Working Groups' Report: The Challenge of Combining Simulation and Verification

Gregory Batt, Jeremy Bradley, Roland Ewald, Francois Fages, Holger Hermans, Jane Hillston, Peter Kemper, Alke Martens, Pieter Mosterman, Flemming Nielson, Oleg Sokolsky, Adelinde Uhrmacher, and all other participants of the Dagstuhl seminar

1 Motivation for the Seminar

Simulation has found widespread use for experimentation and exploration of the possible impacts of a variety of conditions on a system. In contrast, formal verification is concerned with proving or disproving the correctness of a system with respect to a certain property, using mathematical and logical methods.

Historically, simulation and verification evolved as two distinct disciplines. Simulation, artificially constructing parts of the system or its environment, aims to explore the effect of certain design decisions or environmental conditions on system behavior. By contrast, formal verification is concerned with proving or disproving the correctness of a system with respect to a certain property, using mathematical and logical methods.

Despite of these different objectives, the fields of simulation and verification address similar research challenges. In particular, the need for identifying and defining suitable models of a dynamic system under study unifies both research fields, although modeling paradigms and employed validation strategies vary. There are disparate approaches in the fields of simulation and verification for validating timed, probabilistic, and hybrid systems. Both fields address component-based and abstraction-based validation techniques and face some of the same challenges – issues of state space explosion and complexity are common.

Simulation and verification are currently moving closer together. Simulators are extended with bounded model checking techniques for increasing state-space coverage, and recently developed model checking techniques employ event-based simulation techniques. Combinations of explicit-state simulation with symbolic-state exploration should make it possible to not only increase coverage of explicit-state simulation, but also to use localized symbolic simulation to drive simulations to territory in the state space that would otherwise remain unexplored, e.g. [11]. Verification techniques can

Dagstuhl Seminar Proceedings 06161
Simulation and Verification of Dynamic Systems
http://drops.dagstuhl.de/opus/volltexte/2006/724

also be used to generate simple invariants, which are used to restrict the search space for explicit exploration.

Particularly, in application areas like systems biology, researchers of both fields have started identifying common problems and re-usable solutions.

2 Systems Approach

If concerned with modeling, simulating, or verifying a dynamic system, we have to decide what general model class appears most suitable for the system under study. The question has to be answered whether continuous systems models, discrete event systems models, discrete stepwise systems models, or hybrid models that combine continuous and discrete approaches will be utilized [24]. Different model classes can be distinguished referring to the time base (continuous or discrete), the state base (continuous, discrete or both), and the number of interesting state changes within one time interval (finite or infinite) [3].

During the seminar, discrete-event, e.g. [33, 41, 32, 40], and hybrid modeling approaches, e.g. [25, 31, 38, 42], prevailed. However, discrete stepwise approaches, e.g. [26], were also presented. Purely continuous approaches were only discussed in relating deterministic macro models and discrete event micro models, e.g. [29, 27].

3 Modeling, Simulation, Validation, and Verification

The validity of models and the suitability of modeling approaches is dependent on the system under study and the objective of the simulation or verification study. Modeling means structuring knowledge about a given system. Diverse modeling approaches exist, discrete event approaches comprise timed automata [32], DEVS [41], term rewriting [30], and process algebras. The latter embrace various extensions of stochastic PI [29, 36, 29] and PEPA [28, 33]. The extensions of stochastic PI are aimed at facilitating re-use by e.g. introducing abstract data types [36], or visual means for modeling [29]. Following the argumentation line of Wittgenstein "the limits of my language are the limits of my thoughts", the particular strength and limitations of the different modeling approaches were discussed in detail, which helped foster a working group at the seminar (see 6.3).

Although discrete event approaches dominated the seminar, also continuous approaches played a role, e.g. to compare modeling approaches with [29, 27], for a more efficient execution [28, 33], or for combining discrete and continuous approaches [31, 37]. In [37], background traffic is produced by continuous models, whereas that part of the dynamic system that will be explored in detail is represented by a discrete event approach. These types

of hybrid models and, more generally, multi-level models allow to focus on specific parts of the system on demand and are of particular interest for systems biology [41].

Discrete event models do not necessarily mean a more concrete view of the system, even though this is the case if combined with turning from a macro to a micro view of the system under study, as is typically done in systems biology [7]. So it is important to note that "eo ipso" a continuous approach is not more abstract than a discrete event approach, nor vice versa. Similarly, moving from a deterministic to a stochastic model might imply a more abstract or a more concrete view on the system and its processes. How different models are related by abstraction and refinement and what effect this has on the simulation and verification has been subject of the second working group (see 6.2).

