrades as the amounts of meal carbohydrates are increased. To overcome this challenge, an AWPID control strategy for AP systems is proposed.

In this strategy, the proportional gain K_P in (4) is replaced by the following weighted gain:

$$K_P^w = W(S, \bar{S}, t_{IS}) \cdot K_P \tag{6}$$

Where W is the adaptive weight and $\{S, \bar{S}, t_{IS}\}$ are the current state, previous state, and the time in state, respectively. The weight W is calculated using the Finite State Machine (FSM) shown in Fig. 4.2 which accounts for the physiological characteristics of the glucose-insulin interactions. It should be noted that the parameters used in this approach are tuned empirically.

In the traditional clinical practice, it is recommended to have a pre-meal insulin bolus more than thirty minutes before consuming food (Slattery et al., 2018). The reason for this is to mitigate the slow onset and delayed peak response of insulin. These medical facts and practices have inspired the

weight selection for AWPID. The weights are selected to strengthen the early insulin delivery after meals and to constrain the delayed injections. In particular, under normal conditions the glucose level should be around the target value in the state S_0. When having a meal, the glucose level gradually increases towards states $\{S_4, S_5\}$. If the meal is large enough it increases the glucose above the hyperglycemia threshold and reaches to state S_6 where the weight will be initially elevated to 3. If it spends more than t_{th} in this state, the weight will change to zero for a similar amount of time before recovering back to positive values.

When the glucose level drops after spending enough time in hyperglycemia, a flag (PP) is set to signify the postprandial behavior and to make more conservative weight selection to avoid hypoglycemia (transition from S_5 to S_4). When the glucose drops below the target value of G_t, the weight is increased promptly to give a stronger response that aims at preventing hypoglycemia as shown in states $\{S1, S2, S3\}$. This is especially critical because any excessive insulin delivered cannot be compensated as the insulin pump is unable to deliver negative amounts of insulin.

4.4.2 Proposed Look-Ahead PID Controller with Retrospective Estimation Error Correction

Unlike the previous AWPID strategy where the proportional gain is weighted, this approach addresses the temporal behavior of glucose measurements by applying look-ahead techniques to mitigate the long delays in insulin action and glucose measurement. By doing so, the current glucose status in PID equations (4 & 5) is replaced by a future prospective estimated status. To reduce the drawbacks of estimation errors, these errors are observed retrospectively to add corrections towards eliminating the effect of error. In the rest of this section, the development of the proposed LAPID-REC strategy will be presented.

Starting from the classical PID control and by substituting for (5) in (4) and neglecting the integral term, glucose PID control can be described by:

$$IIR_c(t) = IIR_0 + K_P\,G_t - K_P\,G_m(t) - K_D\frac{dG_m(t)}{dt} \tag{7}$$

48
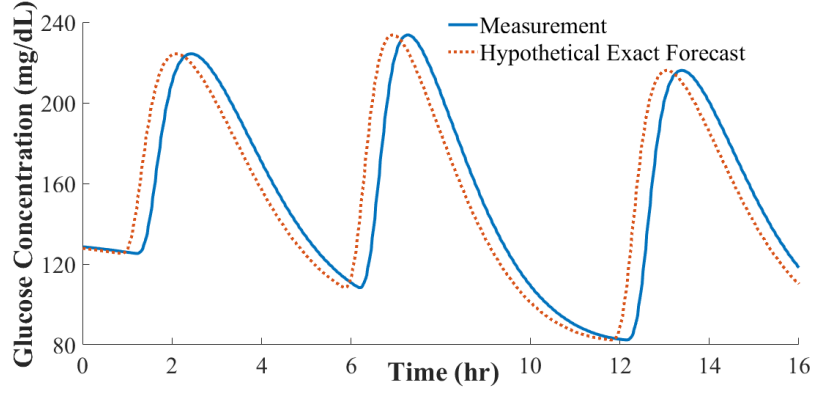
Figure 4.3: Hypothetical Ideal Forecast For Glucose Measurement

In this equation, the current glucose measurement $G_m(t)$ and its derivative are continuously observed to calculate the control action. For the case of LAPID, the current glucose measurement and its derivative are replaced by estimates of a future forecast of the measurement and its derivative, respectively. In the hypothetical ideal case, the estimate will be equal to the time-shifted version of the real measurement as shown in Fig. 4.3 for the glucose measurement. This hypothetical forecast is unattainable in reality but it will be used to visualize the realistic implementations of LAPID.

In reality, future measurements cannot be predicted exactly but can only be estimated with reasonable estimation error. One simple approach that is adopted in this work is by using second order extrapolation to calculate the future forecasts given the current glucose measurement $G_m(t)$, derivative $G'_m(t)$, and acceleration $G''_m(t)$. The estimated prospective glucose measurement G_{est} and derivative G'_{est} after look-ahead time T_{LA} are calculated using the following two equations:

$$G_{est}(t + T_{LA}) = \int_t^{t+T_{LA}} \int_t^{t+T_{LA}} G''_m(t)d\tau \tag{8}$$

$$G'_{est}(t + T_{LA}) = \int_t^{t+T_{LA}} G''_m(t)d\tau \tag{9}$$

A demonstration of glucose prospective estimation against the hypothetical ideal case is shown in Fig. 4.4. It can be seen that the LAPID generates notable estimation errors when the curve bends or reverses its orientation.

The estimation errors arising when applying LAPID can cause insulin over-administration and

49
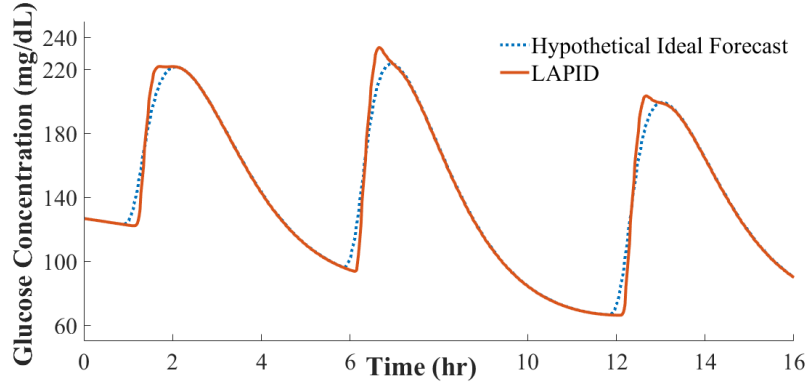
Figure 4.4: LAPID Glucose Estimation

result in undesirable hypoglycemia events. To overcome this shortcoming, retrospective correction is proposed. In this approach, prospective estimations are buffered and compared with the actual measurements when they appear. The estimation error in glucose measurement $E_{G'_m}(t)$ and derivative $E_{G'_m}(t)$ are calculated using the following two equations:

$$E_{G_m}(t) = G_m(t) - G_{est}(t) \qquad (10)$$

$$E_{G'_m}(t) = G'_m(t) - G'_{est}(t) \qquad (11)$$

A simple approach will be to use these estimation errors to correct the current control step and at the same time to equalize the previous disturbed estimate. Before applying them to (7), the estimates calculated in (8) and (9) are substituted with new corrected values as in the following equations:

$$G^{cor}_{est}(t + T_{LA}) = G_{est}(t + T_{LA}) + 2\, E_{G_m}(t) \qquad (12)$$

$$G^{cor'}_{est}(t + T_{LA}) = G'_{est}(t + T_{LA}) + 2\, E_{G'_m}(t) \qquad (13)$$

As can be seen in Fig. 4.5, this estimation curve generates undershoots following estimation error overshoots to equalize the errors. To do further investigation on the proposed methodology, we analyzed the corrected versions of the estimates calculated in (12) and (13) for the current control step.

50
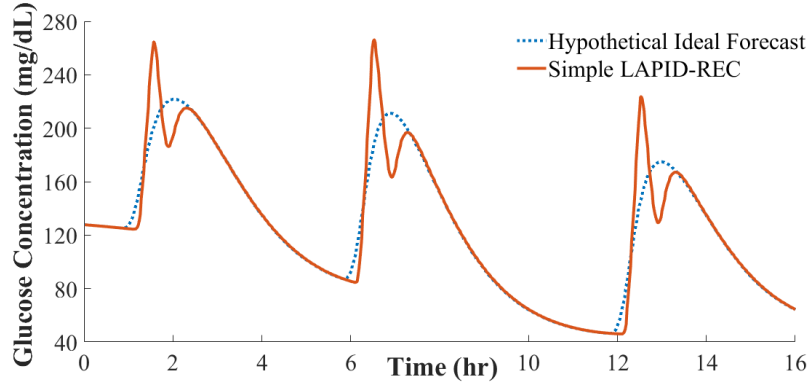
Figure 4.5: Simple LAPID-REC Glucose Estimation

But the original estimates calculated in (8) and (9) are to be buffered for future error calculation. The reason for doing so is to avoid the accumulation of error.

When the control strategy devise negative amounts of insulin, the controller will command insulin rate of zero value which effectively suspends insulin delivery until the glucose starts to increase again. During this period of insulin suspension, the control law becomes invariant for drop of glucose estimates. This dictates that corrections similar to the ones described in (12) and (13) will inevitably equalize low glucose estimates that did not affect the control law in the first place. To overcome this issue, we adopted a LAPID-REC that replaces (10) with:

$$E_{G_m}(t) = G_m(t) - G_{est}^{eff}(t) \tag{14}$$

where the term $G_{est}^{eff}(t)$ refers to the effective glucose value that is used to generate the control law such that low glucose values are scaled up to reach the zero point of control output. Also, (13) is replaced with:

$$G_{est}^{cor'}(t+T_{LA}) = min(G'_{est}(t+T_{LA}), G'_{est}(t+T_{LA}) + 2\, E_{G'_m}(t)) \tag{15}$$

where the $min(.)$ function aims at favouring lower glucose derivatives to avoid un-correctable excessive insulin delivery. Thus the LAPID-REC described by equations (14, 12, and 15) is adopted to conduct analysis in this paper. The simulation shown in Fig. 4.6 demonstrates this proposed LAPID-REC approach.

As a complementary step to demonstrate the relative effectiveness among the above mentioned

51
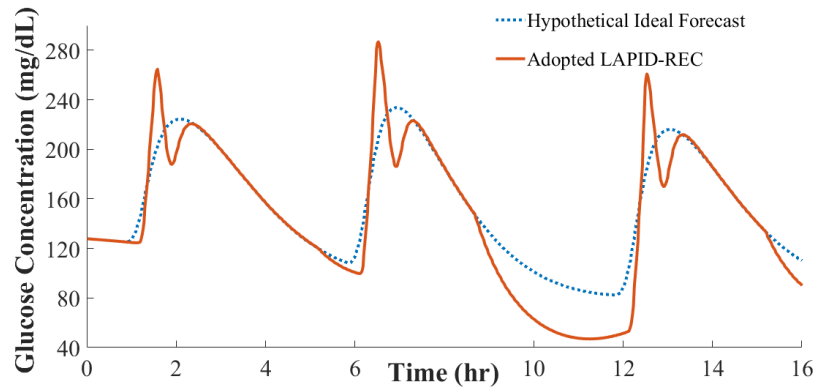
Figure 4.6: Adopted LAPID-REC Glucose Estimation
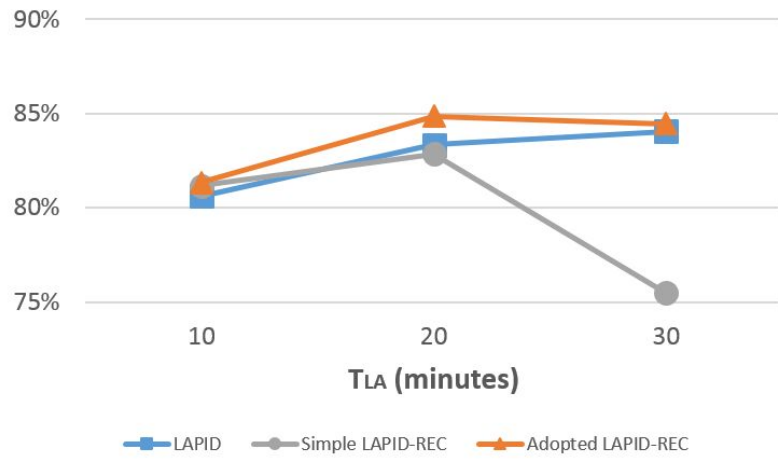
Figure 4.7: Expected Minimum Time in Target $G \, \epsilon \, [70, 180]$ for Different LAPID Schemes

LAPID-based approaches and to tune the adopted approach, A preliminary experiment was conducted on 30 virtual patients with various meal sizes. The resulting expected minimum time-in-target for the three approaches and for different values of look-ahead time T_{LA} are shown in Fig. 4.7. It can be seen clearly that using the adopted LAPID-REC approach with a look-ahead time of 20 minutes yields the best performance among the three LAPID-based approaches.

4.5 Experimental Results

4.5.1 Setup

A NPTA is created to model the various system components in UPPAAL-SMC tool (A. David et al., 2015). The feasibility of using this tool to investigate the safety of medical devices has been demonstrated in (Ma, Rinast, Schupp, & Gollmann, 2014). similar to (Alshalalfah et al., 2019), PTA components are constructed to model both cyber and physical components of the system. These PTAs are instantiated and parallel-composed to form the model for the whole system. The resultant model is then analyzed for safety and performance using SMC where properties specified in a stochastic temporal logic are analyzed using numerical and symbolic methods (Agha & Palmskog, 2018).

In this work, two stochastic safety properties $\{S_A, S_B\}$ are analyzed. These two safety properties are defined using the following two Metric Interval Temporal Logic (MITL) queries, respectively:

- **Pr[t≤1440] ([] G <= 300) ≥ 0.999**: which means: "*the probability of keeping the glucose level below the severe hyperglycemia threshold of* $300(mg/dL)$ *with confidence* 95% *throughout the test period of* $1440\ minutes\ (24\ hours)$ *should not be less than* 0.999."

- **Pr[t≤1440] ([] G >= 50) ≥ 0.999**: which means: "*the probability of keeping the glucose level above the severe hypoglycemia threshold of* $50(mg/dL)$ *with confidence* 95% *throughout the test period of* $1440\ minutes\ (24\ hours)$ *should not be less than* 0.999."

Also, 500 random simulations per patient and per meal carbohydrate range were performed for each control strategy to evaluate their performance. Thirty virtual patients (Man et al., 2014) were analyzed for safety and performance for 24 hours starting from 7:00 AM. Each virtual patient consumes three meals at 8:00 AM, 1:00 PM, and 7:00 PM. Each meal has a random amount of carbohydrates that is uniformly distributed between a minimum and a maximum value. The controller's target glucose level G_t is set to $120(mg/dL)$ and the time threshold t_{th} in Fig. 4.2 is set to $20(minutes)$. The setup of these experiments is summarized in Table 4.1.

Table 4.1: Experiments Setup

# of virtual patients	30
target glucose level	120 (mg/dL)
test duration	{7:00AM - 7:00AM} (24 hrs)
meal times	{8:00AM, 1:00PM, 7:00PM}
consumed meal carbohydrates	$D_{\{1,2,3\}}$ g/(100Kg of body weight), $D_i \in [D_{min}, D_{max}]$

In addition to the error-free normal scenarios, scenarios that involve meal inconsistency where a meal is missed with 50% chance were analyzed. Also, scenarios that engage noisy glucose sensors are analyzed. Even after applying denoising on the raw sensor measurements, the errors are reduced but not eliminated. Additive bounded noise is assumed that have two components: bias and zero-mean. The bias component accounts for deterministic errors caused by imperfections such as loss of calibration. The zero-mean component accounts for all other random noise sources.

The zero-mean random noise is set to random values from the bounded interval $\pm 5 (mg/dL)$. Also, for each run, the bias noise is set to a random value from the interval $\pm 10 (mg/dL)$. Moreover, due to the smoothing effect from denoising, the noise first and second derivatives are constrained to the bounded intervals $\pm 1 (mg/dL/s)$ and $\pm 0.1 (mg/dL/s^2)$, respectively.

Four control techniques are analyzed: PID, AWPID, LAPID-REC, and FL. For the FL controller, the approach described in (Mauseth et al., 2013) is used. When implementing FL controller, the personalization factor for each patient was tuned to generate the best results for that patient.

4.5.2 Results

Table 4.2 shows the performance comparison after running error-free simulations under different control strategies. In these simulations, the virtual patient consumes meals where each meal consists of random relative amounts of carbohydrates anywhere between 0 and 150 $(g/100Kg)$. It can be seen that the proposed LAPID-REC approach supersedes the other techniques in most of the performance figures. It can be noted that the results in Table 4.2 have large deviations from the mean values which is attributed to the large meal variations and the inter-patient variability.

The Control-Variability Grid Analysis (CVGA) of both proposed approaches (AWPID and

Table 4.2: Comparison of the Performance of Different Control Strategies {mean+std} Under Error-Free Simulations, $D_i \in [0, 150](g/100Kg)$

Percent Time Blood Glucose was ↓	PID	AWPID	FL	LAPID-REC
In Target ($70 \leq G \leq 180$)	84.43 ± 11.41	86.07 ± 9.19	84.75 ± 10.28	**89.01 ± 7.11**
In Tight Target ($80 \leq G \leq 140$)	68.00 ± 13.01	65.95 ± 11.07	57.42 ± 25.27	**74.15 ± 8.74**
Above Target ($G > 180$)	11.19 ± 6.46	11.70 ± 6.64	14.41 ± 9.93	**10.85 ± 6.92**
Below Target ($G < 70$)	4.38 ± 7.45	2.23 ± 5.61	0.84 ± 2.64	**0.14 ± 1.01**
In Severe Hyperglycemia ($G > 300$)	0.235 ± 0.875	0.307 ± 1.007	0.068 ± 0.430	**0.049 ± 0.304**
In Severe Hypoglycemia ($G < 50$)	0.900 ± 3.219	0.325 ± 2.148	**0.003 ± 0.118**	0.016 ± 0.303

LAPID-REC) is shown in Fig. 4.8. Following the procedure described in (Magni et al., 2008), each point on the graph represents the upper and lower bounds of the 95% confidence interval for a specific patient. LAPID-REC was able to maintain 97% of the patients in the $A + B$ macrozones. On the other hand, AWPID had only 73% of the patients in the $A + B$ zones and had 27% of the patients in the $Lower\ D$ Zone.

To understand how the control strategies perform with various meal scenarios, different levels of meal consumption were analyzed separately. The average expected minimum time in target ($70 < G < 180$) is shown in Fig. 4.9. For the error-free normal case, the expected value is above 90% for smaller meals under all control strategies. As the amounts of carbohydrates per meal increase, the time-in-target drops with varying rates for the different strategies. For extremely large meals, the average expected value drops to 72% at best under LAPID-REC and 55% at worst under PID. The average value over the different meal sizes is 85% under LAPID-REC, 80% under AWPID, 78% under FL, and 77% under PID, respectively.
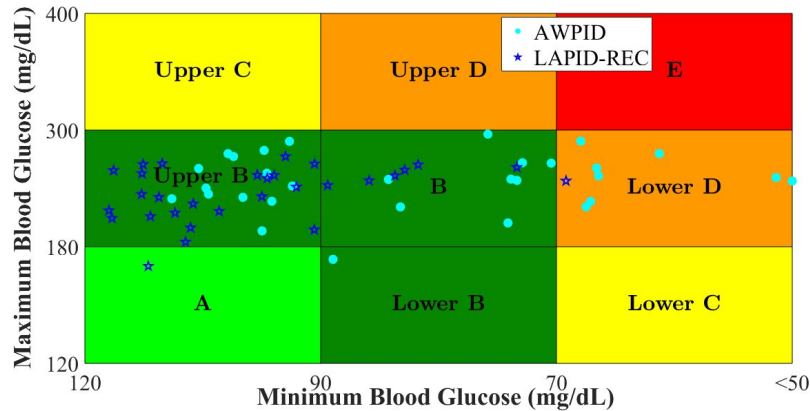
Figure 4.8: Control Variability Grid Analysis of AWPID and LAPID-REC

(a) Error-Free Normal   (b) Under Meal Inconsistency   (c) Under Sensor Noise
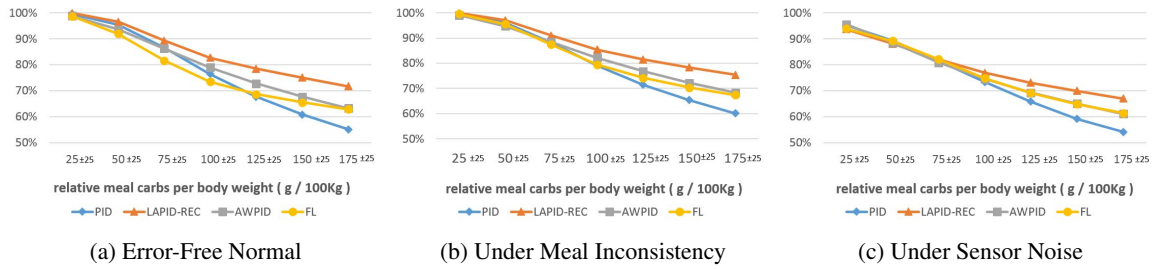
Figure 4.9: Expected Minimum Time in Target $G \in [70, 180]$

When meal inconsistency takes effect with a chance of missing a meal, the average expected minimum time in target increases in general. As can be found in the middle graph of Fig. 4.9, the increase ranges between 2% for LAPID-REC and 5% for FL. The opposite happens when sensor noise is considered as in the right graph of Fig. 4.9. In this case, the average expected time-in-target drops with ratios between 1% for FL and 7% for LAPID-REC. Overall, Fig. 4.9 shows that AWPID outperforms the other techniques in terms of the average expected minimum time-in-target, especially for larger meal sizes.

