Proceedings of the 2016 Winter Simulation Conference
T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, eds.

THE GOAL-HYPOTHESIS-EXPERIMENT FRAMEWORK: A GENERATIVE COGNITIVE DOMAIN ARCHITECTURE FOR SIMULATION EXPERIMENT MANAGEMENT

Levent Yilmaz
Sritika Chakladar
Kyle Doud

Department of Computer Science and Software Engineering
Auburn University
3101 Shelby Center
Auburn, AL 36849, USA

ABSTRACT

Simulation experiments are not conducted in a vacuum. They are performed to address specific research questions that require evaluation of testable hypotheses. However, the connections among goals, hypotheses, and experiments are often characterized in an ad hoc manner. In this paper, we examine symbiotic dependencies among goals, hypotheses, and experiments within the context of computational discovery. Model-Driven Science is advanced as a strategy to facilitate the search process within the operational level of hypotheses and the tactical level of experiments. We discuss the theory of explanatory coherence for evaluating and revising hypotheses while using it as a run-time cognitive model that evolves via experimentation toward an explanatory theory of the system under study.

1 INTRODUCTION

Simulation experiments are central to advancing scientific knowledge (Kleijnen 2007, Teran-Somohano, Smith, Ledet, Yilmaz, and Oğuztüzün 2015). However, experiments are not conducted in isolation; they are performed with specific goals and definite questions in mind. These questions and goals are formulated in the form of hypotheses that drive the experimentation process with the help of models. The role and utility of such computational models as abstract and ideal representations are well acknowledged (Glotzer, Kim, Cummings, Deshmukh, Head-Gordon, Karniadakis, Petzold, Sagui, and Shinozuka 2009). As advancements in model development methodologies reduce the gap between the solution and problem spaces, scientists continue to improve their ability to conceive abstractions that are necessary to effectively address research questions.

However, despite successful automation of the routine aspects of simulation and data management, scientific process continues to require considerable human expertise and effort. To reduce such manual effort, we aim to underline the need for revisiting Model-Driven Engineering (MDE) methods and coupling them with cognitive computing strategies to extend the scope of human intellect and to partner with scientists on a broad range of tasks. These tasks include identifying, prioritizing, and formulating questions, hypothesizing behavioral mechanisms (Darden 2001), defining or generating simulation experiments to test hypotheses, drawing inferences (Dzeroski, Langley, and Todorovski 2007), and evaluating results within an incremental and iterative discovery lifecycle (Bunge 1998). This iterative process calls for explicit models of hypotheses, experiments, and models, along with traceability among them to support computational exploration (Sliwoski, Kothiwale, Meiler, and Lowe 2014).

In supporting computational discovery, reliable models are necessary but not sufficient in addressing complex research problems (Kleijnen, Sanchez, Lucas, and Cioppa 2005) as proper design and management

of experiments are critical to instill confidence in the use of computer simulations (Ewald and Uhrmacher 2014). However, due to under-utilization (Teran-Somohano, Dayibas, Yilmaz, and Smith 2014) of the Design of Experiments (DoE) methodology (Montgomery 2006) and the lack of transparency in the collection and analysis of simulation data, there is an emerging credibility gap in simulation-based science and engineering (Freire, Bonnet, and Shasha 2012). To mitigate these issues, there already exist proposals, including the development of frameworks, guidelines, tools, and methodologies for computational reproducibility. For instance, Simulation Experiment Description Markup Language (SED-ML) was initiated as a community-wide effort that aims to document and improve the communication and sharing of experiments. Simulation experiment description languages (Ewald and Uhrmacher 2014) and model-driven engineering principles (Teran-Somohano, Smith, Ledet, Yilmaz, and Oğuztüzün 2015) are also promoted to streamline the management of experiments.

In this paper, by promoting a Model-Driven Science (MDS) approach and taking into consideration the characteristics of computational discovery, we present a strategy that advocates agility in model development and revision. Unlike Model-Driven Engineering (MDE), which rapidly converges to an authoritative model, our vision of MDS takes into account the iterative nature of the discovery process, requiring evaluation and revision of numerous competing assumptions and constraints until sufficient degree of veridicality is attained. This requires cognitive tools that support the co-evolution and symbiotic adaptation of the hypothesis and experiment spaces as active–learning takes place through experimentation.

To highlight the significance of properly placing simulation experiments in the context of computational discovery, the rest of the paper is structured as follows. In section 2, we overview existing workflow and experiment management systems as well as critical knowledge structures and activities in scientific discovery. Section 3 introduces a conceptual framework based on three central elements in computational research: *Goals*, *Hypotheses*, and *Experiments*. In section 4, we sketch selected elements of a Domain-Specific Language to make progress toward the envisioned computational strategy. Section 6 demonstrates the evaluation of hypotheses in terms of an evolving cognitive run-time model of the phenomena under study. We conclude in section 7 by summarizing the contributions and put forward potential research directions.