Although modeling formalisms have typically roots either in simulation or verification, the same modeling formalisms are used increasingly, e.g. in PEPA. Still, the observation seems to hold that simulation is used for more complex models, if we define complexity in terms of number and heterogeneity of subsystems and their interactions involved. Often simulation is regarded as the technique of last resort when nothing else works.

It seems comparatively easy for the communities of simulation and verification to settle on a joint understanding of models, and important aspects in modeling. Both communities even share controversies, like whether one modeling formalism will suffice or multi-formalism modeling is necessary to address the varying needs in describing complex systems. However, the terms simulation and verification are quite differently used in both communities. In the verification realm, simulation is often used synonymously with continuous simulation, ignoring the rich pool of methods that have been developed for discrete event modeling and simulation. In the context of simulation, *verification* often means simply checking certain properties of the system by analyzing simulation trajectories or the model structure. Verification does neither imply the use of logical descriptions and formal inference engines, nor the derivation of general properties

Whereas a different interpretation of simulation and verification in the different fields as well as the difference of exploited methods still hamper an integration of both areas, the joint desire to contribute to a better understanding of dynamic systems and joint application fields like systems biology help to bridge these differences.

4 A New Area for Verification and Simulation

The goal of systems biology is to analyze the behavior and interrelationships between entities of entire functional biological systems [12]. As the systems under study do not support an easy experimental access and analysis, models

play an important role in gaining insight into the system's behavior and structure [5]. Models are evaluated by analysis, verification, e.g. [14, 20], and simulation. Thereby, as an experimental technique, simulation takes on the role of the *in-silico* pendant to the *in-vivo* and *in-vitro* experiments.

Although continuous modeling approaches still prevail, discrete event and hybrid approaches increasingly gain ground. Generally, continuous and deterministic macro models represented as differential equations can be translated into stochastic, discrete-event micro models – or be executed as such, as Gillespie showed in the 1970s [9]. Gillespie showed that, based on exponential distributions and propensities, chemical reactions can be correctly simulated. With this, the plethora of modeling approaches and verification tools based on Markov Chains have found a new and exciting application field. They form an alternative to the traditional continuous simulation. Moreover, it has to be noted that certain phenomena depend on stochasticity. As shown by [29], sometimes a deterministic model does not reproduce characteristic behavior patterns of a real system, like oscillations, whereas a stochastic model does.

In addition, the combination of discrete and continuous approaches is starting to play a major role. Many mechanisms, e.g. enzymatic reactions, can naturally be described as continuous macro models. However, other phenomena such as gene regulation exhibit stochastic behavior, which is best captured by a discrete event approach. So it seems natural to combine different approaches for describing cellular systems.

Are biological systems by any means different from systems like computer networks [37, 39] or hardware [34], and does their specification, i.e. modeling, or their evaluation in terms of verification or simulation, require special methods? These questions found sufficient interest to be discussed in a separate working group (see 6.1).

5 Combinations of Simulation and Verifications

Interestingly the question of how simulation and verification can be combined was not the subject of the working groups during the seminar, but rather demonstrated in different talks [32, 39, 31]. In combining simulation and verification, different approaches can be distinguished, e.g.

- Verification guides the simulation. Verification is used to determine suitable parameters for executing simulations [32], in this context counter examples play a central role.
- Simulation is used to verify certain properties. Individual trajectories of simulation runs are checked to ascertain whether certain properties hold or are violated [31, 39],

- Verification of certain properties uses many systematic simulation experiments [31] internally. This approach can also be used to identify suitable parameters.

Each of these was the subject of different talks given during the seminar.

5.1 Guiding simulation by verification

In different talks [32, 26] approaches were presented to guide the simulation by verification. One of them [32] uses *model checking* to guide *discrete event simulation*. The mathematical workhorse for this approach is a nondeterministic *and* stochastic model, in particular the model of a stochastic timed automata. These models are rich in modeling power, which makes them well-suited for diverse application areas. However, discrete-event-simulation can be used in a meaningful way only if nondeterminism is resolved by some means. This is where model-checking comes into play, providing examples (or counterexamples) of interesting behavior sequences to be studied via simulation.

5.2 Checking properties of simulation trajectories

Another possibility is to check single trajectories. Using formal run-time verification techniques in conjunction with very detailed simulations of ad hoc routing protocols in the NS network simulator, violations of significant high-level protocol properties could be identified. Utilizing the interplay between the simulation and verification engines facilitates debugging of the model.

5.3 In-between simulation and verification

Two types of models are used in this approach – one model is the typical simulation model that describes the rules of behavior. The other model is comprised of logical statements that refer to general behavior observed of the system under study. The Biochemical Abstract Machine BIOCHAM [4] brings both together.

To a large extent, it can be shown that:

- biochemical systems can be described as transition systems (of different kinds, with discrete or continuous dynamics),
- the biological properties known from experiments can be formalized in Temporal Logic (propositional or with numerical constraints)
- and biological validation amounts to model checking in this setting.