The average expected maximum time above target ($G > 180$) is shown in Fig. 4.10. For the error-free normal case, the expected value ranges between less than 5% for the smaller meals and 9% for the large meals. The only exception is FL which reaches up to 16% above the target range for the largest meal scenario.

(a) Error-Free Normal   (b) Under Meal Inconsistency   (c) Under Sensor Noise
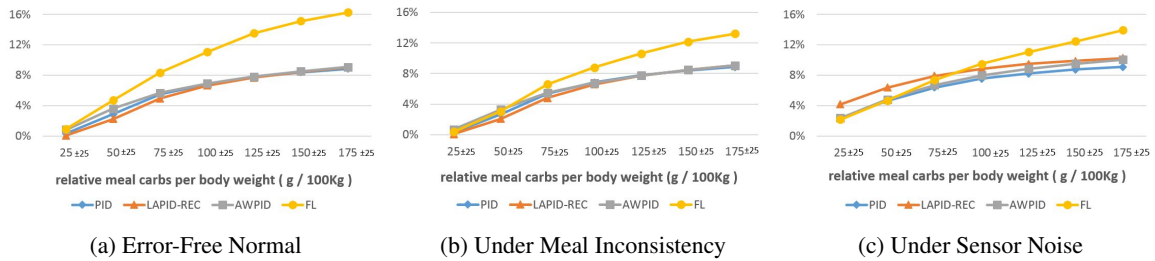
Figure 4.10: Expected Maximum Time Above Target ($G > 180$)

Under meal inconsistency conditions, the expected time above target decreases on average by a ratio of 3% under AWPID and 22% under FL as can be seen in the middle graph of Fig. 4.10. The situation is different when the sensor noise is in effect as shown in the right graph of Fig. 4.10. In this case, except for FL where the average expected time above target decreases by 13%. The

other strategies suffer from increases ranging between 5% for LAPID-REC and 18% for AWPID. Overall, Fig. 4.10 shows that all control strategies follows almost the same behavior in terms of the average expected maximum time above target except FL which spends more time above the target.

The average expected maximum time below target ($G < 70$) is shown in Fig. 4.11. For the error-free normal case on the left, the expected value for smaller meals is below 2% for PID, 1% for AWPID, and 0% for both FL and LAPID-REC. For the largest meal sizes, the time below target increases to 12% and 7% for PID and AWPID, respectively. Still, the expected value remains below 2% and 1% for LAPID-REC and FL, respectively.

(a) Error-Free Normal   (b) Under Meal Inconsistency   (c) Under Sensor Noise
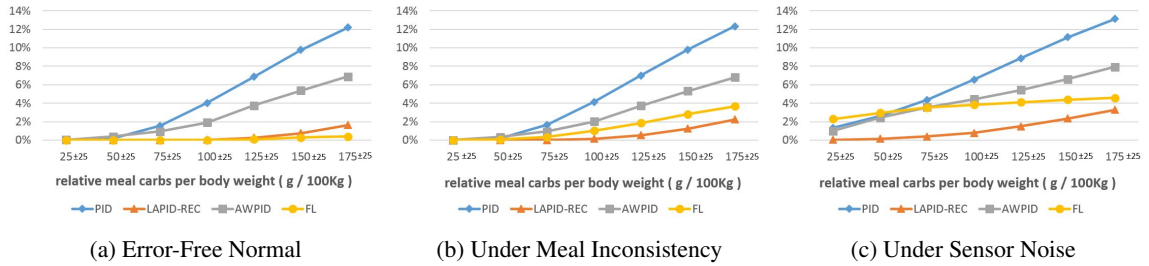
Figure 4.11: Expected Maximum Time Below Target ($G < 70$)

When affected by disturbance of a meal inconsistency that involves a chance for a missed meal, the average expected time below target continue to the same levels for PID and AWPID, increases to above 2% for LAPID-REC, and to around 4% for FL as shown in the middle of Fig. 4.11. On the other side, when affected by sensor noise as in the right graph of Fig. 4.11, these expected values increase towards 13% for PID, 8% for AWPID, 3% for LAPID-REC, and 5% for FL, respectively. Overall, Fig. 4.11 shows that although AWPID reduces the average expected maximum time below target when compared with PID. It is superseded by the FL and LAPID-REC. Under error-free normal scenario, FL performs better than LAPID-REC. But when either sensor noise or meal inconsistency takes effect, LAPID-REC supersedes FL in avoiding time below target.

Fig. 4.12 shows the SMC analysis results on the number of patients satisfying safety property S_A. This safety property is linked to the ability of the control strategy to avoid severe hyperglycemia ($G > 300$) episodes. For the error-free normal case on the left, all controllers tend to satisfy S_A for almost all patients for smaller meals. For larger meals, the number of patients satisfying S_A drops.

For the largest meal, the number drops down to 11 patients at best for LAPID-REC and 7 patients at worst for AWPID. For all control strategies the average number of patients satisfying S_A over the various meal amounts range between 24 for LAPID-REC and 21 for AWPID.
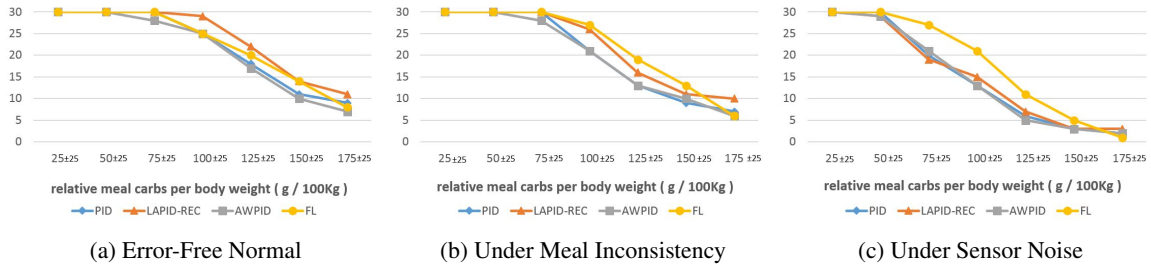
Figure 4.12: Number of Patients with Safety Properties (S_A) Satisfied

Under meal inconsistency, the number of patients satisfying S_A does not change for smaller meals but varies with different levels for larger meals. The drop in the average number per control strategy over all meal amounts varies between two patients for PID and LAPID-REC, one patient for AWPID, and no noticeable drop in FL as can be seen in the middle of Fig. 4.12. When affected by sensor noise, the number of patients satisfying S_A drops significantly with average drop ratios ranging between 19% for FL and 31% for LAPID-REC. The peak drop of 11 patients occurs under an average meal size for LAPID-REC as can be seen in the right graph of Fig. 4.12. Overall, under error-free normal scenario, LAPID-REC performs better in terms of S_A safety over the other techniques. But under sensor noise or even meal inconsistency, FL performs better than the other techniques. AWPID and PID have comparable performance in avoiding severe hyperglycemia.

Fig. 4.13 shows the SMC analysis results on the number of patients satisfying safety property S_B. This safety property is linked to the ability of the control strategy to avoid severe hypoglycemia ($G < 50$) episodes. For the error-free normal case on the left, FL was found to satisfy S_B for all patients under all meal amounts. LAPID-REC satisfied S_B for the patients under small and medium meal sizes with two patients violating safety under the largest meal sizes. Under the largest meal size, AWPID satisfied S_B for only 20 patients and PID satisfied the safety property for only 13 patients.

When affected by meal inconsistency as shown in the middle of Fig. 4.13, the average number

(a) Error-Free Normal (b) Under Meal Inconsistency (c) Under Sensor Noise
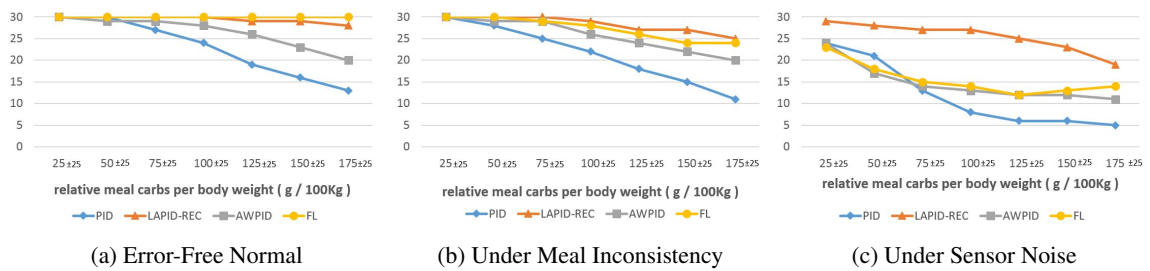
Figure 4.13: Number of Patients with Safety Properties (S_B) Satisfied

of patients satisfying S_B over all meal sizes drops by almost one patient for PID, AWPID, and LAPID-REC, except for FL in which the average number drops by 3 patients. When affected by glucose sensor noise, all control strategies suffer from less patients satisfying S_B. As can be seen in the right graph of Fig. 4.13, even for the smallest meal size the numbers are between 23 and 24 except for LAPID-REC in which 29 patients satisfy S_B. Under the largest meal size, the number drops to 19 for LAPID-REC, 14 for FL, 11 for AWPID, and 5 for PID, respectively. The average value over the various meal sizes is 12 for PID, 25 for LAPID-REC, 15 for AWPID, and 16 for FL controller. Overall, under error-free normal scenario, FL performs ideally to avoid severe hypoglycemia and LAPID-REC performs close to ideal behavior. AWPID shows less performance when compared to these two techniques but still supersedes PID. When affected by sensor noise, LAPID-REC supersedes all the other techniques with an average of 10 more patients satisfying S_C as compared to FL. The performance of FL under sensor shows similar performance as AWPID in terms of avoiding severe hypoglycemia.

4.6 Conclusion

When compared with other closed-loop techniques such as FL controller and model predictive controller, PID controller is favored for industrial artificial pancreas systems due to its few parameters and simplicity. To overcome its limitations, industrial systems implement a hybrid-closed loop controller where the user informs the controller about consumed meals. In this paper, we propose improved PID-based strategies towards a realizable closed-loop artificial pancreas. By design, the

proposed improvements require no personalized parameter tuning beyond the standard PID controller's tuning.

In this paper, we have proposed two improved PID-based control strategies for the closed-loop artificial pancreas. In the first approach, an adaptive weight is applied to the proportional gain of the PID controller. The weights are selected using a finite state machine that accounts for the short term history of glucose measurements. In the second approach, the current glucose measurement and derivative are substituted by prospective estimates of glucose measurement and derivative with retrospective estimation error corrections.

The proposed strategies were evaluated on 30 virtual patients from an FDA-approved model with different meal scenarios under error-free normal conditions, meal inconsistency, and glucose sensor noise. The results have shown that the adaptive weight clearly reduces the hypoglycemic events when compared against PID. However, the performance of the adaptive weighted approach does not exceed that of a carefully-tuned fuzzy logic controller.

In the second approach, a look-ahead PID controller is proposed. By doing so, the current measurement and its derivative in the PID equation are replaced by prospective estimates of a future value. To efficiently deal with the changes of the glucose curve, retrospective estimation error correction is applied. The results have shown that this proposed approach outperforms all the other analyzed techniques in avoiding hypoglycemia episodes. Although it did not completely eliminate the safety issues, it reduced them significantly without adding more personalized tuning. This is especially observable under sensor noise where, for example, under a typical relative meal size between 75 and 125 $(g/100Kg)$, LAPID-REC can satisfy hypoglycemia safety property for 90% of the patients compared to lower than 50% of the patients for the other investigated techniques.

Chapter 5

Article II: A Framework for Modeling and Analyzing Cyber-Physical Systems using Statistical Model Checking

Authors: Abdel-Latif Alshalalfah, Otmane Ait Mohamed, Samir Ouchani

Abstract: The trustworthiness of a cyber-physical system is essential for it to be qualified for utilization in most real-life deployments. This is especially critical for systems that deal with precious human lives. Although these safety-critical systems can be investigated using both experimental testing and model-based verification, accurate models have the potential to permit risk-free mimicking of the system behavior even in the most extreme scenarios. To overcome the CPS modelling and design challenges, the INCOSE/OMG standard System Modeling Language (SysML) is utilized in this work to accurately specify cyber-physical systems. For that, a bounded set of SysML constructs are defined to precisely capture the semantics of continuous-time and discrete-time system behaviors. Then, the SysML constructs are substituted by developing a new algebra, called Enhanced Activity Calculus (EAC). So, EAC helps construct equivalent priced timed automata models by developing a new systematic procedure to correctly translate the SysML models into the statistical model checking tool UPPAAL-SMC inputs. The latter checks whether the system is correct and safe or not. Moreover, the soundness of the developed translation mechanism has been proved and

its effectiveness has been shown on a real use case, namely the artificial pancreas.

5.1   Introduction

Whether human-operated or autonomous, embedded systems are designed to improve the quality of life for people. From embedded computing to distributed systems, Cyber-Physical Systems (CPS) refer to computing systems that interact with control and management objects (Skorobogatjko et al., 2014). As technology advances, CPS is being used in a wide range of applications (Kurzweil, 2004). With the reduction in size and cost of hardware, along with accelerating innovation and advancement in sensor and computational technologies, CPS has been able to spread to all types of applications. Through horizontal expansion, CPS has gained popularity in all types of application. Also, CPS flourished vertically to find a foot in more complex and dependable applications From daily applications, the various success stories have encouraged designers to develop CPS for autonomous control compared to the early systems which required some degree of human interaction (Han et al., 2022; Montanaro et al., 2019). Nowadays, wireless body area networks are utilized to connect devices that observe the status of the physiological dynamics (Y. B. David et al., 2021). As a result, health conditions can be monitored and treated in a timely manner. Patients with chronic diseases will particularly benefit from this. For example, with around half a billion diabetes worldwide (Ogurtsova et al., 2017), an automatic glucose controller is necessary for them to live a normal life while still avoiding the health complications related to their situation.

In order to get approval certificates from the appropriate authorization entities, these systems must prove their safety and robustness under all scenarios (Cummings & Britton, 2020). However, for real-life deployments, only qualified systems must meet these safety requirements. From the first prototype to the final fabricated product, verifying the safety of CPS is a vital step in the development process. The system-level analysis provides feedback early in the design process, and by identifying safety issues early, time and resources are not wasted (Koong, Ng, Ramayah, Koh, & Yoong, 2021). Additionally, the system-level analysis helps understand CPS limitations and define the requirements of CPS components for safe operation. Furthermore, CPS can be verified under extreme scenarios that would be impossible to conduct in real life without taking extraordinary risks

by using appropriate realistic models.

Analyzing systems at the system level is either accomplished through simulation testing or through formal methods. In the former approach, specific input scenarios can be used to evaluate CPS behavior. Yet, it does not give confidence on the state space coverage. On the other hand, formal methods such as model checking (Clarke Jr et al., 2018) provide exhaustive coverage for the whole state space. Unfortunately, formal techniques do not scale well for realistic hybrid systems and suffer from the infamous state space explosion problem (Godefroid, 1996).

As a compromise between these two approaches, Statistical Model Checking (SMC) can be used for verification. Although it does not provide exhaustive coverage for the state space, SMC can be used to introduce statistical guarantees for safety properties with feasible computational resources. In a nutshell, the following are the main contributions of this work.

- Proposing a novel systematic procedure to capture the semantics of SysML-based diagrams and to construct its equivalent PTA models for SMC analysis.

- The effectiveness of the proposed framework to analyze a medical CPS is demonstrated on an artificial pancreas case study. In particular, the safety of the system is verified using SMC to evaluate the ability of three control configurations to mitigate message errors.

Below is an outline of the remainder of the article. The literature review is presented in Section 2, and then Section 3 demonstrates, through an artificial pancreas example, the SysML graphical and textual modeling. Afterwards, Section 4 introduces the new proposed automatic construction of equivalent Priced Timed Automata (PTA) models and proves the soundness of the developed approach. Section 5 illustrates the experiments conducted for model validation and safety verification procedure by an example experiment, and section 6 concludes the article.

5.2 Literature Review

With the growing demand for CPS applications, several research works have investigated the verification and safety analysis problems related generally to CPS. Based on our surveyed initiatives, we have identified two main categories: *Formal verification* and *Simulation* based approaches.

5.2.1 Simulation based approaches

Even before the advent of modern computer systems, the term *Simulation* is known as the process of designing a model of a real system to conduct experiments (Shannon, 1975). These experiments aim at understanding the system's behavior or evaluating a strategy associated with the system. Simulation software tools have flourished with the advent and availability of low-cost computational systems.

Liu et al. (Liu et al., 2017) have used the open-source toolkit MATSim (W Axhausen et al., 2016) to investigate large-scale transportation patterns for shared autonomous vehicles. In their work, agent-based modelling is applied to estimate mode choices between human-driven vehicles, shared autonomous vehicles, and public transit. Following a cost function that takes into account, the out-of-pocket, the trip time, and the waiting time, each driver chooses one of the three options of travel mode. The analysis is done for different fare levels, demographic settings, and shared autonomous vehicles availability to give implications on sustainability.

In (Lakshmanan et al., 2019), an assessment of the safety of leader-follower configurations for autonomous radar semi-trucks is made based on different environmental conditions. The simulation model is developed with the commercial platforms AmeSim, PreScan, and Matlab-Simulink to study the effect of environmental conditions on safety margins in semi-truck convoy platooning. The autonomy in their simulated vehicles is enabled by adopting sensors for radar, global positioning systems, and short-range inter-vehicle communication.

Instead of fully autonomous vehicles, the work in (Arnaout & Arnaout, 2014) addressed semi-autonomous vehicles implementing adaptive cruise control coexisting with regular vehicles and trucks. The vehicles enter the four-lane highway with a user-predefined arrival rate in the microscopic Java-based F.A.S.T. traffic simulator. Their findings show that a high penetration of semi-autonomous vehicles can increase traffic performance, especially under high traffic conditions.

Connected and autonomous vehicles and their impact on road safety are discussed in (Papadoulis et al., 2019). Initially, the simulation software VISSIM is utilized to study a test-bed that mimics a three-lane motorway with traffic statistics measured from a real one in England. A lateral and longitudinal control algorithm is then tested for its ability to reduce traffic conflicts at different

market penetration rates.

From a healthcare perspective, a falsification approach is presented in (Cameron et al., 2015; Sankaranarayanan et al., 2017) to simulate and verify the artificial pancreas controller in a simulation environment. The S-Taliro tool which applies falsification simulations terminates with either finding a safety violation or failure to find, without the explicit guarantee that such one does not exist. Instead, the tool uses robustness metric to predict the distance between simulation outcomes and safety margins.

5.2.2 Formal based approaches

Unlike the numerical simulation approaches which mimic the behavior of real systems, formal methods apply analytical reasoning to derive mathematically-proven properties that characterize the system behavior. These characteristics are not always attainable, but when achieved they provide guaranteed outcomes which is an asset that helps verify safety-critical systems.

In (Kekatos et al., 2017b), piece-wise affine hybrid automata was used to analyze the wind turbine dynamics in SpaceEx verification platform (Frehse et al., 2011). Even though Kekatos et al. reduced some blocks for better scalability, the resulting model contained around 16 million locations, which would hinder the ability to analyze more elaborate systems. However, classical hybrid automata (HA) tools and methodologies suffer from this limitation (Schupp et al., 2015).

The problem of formally analyzing a swarm of robots is handled by Schupp et al. (Schupp et al., 2022). The cooperative decentralized robots are modeled as a hybrid system and investigated by *flowpipe* analysis where the sets of reachable states are iteratively over-approximated (Frehse, 2015). Although the work in (Schupp et al., 2022) deals with a simple model of distributed synchronization, it still causes some scalability challenges that are partially encountered by compositional analysis and optimized transition emulation.

Using a combination of simulations and formal analysis, (Pajic et al., 2012) examines patient-controlled analgesia's safety. So, to analyze the resulting CPS, its detailed behavior is modeled in Simulink. Then, to qualify the CPS for model checking, the continuous dynamics are abstracted away from the system model and then replaced by simple timing constraints with the target to be analyzed in UPPAAL model checker (Behrmann et al., 2004). Additionally, UPPAAL is also

used in (Jiang et al., 2012) to verify control algorithms in a dual chamber implantable pacemaker. Meanwhile, a timed automata representation of the heart and the pacemaker are used to specify the ability of the algorithms to avoid unsafe regions of the state space. The proposed approach covers the whole state space, yet only the state space that is modeled. Thus, this excludes certain control and physiological behaviors that are beyond the expressive power of the modeling language and the computational feasibility of the verification technique. In fact, these behaviors can be skipped in some systems but are essential to correctly analyze hybrid systems with continuous-time variables.

5.2.3 Statistical model checking based approach

SMC consists of observing a number of simulation runs or system executions and using statistical methods to reason about formal properties (Legay et al., 2019).