2 BACKGROUND

In this section, we overview related efforts in workflow and simulation experiment management and then discuss the issues raised by computational scientific discovery. We posit that these issues suggest critical requirements for the development of next generation experiment management systems.

2.1 Workflow and Experiment Management Systems

Scientific workflow systems are widely used to streamline experiments in computational science. These systems support execution of software modules or services to automate and repeat workflows. Among the popular workflow languages are Taverna (Oinn, Addis, Ferris, Marvin, Senger, Greenwood, Carver, Glover, Pocock, Wipat, and Li 2004), Kepler (Altintas, Berkley, Jaeger, Jones, Ludascher, and Mock 2004), and myExperiment, which enable sharing and distribution of platform-specific workflow models.

Experiment management systems have also been proposed to standardize definition of simulation experiments. For instance, Simulation Experiment Description Markup Language (SED-ML) is an XML-based format for encoding simulation scenarios. It supports sharing simulation experiments in the biomedical sciences domain. Simulation Experiment Specification via a Scala Layer (SESSL) (Ewald and Uhrmacher 2014) uses a general purpose language to introduce an internally defined DSL for experiment definition. SESSL allows model specification, definition of replications, the stop condition for the simulation run, the objective, and the optimization method. In Simulation Automation Framework for Experiments (SAFE) (Perrone, Main, and Ward 2012) and NIMROD (Peachey, Diamond, Abramson, Sudholt, Michailova, and Amirriazi 2008), experiment specifications are standardized to enable reproducibility. The ns-3 Experiment Description Language (NEDL) (Riley and Pekley 2011) also aims to meet the demand for a language

capable of explicitly capturing experiment scenarios. Despite their utility, existing experiment specification languages view experiment management independent of the overall scientific discovery context (Teran-Somohano, Smith, Ledet, Yilmaz, and Oğuztüzün 2015) characterized in the previous section. To partly mitigate this issue, experiment lifecycle management and Model-Driven Engineering (MDE) principles and practices, which are introduced in (Teran-Somohano, Dayibas, Yilmaz, and Smith 2014), aim to facilitate derivation and online adaptation of experiment models based on high-level specifications.

2.2 Computational Discovery of Scientific Knowledge

The nature of scientific knowledge structures and the associated activities, including theory formation and revision (Bunge 1998), impose additional requirements on experiment management. Among the critical scientific knowledge structures are concepts, hypotheses (including laws), experiments, and models (grounded on theories). *Concepts* abstract entities in a domain and establish the vocabulary and terminology of the problem domain. A conceptual model includes not only the concepts, but also their properties (attributes) and associations. Models and hypotheses are framed and defined in terms of the domain model. *Hypotheses* define relations and expected regularities among variables, events, processes, and objects. They also characterize assumptions that stretch our information and help us define fruitful questions to study the object of interest. By evaluating testable consequences of hypotheses, *experiments* facilitate validation or falsification of hypotheses using *models* that express statements about the structure and processes underlying the system. Models are defined in terms of the concepts and the initial fundamental mechanistic assumptions (axiomatic hypotheses) that serve as premises of the model.

Of particular interest for the purpose of our study is the relation between hypotheses and experiments. Each scientific inquiry starts with a specific goal (e.g., explain, predict, control) and imply specific problems of certain kind (e.g., which, where, why, whether, functional (how, what)). The answers are hypothesized and tested against empirical or simulated observations. Hypotheses, rather than experiments and data, are at the center of cognitive activity. All the initial assumptions underlying the model of the theory, whether formal or factual, are hypotheses. These include the posited corrigible behavioral mechanisms, which form the basic assumptions of the model. As shown in Table 1, hypothesis can be classified based on their derivation and level of abstraction.

Table 1: Types of Hypotheses

Hypothesis Models		
Criteria	**Type of Hypothesis**	
Derivation Strategy	Non-Constructive	Deductive
		Inductive
	Constructive	Intuitive
		Abductive
Abstraction Level	Low	Phenomenological
	High	Mechanistic

Regardless of how an hypothesis is derived, it may either refer to the observable behavior of the phenomena in terms of relations between the dependent and independent variables or refer to underlying mechanisms (i.e., representational view) that generate the expected regularity (e.g., relational or distributional). Scientific activities operate over knowledge structures to support taxonomy formation, law/hypothesis generation via generalization, and theory formation and revision. Inductive, deductive, and abductive reasoning involve distinct activities to govern the knowledge generation process. In contrast to *inductive discovery* of a law from simulation-generated or observed data, *deductive analysis* uses an explanatory framework (e.g., a mechanistic model) to derive a law and an explanation via the model about how the law is generated from the fundamental axiomatic mechanisms of the model. Explanation relies on abductive reasoning to posit, if necessary, new mechanistic hypotheses that can effectively explain and account for the law.