This approach is used in BIOCHAM, for searching parameter values and learning reaction rules from temporal logic properties. This has been illustrated with models of the cell cycle control.

6 Working Groups

In addition to talks, working groups form an intrinsic part of each Dagstuhl seminar, the themes in this case formed during the first days of the seminar. One of the group addressed the question whether the application area of systems biology requires specific modeling, simulation, and verification tools, and how biological systems differ from engineered ones. Closely related to the question of simulation and verification is the question of refinement and abstraction, which was the subject of the second group. Refinement and abstraction plays a crucial role, both in simulation and verification, but even more so if both approaches are combined, as previous talks showed [31, 32, 39]. Modeling methods have a significant impact on how easily certain phenomena can be described, influence the acceptance in the application community, and the possibilities to be analyzed and simulated. The third working group was dedicated to exploring the potentials and limitations of different modeling approaches.

6.1 Working Group – Why are biological systems difficult to model?

6.1.1 Introduction

Both science and engineering have come to rely extensively on computational models. Models are used to structure, relate and exchange information, they are the basis for simulation and communication. A model reflects a certain interpretation of the investigated system. Moreover, a developed model is most of the time related to a certain purpose, which itself may be reflected in the type of model, in the level of abstraction and aggregation, and in the representation of the model. Models that are simulated over time are used, for example, to test hypotheses about dynamic behavior of systems, and to predict the behavior of the investigated system. Model validation is another basic task to perform since one needs to assess the validity of a model before using it as an explanatory or a prediction tool.

Nowadays, modeling often is related to the development of models that are executed on a computer. Such a computer based simulation of a model can help to analyze the behavior of a dynamic system, which is of significant interest in present-day research. Especially in the investigation of biological systems, the use of computers in modeling, simulation, and verification has pushed research forward and played a key role in the emergence of new disciplines such as systems biology.

Given these developments, computational modeling has become the center of analysis of dynamical systems. As such, it has become a task of critical importance, which justifies efforts that take a relatively long period of time, i.e., years, to obtain a model. To successfully exploit modeling calls for reducing the required effort by structuring the modeling activity and providing the modeler with as much support as possible.

Modeling is an activity that is situated somewhere within the spectrum of theoretical modeling from first principles [8] and data-driven modeling approaches, such as system identification [15]. Clearly, some systems are more difficult to model than others. In contrast to systems that have behavior governed by inherent domain laws that are well studied and understood [17], there are systems that have been found to be much more difficult to capture in a rigid mathematical representation which ought to express the behavior specific enough for the purposes of the modeler. Such systems are, for example, biological systems, the stock market, and the atmosphere. The purpose of modeling such systems may be understanding cell behavior, prediction of cell response to a perturbation, predicting stock market development, and forecasting the weather. This study aims to understand where this discrepancy in modeling difficulty stems from, in particular with respect to systems biology.

6.1.2 Modeling: Control Engineering of Biological Systems?

Modeling is a pervasive activity in science and engineering. In particular in embedded control systems design, modeling is used for both capturing the behavior of the controller as well as the device to be controlled [2]. An embedded control system typically consists of a controller and a controlled device, both of which are modeled (albeit in general by different modeling formalisms). The device under control could be something mechanical, such as a power window [16]. Fairly systematic approaches exist to deal with these models, as there exists a good understanding of the systems they represent. One of the reasons for this is that they have to satisfy a manifold of domain constraints (conservation of energy and momentum, continuity of power, etc.), which are often known in advance. Furthermore, these systems are specifically designed to exhibit an intended behavior.

Whereas engineered systems are designed to adhere to prescribed behavior, biological systems are the result of an evolutionary process. Therefore, biological systems seem to lack the clear structure that can be found in engineered systems. One of the goals of systems biology is to find possibly existent underlying principles of biological systems. These principles may be very different from engineered systems. For example, Rosen argued that biological systems cannot be completely simulated, because the way they are controlling themselves cannot be reproduced on a computer [19, 23]. To some extent, the relation between neural networks and expert systems [18] in

computer science resembles the relation of biological systems and engineered systems. A neural network represents implicit knowledge, its behavior cannot be systematically derived and is difficult to explain.

In contrast, the expert knowledge model is developed based on explicit knowledge, which allows to shed light on the solution process and therefore makes the expert system's behavior observable and reconstructible. Although the position of biological systems with respect to these two extreme cases (information completely implicit or completely explicit) is not yet clearly established, a biological system adapts its behavior similarly to a neural network. A biological system has come about by responding to stimuli so that it would thrive. If it did not produce beneficial behavior, it would wither instead. The evolutionary selection process resulted in implicit knowledge of appropriate responses – a knowledge which is difficult to grasp, as it relies on slight adaptations and not on explicit rules. On the other hand, it is increasingly clear that biological systems obey design principles, such as the use of modularity [5].