After some preliminary works such as the hypothesis testing of modal properties in process algebra (Larsen & Skou, 1991), initial results for SMC had witnessed progress since 2002 (H. L. Younes & Simmons, 2002) with the corresponding term introduced for the first time in 2004 (Sen et al., 2004). Reasoning about reachability problems with SMC algorithms provides mainly guarantees on the probability error bound. Depending on the type of reachability expression being dealt with, the error bound can be calculated by utilizing the appropriate classical mathematics such as Monte Carlo with Chernoff-Hoeffding error bounds (Hérault et al., 2004; Okamoto, 1959) or hypothesis testing using Wald's sequential analysis (Wald, 2004).

Different tools exist that implement SMC algorithms such as PRISM (Kwiatkowska et al., 2011), UPPAAL-SMC (A. David et al., 2015, 2011), BIP (Mediouni et al., 2018), and Ymer (H. L. S. Younes, 2004). Since their inception, SMC tools have been utilized to study many discrete-time and continuous-time systems. To list a few: airplane cabin communication system (Basu et al., 2012), distributed sensor network (Lekidis et al., 2015), energy-aware house heating (A. David, Du, et al., 2012), biological mechanisms of the genetic oscillator (A. David, Larsen, et al., 2012), real-time streaming protocol (Ouchani et al., 2012), artificial pancreas (Alshalalfah et al., 2019; Alshalalfah, Hamad, & Mohamed, 2021), anesthesia control (Alshalalfah, Bany Hamad, & Ait Mohamed, 2020), and coordinated emergency braking system (Alshalalfah & Mohamed, 2020).

5.2.4 Model Construction

In order to analyze the system, it is necessary to first convert the specifications into the modeling language used by the analysis tool. Furthermore, an adequate level of expertise is required to model the system properly when done manually. Furthermore, formal modeling languages tend to be more error-prone due to their low readability. Therefore, the need arises to facilitate the process of constructing formal models by automatically translating high-level models that incur better readability.

In (Kekatos et al., 2017b), the system modeled in Simulink is translated into SpaceEx modeling language in four steps. After the Simulink model is modified to comply with the verification standards, the tool SL2SX (Minopoli & Frehse, 2016) is employed to handle the main translation step and construct a SpaceEx model. Afterwards, compositional syntactic hybridization (Kekatos et al., 2017a) and validation are conducted to achieve a model ready to be analyzed.

An approach to transform Simulink models into UPPAAL-SMC is proposed in (Filipovikj et al., 2016). The work is employed on two automotive use cases for brake-by-wire and an adjustable speed limiter. The Simulink models are first reduced by the flattening procedure. Then, each block is replaced by an equivalent timed automaton composed of three locations: start, offset, and operate. Still, their approach does not implement complex real-valued blocks in UPPAAL-SMC but addresses them in Simulink instead.

Instead of commercial modeling tools, System Modeling Language (SysML) (Specification, 2007) can be used to specify CPS. SysML is the defacto standard modeling language for systems engineering with rich semantics and expressive power sufficient to describe system structures and behaviors at various levels of abstraction (Holt & Perry, 2008). Ouchani et al. (Ouchani et al., 2014) constructed probabilistic automata by converting SysML models. The resulting models were incurred to analyze security properties of the real-time streaming protocol using the probabilistic model checker PRISM (Kwiatkowska et al., 2011).

Compared to the studied initiatives, the main objective of this work is to develop a framework that enables efficient modeling and analysis for CPS. The proposed framework takes system behavior specified using SysML diagrams as input. The novelty of this proposed work is summarized by

the following contributions.

- Defining a bounded set of SysML constructs that are sufficient to capture the behaviors of the CPS discrete-time and continuous-time dynamics.

- Defining textual specification language for SysML by extending the semantics initially developed in (Debbabi, Hassaine, Jarraya, Soeanu, & Alawneh, 2010; Ouchani et al., 2014).

- Proposing a novel systematic procedure to transform the SysML behavioral specifications into PTA. Compared with the previous works that processed models specified in the commercial tool Simulink (Filipovikj et al., 2016; Kekatos et al., 2017b; Minopoli & Frehse, 2016) or did not support modeling physical processes (Filipovikj et al., 2016; Ouchani et al., 2014), this new proposed approach defines a systematic procedure to process SysML models for the CPS and to construct an equivalent PTA model for analysis by supporting more features and expressive powers to specify physical properties like *time*, *rate* and *real-numbers related measurements*.

- The soundness of the proposed approach has been proven and its effectiveness to analyze CPS is demonstrated on an artificial pancreas system.

5.3 The Proposed Framework

Fig. 5.1 provides a brief overview of the proposed framework that runs on the following steps.

① The process starts with the initial identification of the CPS to explore the nature of its application. This step helps specify the system's requirements including the safety properties that have to be met.

② The topology of the system is defined by specifying the functional components of the system which are used to create the SysML block definition diagram. Also, the interactions between the CPS components are used to define the SysML flow diagram. Integrated CPS are formed from continuous real-time components describing physical processes and discrete-time components describing cyber processes.
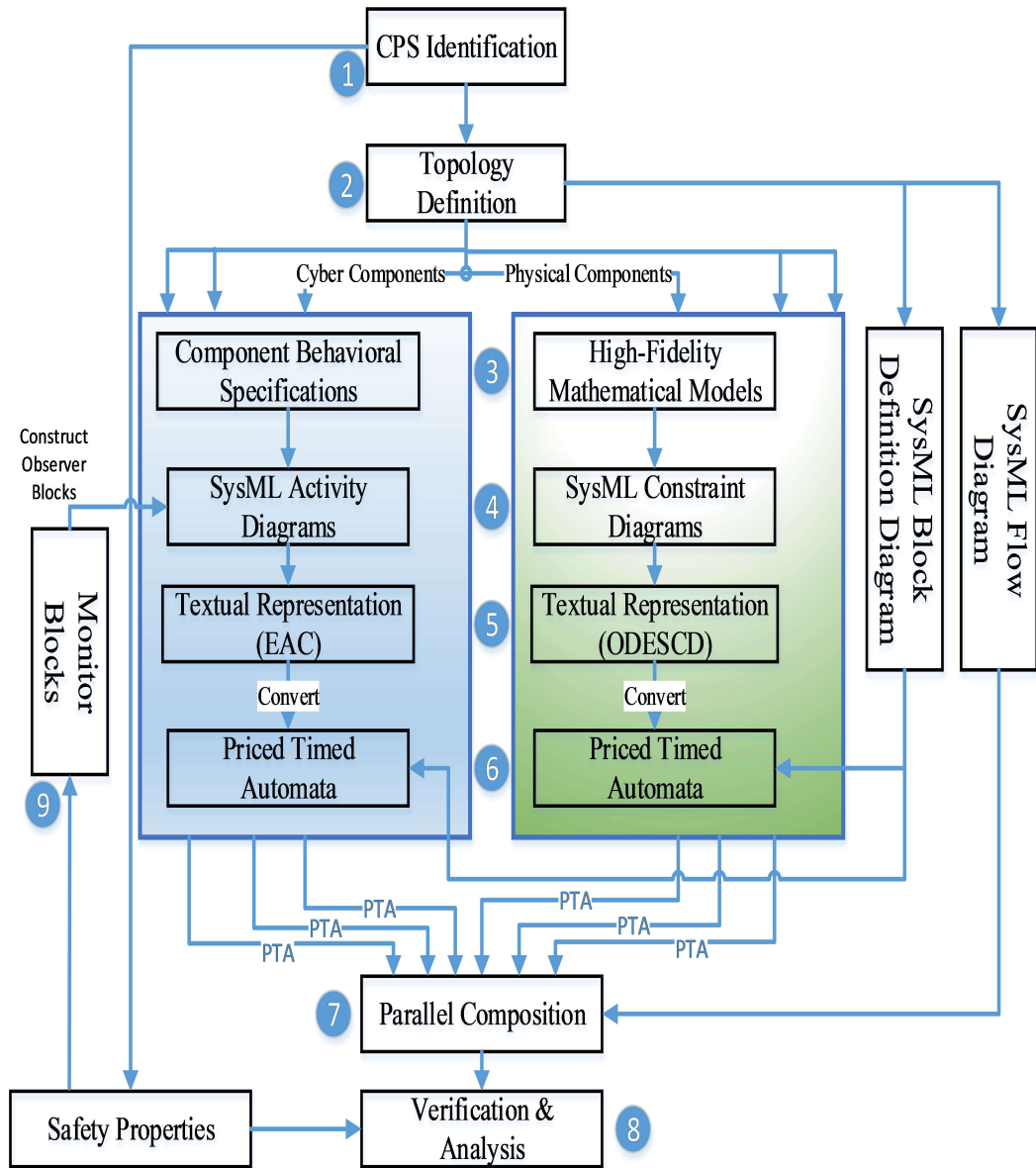
Figure 5.1: The Proposed Framework Workflow

③ By relying on the existing topologies, behavioral models for the physical components are imported in the form of Ordinary-Differential Equations (ODE). Similarly, the cyber components of the system are imported from design specifications in the form of discrete variations.

④ Physical and cyber components are represented using SysML parametric constraint diagrams and activity diagrams, respectively.

⑤ To automate further processing, each of the SysML diagrams are written in textual format. For a constraint diagram describing the physical dynamics, the representation is done using the proposed syntax named Ordinary-Differential Equations of SysML Constraint Diagram (ODESCD). For an activity diagram describing a component's behavior, the representation is done using the proposed Enhanced Activity Calculus (EAC).

⑥ A new systematic algorithm is proposed to convert ODESCD and EAC blocks into equivalent PTA blocks. The SysML block definition diagram, describing the system's structure, specifies the input/output connections of each PTA block.

⑦ The various PTA blocks for physical dynamics and component behaviors are mapped as described by the flow diagram. The parallel composition of all the PTA blocks form the integrated CPS that is processed.

⑧ The analysis tool UPPAAL-SMC is used to analyze the system and verify the safety properties. The framework is demonstrated on an artificial pancreas system alongside a proposed representation of continuous-time and discrete-time dynamics.

⑨ For safety properties that are beyond the expressive power of the query language in use, dedicated monitor blocks are employed to observe specific phenomena. An observer block is then added to the system by developing a behavioral model for that block which is specified using a SysML activity diagram. The new block is then processed as component of the CPS to construct an equivalent PTA model. By adding these monitor blocks, more complex safety properties are simplified and expressed easily in order to be examined for safely.

5.3.1 SysML Graphical and Textual Modeling

SysML for continous-time dynamics

The dynamics of physical processes describe the flow of physical quantities in the real world. These quantities are represented by real-valued real-time variables where the derivative of a variable is equivalent to the change on its associated physical quantity. Therefore, it is common for continuous-time dynamics to be specified by a system of ODEs. SysML constraint diagrams can be used to model ODEs.

Notation 1 (ODE of SysML Constraint Diagram (ODESCD)). ODESCD is defined as a tuple $(X, X^0, K, P, R, F, I, O)$, where:

- X is a set of real-time real-valued differentiable variables,

- X^0 is a set of initial values,

- K is a set of real-valued equation coefficients,

- P is a set of constant real-valued parameters,

- R is a set of real-time real-valued variables,

- $F(X, R)$ is a set of real-valued functions,

- $I \in X \cup K \cup P \cup R$ is a set of input variables, and

- $O \in X$ is a set of output variables.

Definition 1 (Semantics of ODESCD). Let $(X, X^0, K, P, R, F, I, O)$ be a ODESCD, its semantics is defined as the dynamics of a physical system described by a set of ODEs as follows (in this context a subscript in the form of $a_1 \times a_2$ indicates the matrix dimensions).

$$X'_{n \times 1}(t) = K_{n \times n}(X, P, R, t) \, X_{n \times 1} + F_{n \times 1}(X, R) \tag{16}$$

$$X_{n \times 1}(t = 0) = X^0_{n \times 1} \tag{17}$$

$X'_{n\times 1} = [x_1 \, x_2 \ldots x_n]$ is the set of differential variables to be solved, $X^0_{n\times 1} = [x^0_1 \, x^0_2 \ldots x^0_n]$ is the set of initial values for the differential variables, $K_{n\times n}(X, P, R, t)$ is the set of differential equation coefficients which can be constants or functions of constant parameters, real-time variables or time, P is the set of additional constant parameters for the equation, R is the set of additional real-time variables, $F_{n\times 1}(X, R)$ is the additional terms of the ODE, $I \in X \cup K \cup P \cup R$ is the set of input variables which can be parameters or real-time variables, and $O \in X$ is the set of output variables which is a subset of the ODE solution.

In this system, I is defined to utilize variables and parameters that are provided as input to the ODESCD definition, and O is used to export the desired variables from the solution of ODESCD.

❖ **ODESCD example : meal glucose absorption model**

X is a vector representing carbo-hydrate measures in the stomach where Q_{sto1} and Q_{sto2} are the stomach glucose amounts in solid state and liquid state, respectively, and *rag* is the blood glucose rate of appearance. These physical quantities are initially nulled as assigned in X^0. Fig. 5.2 depicts the SysML constraints block diagram for meal absorption variations measures.

$$X = [Q_{sto1} \, Q_{sto2} \, rag]^T \tag{18}$$

$$X^0 = [0 \, 0 \, 0]^T \tag{19}$$

$$\left[K\right] = \begin{bmatrix} -k_{gri} & 0 & 0 \\ k_{gri} & -k_{empt}(Q_{sto1}(t) + Q_{sto2}(t), D_{meal}) & 0 \\ 0 & \frac{f.k_{abs}}{BW} k_{empt}(Q_{sto1}(t) + Q_{sto2}(t), D_{meal}) & -k_{abs} \end{bmatrix}$$

$$k_{empt}(Q, D_{meal}) = \begin{cases} k_{min} + \frac{k_{max} - k_{min}}{2}(tanh(\alpha(Q - b.D_{meal})) \\ \qquad -tanh(\beta(Q - c.D_{meal})) + 2) & D_{meal} > 0 \\ \\ 0 & D_{meal} = 0 \end{cases}$$

$$\alpha = \frac{5}{2.D_{meal}.(1 - b)} \tag{20}$$

$$\beta = \frac{5}{2.D_{meal}.c} \tag{21}$$

$$P = \{k_{gri}, k_{abs}, f, BW, b, c, k_{min}, k_{max}\} \tag{22}$$

$$R = \{cur\_Meal, D_{meal}\} \tag{23}$$

$$F(X, R) = [cur\_Meal(t) \, 0 \, 0]^T \tag{24}$$

$$I = \{cur\_Meal, D_{meal}\} \tag{25}$$

$$O = \{rag\} \tag{26}$$

❖ ODESCD example: glucose-insulin dynamics

X is a vector representing the various physical quantities for the glucose and insulin dynamics all over the body compartments. I_{sc1} and I_{sc2} are the insulin levels in the subcutaneous tissues, X_1 is the insulin in the interstitial fluid, $\{G, G_s, G_t\}$ are the glucose levels in the blood, subcutaneous tissues, and slowly equilibrating tissues respectively. I_p is the plasma insulin, I_l is the portal vein insulin, and I_d is the delayed insulin signal. These physical quantities are initialized as in the vector X^0. Fig. 5.3 depicts the SysML Constraint Block diagrams for Glucose-Insulin variations measures.

$$X = [I_{sc1} \, I_{sc2} \, X_1 \, G_s \, I_1 \, I_d \, I_l \, I_p \, G \, G_t]^T \tag{27}$$

$$X^0 = [I_{sc1_{ss}} \, I_{sc2_{ss}} \, 0 \, G_i \, I_b \, I_b \, I_{l_b} \, I_{p_b} \, G_i \, G_{t_i}]^T \tag{28}$$

«constraint»
Meal Absorption : Equality
$Q_{sto1}' == - k_{gri} \cdot Q\_{sto1} + cur\_Meal(t)$ $Q_{sto2}' == - k_{empt}(Q_{sto1}+Q_{sto2}, D_{meal}) \cdot Q_{sto2} + k_{gri} \cdot Q_{sto1}$ $rag' == - k_{abs} \cdot rag + f \cdot k_{abs} \cdot k_{empt}(Q_{sto1}+Q_{sto2}, D_{meal}) \cdot Q_{sto2} / BW$
Parameters: Input: cur_Meal [*mg/min*] (real time glucose intake) , D_{meal} [*g*] (meal carbs)   Output: rag [*mg/Kg/min*] (real time glucose rate of appearance in the blood)

Figure 5.2: SysML Constraint Block for Meal Absorption

$$[K] = \begin{bmatrix} -(k_d + k_{a1}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ k_d & -k_{a2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -p_{2u} & 0 & 0 & 0 & 0 & \frac{p_{2u}}{V_I} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{T_s} & 0 & 0 & 0 & 0 & \frac{1}{T_s} & 0 \\ 0 & 0 & 0 & 0 & -k_i & 0 & 0 & \frac{k_i}{V_I} & 0 & 0 \\ 0 & 0 & 0 & 0 & k_i & -k_i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -(m_1 + \frac{m_6 \cdot m_1}{1-m_6}) & m_2 & 0 & 0 \\ k_{a1} & k_{a2} & 0 & 0 & 0 & 0 & m_1 & -(m_2 + m_4) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{-k_{p3}}{V_G} & 0 & 0 & -(k_{p2} + k_1) & \frac{k_2}{V_G} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_1 . V_G & -k_2 \end{bmatrix}$$

(29)

$$P = \{I_{sc1_{ss}}, I_{sc2_{ss}}, G_i, I_{l_b}, I_{p_b}, G_{t_b}, G_b, k_d, k_{a1}, k_{a2}, p_{2u}, V_I, I_b, T_s, k_i, V_G, m_1,$$
$$m_6, m_2, m_4, k_{p1}, k_{p2}, k_{p3}, F_{cns}, k_{e1}, k_{e2}, k_1, k_2, V_{m0}, V_{mx}, K_{m0}, K_{mx}\}$$

(30)

$$R = \{rag, IIR\}$$

(31)

$$F(X, R) = [IIR \, , \, 0 \, , \, -p_{2u}.I_b \, , \, 0 \, , \, 0 \, , \, 0 \, , \, 0 \, , \, 0 \, , \, \frac{rag + k_{p1} - F_{cns}}{V_G}$$
$$- k_{e1}.max(0, G - \frac{k_{e2}}{V_G}) \, , \, -\frac{(V_{m0} + V_{mx}.X_1).G_t}{K_{m0} + K_{mx}.X_1 + G_t}]^T$$

(32)

74

«constraint»
Glucose-Insulin Dynamics : Equality

$I_{sc1}' == -(k_d + k_{a1}) . I_{sc1} + IIR$
$I_{sc2}' == k_d . I_{sc1} - k_{a2} . I_{sc2}$
$X_1' == -p_{2u} . X_1 + p_{2u} . (I_p / V_I - I_b)$
$G_s' == -(G_s - G) / T_s$
$I_1' == -k_i . (I_1 - I_p / V_I)$
$I_d' == -k_i . (I_d - I_1)$
$I_l' == -(m_1 + (m_6 . m_1 / (1 - m_6))) . I_l + m_2 . I_p$
$I_p' == -(m_2 + m_4) . I_p + m_1 . I_l + k_{a1} . I_{sc1} + k_{a2} . I_{sc2}$
$G' == (k_{p1} - k_{p2} . G . V_G - k_{p3} . I_d - F_{cns} - k_{e1} . max(0, G . V_G - k_{e2}) - k_1 . G . V_G + k_2 . G_t + rag) / V_G$
$G_t' == -(Gt . (V_{m0} + V_{mx} . X_1)) / (K_{m0} + K_{mx} . X + G_t) + k_1 . G . V_G - k_2 . G_t$

Parameters:
Input: IIR [*pmol/Kg/min*] (subcutaneous insulin infusion rate)
Input: rag [*mg/Kg/min*] (meal glucose rate of appearance in the plasma)
Output: G_s [*mg/dL*] (real time subcutaneous glucose level)
Output: G [*mg/dL*] (real time blood glucose level)

Figure 5.3: SysML Constraint Block for Glucose-Insulin Dynamics
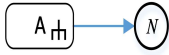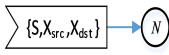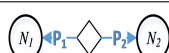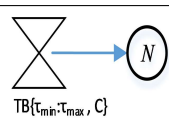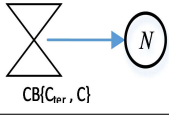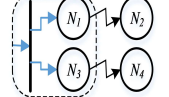
$$I = \{rag, IIR\} \tag{33}$$

$$O = \{G_s, G\} \tag{34}$$

SysML for discrete-time dynamics

Discrete-time dynamics are described by SysML activity diagrams. So, in order to precisely describe CPS and capture exactly its underlying semantics, we develop Enhanced Activity Calculus (EAC) to formally describe SysML activity diagrams by extending NuAC presented in (Debbabi et al., 2010; Ouchani et al., 2014). These enhancements include redefining existing nodes as well as proposing new nodes for time-bounded delay, constraint-bounded delay, and competing events. The

Table 5.1: SysML Enhanced Activity Calculus Nodes Syntax

SysML Term	SysML Activity Diagram Structure	EAC Syntax
Activity Initial Node		$l \mapsto N$
Action Node		$l : ACT(A) \mapsto N$
Call Procedure		$l : CALL_P(A) \mapsto N$
Send Node		$l : \{S, X\} \, ! \mapsto N$
Receive Node		$l : \{S, X_{src}, X_{dst}\} \, ? \mapsto N$
Merge Node		$l : Mrg \mapsto N$
Guarded Branch		$l : B_C(l_{i_1} : (C = C_1) \mapsto N_1,$ $\quad l_{i_2} : (C = C_2) \mapsto N_2, \ldots)$
Probabilistic Branch		$l : B_P(l_{i_1} : (P = P_1) \mapsto N_1,$ $\quad l_{i_2} : (P = P_2) \mapsto N_2, \ldots)$
Time-Bounded Delay Node		$l : D_T B(\tau_{min} : \tau_{max}\,, \; C) \mapsto N$
Constraint-Bounded Delay Node		$l : D_{CB}(C_{ter}\,, \; C) \mapsto N$
Competing Events		$l : Comp\_Events(N_1 \mapsto N_2,$ $N_3 \mapsto N_4, \ldots)$

list of the used activity nodes and their textual EAC representation is shown in Table 5.1.