3 CONCEPTUAL FRAMEWORK FOR EXPERIMENT LIFECYCLE MANAGEMENT

Based on the foregoing discussion on scientific knowledge structures and activities, we aim to place simulation experiments in the context of the overall knowledge discovery process. First, we introduce the Goal-Hypothesis-Experiment (GHE) framework, which helps us structure the process in terms of conceptual, operational, and tactical levels. Following the introduction of the GHE framework, we sketch a conceptual model-driven engineering architecture to support the framework.

3.1 The Conceptual Level: Goals

The scientific method starts with background domain knowledge and involves the following general steps to address a specific *goal*: (1) Formulate well-structured precise questions. (2) Specify testable hypotheses that are grounded in the domain ontology to answer the questions. (3) Generate the logical consequences of assumptions in the form of expected behavior. (4) Design computer simulations to test the underlying assumptions (e.g., mechanistic hypotheses) about the phenomena. (5) Validate the simulation for relevance and reliability. (6) Design experiments, execute them, and interpret results. (7) Evaluate the correctness of the assumptions, and if necessary revise the model, experiments, or the expected behavior. These steps suggest three major activities, taking place at different levels of abstraction.

Table 2: Goal Specification is defined in terms of multiple dimensions.

Dimension	**Example**
Object of study	Immune system influence on hepatic cytochrome P450 regulation
Purpose	Explain/characterize
Issue/Focus	the reason for changes in downstream drug metabolism and hepatotoxicity
Viewpoint	based on the response of hepatic cytochrome P450-regulating mechanisms
Context	when health and/or therapeutic interventions change

The scientific activity starts with a goal, which is defined in relation to a phenomena or object for a variety of reasons, from a point of view, and in relation to a particular context (see Table 3. In computational experiments, from the perspective of scientific discovery, one can aim to characterize, understand, evaluate, predict, or improve the object of the study. For instance, in biomedical sciences, mechanism discovery is of particular interest.

3.2 The Operational Level: Hypotheses

Once a problem has been set up and examined, a solution will be sought. The solution does not start by rushing into experiments, but rather starts with ideas related to experience, expectation, or observations in the form of assumptions, called hypotheses, which are upgraded into laws, resulting in a system of laws, called theories. To characterize the attainment or evaluation of the goal of the study and to support the definition of models, a set of questions are formulated. Hypotheses are generated based on these questions and defined in terms of models of the phenomena or system of interest. The central role of hypotheses are often ignored or overlooked due to pejorative meaning associated with the term “assumption”. Most scientific and real-world activities involve assumptions stretching beyond perceivable information available in a given context. With respect to model-driven and simulation-based knowledge generation activities, we distinguish three types of hypotheses:

- *Phenomenological hypotheses* are intended to make assertions about the relations between inputs and outputs of simulations. If input factors (independent variables) satisfy a set of constraints, the conjecture is that the output (dependent variables) will take a specific form or exhibit specified regularities. Such hypotheses facilitate *analysis* by allowing comparison of system configurations and sensitivity analysis.
- *Mechanistic hypotheses* define theoretically grounded model's fundamental and basic mechanisms that produce and maintain the desired behavior. Conducting experiments to determine the behavioral mechanisms that produce, underlie, and maintain a system behavior is also known as *synthesis*.
- *Control hypotheses* relate to optimization of behavior. Given the mechanisms of the model, the objective is to discern input factors that generate optimal behavior. The search for and the identification of these input factors and their levels constitute the *optimization* process.

Table 3: Illustrative phenomenological and mechanistic hypotheses.

Type	**Example**
Phenomenological	In response to lipopolysaccharide, Kupffer cells down regulate hepatic P450 levels via inflamatory cytokines, thus leading to a reduction in metabolic capacity.
Mechanistic	Inflammatory-induced P450 down-regulation is mediated by proinflammatory cytokines that specifically regulate different yet overlapping subsets of P450s in both humans and rats (Aitken and Morgan 2007). Many of these cytokines are derived from Kupffer cells. While some cytokines down-regulate P450 in primary hepatocytes cultures, others are dependent upon the presence of Kupffer cells (Sunman, Hawke, LeCluyse, and Kashuba 2004). Kupffer cells can be activated by bacterial endotoxin (lipopolysachharide, LPS). An LPS stimulus causes Kupffer cells to release proinflammatory cytokines, triggering P450 down-regulation and the subsequent decrease in drug clearance.