The control engineering system has been developed based on initially identified and fixed rules and will not exhibit behavior other than what is intended. Any other behavior can be interpreted as erroneous. Moreover, whereas the biological system can exhibit unpredictable behavior, the control engineering system should not.

An engineered system is typically constructed to satisfy a plurality of well-defined requirements. In order to produce a system that satisfies these requirements, engineers use certain techniques that allow them to compose complex systems [22]:

- Modularity,
- Partitioning,
- Hierarchical design.

Modularity requires to rigidly define interfaces. This is important in cases where the system as a whole is too complex to understand or to model. Modularization, i.e. to break something down into smaller pieces, shall help to grasp an inherent structure and is a tool to cope with complexity. Interfaces are necessary to define a system as a set of relatively independent entities[1].

Biological systems are often interpreted to present a hierarchical organization and functioning. In the field of systems biology, the hierarchical organization is revealed by studies focusing on different types of networks:

[1]We expressly distinguish between functional and behavioral. Often, a system is considered to consist of two aspects: behavioral and structural. A certain functionality can be attributed to both aspects.

metabolic networks, protein-interaction network, signal-transduction networks, and genetic regulatory networks [1]. It has also been proposed that functional modules are a critical level of biological organization [5]. However, it is not yet clear how general this modular organization is, nor how it could be exploited in a systematic way. Even if biological systems, like engineered systems, present some degree of modularity and hierarchical organization, a major difference between biological and engineered systems is that the separation between the different entities appears to be much less rigid for biological systems. Stated differently, it is in general difficult, if not impossible, to identify well-defined interfaces for biological systems. As a consequence, these characteristics cannot be easily exploited in the modeling and analysis process.

As opposed to biological systems, the interface definitions of an engineered system can be seen as explicit knowledge about the functionality of its entities. Therefore, control engineering employs a well-defined system structure, which consists of a *controller* and a *controlled device*. This allows the controller to fulfill the prescribed behavior by analyzing past output to compute the input [2] etc. This means, the compensating effect typically induced by a controller can be separated from the controlled device.

This concept transfers into the biological world as well (e.g. homeostasis, or see the approach taken in [13]). However, it is not possible to pinpoint as clearly which elements belong to the controller and which belong to the controlled device. Many components are involved in the regulation processes, and regulation typically takes place at different levels (e.g. metabolic and genetic, in the case of the tryptophan biosynthesis pathway). The control is distributed over a large number of heterogeneous components.

Does the inability to separate controller and controlled device increase the intricacy of modeling? As these two entities cannot be clearly separated, one has to deal with them in a combined manner. This is extremely complicated, since effects of interference might occur, which may be difficult to identify. Biased observation is a prominent problem in many domains. For example, the observers of a psychological group experiment must not be part of the group, because this would have an impact on the results of the experiment (e.g., expectancy effects).

Similarly, the inherent control of a biological system will interfere with the measurement exaltation, which complicates modeling, because one has to deal with the whole system. This makes techniques such as redundancy control (e.g., inhibiting the expression of a single gene) very challenging.

6.1.3 Technological Problems

In other domains, the extensive recording of experimental data has been proven to be a viable solution to obtain an understanding of controller and controlled device. This separation is achieved by taking an extensive amount

of measurements from the interfaces of controller and controlled system (numerical isolation).

In the context of biological systems, on a (macro-) molecular level, making observations is particularly challenging. For example, many experiments cannot be conducted *in vivo*, so that *in vitro* measurements are used. Furthermore, a biological system has to be considered as a multiple-input and multiple-output system in terms of control theory (e.g., one protein regulates several genes and *vice versa*), which complicates the interpretation of experimental data.

Even though these problems may be solved in the future, it has to be asked whether numerical isolation of controller and controlled device is possible in principle. Our observation is that with sufficient data of certain critical variables, we *may* be able to numerically analyze the system. Control engineering techniques have been successfully applied to other applications, notably aerospace and automotive control systems. This methodology may well transfer successfully into the biological domain. At least, it should be investigated if this is possible.

6.1.4 Cultural Problems

It is generally accepted that each scientific discipline entertains and prefers specific viewpoints, methodologies, and techniques; in other words, a scientific culture. While this is not a problem in itself, it may complicate the creation of computational models by causing miscommunication and the like.