❖ **EAC example: artificial pancreas**

The artificial pancreas is composed of a sensor (Fig. 5.4) that periodically measures the glucose level, sends it over wireless channel (Fig. 5.5) to the controller. Then, the controller (Fig. 5.6) calculates the required amount of insulin, and the actuator (Fig. 5.7) applies the control action. Lastly, the SysML activity diagram describing the meal scenario is shown in Fig. 5.8.

By substituting the SysML nodes with their textual equivalents following Table 5.1, the EAC
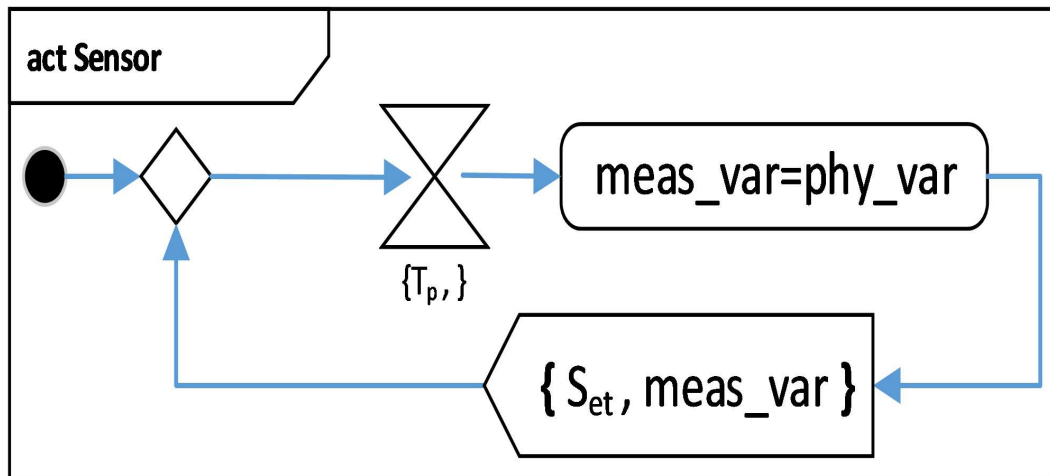
Figure 5.4: SysML Activity Diagram of the Sensor
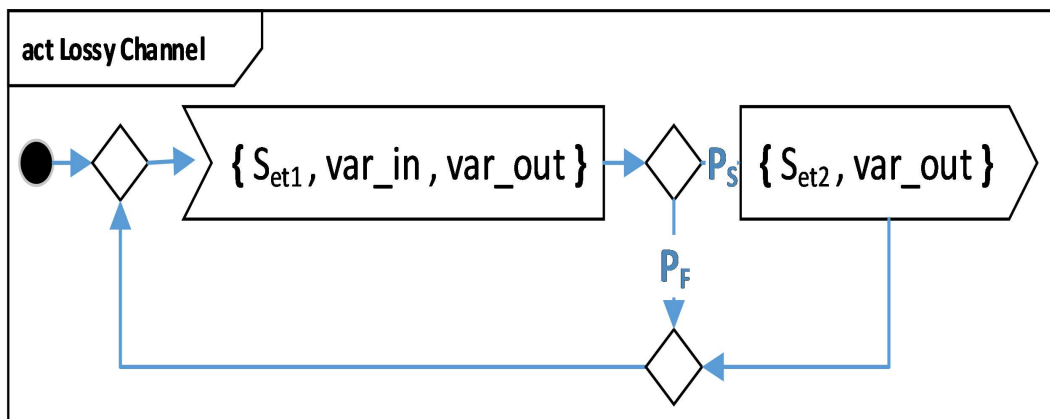
Figure 5.5: SysML Activity Diagram of the Lossy Channel
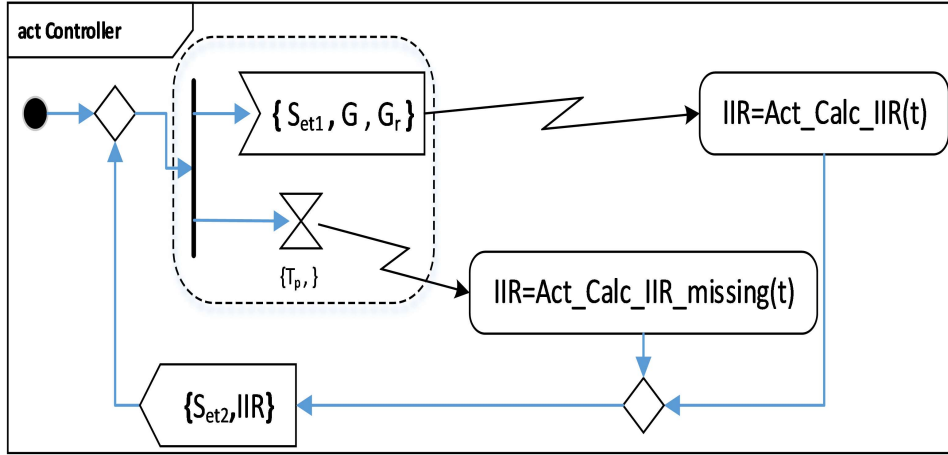
Figure 5.6: SysML Activity Diagram of the Controller
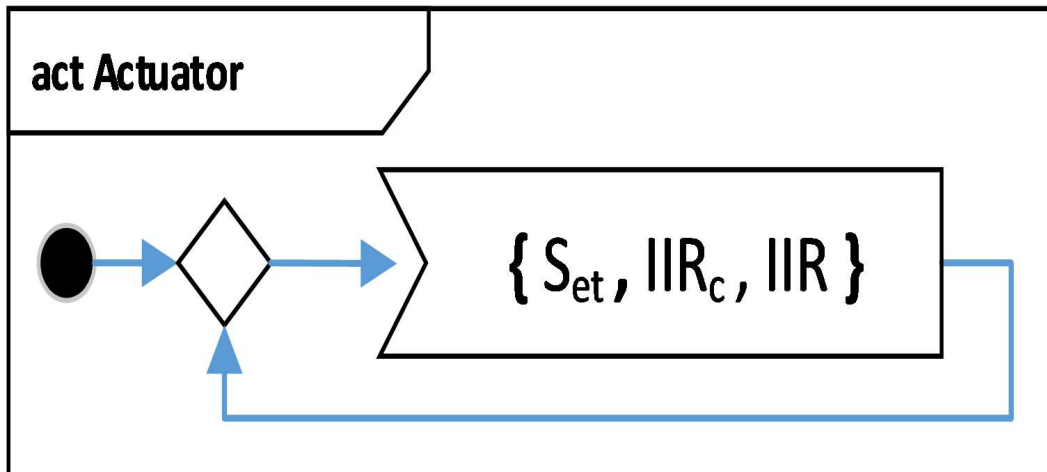
Figure 5.7: SysML Activity Diagram of the Actuator
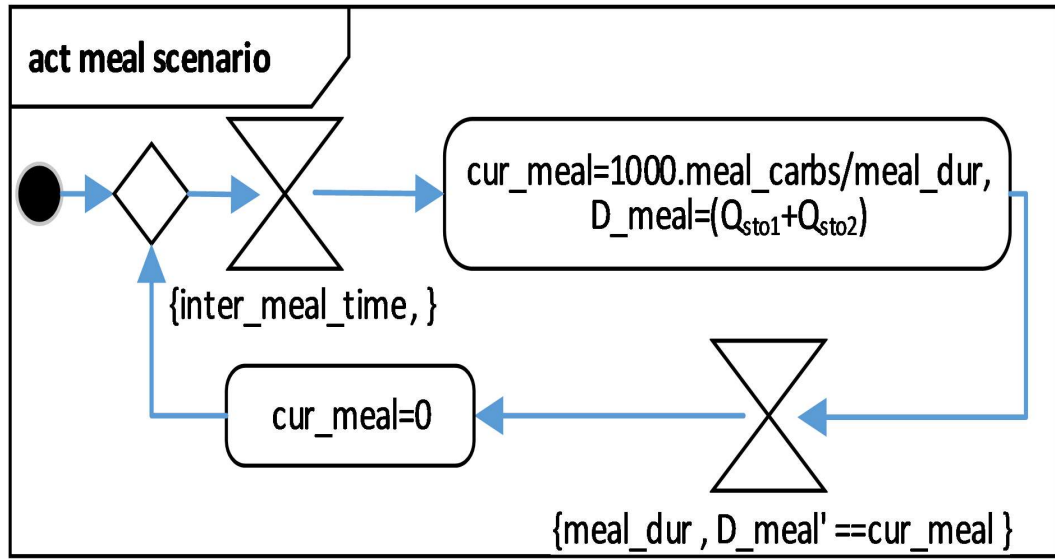
Figure 5.8: SysML Activity Diagram of the Meal Scenario

representation of these activity diagrams is shown below.

$$Act\_Sensor = l \mapsto l_1 : Mrg \mapsto N_1$$

$$N_1 = l_2 : D_{TB}(T_p,) \mapsto l_3 : ACT(meas\_var = phy\_var) \mapsto l_4 : \{S_{et}, meas\_var\}! \mapsto l_1$$

$$Act\_Channel_{lossy} = l \mapsto l_1 : Mrg \mapsto N_1$$

$$N_1 = l_2 : \{S_{et1}, var\_in, var\_out\}? \mapsto l_3 : B_P(l_4 : (P = P_S) \mapsto N_2, l_5 : (P = P_F)$$

$$\mapsto l_6 : Mrg \mapsto l_1)$$

$$N_2 = l_7 : \{S_{et2}, var\_out\}! \mapsto l_6$$

$$Act\_Ctrl = l \mapsto l_1 : Mrg \mapsto N_1$$

$$N_1 = l_2 : Comp\_Events(l_3 : \{S_{et1}, G, G_r\}? \mapsto N_2, l_4 : D_{TB}(Tp,) \mapsto N_3)$$

$$N_2 = l_5 : CALL_P(IIR = Act\_Calc\_IIR(t)) \mapsto l_6 : Mrg \mapsto N_4$$

$$N_3 = l_7 : CALL_P(IIR = Act\_Calc\_IIR\_missing(t)) \mapsto l_6$$

$$N_4 = l_8 : \{S_{et2}, IIR\}! \mapsto l_1$$

$$Act\_Actuator = l \mapsto l_1 : Mrg \mapsto l_2 : \{S_{et}, IIR_c, IIR_r\}? \mapsto l_3 : ACT(IIR = IIR_r) \mapsto l_1$$

$$Act\_meal\_scenario = l \mapsto l_1 : Mrg \mapsto l_2 : D_{TB}(inter\_meal\_time,) \mapsto N_1$$

$$N_1 = l_3 : ACT(cur\_meal = 1000 * meal\_carbs/meal\_dur, D\_meal =$$

$$(Q_{sto1} + Q_{sto2})/1000) \mapsto N_2$$

$$N_2 = l_4 : D_{TB}(meal\_dur, D\_meal' == cur\_meal/1000)$$

$$\mapsto l_5 : ACT(cur\_meal = 0) \mapsto l_1$$

❖ **CPS architecture and flow for artificial pancreas**

The SysML block definition diagram shown in Fig. 5.9 defines the blocks and their input/output ports. Also, the mapping of the blocks and the variables as well as the flow of information among these blocks are defined in the flow internal block diagram shown in Fig. 5.10.

5.4 CPS Semantics

The system behavior should be represented in the suitable formality that matches the language of the analysis tool. To do so, the SysML components are converted into a network of equivalent
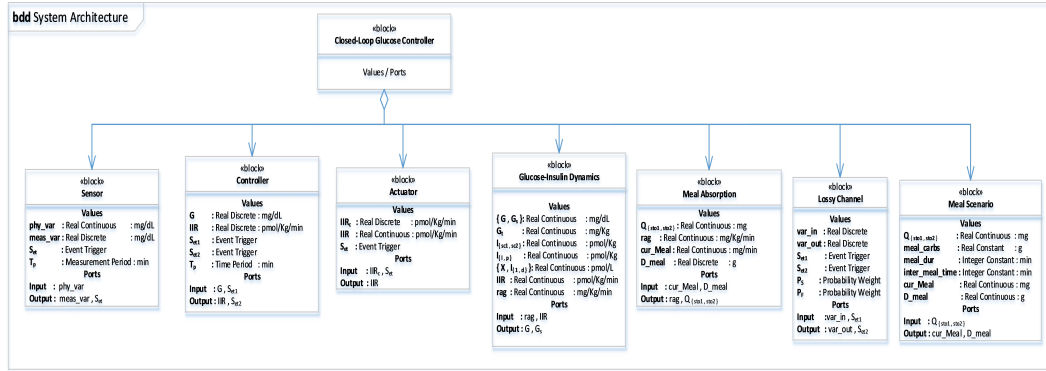
Figure 5.9: SysML Architectural Block Definition Diagram of the Closed-Loop Glucose Control System

PTA models. In the following, the PTA is defined and the new proposed automated conversion procedure is presented.

Definition 2 (PTA). A PTA for CPS is a tuple

$(L, l_0, L_{lbl}, L_{IP}, L_{OP}, E, X, V_g, INV(X, VAR), A(V_g), G(X, V_g), S_{et}, P_r)$, where:

- L is a finite set of locations,

- $l_0 \in L$ is the initial location,

- L_{lbl} is a set of labels,

- L_{IP} is a finite set of input ports,

- L_{OP} is a finite set of output ports,

- E is a finite set of edges,

- X is a finite set of clocks,

- VAR is a finite set of general-type variables,

- $INV(X, V_g)$ is a finite set of invariants over PTA clocks X and variables V_g, $A(V_g)$ is a finite set of actions on the variables V_g,

- $G(X, V_g)$ is a finite set of atomic propositions on PTA clocks X and variables V_g, S_{et} is a finite set of synchronization event triggers, and

81

- P_r is a finite set of probabilistic weights.

Definition 3 (Semantics of CPS). Let $(L, l_0, L_{lbl}, L_{IP}, L_{OP}, E, X, V_g,$ $INV(X, VAR), A(V_g), G(X, V_g), S_{et}, P_r)$ be a PTA for CPS. The semantics are defined as a hybrid transition system composed of a set of locations L interconnected by a set of edges E through sets of input ports IP and output ports OP, where:

- Locations $L = \{l_1, l_2, \ldots, l_{n1}\}$, where the ith location $l_i \in L$ labelled $label_i \in L_{lbl}$ having the invariant constraints $inv_i \in INV$ and connected to the input port x_{ip} and the output ports X_{op} is referred as $l_i(label_i, inv_i, x_{ip}, X_{op})$.

- Edges $E = \{e_1, e_2, \ldots, e_{n2}\}$, where the ith edge running the action $a \in A$ and triggering the synchronization event $s_{et} \in S_{et}$, and connected to the output port x_{op} and input port x_{ip} is referred as $e_i = \{a, s_{et}, x_{op}, x_{ip}\}$.

- Input ports $L_{IP} = \{l_{ip_1}, l_{ip_1}, \ldots, l_{ip_{n1}}\}$, where the ith input port $l_{ip_i} \in L_{IP}$ sourcing from incoming edges X_e towards the ith location $l_i \in L$ and applying the action $a \in A$ is defined as $l_{ip_i} = \{a, X_e, i\}$.

- Output ports $L_{OP} = \{l_{op_1}, l_{op_1}, \ldots, l_{op_{n3}}\}$, where the kth output port $l_{op_k} \in L_{OP}$ sourcing from the ith location L_i towards the jth edge e_j, guarded by the atomic proposition $g \in G$, triggered by the event trigger $s_{et} \in S_{et}$, and having the probabilistic weight $p_r \in P_r$ is defined as $l_{op_k} = \{g, s_{et}, p_r, i, j\}$.

PTAs traverse sequentially through output ports towards edges, followed by input ports towards the next location, starting at an initial location denoted by l_0. In the case of the PTA being at location l_i, the invariant inv_i must be satisfied as long as the PTA is at location L_i. Similarly, an output port that has a guard g with respect to its traversal can only be traversed if this guard g has been satisfied. An output port with an event trigger s_{et} is synchronized with another PTA, so that the output port is only traversed when it is activated by the corresponding event trigger on the edge of the other PTA. Furthermore, an output port can be traversed among other output ports in a probabilistic manner by assigning a probability weight p_r to each of the possible candidates for traversal of the output port.
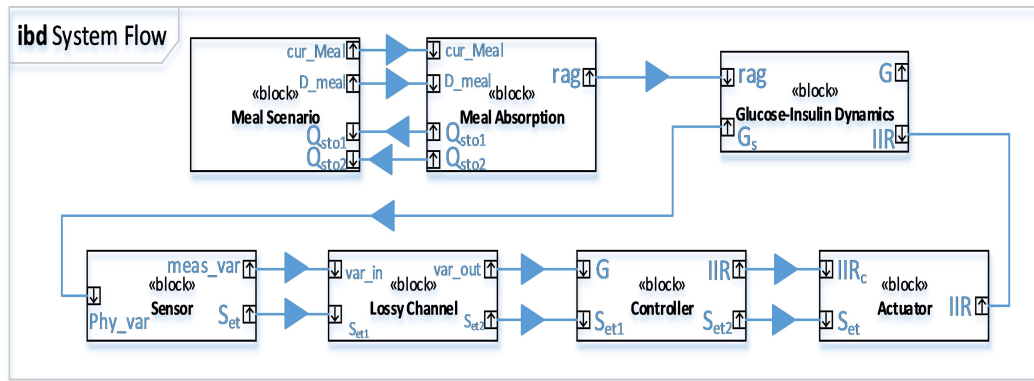
Figure 5.10: SysML Flow Internal Block Diagram of the Closed-Loop Glucose Control System.

5.4.1 Converting SysML into Equivalent PTA

In order to analyze the CPS described in SysML, it is necessary to model the hybrid system in PTA. So, SysML blocks are translated into equivalent PTA blocks which are parallel-composed to construct the hybrid system's global behavior. The synchronization of actions and the transfer of values are specified using shared variables.

The template of each PTA is instantiated with its input/output parameters properly defined. The SysML flow internal block diagram (as in Fig. 5.10) is consulted to define global variables for the parameters connecting the PTA components of the system. When instantiating a PTA template, the parameters are passed by-reference except for constant parameters that are passed by-value. Instead, those constants can be defined as local variables in the PTA. The following rules govern the definition of variables in PTA models.

- Continuous real-valued parameters are defined using clock variables.

- Discrete real-valued parameters are defined using floating point variables.

- An event trigger should be activated whenever a discrete variable is updated, so that the other PTAs are notified about the new update.

- Discrete integer parameters are defined as integer variables and are passed between PTAs similar to the floating point variables.

- When assigning or initializing a numerical variable, it can be evaluated to a single value or to a range of values for a uniformly-distributed random assignment.

❖ **Converting SysML EAC into PTA**

This part presents the detailed procedure for constructing a PTA block that represents a SysML EAC block. Alongside the description of the conversion steps, an illustrative example is provided for converting the $Act\_Channel_{lossy}$ block from EAC into PTA.

- The first step is to merge all the EAC nodes into the main EAC construct. This is done by iterating through the auxiliary constructs (N_x) and substituting for them in the main construct as depicted in Fig. 5.11.

$$Act\_Channel_{lossy} = l \mapsto l_1 : Mrg \mapsto \boxed{N_1}$$
$$N_1 = l_2 : \{S_{et1}, var\_in, var\_out\}? \mapsto l_3 : B_P(l_4 : (P = P_S) \mapsto \boxed{N_2},$$
$$l_5 : (P = P_F) \mapsto l_6 : Mrg \mapsto l_1)$$
$$N_2 = l_7 : \{S_{et2}, var\_out\}! \mapsto l_6$$
$$\Downarrow \Downarrow \Downarrow$$
$$Act\_Channel_{lossy} = l \mapsto l_1 : Mrg \mapsto \boxed{l_2 : \{S_{et1}, var\_in, var\_out\}? \mapsto l_3 : B_P(l_4 : (P = P_S)}$$
$$\boxed{\mapsto \boxed{N_2}, l_5 : (P = P_F) \mapsto l_6 : Mrg \mapsto l_1)}$$
$$N_2 = l_7 : \{S_{et2}, var\_out\}! \mapsto l_6$$
$$\Downarrow \Downarrow \Downarrow$$
$$Act\_Channel_{lossy} = l \mapsto l_1 : Mrg \mapsto l_2 : \{S_{et1}, var\_in, var\_out\}? \mapsto l_3 : B_P(l_4 : (P = P_S)$$
$$\boxed{\mapsto l_7 : \{S_{et2}, var\_out\}! \mapsto l_6}, l_5 : (P = P_F) \mapsto l_6 : Mrg \mapsto l_1)$$

Figure 5.11: EAC Lossy Channel Example - Merging Nodes

- Connecting the EAC terms so that each arrow is uniquely identified as presented in Fig. 5.12.