3.3 The Tactical Level: Experiment

Experiments involve concrete procedures and metrics to answer the questions posited and to validate or refute the related assumptions and hypotheses. Outcomes of experiments feed back into the process to facilitate revision of goals, models, questions, and experiments. An experiment may have many factors, each of which might be assigned a range of values, called the levels of the factor in DOE terminology. Factors are classified into types, including quantitative/qualitative, discrete/continuous, and controllable/uncontrollable.

The GHE framework aims to provide a mechanism for defining and interpreting operational questions and measurable experiments for the conceptual research questions of interest. The framework is construed in the form of a hierarchical structure starting with a goal (specifying the problem, the purpose of the study, object to be measured, and viewpoint from which the measure is taken). The goal is decomposed into multiple hypotheses (formulated in the form of questions and assumptions), which refine the issue underlying the problem into its major components. Each question is then mapped onto experiments that aim to address the questions. Experiments need to be conducted in a way to discriminate between rival hypotheses. However, within the current state of the art, simulation tools and techniques are not structured to support seamless navigation and traceability between these levels. To mitigate this issue, we propose to take steps toward supporting the GHE framework by leveraging the principles and practices of model-driven engineering, domain-specific languages, and the intelligent agent technology.

4 A COMPUTATIONAL STRATEGY TO SUPPORT THE GHE FRAMEWORK

Model-Driven Engineering (MDE) has emerged as a practical and unified methodology to manage complex simulation systems development by bringing model-centric thinking to the fore. The use of platform independent domain models along with explicit transformation models facilitates deployment of simulations across a variety of platforms. While the utility of MDE principles in simulation development is now widely recognized, its benefits for experimentation have not yet received sufficient attention.

In (Yilmaz 2015), a conceptual framework is presented to integrate MDE, agent models, and product-line engineering to manage the overall lifecycle of a simulation experiment. Building on this framework, the major elements of our proposed strategy are shown in the component architecture presented in Figure 1. In the component architecture, the experiment and simulation model spaces are tightly coupled to orchestrate the co-evolution of simulation and experiment spaces as learning takes place. Next, we overview these components to open a discussion about their potential contributions to the process of computational discovery.

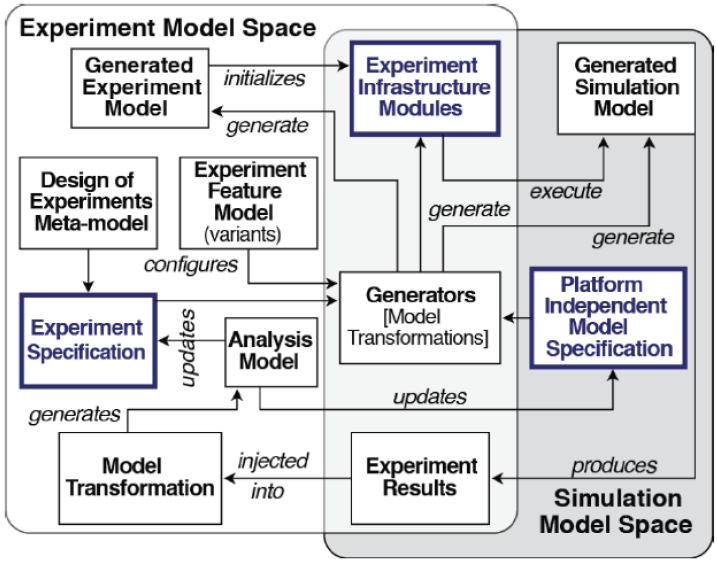

Figure 1: Experiment Management Framework

4.1 DSLs for Experiment and Hypothesis Modeling

For generating experiment specifications from research questions and hypotheses, the DOE methodology in simulation experiment design (Kleijnen, Sanchez, Lucas, and Cioppa 2005) provides a structured basis for automation. The DOE ontology defines the vocabulary and grammar. i.e., the abstract syntax for building the experiment domain model. To support the instantiation of the experiment specifications conforming to the DOE metamodel, a suitable Domain Specific Language is needed. The experiment model defined by the DSL needs to be configured with the aspects specified in an experiment feature model. An experiment design can have various mandatory, alternative, and optional features, which are prominent attributes that facilitate modeling variants of experiments to support different objectives (Sanchez, Sanchez, and Wan 2014). For instance, the type of the experiment design (e.g., factorial, fractional factorial), the optimization strategy (e.g., evolutionary strategy vs. simulated annealing), and the analysis method (e.g., ANOVA vs. MANOVA) are potential features that collectively define plausible configurations of an experiment.