With respect to the application of modeling and simulation in control engineering, a mathematically concise description of a system's entities can be regarded as the common ground of both domains (e.g., finite state machines, FSM, to model controllers, ordinary differential equations, ODE, to model controlled devices, or hybrid approaches to combine both [21]). The availability of suitable modeling formalisms to combine computer science, control engineering, and physics is made possible by the *mathematical* foundations employed in these domains. Computer science, for example, relies heavily on discrete mathematics (e.g., graph theory, combinatorics, etc.). Similarly, physics models are often expressed by a set of equations (e.g., ODEs). In biology, however, it seems as if there is no unique formalism which is able to capture the behavior of the systems thoroughly. For example, for modeling of genetic regulatory networks, formalisms as diverse as graphs, logical models, ODEs, or stochastic master equations have been proposed [5]. One may wonder whether this lack of unifying formalism reflects the immaturity of systems biology or the tremendous complexity of biological systems.

Even though there are formalisms to describe a certain aspect of a biological system, a biologist who is untrained with respect to mathematics may fail to formalize a biological system, whereas a computer scientist may not

have a sufficient understanding of the biological system, and, therefore also be unable to formalize it. The present approach to education exacerbates this problem by relying on very domain-specific courses, instead of regarding interdisciplinarity as a prerequisite for future research in these areas.

6.1.5 Conclusions

This study identified several problems that hamper the application of computational modeling, simulation, and verification for biological systems. First of all, intrinsic problems of these systems have been discussed. Then, some technological issues were briefly described and it was proposed to investigate the feasibility of control engineering methods in the context of biological systems. A number of reasons for difficulties in understanding and communicating between researchers with different backgrounds have been pointed out, as they were experienced in our working group as well (e.g., the scientific perspective in contrast to the engineer's point of view).

Still, the new application area of modeling, simulation, and verification may also be extremely beneficial in different contexts: While biologists try to gain deeper insights into the biological systems they study, these insights may be used to engineer systems that exhibit similar properties, such as robustness and adaptivity.

6.2 Working Group - Refinement and Abstraction

6.2.1 State of the art

Abstractions are useful. They save space and time both in simulation and verification, as well as improving clarity by concentrating on relevant aspects of behavior.

During this Dagstuhl seminar, the usefulness and potential of abstraction has become apparent in a number of presentations:

- Simulation trace analysis by verification techniques [31, 39]
- Abstraction refinement from verification and derivation of conceptual model from detailed simulation [31]
- Pruning of search space by exploiting abstract model of hybrid or discrete system [38, 26] by guiding simulation via generating schedules [32]
- Switching between levels of abstraction dynamically - possibly property guided.
- Refinement of an abstraction by counterexamples as a form of simulation

Known successful applications of abstractions include conversions between discrete (stochastic) and continuous, integer to reals (relaxation etc.), finite to infinite (queuing theory), infinite to finite (integer and real to boolean).

6.2.2 The "V" in abstraction and refinement

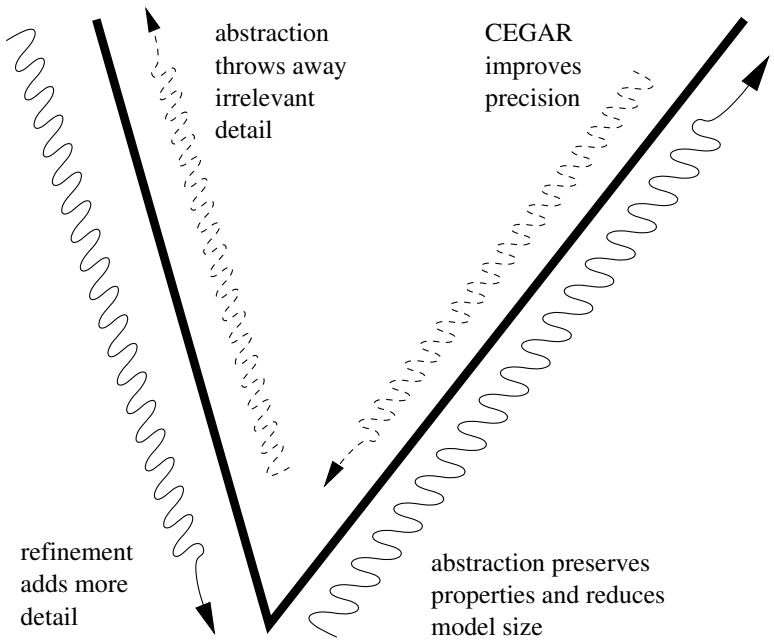

Figure 1: The V in modeling and analysis

The process of modeling and analysis can be visualized by the following V-shaped diagram (not to be confused with the other V diagrams out there). V reconciles two notions of abstraction and refinement used in modeling, simulation, and verification.