- Handling branching terms (B_P or $Comp\_Events$) and replicating the EAC construct, so that each branching term has only one path at a time. This is done by iterating through the branching terms and taking one branch at a time as shown in Fig. 5.13.

$$\Downarrow \Downarrow \Downarrow$$

$$Act\_Channel_{lossy} = l \overset{1}{\mapsto} l_1 : Mrg \overset{2}{\mapsto} l_2 : \{S_{et1}, var\_in, var\_out\}? \overset{3}{\mapsto} l_3 : B_P(l_4 : (P = P_S)$$
$$\overset{4}{\mapsto} l_7 : \{S_{et2}, var\_out\}! \overset{5}{\mapsto} l_6, l_5 : (P = P_F) \overset{6}{\mapsto} l_6 : Mrg \overset{7}{\mapsto} l_1)$$

Figure 5.12: EAC Lossy Channel Example - Labeling Arrows

$$Act\_Channel_{lossy} = l \overset{1}{\mapsto} l_1 : Mrg \overset{2}{\mapsto} l_2 : \{S_{et1}, var\_in, var\_out\}? \overset{3}{\mapsto} l_3 : B_P($$
$$l_4 : (P = P_S) \overset{4}{\mapsto} l_7 : \{S_{et2}, var\_out\}! \overset{5}{\mapsto} l_6 , l_5 : (P = P_F)$$
$$\overset{6}{\mapsto} l_6 : Mrg \overset{7}{\mapsto} l_1)$$
$$Act\_Channel_{lossy}\text{-}\boxed{Path①} = l \overset{1}{\mapsto} l_1 : Mrg \overset{2}{\mapsto} l_2 : \{S_{et1}, var\_in, var\_out\}? \overset{3}{\mapsto} l_3 :$$
$$B_P\text{-}\boxed{Branch}(l_4 : (P = P_S) \overset{4}{\mapsto} l_7 : \{S_{et2}, var\_out\}! \overset{5}{\mapsto} l_6)$$
$$\Downarrow \Downarrow \Downarrow$$
$$Act\_Channel_{lossy}\text{-}\boxed{Path②} = l \overset{1}{\mapsto} l_1 : Mrg \overset{2}{\mapsto} l_2 : \{S_{et1}, var\_in, var\_out\}? \overset{3}{\mapsto} l_3 :$$
$$B_P\text{-}\boxed{Branch}(l_5 : (P = P_F) \overset{6}{\mapsto} l_6 : Mrg \overset{7}{\mapsto} l_1)$$

Figure 5.13: EAC Lossy Channel Example - Branches Handling

- Building the PTA skeleton using the procedure described in Algorithm 1. The resulting skeleton for $Act\_Channel_{lossy}$ example is shown in Fig. 5.14.

Algorithm 1 Construction of PTA Skeleton.

---

for each: EAC_Path

1: $prev\_Node = \emptyset$ ▷ The first node of a path has no predecessor.

for each: $EAC\_Node \in EAC\_Path$

2: **if** $EAC\_Node \in \{Mrg, Comp\_Events, B_P, D(*), \{*, *\}?, (P = *)\}$ **then**

3:     $EAC\_Type = LOCATION$

4: **else if** $EAC\_Node \in \{\mapsto, ACT, CALL_P, \{*, *\}!\}$ **then**

5:     $EAC\_Type = EDGE$

6: **end if**

7: **if** EAC_Node processed before **then**

8:     $cur\_Node = PTA\_Node[EAC\_Node]$ ▷ Traverse through the node.

9:     $cur\_Node.prev.addMember(prev\_Node)$ ▷ Create a new input port for the node.

10:     $prev\_Node.next.addMember(cur\_Node)$ ▷ Create a new output port for the node.

11: **else if** $EAC\_Type == prev\_Node.type$ **then**

12:     $cur\_Node.EAC.addMember(EAC\_Node)$ ▷ A compliment for the previous node.

13: **else**   ▷ A node not processed yet.

14:     $cur\_Node = create\_Node(type = EAC\_Type)$ ▷ Create the node.

15:     $cur\_Node.EAC.addMember(EAC\_Node)$ ▷ Traverse through the node.

16:     $cur\_Node.prev.addMember(prev\_Node)$ ▷ Create an input port.

17:     $prev\_Node.next.addMember(cur\_Node)$ ▷ Create an output port.

18: **end if**

- For each location node that has non-empty $prev$ field, insert an input port. For locations with $next$ field, insert an output port per edge node that is outgoing from the location. In the following steps, when an EAC term is linked to an output port, the one that is connected to the location where the EAC belongs is identified. In case the location is attached to two or more output ports, the sequence of EAC terms in the path construct is used to identify the corresponding output port. Moreover, an EAC node that shows up in more than one path is only converted once at its first appearance.

- Replacing the following EAC terms with their equivalent PTA terms.

  ○ EAC term l signifies the location as an initial location.

$$Node_{ID} = [\quad type,\qquad\qquad prev,\quad next,\qquad\qquad EAC\quad]$$
$$Node_1 = [\quad LOCATION,\qquad \emptyset,\quad 2,\qquad\qquad l\quad]$$
$$Node_2 = [\quad EDGE,\qquad\qquad 1,\quad 3,\qquad\qquad \overset{1}{\mapsto}\quad]$$
$$Node_3 = [\quad LOCATION,\qquad \{2,11\},\quad 4,\qquad\qquad l_1\quad]$$
$$Node_4 = [\quad EDGE,\qquad\qquad 3,\quad 5,\qquad\qquad \overset{2}{\mapsto}\quad]$$
$$Node_5 = [\quad LOCATION,\qquad 4,\quad 6,\qquad\qquad l_2\quad]$$
$$Node_6 = [\quad EDGE,\qquad\qquad 5,\quad 7,\qquad\qquad \overset{3}{\mapsto}\quad]$$
$$Node_7 = [\quad LOCATION,\qquad 6,\quad \{8,10\},\qquad \{l_3, l_4, l_5\}\quad]$$
$$Node_8 = [\quad EDGE,\qquad\qquad 7,\quad 9,\qquad \{\overset{4}{\mapsto}, l_7, \overset{5}{\mapsto}\}\quad]$$
$$Node_9 = [\quad LOCATION,\qquad \{8,10\},\quad 11,\qquad\qquad l_6\quad]$$
$$Node_{10} = [\quad EDGE,\qquad\qquad 7,\quad 9,\qquad\qquad \overset{6}{\mapsto}\quad]$$
$$Node_{11} = [\quad EDGE,\qquad\qquad 9,\quad 3,\qquad\qquad \overset{7}{\mapsto}\quad]$$

Figure 5.14: EAC Lossy Channel Example - Building Skeleton.

○ $D_{TB}(\tau_{min} : \tau_{max} , C)$: Declare a clock variable t, Add a reset for the clock $(t = 0)$ to the input port action, Add the following constraint $(t \leq \tau_{max} \,\&\&\, C)$ to the invariants inv of the location, and add the following $(t \geq \tau_{min})$ to the guard g of the output port.

○ $\{S, X_{src}, X_{dst}\}?$: Add the event trigger $S?$ to the respective field s_{et} of the output port, and add the assignment $(X_{dst} = X_{src})$ to the action of the edge outgoing from the output port.

○ $\{S, X\}!$: Add this event trigger $S!$ to the respective event trigger field s_{et} of the containing edge.

○ $(P = p_x)$: Add the following probabilistic weight to the corresponding field p_r of the output port.

○ $ACT(A)$: Add the action A to the corresponding field a of the edge.

○ $CALL_P(A)$: Add the behavior call $A()$ to the action field a of the edge.

The results shown in Fig. 5.15 are obtained when applying the above rules on the $Act\_Channel_{lossy}$ example:

87

$$loc_1(label_1, \phi, \phi, op_1)$$
$$loc_2(label_2, \phi, ip_2, op_2)$$
$$loc_3(label_3, \phi, ip_3, op_3)$$
$$loc_4(label_4, \phi, ip_4, \{op_{(4,1)}, op_{(4,2)}\})$$
$$loc_5(label_5, \phi, ip_5, op_5)$$
$$ip_2(\phi, \{e_1, e_6\}, loc_2)$$
$$ip_3(\phi, e_2, loc_3)$$
$$ip_4(\phi, e_3, loc_4)$$
$$ip_5(\phi, \{e_4, e_5\}, loc_5)$$
$$op_1(\phi, \phi, \phi, loc_1, e_1)$$
$$op_2(\phi, \phi, \phi, loc_2, e_2)$$
$$op_3(\phi, S_{et_1}?, \phi, loc_3, e_3)$$
$$op_{(4,1)}(\phi, \phi, P_S, loc_4, e_4)$$
$$op_{(4,2)}(\phi, \phi, P_F, loc_4, e_5)$$
$$op_5(\phi, \phi, \phi, loc_5, e_6)$$
$$e_1(\phi, \phi, op_1, ip_2)$$
$$e_2(\phi, \phi, op_2, ip_3)$$
$$e_3(var_{out} = var_{in}, \phi, op_3, ip_4)$$
$$e_4(\phi, S_{et_2}!, op_{(4,1)}, ip_5)$$
$$e_5(\phi, \phi, op_{(4,2)}, ip_5)$$
$$e_6(\phi, \phi, op_5, ip_2)$$

Figure 5.15: EAC Lossy Channel Example - Replacing EAC with PTA Terms.

- After each EAC receive node, insert a new location between the event trigger and the signal sampling. Also, a new location is added when an output port with a probabilistic weight is directly followed by an edge with an EAC send node. This is done so that the send node is separated from the output port. When applying this on the $Act\_Channel_{lossy}$, the results look like Fig. 5.16

- Divide the locations into transient and regular (time-consuming) locations. A regular location is identified by having either a guard or an event trigger on the output port, or by having a non-empty invariant field. For the $Act\_Channel_{lossy}$ example, all the locations are transient except location loc_3 which has an event trigger on the output port.

- The rate of all local clocks should be identified on all regular locations. Therefore, if a clock is not supposed to evolve in a specific regular location, its evolution rate should be assigned to 0 in the invariants field of that location.

- When exporting the PTAs into an XML file compatible with UPPAAL-SMC analyzer, transient locations are specified as $urgent$ locations except for the following:

  ○ A location which emits output ports with probabilistic weights (location loc_4 in $Act\_Channel_{lossy}$ example) is defined as an $anchor$ point (for syntax compatibility).

  ○ The first location following a receive node (location loc_6 in $Act\_Channel_{lossy}$ example) should be set to $committed$ for synchronization correctness (semantic compatibility).

The resulting PTA diagram for the above transformed lossy channel is depicted in Fig. 5.17. This PTA initializes at the location loc_1. This location is urgent which means that no time progress and hence the PTA will move instantly through the output port op_1, the edge e_1, the input port ip_2 to the next location loc_2. This location is also an urgent location and hence the PTA will move through the output port op_2, edge e_2, and the input port ip_2 towards the location loc_3. The output port op_3 is activated by the event trigger S_{et1}? which is controlled by another PTA (the sensor in this case). Then, this sensor activates the event trigger S_{et} to send a new measurement (the variable var_{in}) through the wireless channel. When triggered by the event trigger S_{et1}, the lossy channel PTA moves through the output port op_3, the edge e_7, and the input port ip_6 towards the committed

$$loc_1(label_1, \phi, \phi, op_1)$$
$$loc_2(label_2, \phi, ip_2, op_2)$$
$$loc_3(label_3, \phi, ip_3, op_3)$$
$$loc_4(label_4, \phi, ip_4, \{op_{(4,1)}, op_{(4,2)}\})$$
$$loc_5(label_5, \phi, ip_5, op_5)$$
$$loc_6 \ (label_6, \phi, ip_6, op_6)$$
$$loc_7 \ (label_7, \phi, ip_7, op_7)$$
$$ip_2(\phi, \{e_1, e_6\}, loc_2)$$
$$ip_3(\phi, e_2, loc_3)$$
$$ip_4(\phi, e_3, loc_4)$$
$$ip_5(\phi, \{e_8, e_5\}, loc_5)$$
$$ip_6 \ (\phi, e_7, loc_6)$$
$$ip_7 \ (\phi, e_4, loc_7)$$
$$op_1(\phi, \phi, \phi, loc_1, e_1)$$
$$op_2(\phi, \phi, \phi, loc_2, e_2)$$
$$op_3(\phi, S_{et_1}?, \phi, loc_3, e_7)$$
$$op_{(4,1)}(\phi, \phi, P_S, loc_4, e_4)$$
$$op_{(4,2)}(\phi, \phi, P_F, loc_4, e_5)$$
$$op_5(\phi, \phi, \phi, loc_5, e_6)$$
$$op_6 \ (\phi, \phi, \phi, loc_6, e_3)$$
$$op_7 \ (\phi, \phi, \phi, loc_7, e_8)$$
$$e_1(\phi, \phi, op_1, ip_2)$$
$$e_2(\phi, \phi, op_2, ip_3)$$
$$e_3(var_{out} = var_{in}, \phi, op_6, ip_4)$$
$$e_4(\phi, \phi, op_{(4,1)}, ip_7)$$
$$e_5(\phi, \phi, op_{(4,2)}, ip_5)$$
$$e_6(\phi, \phi, op_5, ip_2)$$
$$e_7 \ (\phi, \phi, op_3, ip_6)$$
$$e_8 \ (\phi, S_{et_2}!, op_7, ip_5)$$

Figure 5.16: EAC Lossy Channel Example - Inserting Locations

location loc_6. Like the urgent location, a committed location freezes time but also synchronizes the PTAs so that the correct sequence of actions takes place. In this PTA, it is required so that the up-to-date version of the measurement value var_{in} is read.

The PTA moves through op_6 towards the edge e_3 where the measurement is sampled, and then through the input port ip_4 to the location loc_4 which is a probabilistic branching point. Then, the PTA will take a branch depending on probability weights. At one branch, the message will get lost and so the PTA takes the output port $op_{(4,2)}$ towards the edge e_5 and the input port ip_5 to reach the location loc_5. In the other branch, the measurement is successfully relayed so the other PTA (the controller in this case) is notified with the event trigger $S_{et2}!$, so the PTA moves through $op_{(4,1)}$, e_4, ip_7 to the transient location loc_7 towards the output port op_7 and the edge e_8 (where $S_{et2}!$ is activated) to the input port ip_5 while merging with the other branch in the location loc_5. Finally, the PTA moves via the output port op_5 and the edge e_6 through the input port ip_2 to merge in the location loc_2.
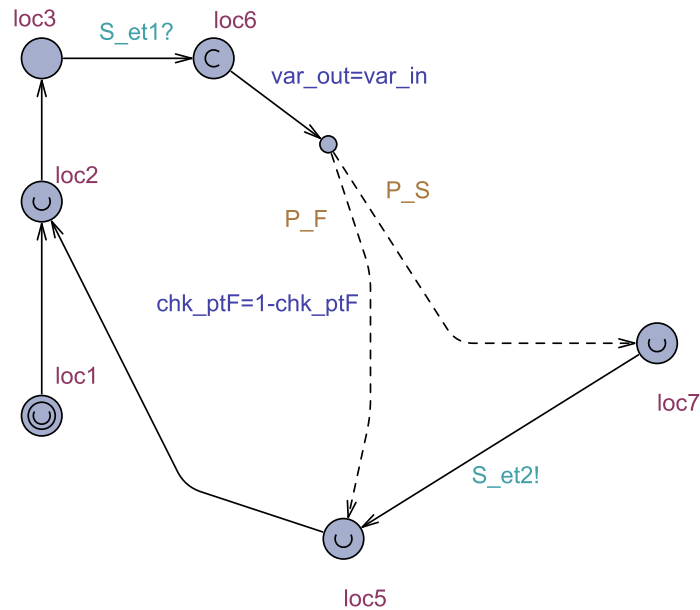
Figure 5.17: The Resulting PTA Diagram for the Lossy Channel

❖ **modeling ODESCD using PTA**

The same rules apply to convert ODESCD into PTA where the ODE variables X are defined as clock variables. The PTA is composed of one location where the rates of the ODE variables X are

assigned using equality constraints in the invariant field of the main location. If some variables or parameters are initialized with random values, an additional transient initial location is added with the variables assigned in the edge connecting the initial location to the main operational location.

5.4.2 Soundness

After presenting the semantics of CPS and PTA, we prove the soundness of the developed framework. First, lets Γ to be a function denoting Algorithm 1. Now, we prove the soundness of the transformation by showing that Γ guarantees the integrity of the CPS design, i.e. no added, modified, or excluded behavior. Thus, an equivalent PTA behavioral model is produced. Then, we show that the soundness proves the satisfiability preservation of MILT expressions when applying Γ.

As depicted in Fig. 5.18, we have to show the nature of the relation $\mathscr{R}$, that compares both PTA^{cps} and PTA^f constructed through EAC and PTA semantics rules respectively, while preserving both behaviours. Indeed, the relation $\mathscr{R}$ could be determined by comparing the semantics of each term in EAC and the semantics of its image obtained by the function Γ. Since the goal is to guarantee the behaviour integrity of PTA^{cps} and the resulting PTA^f should not differ from PTA^{cps}, Lemma 1 proves that $\mathscr{R}$ is a bisimulation relation.
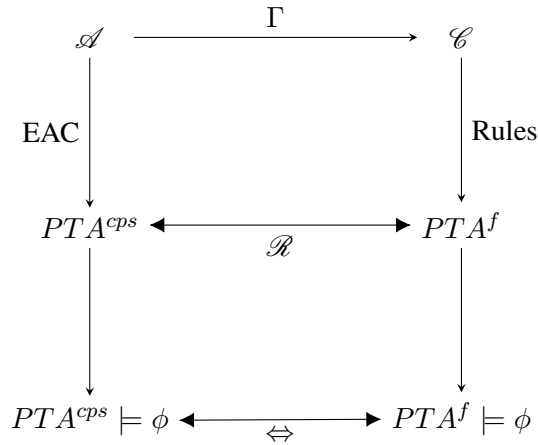
Figure 5.18: The Transformation Soundness Schema.

Lemma 1. *The binary relation $\mathscr{R}$, is a bisimulation, whenever $S\mathscr{R}\hat{S}$, satisfies the following.*

(1) If $S \xrightarrow{\alpha} S'$ then $\exists \hat{S}'$ such that $\hat{S} \xrightarrow{\alpha} \hat{S}'$ and $S' \equiv_{\mathscr{R}} \hat{S}'$.

(2) If $\hat{S} \xrightarrow{\alpha} \hat{S}'$ then $\exists S$ such that $S' \xrightarrow{\alpha} S'$ and $\hat{S}' \equiv_{\mathscr{R}} S'$.

Proof. Let's consider $A \in PTA^{cps}$ and $B \in PTA^f$ where $\Gamma(A) = B$. So, by induction on EAC terms, we prove that $\mathscr{R}$ is a bisimulation binary relation as follows.

- When $A = i \rightarrow \mathscr{N}$, then based on the rule $\exists S \xrightarrow{\alpha} S'$ such that $S = i \rightarrowtail \mathscr{N}$ and $S' = \overline{i \rightarrowtail \mathscr{N}}$, we will have, $\Gamma(A) = \Gamma(i \rightarrow \mathscr{N}) = initial\ to\ i$. Thus, $initial \wedge \neg i \xrightarrow{\alpha} \neg initial \wedge i \in B^s$. Then, $PTA^{cps} \mathscr{R} PTA^f$ when $A = i \rightarrow \mathscr{N}$.

- For $\{S, X\}! \rightarrow \mathscr{N}$, then $\overline{X \rightarrowtail \mathscr{N}} \longrightarrow \{S, X, X'\}! \rightarrowtail \overline{\mathscr{N}} \in PTA^{cps}$. Also, $\Gamma(A) = resource < v > \rightarrow \mathscr{N}$ which means $resource_v \wedge \neg \mathscr{N} \xrightarrow{prt} \neg resource_v \wedge \mathscr{N} \in B^s$. So, $PTA^{cps} \mathscr{R} PTA^f$.

- In the case of $A = resource?\, v \rightarrow \mathscr{N}$, we have $\overline{resource?\, v \rightarrowtail \mathscr{N}} \longrightarrow resource?\, v \rightarrowtail \overline{\mathscr{N}} \in B^{sn}$. Thus, $PTA^{cps} \mathscr{R} PTA^f$.

- If $A = resource!\, v \rightarrow \mathscr{N}$, we have $\overline{resource!\, v \rightarrowtail \mathscr{N}} \longrightarrow resource!\, v \rightarrowtail \overline{\mathscr{N}} \in B^{sn}\ R\ resourceout_v \wedge \exists v \wedge \neg \mathscr{N} \xrightarrow{prt} \neg resourceout_v \wedge \mathscr{N} \in B^s$.

- By considering $A = resource \uparrow expression \rightarrow \mathscr{N}$, then $\overline{resource \uparrow expression \rightarrowtail \mathscr{N}} \longrightarrow resource \uparrow expression \rightarrowtail \overline{\mathscr{N}} \in B^{sn}$. As a result, we have $resource_v \wedge \neg \mathscr{N} \xrightarrow{prt} \neg resource_v \wedge v = newvalue \wedge \mathscr{N} \in B^s$, which means $PTA^{cps} \mathscr{R} PTA^f$.