4.2 Agent-assisted Experiment Specification Generation

An experiment design agent evaluates questions of interest to generate an experiment design that is not only effective in discriminating rival hypotheses, but also efficient in covering the parameter space of the system. A trade-off analysis between the number of design points and the number of replicates per design point are carried out in relation to the type of experiment being conducted. Consider, for instance, two options: one with many replicates per design point, and another with more design points with fewer replicates. The first option enables explicit estimation of response variances that can vary across scenarios. If the primary objective is to find a robust system design, then some replication at every design point is essential. If the goal is to understand the system behavior, this requires understanding the variance, again mandating replication. However, if the goal is that of comparing systems and a constant variance can be assumed, then this constant can be estimated using classic ordinary least squares regression. Replication is then of less concern, and the second option (exploring more design points/scenarios) can be a better way to spend limited computer resources.

4.3 Agent-monitored Experiment Orchestration and Update

Experiment orchestration involves experiment design adaptation capabilities so that factors that are not significant in explaining the differences in the dependent variables are reclassified as control variables, and, if necessary, design schema can adapt as experimentation moves from variable screening to factor analysis. The aggregation of results for effective analysis and communication is a critical step. Regression trees and Bayesian networks are effective ways of communicating which factors are most influential on the performance measures.

Model updating based on the analyses performed by an Experiment Orchestration Agent is the next step. We consider two types of updates: (1) experiment model (space) update and (2) simulation model (mechanistic hypothesis space) update. Adaptation of an experiment occurs at multiple levels. Based on sequential experiment results, specific factors are identified as significant, while others are classified as control variables. The reduction in the number of pertinent factors triggers a more detailed analysis of the levels of relevant factors. Such changes in the direction of exploration of the parameter space do not require an update in the experiment schema. However, higher-order experiment schema (metamodel) and search strategy adaptation may be necessary when the observed response surface complexity and the change in the number of factors trigger, for example, an update from a Central Composite Design to a Latin Hypercube design. Schema adaptation can be followed by a complete schema revision, requiring a new experiment model consistent with the evolving focus of the experiment.

5 CASE STUDY: A DOMAIN-SPECIFIC LANGUAGE FOR THE GHE FRAMEWORK

Our Model-Driven approach to experiment management is driven by explicit specification of goals, hypotheses, and experiments. We studied and modeled different types of hypotheses which allow the user to ask questions about the model or the system under study. In the context of Design of Experiments, hypotheses can be defined as: mechanistic hypotheses, relational hypotheses, and constraints. Relational hypotheses deal with the impact of changes in inputs on the outputs. Mechanistic hypotheses deal with the mechanisms of the model. Constraints are the temporal or spatial properties that are to be verified or falsified through the experiment. For illustration purposes, the evolving DSL is used to define experiments for an agent-based In Silico Hepatocyte Culture (ISHC) model (Petersen, Ropella, and Hunt 2014), which we replicated to explore the concepts proposed in this study.

The **model** section of the specification consists of the model's name, the conjectured mechanisms, the events, and the factor parameters. Mechanisms consist of the processes that plausibly take place in the simulated system. If they are confirmed via experiments, they become possible mechanisms, which in turn become actual. Events define the path for tracing the functions that evaluate elements of the evidences or constraints. Parameters are the inputs to the model; they have properties, which impact the response of the

simulation run.

```
model ISHC is {
    mechanism M1 : 'Cytokines is produced in Kupffer cells when
        inflammatoryAgent is greater than inflammatory threshold'
    mechanism M2 : 'Cytokines are not produced when noOfCytokines
        is greater than cytokine threshold'
    mechanism M3 : 'Hepatocytes down-regulate P450 levels in
        response to cytokines '

    parameters are :
        Metabolite1: type Solute with properties {bindable:false,
                          pBind : 0.05}
        Metabolite2: type Solute with properties {bindable : false,
                          pMetabolize :0.6}
        KupfferCell: type Cell  with properties
                {inflammatoryThreshold: 0.87 , cytokineThreshold :0.9}
}
```

