Simulation Validation for Societal Systems

Alex Yahja

September 2006

CMU-ISRI-06-119

School of Computer Science
Institute for Software Research International
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee

Dr. Kathleen Carley, Chair
Dr. Norman Sadeh
Dr. Douglas Fridsma M.D.
Dr. Elizabeth Casman

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

This research was funded in part by the DARPA under the Scalable Biosurveillance Systems project f30602-01-2-0550, the DOD 06806-01, the DHS EMW-2004-GR-0816 with ODU, and the ONR under the Biodefense project N00014-06-1-0252. Additional support was provided by CASOS at Carnegie Mellon University. This work also leveraged work supported by the NSF on multi-agent modeling. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect those of the sponsors.

Keywords: simulation, multi-agent, social-network, validation, model improvement, knowledge-based systems, ontology, inference, knowledge management, causality, hypothesis building, hypothesis testing, experiment, empirical artificial intelligence

Abstract

Simulation models, particularly those used for evaluation of real world policies and practices, are growing in size and complexity. As the size and complexity of the model increases so does the time and resources needed to validate the model. Multi-agent network models pose an even greater challenge for validation as they can be validated at the individual actor, the network, and/or the population level. Validation is crucial for acceptance and use of simulations, particularly in areas where the outcomes of the model will be used to inform real world decisions. There are however, substantial obstacles to validation. The nature of modeling means that there are implicit model assumptions, a complex model space and interactions, emergent behaviors, and uncodified and inoperable simulation and validation knowledge. The nature of the data, particularly in the realm of complex socio-technical systems poses still further obstacles to validation. These include sparse, inconsistent, old, erroneous, and mixed scale data. Given all these obstacles, the process of validating modern multi-agent network simulation models of complex socio-technical systems is such a herculean task that it often takes large groups of people years to accomplish. Automated and semi-automated tools are needed to support validation activities and so reduce the time and number of personnel needed.

This thesis proposes such a tool. It advances the state of the art of simulation validation by using knowledge and ontological representation and inference. Advances are made at both conceptual and implementation or tool level.

A conceptualization is developed on how to construct a reasoning system for simulation validation. This conceptualization sheds light on the relationships between simulation code, process logic, causal logic, conceptual model, ontology, and empirical data and knowledge. In particular, causal logic is employed to describe the cause-and-effect relationships in the simulation and "if-then" rules closely tied to the cause-and-effect relationships encode how causal parameters and links should change given empirical data. The actual change is based on minimal model perturbations. This conceptualization facilitates the encoding of simulation knowledge and the automation of validation. As a side effect, it also paves a way for the automation of simulation model improvement.

Based on this conceptualization, a tool is developed. This tool, called WIZER for What-If Analyzer, was implemented to automate simulation validation. WIZER makes the model assumptions explicit, handles a complex model space and interactions, captures emergent behaviors, and facilitates codification and computer-processing of simulation and validation data. WIZER consists of four parts: the Alert WIZER, the Inference Engine, the Simulation Knowledge Space module, and the Empirical/Domain Knowledge Space module.

The Alert WIZER is able to characterize simulation data with the assistance from statistical tools it can semantically control, compare the data to the empirical data, and produce symbolic or semantic categorization of both the data and the comparison. The Inference Engine is able to perform both causal and "if-then" rule inferences. The causal inferences capture the core workings of the simulations, while the "if-then" rule inferences hint at which model parameters or links need change given the symbolic categories from the Alert WIZER. Both kinds of rule inferences have access to ontology.

The Inference Engine is in the form of a forward-chaining production system but with knowledge-based and ontological conflict resolution. It performs minimal model perturbations based on knowledge bases and ontology. The perturbations result in new parameter values and/or meta-model values best judged to move the simulator closer to validity for the next cycle of simulation. Both the simulation knowledge space and the domain knowledge space are in the form of a graph, with nodes representing entities, edges representing relationships, and node attributes representing properties of the entities. Knowledge-based and ontological reasoning is performed on both knowledge spaces. A simple hypothesis can be formed by search and inference in the knowledge bases and ontologies.

Several validation scenarios on two simulation models are used to demonstrate that WIZER is general enough to be able to assist in validating diverse models. The first model is BioWar, a city-scale multi-agent social-network of weaponized disease spread in a demographically realistic population with naturally-occurring diseases. The empirical data used for the WIZER validation of BioWar comes from the National Institute of Allergy and Infectious Disease and other sources. The second model is CONSTRUCT, a model for co-evolution of social and knowledge networks under diverse communication scenarios. The empirical data used for the WIZER validation of CONSTRUCT comes from Kapferer's empirical observation of Zambia's tailor-shop's workers and management.

The results of BioWar validation exercise show that the simulated annual average influenza incidence and the relative timing of the peaks of incidence, school absenteeism, and drug purchase curves can be validated by WIZER in a clear and concise manner. The CONSTRUCT validation exercises produce results showing that the simulated average probability of interaction among workers and the relative magnitude of the change of the simulated average probability of interaction between different groups can be matched against empirical data and knowledge by WIZER. Moreover, the results of these two validation exercises indicate the utility of the semantic categorization ability of the Alert WIZER and the feasibility of WIZER as an automated validation tool. One specific CONSTRUCT validation exercise indicates that "what-if" questions are facilitated by WIZER for the purpose of model-improvement, and that the amount of necessary search is significantly less and the focus of that search is significantly better using WIZER than using Response Surface Methodology.