- For the decision term $A = D(g_{v1}, \mathscr{N}_1, \mathscr{N}_2)$, we differentiate two cases:

  (1) When $\neg g_{v1} \models \top$, we have $\overline{D(g_{v1}, \mathscr{N}_1, \mathscr{N}_2)} \xrightarrow{\neg g_{v1}} D(g_{v1}, \mathscr{N}_1, \overline{\mathscr{N}_2}) \in B^{sn}$ by relying on the decision rule. Also, we have: $\Gamma(A) = \{$on prt_i from source to $\mathscr{N}$ provided $g_{v_i} = eval(v_i) : i \in \{1, 2\}\}$. Also, since $\neg g_{v1} \models \top$, we have: $\overline{D(g_{v1}, \mathscr{N}_1, \mathscr{N}_2)} \xrightarrow{\neg g_{v1}} D(g_{v1}, \mathscr{N}_1, \overline{\mathscr{N}_2}) \in B^s$.

  (2) For the other case, when $g_{v1} \models \top$, we have shown that $D(g_{v1}, \mathscr{N}_1, \mathscr{N}_2) \equiv D(\neg g_{v1}, \mathscr{N}_2, \mathscr{N}_1)$. Thus, $PTA^{cps} \mathscr{R} PTA^f$.

We have shown that for each EAC term, we have $PTA^{cps} \mathscr{R} PTA^f$ in which result that $\mathscr{R}$, is a bisimulation relation and it is symmetric. $\qquad\square$

Based on the illustration presented in Fig 5.18:sound, the transformation's objective is to verify functional properties of the generated PTA model and then infer satisfiability results for the CPS design. Using Lemma 1, Proposition 1 demonstrates how the properties expressed in MITL logic can be satisfied.

Proposition 1. $\forall A \in PTA^{cps}, B \in PTA^f$ *s.t.* $\Gamma(A) = B$, *we have:* $\forall \phi \in MITL : PTA^f \models \phi \implies PTA^{cps} \models \phi$.

Proof. By induction on MITL terms, we prove that $B \models \phi \implies A \models \phi$.

(1) First, let's consider the state formulae $\phi = \varphi_1 \wedge \varphi_2$ where $B \models \phi$. Now, we show the satisfiability of ϕ on A for the following EAC terms.

- For $A = i \to \mathcal{N}$, we have $i \to \mathcal{N} \xrightarrow{\alpha} \overline{i \rightarrowtail \mathcal{N}}$ R $initial \wedge \neg i \xrightarrow{\alpha} \neg initial \wedge i$. If $initial \wedge \neg i \models \varphi_1 \wedge \varphi_2$ means $initial \wedge \neg i = \varphi_1 \wedge \varphi_2$. Thus, $i \to \mathcal{N} \models \phi$, and, $B \models \phi$

- For $A = resource < v > \to \mathcal{N}$ when $\neg resource_v \wedge \mathcal{N} \models \varphi_1 \wedge \varphi_2$, we have $resource < v > \rightarrowtail \overline{\mathcal{N}} \models \varphi_1 \wedge \varphi_2$. Then, $B \models \phi$.

- For $A = resource? v \to \mathcal{N}$, then $B \models \phi$ $resourcein_v \wedge v = newvalue \wedge \neg \mathcal{N} \xrightarrow{prt} \neg resourcein_v \wedge \mathcal{N} \models \phi$. Thus, we have $\overline{resource? v \rightarrowtail \mathcal{N}} \longrightarrow resource? v \rightarrowtail \overline{\mathcal{N}} \models \phi$. Consequently, $B \models \phi$.

(2) Now, we consider the path formulae $P_{\bowtie p}[\psi]$. So, since EAC does not support probabilistic decisions and has only deterministic ones, $P_{\geq 1}[\psi]$ means ψ else we consider the case of $P_{\leq 0}[\psi]$. Then, we prove by induction on the path operators that $PTA^{cps} \models \phi$ when $PTA^f \models \phi$ as follows.

- For $\phi = N\varphi$, $B \models \phi$ means $\exists \hat{S} \xrightarrow{\alpha} \hat{S}' \in B^n$ such that $\hat{S}' \models \varphi$. In addition, since $\mathcal{R}$ is symmetric, then $\exists S \xrightarrow{\alpha} S' \in B$ such that: $S' \models \varphi$.

- For $\phi = \varphi_1 \cup^t \varphi_2$, we have $\exists \hat{S}_1 \xrightarrow{\alpha} \cdots \to \hat{S}'_t \subseteq B^n$ such that $\hat{S}_{i:i<t'} \models \varphi_1$ and $\hat{S}_2 \models \varphi_2$. Also, $\mathcal{R}$ is symmetric and $\exists S_1 \xrightarrow{\alpha} \cdots \to S'_t \subseteq B^{sn}$ where $S_i \mathcal{R} \hat{S}_i : 0 < i \leq t$. Thus, $B \models \phi$.

Based on the previous proof, we have shown that for each EAC and MITL term, $\mathscr{R}$ always preserves the satisfiability of MITL formulae. Consequently, $B \models \phi \implies A \models \phi$ for all ϕ expressed in MITL when $PTA^{cps} \mathscr{R} PTA^{f}$. $\qquad\square$

5.5   Experimentation

This section shows the effectiveness of the proposed framework by first validating the transformation algorithm. Then, the proposed approach is used to demonstrate how the safety of the obtained model can be examined by statistical model checking over a list of selected functional and safety requirements.

5.5.1   Validation of the Conversion Procedure

To demonstrate the correctness of the proposed approach, the resulting PTA models are validated. A set of properties constraining the functional behavior of each system component are specified. Random simulations of the resulting PTA model are conducted and trace log analysis is applied on the results to evaluate their satisfaction of the behavioral properties. By proving that all the properties are satisfied, we increase the confidence in the resulting PTA models to be valid representations of the CPS components. The PTAs representing ODESCD are validated by comparing the values of the ODE variables against a mathematical ODE solver. The following steps demonstrate the model validation for the PTAs representing EAC.

5.5.2   Validation of the Conversion Procedure

In order to demonstrate the correctness of the proposed approach, PTA models are validated. Properties are specified for each component of the system that constrain its functional behavior. To evaluate whether the resulting PTA model meets the behavioral properties, random simulations are conducted and trace log analysis is applied to the results. The resulting PTA models are more likely to be valid representations of the CPS components when all the properties are satisfied.

By comparing the values of the ODE variables with a mathematical ODE solver, PTAs representing ODESCD are validated. In the case of the ODESCDs describing meal absorption and

glucose-insulin dynamics, multiple simulations are conducted on 10 virtual patients for 24 hours under various meal scenarios. The PTAs for these ODESCDs that are constructed using the above automatic procedure are simulated.

The trace logs of the physical variables are compared against our ODE solver developed in Matlab and errors are recorded. The absolute errors of variable samples are divided by the variable root mean square to get the relative absolute errors. The percentage mean and standard deviation (std) of these relative absolute errors are depicted in Table 5.2. It can be noted that the relative errors are negligible and hence demonstrate the correctness of the proposed procedure.

Table 5.2: Meal and Glucose-Insulin Dynamics ODESCD Variables (Results Against a Mathematical Solver).

Variable Identifier	Relative Absolute Error {mean+std}
Q_{sto1}	$0.018\% \pm 0.007\%$
Q_{sto2}	$0.027\% \pm 0.012\%$
rag	$0.028\% \pm 0.012\%$
I_{sc1}	$0.166\% \pm 0.049\%$
I_{sc2}	$0.117\% \pm 0.039\%$
X_1	$0.219\% \pm 0.029\%$
G_s	$0.164\% \pm 0.203\%$
I_1	$0.071\% \pm 0.030\%$
I_d	$0.047\% \pm 0.027\%$
I_l	$0.118\% \pm 0.040\%$
I_p	$0.118\% \pm 0.040\%$
G	$0.165\% \pm 0.200\%$
G_s	$0.180\% \pm 0.209\%$

For the case of cyber components which are specified by EAC, the following steps demonstrate the model validation for this type of PTAs.

- Sensor: The sensor PTA shown in Fig. 5.19-a has three locations. It periodically waits in loc_3 before sampling the subcutaneous glucose measurement phy_var into the variable $meas\_var$. The edge originating from loc_3 to loc_2 synchronizes the sensor with the lossy channel by means of the event trigger S_{et}.

  ○ A new measurement is sent periodically every T_p minutes: to check on this property, a new binary flag variable is added to the PTA (chk_pt_1 in the sensor PTA shown in the

graph of Fig. 5.19-a). The variable is marked whenever a measurement is sent. This can be achieved by flipping the value of the variable in an ACT term at the same edge as the send term (the edge goes from *loc3* to *loc2*). The variable is monitored on random simulations and its value should be flipped periodically every T_p minute.

(a) Sensor PTA   (b) Lossy Channel PTA
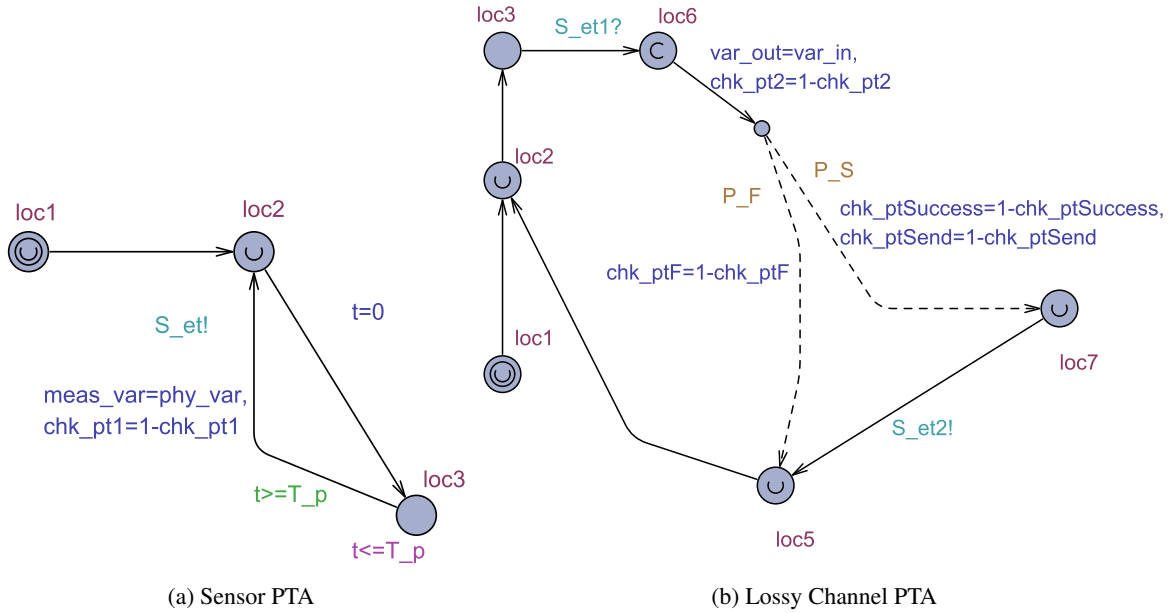
Figure 5.19: A Part of the Sensor's PTA Communication Network.

○ Whenever a measurement is sent, its value should be equal to the most recent sample of the physical variable monitored. Then, the value of the measurement is examined in particular whenever the binary flag, defined above, is flipped.

○ The mapping of all the variables that are shared with other PTAs should be validated as well. In particular, the variables (phy_var, S_{et}, $meas\_var$) in the Sensor PTA are examined against G_s in the glucose-insulin dynamics PTA and (S_{et1}, var_in) in the $Act\_Channel_{lossy}$ PTA, respectively. For a properly mapped system, the values of the variables in a PTA should be matched to their corresponding ones in all other PTAs at any time.

• Channel$_{lossy}$: The PTA shown in Fig. 5.19-b has seven locations where the edge from loc_3

towards loc_6 synchronizes with the sensor PTA to receive the measurement value as an input variable var_{in}. Similarly, the edge from loc_7 to loc_5 synchronizes with the controller PTA to send the measurement value as an output variable var_{out}.

- ○ For every received measurement, the PTA will either successfully relay the measurement to the controller with probability P_S or fail with probability P_F. To check on this, binary flags are marked (flipped) on the corresponding edges for success and failure ($chk\_pt_{Success}$ and chk_pt_F in the graph of Fig. 5.19-b). These binary flags are monitored for random simulations over various probabilistic weights.

- ○ A measurement is sent to the controller if and only if the edge with P_S probabilistic weight is traversed. This can be checked by examining the corresponding binary flags.

- ○ Whenever a measurement is sent to the controller (S_{et2} is activated), the value of the measurement (var_out) should be equal to the value of the sample received from the sensor (var_in).

- ○ To validate the mapping of variables, the values of the variables (S_{et2}, var_out) should be equal to the values of the corresponding variables in the controller PTA (S_{et1}, G), respectively.

- Controller: The PTA shown in Fig. 5.20-a has five locations where the edge from loc_3 towards loc_5 synchronizes with the lossy channel PTA to receive the glucose measurement value as an input variable G. Similarly, the edge from loc_4 to loc_2 synchronizes with the actuator PTA to send the control value as an output variable IIR.

- ○ For each measurement delivered (S_{et1} activated), the PTA will read the measurement value G and use it to calculate a new Insulin Infusion Rate (IIR) using the standard Proportional-Integral-Derivative (PID) control (Alshalalfah et al., 2021; Laxminarayan, Reifman, & Steil, 2012). This new calculated value of IIR should be sent to the actuator by activating the event trigger S_{et2}.

- ○ If the time since the last delivered measurement exceeds the control period T_p, the value of the variable IIR is zeroed and the event trigger S_{et2} is activated to command insulin

(a) The Controller PTA.
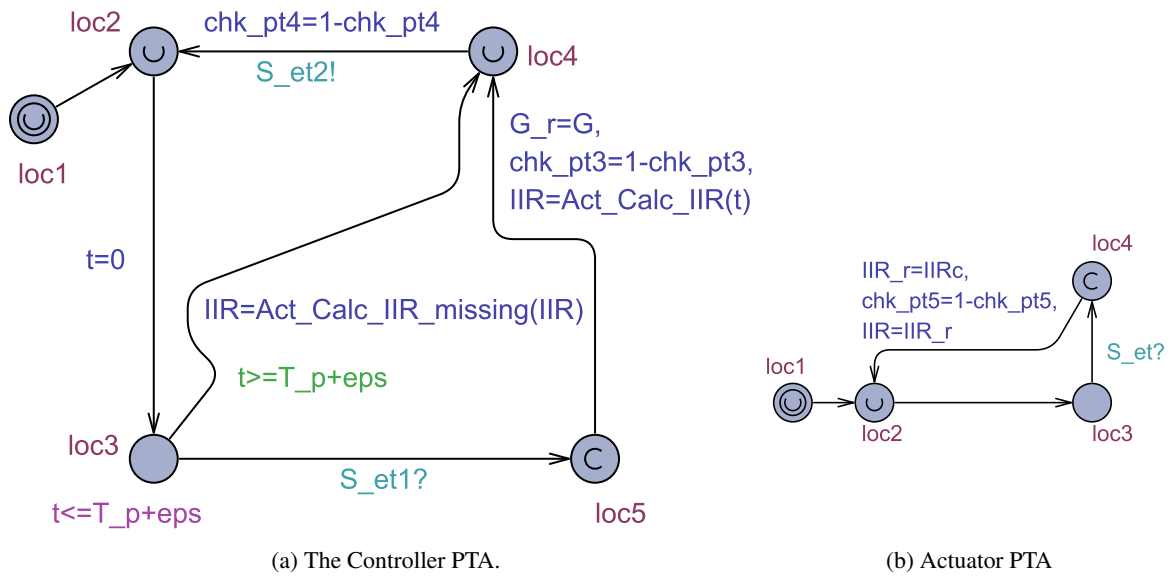
(b) Actuator PTA

Figure 5.20: A Part of the Artificial Pancreas Control Network.

delivery suspension.

○ To validate the mapping of variables, the values of the variables (S_{et2}, IIR) should be equal to the values of the corresponding variables in the actuator PTA (S_{et}, IIR_c), respectively.

• Actuator: The PTA shown in Fig. 5.20-b has four locations where the edge from loc_3 towards loc_4 synchronizes with the controller PTA to receive the control value as an input variable IIR_c. The actuator then modifies the corresponding physical values in the glucose-insulin dynamics PTA through the output variable IIR.

○ Whenever a new infusion rate value IIR_c control command from the controller PTA is received (S_{et} activation), the actuator should update the value of the physical real-time variable IIR.

○ To verify the mapping of variables, the values for the variables IIR in both PTAs, actuator and glucose-insulin dynamics, should be equal at all times.

• Meal Scenario: This PTA is used to assign the input variables of the meal absorption model such as the carbohydrate amounts and the inter-meal times.

- Each of the variables ($meal\_carbs$, $meal\_dur$, $inter\_meal\_time$) takes a value ranging between the configured minimum and maximum with uniform distribution. Based on the histogram of the variables, this can be validated.

- The PTA should generate the values of the real-time variables (cur_meal, D_{meal}) complying with the right amounts of insulin-carbs, meal durations, and inter-meal times.

- Validation for the mapping of the variables (cur_meal, D_{meal}, Q_{sto1}, Q_{sto2}) with their corresponding variables in the meal absorption PTA.

- Meal Absorption & Glucose-Insulin Dynamics:

  - The variables of the ODEs for both PTAs are observed and compared using our ODE simulator. The values for all variables should be identical to the ones calculated by the mathematical ODE solver developed in Matlab except for marginal numerical computational errors, e.g. precision.

5.6 Model Verification

PTAs are constructed for all the CPS components and are exported to a file for verification and analysis. This file is loaded into UPPAAL-SMC. A network of PTAs is created by instantiating and parallel-composing the PTA blocks using the UPPAAL-SMC. The tool performs hypothesis testing on queries specified by Metric Interval Temporal Logic (MITL). Also, monitor-based verification (Bulychev et al., 2012) could be used to specify more complicated queries using simpler expressions or for queries that are beyond the expressive power of MITL query language.

To demonstrate the use of the proposed framework to analyze real-life systems, UPPAAL-SMC is utilized to investigate safety properties of the artificial pancreas CPS that is supposed to regulate the blood glucose levels using a pre-configured closed-loop control strategy. A good control strategy would be able to satisfy safety properties under normal conditions. Moreover, it would accommodate disturbances and minimize the side effects of faults.

Using this system, the sensor periodically transmits measurements to the controller over a wireless channel, but wireless packet transmission failure can cause measurements to be missing. Missing measurements can be handled using different control approaches. With the proposed SMC modeling and analysis, it is possible to evaluate whether each control approach can preserve safety properties at various error rates.

Whenever the controller receives a measurement, it calculates the required insulin rate using the standard PID. For a missing measurement, the controller will behave in one of three ways.

- Sustain: The controller will keep configuring the last valid calculated insulin rate until a new valid measurement is received.

- Suspend: The controller will stop insulin delivery until a new valid measurement is received.

- Revert: The controller will revert to a low value which is equal to the PID controller basal insulin rate until a new valid measurement is received.

The analysis is conducted on a database of 10 adult patients publicly accessible (Man et al., 2014). Each patient receives random meals of $(20 - 50)$ $grams$ carbohydrates each. Per patient, the analysis evaluates whether or not the controller satisfies safety properties for each of the three control configurations: sustain, suspend or revert. The following two safety properties are defined for analysis.

- S_A: At all times, the blood glucose levels should not cross the boundaries of severe minimum and maximum values of 50 mg/dL and 300 mg/dL, respectively.

- S_B: Whenever the glucose elevates to values higher than the threshold of 180 mg/dL, it should restore its value to normal range below this threshold within a maximum of two and a half hours.