The **hypothesis** section consists of *relational hypotheses*, *mechanistic hypotheses* and *expected regularities and constraints*. Phenomenological hypotheses deal with the impact of inputs on outputs. Mechanistic hypotheses reference the behavioral mechanisms defined in the model section. Expected regularities are the temporal and spatial properties that are to be verified or falsified. It is stated in terms of high-level finite state property patterns, which are translated by the language processor into Linear Temporal Logic. The coherence model describes the explanatory coherence relations (see section 6) between the hypotheses and the expected behavior.

```
goals {
    object of study : 'Immune system influence on hepatic
                        cytochrome P450 regulation'
    purpose : 'Explain / characterize'
    focus : 'the reason for changes in downstream drug metabolism
                and hepatotoxicity'
    view point : 'based on the response of hepatic
                cytochrome P450- regulating mechanisms'
    context : 'when health and/or therapeutic interventions change.'
}
hypotheses  {
    phenomenological hypotheses {
            H1 : [if LPS is added to Kupffer cells
                        then hepatic P450 is DECREASED]
            H2 : [if LPS is added to Kupffer cells
                        then noOfCytokines is INCREASED]}
}

evidence  {
    E1: inflammation occurs after inflamAgent > inflamThreshold
        confidence : 0.5
    E2: inflammation is absent after cytokine < cytokineThreshold
        confidence : 0.5}
```

```
coherence model {
    (EXPLAIN (H1)(E1))
    (EXPLAIN (H1,H2)(E2))
    (ANALOGOUS (H1)(H2))
}
```

Experiment – The metamodel (abstract syntax) for the experiment section encompasses the structural elements of an experiment which includes the experiment's design and performance measure. Based on the model's parameters and their levels, as well as the hypothesis and goal of the experiment, a design is created that is used in subsequent steps of the experiment life-cycle. The experimental design is defined by the dependent variables, the control variables, the independent variables. Based on this design, one can characterize the design matrix, which specifies the actual experimental runs, that is, the combination of factor levels.

```
experiment ISHC
    design {
        designType: FulFactorial
            variables {
                independent variables {
                    Metabolite2 levels : LOW where
                        LOW is in the range 100 to 200
                    KupfferCell levels : 100 to 700 with step 50
                }

                control variables {
                    Metabolite1 levels : 1000
                }

                dependent variables {
                    cytokines : type Integer
                }
            }
    }
```

6 COGNITIVE COMPUTING AS AN AID TO SUPPORT COMPUTATIONAL DISCOVERY

To support experimentation within the experiment and hypothesis spaces, there is a need to conjecture plausible explanations for the targeted behavior and discern a coherent set of mechanisms that collectively work together to generate it. To this end, we explore the use of explanatory coherence theory (Thagard 1989) toward developing a strategy. Our implementation technique uses a self-organizing coherence maximization approach to discern the combination of mechanisms that fit together to exhibit the desired behavior.

To incorporate the theory of explanatory coherence into our framework, we defined in the DSL features that represent the hypotheses and evidences, along with initial facilitation and inhibition relations among them. These relations are subject to change based on experiment results. The evidences or expected behavior are presented in terms of finite state verification patterns, which are compiled by the DSL into Linear Temporal Logic. Using the SPIN model checker, we examine if the evidence is supported by the mechanisms of the model.

Coherence theory builds on the observed data to establish relations among specified propositions. Coherence between two propositions is achieved if any of the following is true: (1) P is part of the

explanation of Q. (2) Q is part of the explanation of P. (3) P and Q are together part of the explanation of some R. (4) P and Q are analogous in the explanations they respectively give of some R and S. For illustrative purposes, in Figure 2, we present a hypothetical coherence network that is comprised of evidence and hypothesis nodes. In this example, hypotheses $H1$ and $H2$ together explain the evidence $E1$. The evidence can be represented by a predicate or expected pattern, whereas hypotheses are the behavioral mechanisms that when enacted generate model behavior consistent with the evidence. That is, $H1$ and $H2$ explain the evidence $E1$. An edge with a solid thin line indicates facilitation or explanation relation among two nodes, whereas a thick line denotes an inhibition relation. For instance, $H8$ supports or contributes to $H4$ and $H2$, whereas it inhibits $E4$. Synthesizing a model that is capable of and effective in explaining/generating a set of target behaviors can be viewed in terms of the coherence problem.