The two sides of the diagram represent different tradeoffs in the modeling process, of different significance for simulation and verification. The left side reflects the amount of detail in a model. Abstract models have little detail, keeping them small but imprecise; additional detail increases predictive power of the model at the cost of increase in the model size and hence computational requirements of the model. In simulation, the accuracy of model predictions is of primary importance, and the associated increase in the size of the model is relatively tolerable. Thus, a typical modeling scenario starts with an abstract model and moves downward along the left side of the diagram until sufficient detail has been added. The reverse direction plays a secondary but important role in making a sufficiently predictive model more computationally feasible. The right side of the diagram

represents abstraction in the sense of abstract interpretation, which is an all-important technique in verification. Abstract interpretation significantly reduces model size, yet allows interpretation of verification results: a property, satisfied in the abstract model, is also satisfied by the concrete model. The reverse is not true, however, resulting in spurious counterexamples. A typical verification scenarion, then, starts with a concrete model and performs abstraction trying to reduce the model size that is manageable by verification tools. The reverse direction here also plays secondary but important role: counterexample-guided abstraction refinement (CEGAR) hels to add just enough detail into the abstract model to get rid of spurious counterexamples.

6.2.3 What is lacking from the current approaches?

- Abstraction aimed at improved analyzability is often not doing a good enough job.
- Moving between abstraction levels is often in part magic.
- Advantages of abstraction in simulation are under-explored.
- There is a distance between different approaches
  - incompatible abstractions
  - conflicting assumptions
  - different purposes for abstraction

  Often these purposes and assumptions are not documented, which makes it harder to interpret results obtained via different level models.
- There is no hope for a general-purpose abstraction, because abstraction by definition throws away details. It is always a property-driven abstraction. The idea of taking the property explicitly has not been taken seriously in simulation but it has in verification (cp. artial order techniques).
- The use of abstraction is not yet an engineering discipline. It is an art, which requires ingenuity and custom-built solutions in most new applications. At the same time, there are scientific disciplines - in particular, control theory - where abstraction as means of model reduction is well understood and widely practiced.
- Some people dream of a hierarchical family of models with well-understood embeddings between them. An example of such hierarchy are finite/timed/hybrid/stochastic automata.

6.3 Working group – Composition and communication and their role in modeling and analysis of dynamic systems

The focus of our group was the consideration of the features of modeling formalisms which allow us to capture interesting and commonly occurring features of dynamic systems. Among the issues which we wanted to discuss were:

- Compositionality,
- Communication patterns,
- Quantitative representation of dynamics.

For each of these points we considered the modeling expression aspects but also the differences that they incur for model evaluation and analysis.

6.3.1 State-of-the-Art

All state-of-the-art formalisms support some form of compositionality, at least at the model construction level. For example,

parallel composition	all process algebras, DEVS, state charts, automata
sequential composition	some process algebras, dynamic DEVS, hybrid automata, *composite states?*

In process algebras, a key feature is the exploitation of the compositionality during analysis. This works well in classical process algebras, but is more difficult in stochastic process algebras where, in general, exploitation of the compositional structure depends on stricter structural conditions.

In addition, in process algebras the compositionality plays a role in abstraction/refinement. You can take any component and replace it by another component with the same observable behavior and know by congruence that the complete model behavior will also be observably equivalent. This can be used for either abstraction or refinement.

In simulation, the compositionality can be used for partitioning the model for parallel discrete event simulation, and for validation of components in isolation. However, note that the consequent "validation" of the complete model is then a matter of trust rather than a formal congruence relation as in process algebras.

Process algebra and model checking techniques can be based on synchronous communication, whereas simulation-oriented formalisms are typically based on asynchronous communication. The difference, at least partly, comes from the focus on *activities* and *communication* or *events*. In discrete-event simulation, the events are the driving force of the model evolution and communication is viewed as the means of event dissemination. In contrast

in the process algebra, communication is a primitive used to capture synchronized activity.

Discrete event simulation has more flexibility to add more features, because the model description formalisms are more expressive, but there is a price to pay for this in terms of problems encountered in achieving coverage and interpretation.

6.3.2 Comparisons

We identified two different forms of abstraction: *modeling abstraction* and *semantic abstraction*. By semantic abstraction we mean a notion of organizational abstraction which chooses the level of discrimination at which you want to represent your system. For example, at the individual or population level. This is somehow orthogonal to the choice of the modeling abstraction, or degree of detail that you include in your model. There is also an option to model at a single level but to execute different parts of your model using different semantics. For example, this is sometimes done to handle stiff systems, where fast actions are modeled in continuous deterministic semantics but the slower actions are captured by discrete stochastic semantics.

For certain phenomena in systems biology, such as binding, synchrony is more natural and some effort is needed to capture this successfully in (asynchronous) discrete event simulation.

If you look at the classical domain, where we use process algebras to describe computer interactions and you have a mediating layer, the network, then low-level modeling seems to naturally include asynchrony and high-level modeling seems to naturally include synchrony. When you move to the systems biology domain, this appears to be reversed: in high-level models asynchrony seems natural and for low-level models synchrony seems natural.