The first safety property S_A is straightforward and can be described using the following MITL query:

Pr[t$\leq$1440] ([] G $>=$ 50 && G $<=$ 300) $\geq$ 0.99

This property specifies that throughout the test duration of one day (1440 minutes) the blood glucose levels should be limited between 50 mg/dL and 300 mg/dL with a probability above or equal 99%. On the other side, the second safety property S_B is too elaborate to describe in a query using MITL. Instead, a monitor PTA is designed to observe the time duration for each time the glucose level elevates above 180 mg/dL as shown in Fig. 5.21. Having this variable (tg_{180}) assigned, the safety property S_B is described using the following MITL property.

$$\textbf{Pr[t} \leq \textbf{1440] ([] } tg_{180} <= \textbf{150)} \geq \textbf{0.99}$$

This property is satisfied if and only if a high glucose incidence would recover to normal range within two and a half hours maximum with at least 99% probability. It should be noted that the monitor PTA is constructed by creating a SysML activity diagram characterizing its behavior as shown in Fig. 5.22 and applying the new proposed automatic procedure to convert the EAC description into a PTA component that is parallel-composed with the other PTAs in UPPAAL-SMC tool.

$$Act\_Monitor = l \mapsto l_1 : B_C(l_2 : (C = G{>}180) \mapsto N_1, l_3 : (C = G \leq 180) \mapsto N_2)$$

$$N_1 = l_4 : D_{CB}(G{<}180, G \geq 180 - 1 \&\& tg'_{180} == 1)$$

$$\mapsto l_5 : Act(tg_{180} = 0) \mapsto l_6 : D_{CB}(G{>}180 - 1, G \leq 180) \mapsto l_4$$

$$N_2 = l_6$$

The percentage of the patients with violations for each safety property is shown in Fig. 5.23. No violations exist in the absence of message errors. When message errors are introduced, the three control configurations result in varying behaviors. For safety property S_A, message errors result in a gradual increase of violations on *sustain* and *suspend* approaches. However, the *revert* approach preserves the safety property S_A on all patients with message errors up to 50%. For safety property S_B, the *suspend* approach fails on timely recovery of normal glucose levels in the existence of message errors. The other configurations, *sustain* and *revert*, avoid S_B violations with message
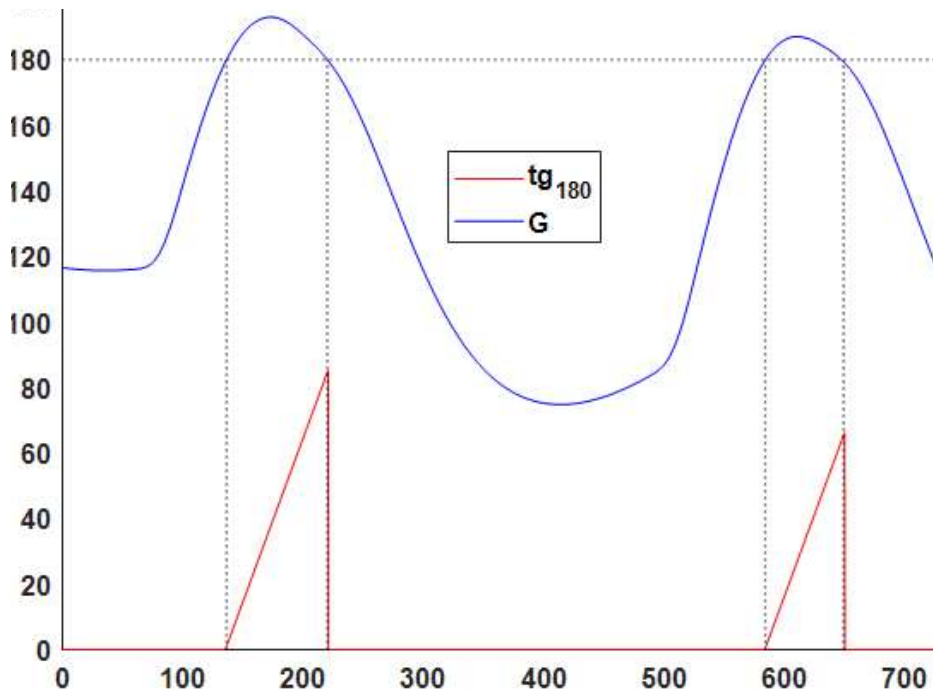
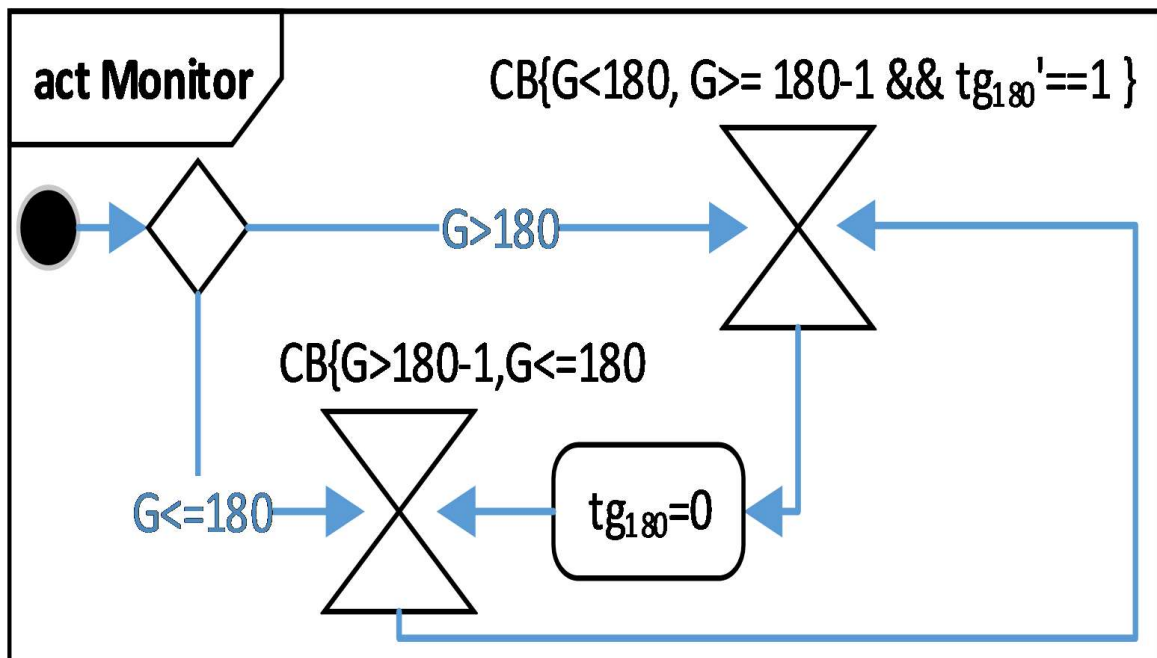Figure 5.21: The Duration of Time Where Glucose Exceeds 180 (mg/dL) $\{tg_{180}\}$

Figure 5.22: SysML Activity Diagram of the Monitor

errors as high as 30%. When the error rate exceeds that level, violations start to occur with the *revert* approach suffering more violations.

5.6.1 Discussion

To understand the experimental results, the following facts should be noted.

- In the absence of message errors, the three control configurations fall back to being the same standard PID controller.

- The analyzed artificial pancreas is a single hormone unidirectional controller (as opposed to dual-hormone systems (Haidar et al., 2015)). This implies that it can deliver more insulin to counteract the excessive glucose levels, but it can only counteract low glucose levels by suspending the insulin delivery and waiting for the pre-delivered insulin to get consumed by the physiological processes inside the body.

Putting this in mind can explain the results on safety property S_A (left graph in Fig. 5.23), where the *sustain* approach accidentally delivers excessive insulin amounts that can cause glucose drops below 50 mg/dL even at low message error rates. On the contrary, the *suspend* approach stops insulin delivery and can make it up by restarting insulin delivery when valid messages are received again. However, when the message error rate increases, there is a chance that the *suspend* approach might fail to prevent large glucose levels above 300 mg/dL. Instead of completely halting the insulin delivery, the *revert* continues delivering small amounts of insulin to make a balance between the two other approaches and avoid extreme highs and lows of glucose. The same concept explains the results in the right graph of Fig. 5.23 where the *sustain* approach provides better performance in avoiding long times with glucose levels above 180 mg/dL as opposed to the *suspend* approach which fails to avoid that. The *revert* approach provides performance similar to the *sustain* approach except for high message error rates where the violations start to increase when utilizing the *revert* approach.

Figure 5.23: Results for Safety Properties Violations: S_A (left) and S_B (right)

5.7 Conclusion

In this work, a framework is proposed to formally model and automatically analyze cyber-physical systems using statistical model checking. The framework takes models specified using SysML modeling language as SysML diagrams. The latters are then represented in textual format using the proposed enhanced activity calculus and ordinary-differential equations of SysML constraint diagrams. Then, these textual representations of the model components are fed into a new proposed conversion algorithm that automatically transforms them into equivalent priced timed automata. Thus, the resulting model is fed into UPPAAL-SMC statistical model checking tool which parallel-composes all the system components and verifies the system behaviors. The use of the proposed framework to verify safety properties is demonstrated on an artificial pancreas case study.

The proposed framework can be used to verify the safety of cyber-physical systems and gain insight into their most critical behaviors at an early stage of the design process, thus saving valuable time and money. Ultimately, it promotes the integration of real-life problems into model-based analysis and allows experimenting a variety of scenarios without compromising participant safety. This is especially crucial when dealing with systems that involve human life, whether directly as in biomedical systems or indirectly as in automotive systems. In the near future, we target to improve the framework to cover more issues, mainly:

- Develop a library of different CPS components and applications.

- Model more cyber-physical systems with a focus on faults and security threats.

105

- Before the CPS deployment, we target also to automatically generate the source code related to the modeled and analyzed CPS.

- Provide guidance to correct the CPS whenever a property has not been satisfied.

- Establish a mechanism for defining CPS complex requirements automatically and easily.

Chapter 6

Conclusion and Future Work

The availability of affordable hardware components and advanced software tools fuels the development of cyber-physical systems. With the huge influx of system designs that promise to grant solutions for the various applications, it is required to guarantee that such designs will not compromise safety. By utilizing system-level safety analysis, only safe designs are implemented into real system prototypes for further verification.

In this research project, a framework is proposed for system-level analysis of safety-critical cyber-physical systems. The framework processes models specified with system modeling language. A systematic procedure is proposed to construct formal models that are analyzed using statistical model checking. The analysis can be used to select designs for safety and to guide proposing enhancements as well. The usage of the analysis methodology is demonstrated on a set of biomedical and automotive systems. Moreover, new improved control strategies with safety enhancements are proposed for the closed-loop artificial pancreas.

The developed framework can be extended to cover more features of cyber-physical systems such as the physicality laws. Also, it can be used to consider decentralized archetictural paradigms. Moreover, specific libraries can be developed to model potential faults, security threats, and environmental effects.

Publications

Refereed Journals

- **Jr1** Alshalalfah, A.-L., Bany Hamad, G. and Ait Mohamed, O., Towards safe and robust closed-loop artificial pancreas using improved PID-based control strategies. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(8), pp.3147-3157, 2021.

- **Jr2** Alshalalfah, A.-L., Ait Mohamed, O. and Ouchani, S., A Framework for modeling and analyzing cyber-physical systems using statistical model checking. **Accepted** to *Internet of Things* on **Feb 19, 2023** (Manuscript ID: IOT-D-22-00776).

Refereed Conferences

- **Cf1** Alshalalfah, A.-L., Bany Hamad, G., and Ait Mohamed, O., Towards system level security analysis of artificial pancreas via uppaal-smc. *International Symposium on Circuits and Systems (ISCAS)* (pp. 1–5), IEEE 2019.

- **Cf2** Alshalalfah, A.-L., Bany Hamad, G., and Ait Mohamed, O., Towards safe and robust closed-loop artificial pancreas using adaptive weighted PID control strategy. *18th International New Circuits and Systems Conference (NEWCAS)* (pp. 146-149), IEEE 2020. (**Best Paper Award**).

- **Cf3** Alshalalfah, A.-L., Bany Hamad, G., and Ait Mohamed, O., System-level analysis of closed-loop anesthesia control under temporal sensor faults via uppaal-smc. *42nd Annual International Conference of IEEE Engineering in Medicine & Biology Society (EMBC)* (pp.

2508-2511), IEEE 2020.

- **Cf4**   **Alshalalfah, A.-L.**, and Ait Mohamed, O., System-level modeling and safety analysis of vehicular coordinated emergency braking under degraded wireless connectivity using priced timed automata. *27th International Conference on Electronics, Circuits and Systems (ICECS)* (pp. 1-4), IEEE 2020.

References

Agha, G., & Palmskog, K. (2018). A survey of statistical model checking. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, *28*(1), 1–39.

AH, K. (1964). Automation control of blood sugar. i. a servomechanism for glucose monitoring and control. *The American journal of medical electronics*, *3*, 82–86.

Aki, M., Zheng, R., Yamabe, S., Nakano, K., Suda, Y., Suzuki, Y., . . . Sakuma, A. (2014). Safety testing of an improved brake system for automatic platooning of trucks. *International journal of intelligent transportation systems research*, *12*(3), 98–109.

Alshalalfah, A.-L., Bany Hamad, G., & Ait Mohamed, O. (2019). Towards system level security analysis of artificial pancreas via uppaal-smc. In *International symposium on circuits and systems (iscas)* (pp. 1–5).

Alshalalfah, A.-L., Bany Hamad, G., & Ait Mohamed, O. (2020). System-level analysis of closed-loop anesthesia control under temporal sensor faults via uppaal-smc. In *42nd annual international conference of the engineering in medicine & biology society (embc)* (pp. 2508–2511).

Alshalalfah, A.-L., Hamad, G. B., & Mohamed, O. A. (2021). Towards safe and robust closed-loop artificial pancreas using improved pid-based control strategies. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *68*(8), 3147–3157.

Alshalalfah, A.-L., & Mohamed, O. A. (2020). System-level modeling and safety analysis of vehicular coordinated emergency braking under degraded wireless connectivity using priced timed automata. In *27th international conference on electronics, circuits and systems (icecs)* (pp. 1–4).

Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T. A., Ho, P.-H., Nicollin, X., . . . Yovine, S.

(1995). The algorithmic analysis of hybrid systems. *Theoretical computer science*, *138*(1), 3–34.

Alur, R., & Dill, D. L. (1994). A theory of timed automata. *Theoretical computer science*, *126*(2), 183–235.

Arnaout, G. M., & Arnaout, J.-P. (2014). Exploring the effects of cooperative adaptive cruise control on highway traffic flow using microscopic traffic simulation. *Transportation Planning and Technology*, *37*(2), 186–199.

Asarin, E., Maler, O., & Pnueli, A. (1995). Reachability analysis of dynamical systems having piecewise-constant derivatives. *Theoretical computer science*, *138*(1), 35–65.

Basu, A., Bensalem, S., Bozga, M., Delahaye, B., & Legay, A. (2012). Statistical abstraction and model-checking of large heterogeneous systems. *International Journal on Software Tools for Technology Transfer*, *14*(1), 53–72.

Beckert, B., Hähnle, R., & Schmitt, P. H. (2007). *Verification of object-oriented software. the key approach: Foreword by k. rustan m. leino* (Vol. 4334). Springer.

Behrmann, G., David, A., & Larsen, K. G. (2004). A tutorial on uppaal. *Formal methods for the design of real-time systems*, 200–236.

Bequette, B. W. (2005). A critical assessment of algorithms and challenges in the development of a closed-loop artificial pancreas. *Diabetes technology & therapeutics*, *7*(1), 28–47.

Bergenhem, C., Meinke, K., & Ström, F. (2018). Quantitative safety analysis of a coordinated emergency brake protocol for vehicle platoons. In *International symposium on leveraging applications of formal methods* (pp. 386–404).

Bilstrup, K., Uhlemann, E., Ström, E., & Bilstrup, U. (2009). On the ability of the 802.11 p mac method and stdma to support real-time vehicle-to-vehicle communication. *EURASIP Journal on Wireless Communications and Networking*, *2009*(1), 902414.

Böhm, A., Jonsson, M., Kunert, K., & Vinel, A. (2014). Context-aware retransmission scheme for increased reliability in platooning applications. In *International workshop on communication technologies for vehicles* (pp. 30–42).

Bulychev, P., David, A., Guldstrand Larsen, K., Legay, A., Li, G., Bøgsted Poulsen, D., & Stainer, A. (2012). Monitor-based statistical model checking for weighted metric temporal logic. In

International conference on logic for programming artificial intelligence and reasoning (pp. 168–182).

Cameron, F., Fainekos, G., Maahs, D. M., & Sankaranarayanan, S. (2015). Towards a verified artificial pancreas: Challenges and solutions for runtime verification. In *Runtime verification* (pp. 3–17).

Chen, X., Dutta, S., & Sankaranarayanan, S. (2017). Formal verification of a multi-basal insulin infusion control model. In *Arch@ cpsweek* (pp. 75–91).

Clarke Jr, E. M., Grumberg, O., Kroening, D., Peled, D., & Veith, H. (2018). *Model checking*. MIT press.

Claybrook, J., & Kildare, S. (2018). Autonomous vehicles: No driver... no regulation? *Science*, *361*(6397), 36–37.

Clopper, C. J., & Pearson, E. S. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, *26*(4), 404–413.

Cummings, M., & Britton, D. (2020). Regulating safety-critical autonomous systems: past, present, and future perspectives. In *Living with robots* (pp. 119–140). Elsevier.

Dalla Man, C., Camilleri, M., & Cobelli, C. (2006). A system model of oral glucose absorption: validation on gold standard data. *IEEE Transactions on Biomedical Engineering*, *53*(12), 2472–2478.

Dalla Man, C., Rizza, R. A., & Cobelli, C. (2007). Meal simulation model of the glucose-insulin system. *IEEE Transactions on biomedical engineering*, *54*(10), 1740–1749.

Dassau, E., Renard, E., Place, J., Farret, A., Pelletier, M.-J., Lee, J., ... Zisser, H. C. (2017). Intraperitoneal insulin delivery provides superior glycaemic regulation to subcutaneous insulin delivery in model predictive control-based fully-automated artificial pancreas in patients with type 1 diabetes: a pilot study. *Diabetes, Obesity and Metabolism*, *19*(12), 1698–1705.

David, A., Du, D., Larsen, K. G., Mikučionis, M., & Skou, A. (2012). An evaluation framework for energy aware buildings using statistical model checking. *Science China information sciences*, *55*(12), 2694–2707.

David, A., Larsen, K. G., Legay, A., Mikučionis, M., & Poulsen, D. B. (2015). Uppaal smc tutorial. *International journal on software tools for technology transfer*, *17*(4), 397–415.

David, A., Larsen, K. G., Legay, A., Mikučionis, M., Poulsen, D. B., & Sedwards, S. (2012). Runtime verification of biological systems. In *International symposium on leveraging applications of formal methods, verification and validation* (pp. 388–404).

David, A., Larsen, K. G., Legay, A., Mikučionis, M., & Wang, Z. (2011). Time for statistical model checking of real-time systems. In *International conference on computer aided verification* (pp. 349–355).

David, Y. B., Geller, T., Bistritz, I., Ben-Gal, I., Bambos, N., & Khmelnitsky, E. (2021). Wireless body area network control policies for energy-efficient health monitoring. *Sensors*, *21*(12), 4245.

Debbabi, M., Hassaine, F., Jarraya, Y., Soeanu, A., & Alawneh, L. (2010). *Verification and validation in systems engineering: assessing uml/sysml design models*. Springer Science & Business Media.

Eleveld, D., Colin, P., Absalom, A., & Struys, M. (2018). Pharmacokinetic–pharmacodynamic model for propofol for broad application in anaesthesia and sedation. *British journal of anaesthesia*, *120*(5), 942–959.

Eleveld, D. J., Proost, J. H., Cortinez, L. I., Absalom, A. R., & Struys, M. M. (2014). A general purpose pharmacokinetic model for propofol. *Anesthesia & Analgesia*, *118*(6), 1221–1237.

El-Khatib, F. H., Russell, S. J., Nathan, D. M., Sutherlin, R. G., & Damiano, E. R. (2010). A bihormonal closed-loop artificial pancreas for type 1 diabetes. *Science translational medicine*, *2*(27), 27ra27–27ra27.

Evans, M., Ceriello, A., Danne, T., De Block, C., DeVries, J. H., Lind, M., . . . Wilmot, E. G. (2019). Use of fast-acting insulin aspart in insulin pump therapy in clinical practice. *Diabetes, Obesity and Metabolism*, *21*(9), 2039–2047.

Facchinetti, A., Del Favero, S., Sparacino, G., Castle, J. R., Ward, W. K., & Cobelli, C. (2013). Modeling the glucose sensor error. *IEEE Transactions on Biomedical Engineering*, *61*(3), 620–629.

Facchinetti, A., Sparacino, G., & Cobelli, C. (2009). An online self-tunable method to denoise cgm sensor data. *IEEE Transactions on Biomedical Engineering*, *57*(3), 634–641.

Facchinetti, A., Sparacino, G., & Cobelli, C. (2020). Cgm filtering and denoising techniques. In

Glucose monitoring devices (pp. 203–218). Elsevier.

Fagnant, D. J., & Kockelman, K. (2015). Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, *77*, 167–181.

Filipovikj, P., Mahmud, N., Marinescu, R., Seceleanu, C., Ljungkrantz, O., & Lönn, H. (2016). Simulink to uppaal statistical model checker: Analyzing automotive industrial systems. In *International symposium on formal methods* (pp. 748–756).

Frehse, G. (2015). An introduction to hybrid automata, numerical simulation and reachability analysis. In *Formal modeling and verification of cyber-physical systems: 1st international summer school on methods and tools for the design of digital systems, bremen, germany, september 2015* (pp. 50–81). Springer.

Frehse, G., Guernic, C. L., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., . . . Maler, O. (2011). Spaceex: Scalable verification of hybrid systems. In *International conference on computer aided verification* (pp. 379–395).

Gallagher, J. D. (1999). Pacer-induced artifact in the bispectral index during cardiac surgery. *Anesthesiology: The Journal of the American Society of Anesthesiologists*, *90*(2), 636–636.

Gepts, E., Camu, F., Cockshott, I., & Douglas, E. (1987). Disposition of propofol administered as constant rate intravenous infusions in humans. *Anesthesia and analgesia*, *66*(12), 1256–1263.

Godefroid, P. (1996). *Partial-order methods for the verification of concurrent systems: an approach to the state-explosion problem.* Springer.

Haidar, A., Legault, L., Messier, V., Mitre, T. M., Leroux, C., & Rabasa-Lhoret, R. (2015). Comparison of dual-hormone artificial pancreas, single-hormone artificial pancreas, and conventional insulin pump therapy for glycaemic control in patients with type 1 diabetes: an open-label randomised controlled crossover trial. *The lancet Diabetes & endocrinology*, *3*(1), 17–26.

Han, J., Davids, J., Ashrafian, H., Darzi, A., Elson, D. S., & Sodergren, M. (2022). A systematic review of robotic surgery: From supervised paradigms to fully autonomous robotic approaches. *The International Journal of Medical Robotics and Computer Assisted Surgery*, *18*(2), e2358.