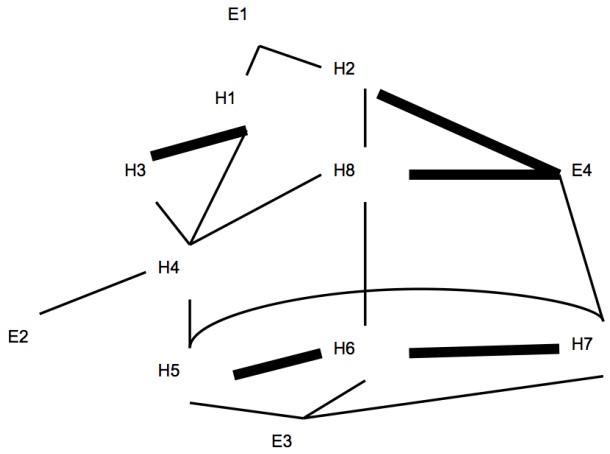

Figure 2: An Illustrative Coherence Network

The Coherence Problem: The coherence problem is defined as follows: We define a finite set of elements e_i and two disjoint sets, $C+$ of positive constraints, and $C-$ of negative constraints, where a constraint is specified as a pair (e_i, e_j) and weight w_{ij}. The set of elements are partitioned into two sets, A (accepted) and R (rejected), and $w(A,R)$ is defined as the sum of the weights of the satisfied constraints. A satisfied constraint is defined as follows: (1) if (e_i, e_j) is in $C+$, then e_i is in A if and only if e_j is in A. (2) if (e_i, e_j) is in $C-$, then e_i is in A if and only if e_j is in R. The underlying dynamics of coherence maximization is akin to simultaneous firing of neurons. Each unit receives input from every other unit that it is connected. The inputs are then moderated by the weights of the link from which the input arrives. The activation value of a unit is updated as a function of the weighted sum of the inputs it receives. The process continues until the activation values of all the units settle by no longer changing over a pre-specified limit. More formally, if we define the activation level of each node j as a_j, where a_j ranges from -1 (rejected) and 1 (accepted), the update function for each unit is as follows:

$$a_j(t+1) = \begin{cases} a_j(t)(1-\theta) + net_j(M - a_j(t)), & \text{if } net_j > 0 \\ a_j(t)(1-\theta) + net_j(a_j(t) - m), & \text{otherwise} \end{cases}$$

The variable θ is a decay parameter that decrements the activation level of each unit at every cycle. In the absence of input from other units, the activation level of the unit gradually decays. In the equation, m is the minimum activation and M is the maximum activation; net_j is the net input to a unit, defined by the following equation: $\sum_i w_{ij} a_i(t)$. These computations are carried out for every unit until the network reaches an equilibrium. Nodes with positive activation levels at the equilibrium state are discerned as maximally coherent propositions. For experimentation purposes, the design of the network can be calibrated and fine tuned to alter the weights of individual links representing the significance of the constraints. Furthermore,

initial activation levels of the propositions and initial levels of evidential support can be set to provide priority or higher weight to specific evidences and hypotheses. We have implemented the above algorithm and tested the coherence maximization algorithm on networks of various size. The next step is to incorporate the algorithm into the run-time engine of the experiment management system.

7 CONCLUSIONS

The vision and the strategy presented herein aims to provide a pathway toward a reference implementation of the GHE experiment management system. As we continue to expand the DSL with features that improve the use of cognitive computing in experiment management and to increase the degree of automation in mapping goals and questions to experiment designs, the role of agent technology (in orchestrating experiments and drawing inferences from the results) requires further exploration. In extending the DSL, our focus shifts toward bringing precision to the specification of mechanisms and introducing the transformation algorithms that map hypotheses to experiment designs. The search within the experiment space requires strategies that can effectively discriminate between competing hypotheses. The precision in the specification of mechanisms will allow weaving them to synthesize the simulation model, which will evolve based on recommendations rendered by the cognitive explanatory coherence system.

REFERENCES

Aitken, A. E., and E. T. Morgan. 2007. "Gene-specific Effects of Inflammatory Cytokines on Cytochrome P450 2C, 2B6 and 3A4 mRNA Levels in Human Hepatocytes". *Drug Metabolism and Disposition: The Biological Fate of Chemicals* 35 (9): 1687–93.

Altintas, I., C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock. 2004. "Kepler: An Extensible System for Design and Execution of Scientific Workflows". In *Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004.*, 423–424: IEEE.

Bunge, M. 1998. *Philosophy of Science: From Problem to Theory*. Transaction Publishers.

Darden, L. 2001. "Discovering Mechanisms: A Computational Philosophy of Science PerspectiveTitle". In *Discovery Science. Volume 2226 of the series Lecture Notes in Computer Science*, 3–15. Springer Berlin Heidelberg.

Dzeroski, S., P. Langley, and L. Todorovski. 2007. "Computational Discovery of Scientific Knowledge". In *Computational Discovery of Scientific Knowledge, Introduction, Techniques, and Applications in Environmental and Life Sciences*, 1–14.