We identified a number of exemplar systems which we felt would be useful to discriminate the modeling capabilities of the different formalisms. There were:

- competitive binding,
- complexation,
- n-ary reactions.

6.3.3 Challenges

A major challenge is to try to ensure that compositionality is preserved as we enrich the formalisms to address the particular needs of new application domains.

Another significant challenge is the construction of quantitative synchronizations in a process algebra setting (and other discrete event formalisms)

which reflect the physical characteristics of the dynamic system, e.g. shape and orientation of cooperating entities, properties of the environment in which the synchronization is happening (temperature, pressure, acidity).

A first approach in this direction can be seen in the work on beta binders [6]. Of course, care is also needed to avoid over-complicating models (and modeling formalisms). Currently quantification in the formal system descriptions is focused on timing and reward-based results, and considerable extensions to existing theory will be needed in order to incorporate other aspects.

We are already starting to see both micro and macro view models emerging, but the details of the relationships between them have not yet been fully resolved. A better understanding here would allow more work on integration of the two views and multi-scale modeling.

Both temporal and spatial heterogeneity are needed for some situations in the systems biology domain (*and others?*). For example, currently models are built assuming a single kinetic semantics for interaction. However there are situations where you would want to be able to mix kinetics, in different interactions within a single system, but also possibly with respect to a particular interaction over time.

7 Conclusions

Computational biology holds a lot of appeal for the simulation and verification community alike. Many of the approaches that have been successfully applied for analyzing computer networks and other systems seem to lend themselves to biological applications. However, whether modularity and hierarchical design are suitable means for constructing biological systems is still controversially discussed [10]. Characteristics of biological systems, requiring highly dynamic multi-resolution and multi-time scale models, challenge methodological developments in modeling, simulation, and verification likewise and thus are likely to propel research in these areas.

Not only due to new emerging application areas like computational biology, research on simulation and verification is currently moving closer together. Joint problems and interest, e.g. how to deal with abstractions, the need for adequate modeling mechanisms, including means for composition and interaction, unify both areas and form the basis of fruitful discussions and joint research.

Compositionality seems one of the central requirements, and ensuring that compositionality is preserved as we enrich the formalisms to address the particular needs of new application domains is one of the challenges. However, more than mere compositionality is needed to soundly relate micro and macro views of dynamic systems as required in computational biology. By micro and macro views, different abstraction levels of the system are

provided, and to move between them is often in part magic. Their combination in one model is hampered by incompatible abstractions, being based on conflicting assumptions and different purposes. Often these purposes and assumptions are not well-documented, which makes it harder to interpret results obtained via different modeling levels. As abstraction is a property-driven endeavor, these properties deserve more attention to facilitate the abstraction process, inter-relating different abstractions, and the re-use and combination of models at different levels of abstraction.

Another significant challenge is the construction of interaction patterns which reflect the physical characteristics of the dynamic system and properties of the environment in which the synchronization is happening without burdening the modeling formalism. For many cell biological systems, synchronous interaction patterns seem currently most natural. However, whether synchronous or asynchronous interactions patterns are more suitable will finally depend on the system and again the level of abstraction. In computer interactions low-level modeling seems to naturally include asynchrony and high-level modeling seems to naturally include synchrony, whereas in computational biology this appears to be reversed.

To conclude, this interdisciplinary application area, the variety of methods and tools available, and the diverse methodological challenges still to be addressed, makes the subject of how to build and analyze models of dynamic systems one of the fascinating areas for computer scientists and should be an integral part of any standard CS curriculum. Crossing the boundaries between traditional simulation and verification, research in the methodological realm will be propelled, and new insights into the dynamic system under study will be achieved. Thus, the number of research projects that combine simulation and verification and explore the inter-relationship is likely to increase significantly within the near future.

8 Literature

8.1 General Literature

[1] E. Alm and A.P. Arkin. Biological networks. *Current Opinion in Structural Biology*, 13(2):193–202, 2003.

[2] Karl J. Åström and Björn Wittenmark. *Computer Controlled Systems: Theory and Design*. Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[3] F. E. Cellier. *Continuous System Modeling*. Springer, New York, 1992.

[4] Nathalie Chabrier-Rivier, François Fages, and Sylvain Sollman. The biochemical abstract machine BIOCHAM. In *Proceedings of the 2nd International Workshop on Computational Methods in Systems Biology, May 26-28, 2004, Paris, France*, 2004.

[5] H. de Jong. Modeling and Simulation of Genetic Regulatory Systems: A Literature Review. *Journal of Computational Biology*, 9(1):67–103, 2002.

[6] P. Degano, D. Prandi, C. Priami, and P. Quaglia. Beta-binders for biological quantitative experiments. Submitted for publication, 2005.