Hasan, S., Balador, A., Girs, S., & Uhlemann, E. (2019). Towards emergency braking as a fail-safe

state in platooning: A simulative approach. In *90th vehicular technology conference (vtc)* (pp. 1–5).

Haugen, F. (2010). *Ziegler-nichols' open-loop method. techteach, 1-7.*

Henzinger, T. A., Kopke, P. W., Puri, A., & Varaiya, P. (1995). What's decidable about hybrid automata? In *Proceedings of the twenty-seventh annual acm symposium on theory of computing* (pp. 373–382).

Hérault, T., Lassaigne, R., Magniette, F., & Peyronnet, S. (2004). Approximate probabilistic model checking. In *International workshop on verification, model checking, and abstract interpretation* (pp. 73–84).

Hernández, A. G. G., Fridman, L., Escobar, A. E., Davila, J., Leder, R., Monsalve, C. R., ... Hernández, A. L. (2013). Robust control for propofol induced anesthesia based on second-order sliding-mode control. In *Conference on decision and control* (pp. 2864–2869).

Holt, J., & Perry, S. (2008). *Sysml for systems engineering* (Vol. 7). IET.

Hovorka, R., Kumareswaran, K., Harris, J., Allen, J. M., Elleri, D., Xing, D., ... others (2011). Overnight closed loop insulin delivery (artificial pancreas) in adults with type 1 diabetes: crossover randomised controlled studies. *Bmj*, *342*.

Hu, R., & Li, C. (2015). An improved pid algorithm based on insulin-on-board estimate for blood glucose control with type 1 diabetes. *Computational and mathematical methods in medicine*, *2015*.

Huyett, L. M., Dassau, E., Zisser, H. C., & Doyle, F. J. (2018). Glucose sensor dynamics and the artificial pancreas: the impact of lag on sensor measurement and controller performance. *IEEE Control Systems Magazine*, *38*(1), 30–46.

Huyett, L. M., Dassau, E., Zisser, H. C., & Doyle III, F. J. (2015). Design and evaluation of a robust pid controller for a fully implantable artificial pancreas. *Industrial & engineering chemistry research*, *54*(42), 10311–10321.

Immler, F. (2015). Tool presentation: Isabelle/hol for reachability analysis of continuous systems. In *Arch@ cpsweek* (pp. 180–187).

International Organization for Standardization. (2018).
ISO 26262: Road Vehicles—Functional Safety. Geneva.

Isufi, E., Loukas, A., Simonetto, A., & Leus, G. (2016). Autoregressive moving average graph filtering. *IEEE Transactions on Signal Processing*, *65*(2), 274–288.

Jiang, Z., Pajic, M., Moarref, S., Alur, R., & Mangharam, R. (2012). Modeling and verification of a dual chamber implantable pacemaker. In *International conference on tools and algorithms for the construction and analysis of systems* (pp. 188–203).

Johansen, J. W. (2006). Update on bispectral index monitoring. *Best practice & research Clinical anaesthesiology*, *20*(1), 81–99.

Kekatos, N., Forets, M., & Frehse, G. (2017a). Constructing verification models of nonlinear simulink systems via syntactic hybridization. In *2017 ieee 56th annual conference on decision and control (cdc)* (pp. 1788–1795).

Kekatos, N., Forets, M., & Frehse, G. (2017b). Modeling the wind turbine benchmark with pwa hybrid automata. *EPiC Series in Computing*, *48*, 100–113.

Kokubugata, H., Kawashima, H., Fukui, R., & Kamata, G. (2022). Speed control of inflow vehicles for merging support on motorways with limited i2v communication.

Koong, J. K., Ng, G. H., Ramayah, K., Koh, P. S., & Yoong, B. K. (2021). Early identification of the critical view of safety in laparoscopic cholecystectomy using indocyanine green fluorescence cholangiography: A randomised controlled study. *Asian Journal of Surgery*, *44*(3), 537–543.

Koopman, P., & Wagner, M. (2016). Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, *4*(1), 15–24.

Kovatchev, B. P., Renard, E., Cobelli, C., Zisser, H. C., Keith-Hynes, P., Anderson, S. M., . . . others (2014). Safety of outpatient closed-loop control: first randomized crossover trials of a wearable artificial pancreas. *Diabetes care*, *37*(7), 1789–1796.

Kurzweil, R. (2004). The law of accelerating returns. In *Alan turing: Life and legacy of a great thinker* (pp. 381–416). Springer.

Kwiatkowska, M., Norman, G., & Parker, D. (2011). PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan & S. Qadeer (Eds.), *Proc. 23rd international conference on computer aided verification (cav'11)* (Vol. 6806, pp. 585–591). Springer.

Lafferriere, G., Pappas, G. J., & Sastry, S. (2000). O-minimal hybrid systems. *Mathematics of control, signals and systems*, *13*(1), 1–21.

Lakshmanan, S., Yan, Y., Baek, S., & Alghodhaifi, H. (2019). Modeling and simulation of leader-follower autonomous vehicles: environment effects. In *Unmanned systems technology xxi* (Vol. 11021, p. 110210J).

Lal, R. A., Ekhlaspour, L., Hood, K., & Buckingham, B. (2019). Realizing a closed-loop (artificial pancreas) system for the treatment of type 1 diabetes. *Endocrine reviews*, *40*(6), 1521–1546.

Larsen, K. G., & Skou, A. (1991). Bisimulation through probabilistic testing. *Information and computation*, *94*(1), 1–28.

Laxminarayan, S., Reifman, J., & Steil, G. M. (2012). Use of a food and drug administration-approved type 1 diabetes mellitus simulator to evaluate and optimize a proportional-integral-derivative controller. *Journal of diabetes science and technology*, *6*(6), 1401–1412.

LeBlanc, H., Chauvet, D., Lombrail, P., & Robert, J. J. (1986). Glycemic control with closed-loop intraperitoneal insulin in type i diabetes. *Diabetes Care*, *9*(2), 124–128.

Lee, D., & Hess, D. J. (2020). Regulations for on-road testing of connected and automated vehicles: Assessing the potential for global safety harmonization. *Transportation Research Part A: Policy and Practice*, *136*, 85–98.

Legay, A., Lukina, A., Traonouez, L. M., Yang, J., Smolka, S. A., & Grosu, R. (2019). Statistical model checking. In *Computing and software science* (pp. 478–504). Springer.

Lekidis, A., Bourgos, P., Djoko-Djoko, S., Bozga, M., & Bensalem, S. (2015). Building distributed sensor network applications using bip. In *Sensors applications symposium (sas)* (pp. 1–6).

Li, S. E., Deng, K., Zheng, Y., & Peng, H. (2015). Effect of pulse-and-glide strategy on traffic flow for a platoon of mixed automated and manually driven vehicles. *Computer-Aided Civil and Infrastructure Engineering*, *30*(11), 892–905.

Liang, K.-Y., Mårtensson, J., & Johansson, K. H. (2015). Heavy-duty vehicle platoon formation for fuel efficiency. *IEEE Transactions on Intelligent Transportation Systems*, *17*(4), 1051–1061.

Liu, J., Kockelman, K. M., Boesch, P. M., & Ciari, F. (2017). Tracking a system of shared autonomous vehicles across the austin, texas network using agent-based simulation. *Transportation*, *44*(6), 1261–1278.

Ma, X., Rinast, J., Schupp, S., & Gollmann, D. (2014). Evaluating on-line model checking in uppaal-smc using a laser tracheotomy case study. In *5th workshop on medical cyber-physical*

systems.

Magni, L., Raimondo, D. M., Bossi, L., Dalla Man, C., De Nicolao, G., Kovatchev, B., & Cobelli, C. (2007). *Model predictive control of type 1 diabetes: an in silico trial.* SAGE Publications.

Magni, L., Raimondo, D. M., Man, C. D., Breton, M., Patek, S., De Nicolao, G., ... Kovatchev, B. P. (2008). Evaluating the efficacy of closed-loop glucose regulation via control-variability grid analysis. *Journal of diabetes science and technology*, *2*(4), 630–635.

Mamdani, E. H. (1974). Applications of fuzzy algorithms for control of simple dynamic plant. *Proc. Iee*, *121*, 1585–1588.

Man, C. D., Micheletto, F., Lv, D., Breton, M., Kovatchev, B., & Cobelli, C. (2014). The uva/padova type 1 diabetes simulator: new features. *Journal of diabetes science and technology*, *8*(1), 26–34.

Marsh, B., White, M., Morton, N., & Kenny, G. (1991). Pharmacokinetic model driven infusion of propofol in children. *BJA: British Journal of Anaesthesia*, *67*(1), 41–48.

Mauseth, R., Hirsch, I. B., Bollyky, J., Kircher, R., Matheson, D., Sanda, S., & Greenbaum, C. (2013). Use of a "fuzzy logic" controller in a closed-loop artificial pancreas. *Diabetes technology & therapeutics*, *15*(8), 628–633.

Mediouni, B. L., Nouri, A., Bozga, M., Dellabani, M., Legay, A., & Bensalem, S. (2018). 𝒮BIP 2.0: Statistical model checking stochastic real-time systems. In *International symposium on automated technology for verification and analysis* (pp. 536–542).

Messori, M., Incremona, G. P., Cobelli, C., & Magni, L. (2018). Individualized model predictive control for the artificial pancreas: In silico evaluation of closed-loop glucose control. *IEEE Control Systems Magazine*, *38*(1), 86–104.

Minopoli, S., & Frehse, G. (2016). Sl2sx translator: from simulink to spaceex models. In *Proceedings of the 19th international conference on hybrid systems: Computation and control* (pp. 93–98).

Montanaro, U., Dixit, S., Fallah, S., Dianati, M., Stevens, A., Oxtoby, D., & Mouzakitis, A. (2019). Towards connected autonomous driving: review of use-cases. *Vehicle system dynamics*, *57*(6), 779–814.

Murthy, D. K., & Masrur, A. (2016). Braking in close following platoons: The law of the weakest.

In *Euromicro conference on digital system design (dsd)* (pp. 613–620).

Myles, P. S., & Cairo, S. (2004). Artifact in the bispectral index in a patient with severe ischemic brain injury. *Anesthesia & Analgesia*, *98*(3), 706–707.

Mysore, V., & Pnueli, A. (2005). Refining the undecidability frontier of hybrid automata. In *International conference on foundations of software technology and theoretical computer science* (pp. 261–272).

Ogurtsova, K., da Rocha Fernandes, J., Huang, Y., Linnenkamp, U., Guariguata, L., Cho, N. H., . . . Makaroff, L. (2017). Idf diabetes atlas: Global estimates for the prevalence of diabetes for 2015 and 2040. *Diabetes research and clinical practice*, *128*, 40–50.

Okamoto, M. (1959). Some inequalities relating to the partial sum of binomial probabilities. *Annals of the institute of Statistical Mathematics*, *10*(1), 29–35.

Ouchani, S., Jarraya, Y., Mohamed, O. A., & Debbabi, M. (2012). Probabilistic attack scenarios to evaluate policies over communication protocols. *J. Softw.*, *7*(7), 1488–1495.

Ouchani, S., Mohamed, O. A., & Debbabi, M. (2014). A formal verification framework for sysml activity diagrams. *Expert Systems with Applications*, *41*(6), 2713–2728.

Pajic, M., Mangharam, R., Sokolsky, O., Arney, D., Goldman, J., & Lee, I. (2012). Model-driven safety analysis of closed-loop medical systems. *IEEE Transactions on Industrial Informatics*, *10*(1), 3–16.

Papadoulis, A., Quddus, M., & Imprialou, M. (2019). Evaluating the safety impact of connected and autonomous vehicles on motorways. *Accident Analysis & Prevention*, *124*, 12–22.

Peters, T., & Haidar, A. (2018). Dual-hormone artificial pancreas: benefits and limitations compared with single-hormone systems. *Diabetic Medicine*, *35*(4), 450–459.

Pfeiffer, E.-F., Thum, C., & Clemens, A. (1974). The artificial beta cell-a continuous control of blood sugar by external regulation of insulin infusion (glucose controlled insulin infusion system). *Hormone and Metabolic Research*, *6*(05), 339–342.

Platzer, A., & Quesel, J.-D. (2008). Keymaera: A hybrid theorem prover for hybrid systems (system description). In *International joint conference on automated reasoning* (pp. 171–178).

Poitout, V., Moatti-Sirat, D., Reach, G., Zhang, Y., Wilson, G., Lemonnier, F., & Klein, J. (1993). A glucose monitoring system for on line estimation in man of blood glucose concentration

using a miniaturized glucose sensor implanted in the subcutaneous tissue and a wearable control unit. *Diabetologia*, *36*(7), 658–663.

Rajamani, R. (2011). *Vehicle dynamics and control*. Springer Science & Business Media.

Rajamani, R., Tan, H.-S., Law, B. K., & Zhang, W.-B. (2000). Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons. *IEEE Transactions on Control Systems Technology*, *8*(4), 695–708.

Renard, E. (2008). *Insulin delivery route for the artificial pancreas: subcutaneous, intraperitoneal, or intravenous? pros and cons.* SAGE Publications.

Sahinovic, M. M., Struys, M. M., & Absalom, A. R. (2018). Clinical pharmacokinetics and pharmacodynamics of propofol. *Clinical pharmacokinetics*, *57*(12), 1539–1558.

Sankaranarayanan, S., Kumar, S. A., Cameron, F., Bequette, B. W., Fainekos, G., & Maahs, D. M. (2017). Model-based falsification of an artificial pancreas control system. *ACM SIGBED Review*, *14*(2), 24–33.

Schiavon, M., Dalla Man, C., & Cobelli, C. (2017). Modeling subcutaneous absorption of fast-acting insulin in type 1 diabetes. *IEEE Transactions on Biomedical Engineering*, *65*(9), 2079–2086.

Schnider, T. W., Minto, C. F., Gambus, P. L., Andresen, C., Goodale, D. B., Shafer, S. L., & Youngs, E. J. (1998). The influence of method of administration and covariates on the pharmacokinetics of propofol in adult volunteers. *Anesthesiology: The Journal of the American Society of Anesthesiologists*, *88*(5), 1170–1182.

Schupp, S., Ábrahám, E., Chen, X., Makhlouf, I. B., Frehse, G., Sankaranarayanan, S., & Kowalewski, S. (2015). Current challenges in the verification of hybrid systems. In *International workshop on design, modeling, and evaluation of cyber physical systems* (pp. 8–24).

Schupp, S., Leofante, F., Behr, L., Ábrahám, E., & Taccella, A. (2022). Robot swarms as hybrid systems: Modelling and verification. *arXiv preprint arXiv:2207.06758*.

Sen, K., Viswanathan, M., & Agha, G. (2004). Statistical model checking of black-box probabilistic systems. In *International conference on computer aided verification* (pp. 202–215).

Shannon, R. E. (1975). *Systems simulation; the art and science* (Tech. Rep.).

Shichiri, M., Yamasaki, Y., Kawamori, R., Hakui, N., & Abe, H. (1982). Wearable artificial endocrine pancreas with needle-type glucose sensor. *The Lancet*, *320*(8308), 1129–1131.

Skorobogatjko, A., Romanovs, A., & Kunicina, N. (2014). State of the art in the healthcare cyber-physical systems. *Information Technology and Management Science*, *17*(1), 126–131.

Slattery, D., Amiel, S., & Choudhary, P. (2018). Optimal prandial timing of bolus insulin in diabetes management: a review. *Diabetic Medicine*, *35*(3), 306–316.

Specification, O. A. (2007). *Omg systems modeling language (omg sysml™), v1. 0.* Object Management Group.

Steil, G., Rebrin, K., & Mastrototaro, J. J. (2006). Metabolic modelling and the closed-loop insulin delivery problem. *Diabetes research and clinical practice*, *74*, S183–S186.

Steil, G. M. (2013). Algorithms for a closed-loop artificial pancreas: the case for proportional-integral-derivative control. *Journal of diabetes science and technology*, *7*(6), 1621–1631.

Todd, J. A. (2010). Etiology of type 1 diabetes. *Immunity*, *32*(4), 457–467.

Van Belle, T. L., Coppieters, K. T., & Von Herrath, M. G. (2011). Type 1 diabetes: etiology, immunology, and therapeutic strategies. *Physiological reviews*, *91*(1), 79–118.

Vladimerou, V., Prabhakar, P., Viswanathan, M., & Dullerud, G. (2008). Stormed hybrid systems. In *International colloquium on automata, languages, and programming* (pp. 136–147).

Wald, A. (2004). *Sequential analysis*. Courier Corporation.

W Axhausen, K., Horni, A., & Nagel, K. (2016). *The multi-agent transport simulation matsim*. Ubiquity Press.

Weaver, K. W., & Hirsch, I. B. (2018). The hybrid closed-loop system: evolution and practical applications. *Diabetes technology & therapeutics*, *20*(S2), S2–16.

Younes, H. L., & Simmons, R. G. (2002). Probabilistic verification of discrete event systems using acceptance sampling. In *International conference on computer aided verification* (pp. 223–235).

Younes, H. L. S. (2004). *Verification and planning for stochastic processes with asynchronous events*. Carnegie Mellon University.

Zheng, R., Nakano, K., Yamabe, S., Aki, M., Nakamura, H., & Suda, Y. (2014). Study on

emergency-avoidance braking for the automatic platooning of trucks. *Transactions on Intelligent Transportation Systems*, *15*(4), 1748–1757.

Zheng, R., Nakano, K., Yamabe, S., & Suda, Y. (2013). Safety evaluation of system failures in formation and separation processes of automatic platooning of trucks. In *20th its world congress, japan.*

Ziegler, J. G., Nichols, N. B., et al. (1942). Optimum settings for automatic controllers. *trans. ASME*, *64*(11).

Appendix A

Meal Absorption Model

The following system of equations govern the glucose rate of appearance (r_{ag}) from ingestion of meal:

$$Q'_{sto1} = -k_{gri}.Q_{sto1} + meal_{intake}(t) \tag{A.1}$$

$$Q'_{sto2} = -k_{empt}(Q_{sto1} + Q_{sto2}).Q_{sto2} + k_{gri}.Q_{sto1} \tag{A.2}$$

$$Q'_{gut} = -k_{abs}.Q_{gut} + k_{empt}(Q_{sto1} + Q_{sto2}).Q_{sto2} \tag{A.3}$$

$$k_{empt}(q) = k_{min} + \frac{(k_{max} - k_{min})}{2}.[tanh(\alpha.(q - b.D)) - \\ tanh(\beta.(q - c.D)) + 2] \tag{A.4}$$

$$r_{ag} = \frac{f.k_{abs}.Q_{gut}}{BW} \tag{A.5}$$

where $meal_{intake}$ is the amount of glucose intake over time (mg), r_{ag} is the glucose rate of appearance in the blood (mg/Kg/s), Q_{sto1} is the glucose in the stomach in solid state (mg), Q_{sto2} is the glucose in the stomach in liquid state (mg), Q_{gut} is the glucose in the intestine (mg), BW is the body weight (Kg), f and the K_x parameters are patient-specific constants.

Appendix B

Glucose-Insulin Response Model

The following system of equations govern the physiological dynamics of glucose and insulin in the body of T1D patients in the existence of subcutaneous insulin injection:

$$I'_{sc1} = -(k_d + k_{a1}).I_{sc1} + IIR(t) \tag{B.1}$$

$$I'_{sc2} = k_d.I_{sc1} - k_{a2}.I_{sc2} \tag{B.2}$$

$$X' = -p_{2u}.X + p_{2u}.(\frac{I_p}{V_I} - I_b) \tag{B.3}$$

$$Gs' = -\frac{1}{T_s}.(G_s - \frac{G_p}{V_G}) \tag{B.4}$$

$$I'_1 = -k_i.(I_1 - \frac{I_p}{V_I}) \tag{B.5}$$

$$I'_d = -k_i.(I_d - I_1) \tag{B.6}$$

$$I'_l = -(m_1 + \frac{m_6.m_1}{1 - m_6}).I_l + m_2.I_p \tag{B.7}$$

$$I'_p = -(m_2 + m_4).I_p + m_1.I_l + k_{a1}.I_{sc1} + k_{a2}.I_{sc2} \tag{B.8}$$

$$G'_p = k_{p1} - k_{p2}.G_p - k_{p3}.I_d - k_{e1}.max(0, G_p - k_{e2}) -$$
$$F_{cns} - k_1.G_p + k_2.G_t + r_{ag} \tag{B.9}$$

$$G'_t = -\frac{V_{m0} + V_{mx}.X}{K_{m0} + K_{mx}.X + G_t}.G_t + k_1.G_p - k_2.G_t \tag{B.10}$$

$$G = \frac{G_p}{V_G} \tag{B.11}$$

where I_{sc1} and I_{sc2} are the amounts of non-monomeric and monomeric insulin in the subcutaneous space (pmol/Kg) respectively, IIR is the exogenous insulin injection rate (pmol/Kg/min), X is the insulin concentration in the interstitial fluid, G and G_s are the blood and subcutaneous glucose concentrations (mg/dL) respectively, G_p and G_t are the glucose amounts in the plasma and the slowly equilibrating tissues (mg/Kg) respectively, I_p is the plasma insulin concentration, I_l is the portal vein insulin concentration, I_d is the delayed insulin signal, $\{F_{cns}, k_x, m_x, p_x\}$ are parameters for patient-specific constants, and r_{ag} is the glucose rate of appearance (mg/Kg/s) from meal ingestion.