Ewald, R., and A. Uhrmacher. 2014. "SESSL: A Domain-Specific Language for Simulation Experiments". *ACM Transactions on Modeling and Simulation* 24 (2): 1–25.

Freire, J., P. Bonnet, and D. Shasha. 2012. "Computational Reproducibility: State-of-the-Art, Challenges, and Database Research Opportunities". In *Proceedings of the 2012 International Conference on Management of Data - SIGMOD '12*, 593–596. New York, New York, USA: ACM Press.

Glotzer, S., S. Kim, P. Cummings, A. Deshmukh, M. Head-Gordon, G. Karniadakis, L. Petzold, C. Sagui, and M. Shinozuka. 2009. *Research and Development in Simulation-based Engineering and Science*. WTEC Technical Report.

Kleijnen, J. P. C. 2007. *Design and Analysis of Simulation Experiments*. Springer.

Kleijnen, J. P. C., S. M. Sanchez, T. W. Lucas, and T. M. Cioppa. 2005. "State-of-the-Art Review: A User's Guide to the Brave New World of Designing Simulation Experiments". *INFORMS Journal on Computing* 17 (3): 263–289.

Montgomery, D. C. 2006. *Design and Analysis of Experiments*. John Wiley and Sons.

Oinn, T., M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, and P. Li. 2004. "Taverna: A Tool for the Composition and Enactment of Bioinformatics Workflows". *Bioinformatics* 20 (17): 3045–54.

Peachey, T., N. Diamond, D. Abramson, W. Sudholt, A. Michailova, and S. Amirriazi. 2008. "Fractional Factorial Design for Parameter Sweep Experiments Using Nimrod/E". *Scientific Programming* 16 (2-3): 217–230.

Perrone, L. F., C. S. Main, and B. C. Ward. 2012. "SAFE: Simulation Automation Framework for Experiments". In *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. Uhrmacher, 1–12. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Petersen, B. K., G. E. P. Ropella, and C. A. Hunt. 2014. "Toward Modular Biological Models: Defining Analog Modules based on Referent Physiological Mechanisms". *BMC Systems Biology* 8:95–116.

Riley, G., and J. Pekley. 2011, mar. "An XML Experiment Description Language for ns-3". In *SimuTools'11 Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, 447–453: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Sanchez, S. M., P. J. Sanchez, and H. Wan. 2014. "Simulation Experiments: Better Insights by Design". In *Proceedings of the 2014 Summer Simulation Multiconference*, 53–60. Society for Computer Simulation International.

Sliwoski, G., S. Kothiwale, J. Meiler, and E. W. Lowe. 2014. "Computational Methods in Drug Discovery". *Pharmacological Reviews* 66 (1): 334–95.

Sunman, J. A., R. L. Hawke, E. L. LeCluyse, and A. D. M. Kashuba. 2004. "Kupffer Cell-mediated IL-2 Suppression of CYP3A Activity in Human Hepatocytes.". *Drug Metabolism and Disposition: The Biological Fate of Chemicals* 32 (3): 359–63.

Teran-Somohano, A., O. Dayibas, L. Yilmaz, and A. Smith. 2014. "Toward a Model-Driven Engineering Framework for Reproducible Simulation Experiment Lifecycle Management". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A.Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 2726–2737. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Teran-Somohano, A., A. E. Smith, J. Ledet, L. Yilmaz, and H. Oğuztüzün. 2015. "A Model-Driven Engineering Approach to Simulation Experiment Design and Execution". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 2632–2643. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Thagard, P. 1989. "Explanatory Coherence". *Behavioral and Brain Sciences* 12:435–502.

Yilmaz, L. 2015. "Toward Agent-Assisted and Agent-Monitored Model-Driven Simulation Engineering". In *Concepts and Methodologies for Modeling and Simulation*, 3–18. Springer.

AUTHOR BIOGRAPHIES

LEVENT YILMAZ is Professor of Computer Science and Software Engineering at Auburn University with a joint appointment in Industrial and Systems Engineering. He holds M.S. and Ph.D. degrees in Computer Science from Virginia Tech. His research interests are in agent-directed simulation, cognitive computing, and model-driven science and engineering for complex adaptive systems. He is the founding organizer and general chair of the Agent–Directed Simulation Conference series. His email address is yilmaz@auburn.edu.

SRITIKA CHAKLADAR is a Graduate Student at the Department of Computer Science and Software Engineering in Auburn University. Her research interests are in model-driven engineering and computational biomedical research. Her email address is szc0098@auburn.edu.

KYLE DOUD is a Graduate Student at the Department of Computer Science and Software Engineering in Auburn University. His research interests are in model-driven engineering and formal methods. His email address is krd0015@auburn.edu.