[7] D. Degenring, M. Röhl, and A.M. Uhrmacher. Discrete event, multi-level simulation of metabolite channeling. *BioSystems*, 2004.

[8] Gottfried Falk and Wolfgang Ruppel. *Energie und Entropie: Eine Einführung in die Thermodynamik*. Springer-Verlag, Berlin, Heidelberg, New York, 1976. ISBN 3-540-07814-2.

[9] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[10] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402(6761):C47–C52, 1999.

[11] Pei-Hsin Ho, Thomas Shiple, Kevin Harer, James Kukula, Robert Damiano, Valeria Bertacco, Jerry Taylor, and Jiang Long. Smart simulation using collaborative formal and simulation engines. *iccad*, 00:120, 2000.

[12] H. Kitano. Systems Biology: A Brief Overview. *Science*, 295(5560):1662–1664, 2002.

[13] A. Kremling, K. Jahreis, J. W. Lengeler, and E. D. Gilles. The Organization of Metabolic Reaction Networks: A Signal Oriented Approach to Cellular Models. *Metabolic Engineering*, 2(3):190–200, 2000.

[14] M. Kwiatkowska, O. Tymchishyn, G. Norman, E. Gaffney, and J. Heath. Computational modelling of signalling pathways: Comparing simulation, verification and differential equation approaches. In L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, editors, *Proc. of the 2006 Winter Simulation Conference*, Monterey, California, USA, 2006.

[15] Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall, Englewood Cliffs, NJ, 2 edition, 1998.

[16] Pieter J. Mosterman, Janos Sztipanovits, and Sebastian Engell. Computer automated multiparadigm modeling in control systems technology. *IEEE Transactions on Control System Technology*, 12(2), March 2004.

[17] Henry M. Paynter. *Analysis and Design of Engineering Systems*. The M.I.T. Press, Cambridge, Massachusetts, 1961.

[18] Elaine Rich and Kevin Knight. *Artificial Intelligence*. McGraw-Hill, Inc, New York, New York, second edition, 1991. ISBN 0-07-052263-4.

[19] Robert Rosen. *Life itself: a comprehensive inquiry into the nature, origin, and fabrication of life*. Columbia University Press, 1991.

[20] C. Talcott. Pathway logic: A logical approach to modeling cellular processes. In L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, editors, *Proc. of the 2006 Winter Simulation Conference*, Monterey, California, USA, 2006.

[21] Frits W. Vaandrager and Jan H. van Schuppen, editors. *Hybrid Systems: Computation and Control*, volume 1569 of *Lecture Notes in Computer Science*. Springer-Verlag, March 1999.

[22] K.C.J. Wijbrans. *Twente Hierarchical Embedded Systems Implementation by Simulation: a structured method for controller realization*. PhD dissertation, University of Twente, Enschede, The Netherlands, 1993. ISBN 90-9005933-4.

[23] Olaf Wolkenhauer. Systems biology: The reincarnation of systems theory applied in biology? *Briefings in Bioinformatics*, 2(3):258–270, 2001.

[24] B.P. Zeigler, H. Praehofer, and Kim T.G. *Theory of Modeling and Simulation*. Academic Press, London, 2000.

8.2 Dagstuhl Contributions

[25] Gregory Batt. Formal verification of hybrid models of genetic regulatory networks. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[26] Jörg Bauer. Abstract interpretation of graph transformation. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[27] Luca Bortolussi. From pi-calculus to differential equations and return. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[28] Jeremy Bradley. Why I'm always late!: using stochastic process algebras to model the circadian clock. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[29] Luca Cardelli. Artificial biochemistry. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[30] Matteo Cavaliere. Modeling (and simulating) biological processes with stochastic multiset rewriting. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[31] François Fages. Biological validation as model checking. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[32] Holger Hermanns. Guiding simulation by model checking. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[33] Jane Hillston. Population models from PEPA descriptions. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[34] Hardi Hungar. Experiences with verification of dynamic systems. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[35] Peter Kemper. Verification of simulation models - experiences and challenges. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[36] Celine Kuttler. Bacterial gene expression in (yet) a(nother) pi calculus. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[37] David M. Nicol. Network simulation performance optimizations, and the need for validation. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[38] Stefan Ratschan. Safety verification of hybrid systems by constraint propagation based abstraction refinement and integrated falsification. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[39] Oleg Sokolsky. Formal analysis of network simulations. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[40] Boleslaw Szymanski. Verifying spatially-explicit simulation of virulence evolution via analytical models. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[41] Adelinde M. Uhrmacher. Component-based modeling and simulation - an exploration based on james ii. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.

[42] Silke Wagner. Model checking of hybrid systems: From reachability towards stability. In *Simulation and Verification of Dynamic Systems*, number 06161 in Seminar Report. Schloß Dagstuhl, April 2006.