Tools such as WIZER can significantly reduce the time for validation of large scale simulation systems. Such tools are particularly valuable in fields where multi-agent systems are needed to model heterogeneous populations and diverse knowledge, such as organizational theory, management, knowledge management, biomedical informatics, modeling and simulation, and policy analysis and design.

Acknowledgments

I would like to thank my committee, Kathleen Carley, Norman Sadeh, Douglas Fridsma, and Elizabeth Casman, for their comments, suggestions, and guidance through the process of finishing this dissertation. Kathleen Carley is a great advisor. Without her guidance, I would not be here today.

I would also like to thank Granger Morgan and Mitchell Small for a memorable Engineering and Public Policy experience. I am grateful for David Krackhardt for his lively Social Networks lectures. I would like to thank John Anderson for his reference.

I would like to thank my friends for all the support and encouragement: Peter Simon, Craig Schreiber, Neal Altman, Natalia Kamneva, Boris Kaminsky, Demian Nave, Eric Malloy, Virginia Bedford, Ju-Sung Lee, Terrill Frantz, Keith Hunter, Jeffrey Reminga, Li-Chiou Chen, Michael Ashworth, Yuqing Ren, George Davis, and Jana Diesner.

Monika DeReno deserves kudos for her administrative work. Jennifer Lucas, Victoria Finney, Anita Connelly, Rochelle Economou, and Sharon Burks are always helpful.

From my former life, I would like to thank Stewart Moorehead for personal help after my sports injury when I was virtually alone.

Many more people in my life have ultimately contributed to this work than I can name here. If you feel left out of these acknowledgments, consider yourself acknowledged and let me know.

My parents deserve all the love in the world for their love. I can only hope I would live up to their life example. I am fortunate to have extended families lending support and encouragement, thank you all.

And finally, I thank my wife for her timeless love and companionship, and for providing much needed balance in my life.

Contents

List of Tables

List of Figures

Chapter I: Introduction

Validation is a critical problem for the use of simulations in policy design and policy making. Many crucial real world problems are complex and simulations provide a means to understand them. Validation is a very different notion from verification. In validation, the focus is in how to build the right product, while in verification the focus is in how to build the product right. Except for simulations accredited via a labor-intensive process of verification, validation, and accreditation (VV&A), most people do not trust simulation results. Curiously enough, there is an additional step – the accreditation step – that needs to be performed after the validation step in the VV&A process. If a simulation model is certified valid, why is accreditation needed? This means the validation step is still perceived to potentially produce invalid results or mismatches in application. Thus it is crucial to get the validation process right.

Modeling and simulation is becoming a useful scientific tool. Unlike the scientific problems of previous eras, most problems of consequence today are complex and rich in data, rendering less likely that a lone scientist with paper and pencil would be able to solve them. This is particularly evident in biomedical and social sciences. As the complexity of modeling and simulation – and the size of simulations – increases, assessing whether the models and simulations are valid is becoming an indispensable element of the development process. Moreover, due to the size of the validation task, it is necessary to have automated tools for the validation of models and simulations. Model assessment – determining how valid and robust a model is – is becoming a major concern. For example, NATO argued that identifying reliable validation methods for electronic medical surveillance systems is a critical research area (Reifman et al. 2004). From the policy maker perspective, the main question is whether the simulation is valid enough to answer the policy questions. Indeed, lack of confidence in the validity of simulations leads to a debate whether simulations mean anything substantial or even anything at all as a basis for business and policy decisions. There are organizations dedicated to doing VV&A, but there is a question of whether VV&A is objective and doing VV&A this way consumes a lot of time and resources. Here the automation of validation comes into play. Automation requires all assumptions and inferences be made explicit and operable, and lends to the assessment of the robustness of simulation scenarios.

One area of science that needs better modeling and simulation is Social Sciences, especially for societal modeling. Societal modeling is complex due to the many layers of physical reality affecting society and the interactions within and between the layers – with the emergence of social patterns and norms from the interactions. The biological layer of physical reality, for example, includes the neural basis for social interaction (Frith and Wolpert 2004). At the sociological layer, computational modeling and analysis (Axelrod 1997, Carley and Prietula 1999, Epstein and Axtell 1996, Prietula et al. 1998) – including the simulation component – has emerged as a useful tool.

Computational modeling and analysis can handle socio-technical problems with complex, dynamic, and interrelated parts, such as natural disaster response and disease outbreak response, which occur within a context constrained by social, organizational, geographical, regulatory, financial, and other factors. It can handle the emergence of social patterns from individual interactions. Modeling a person as an agent and social relationships as networks is part of computational modeling. The former takes the form of multi-agent models (Weiss 1999, Lucena et al. 2004, Nickles et al. 2004, Dastani et al. 2004); the latter takes the form of social network analysis (Wasserman and Faust 1994). A related modeling field is Artificial Life (Capcarrere et al. 2005), which deals with the processes of life and how to better understand them by simulating them with computers.

The use of computational modeling and analysis has increased rapidly. However, the implicit assumptions and abstractions, changes in reality, and human cognitive limitations make calibration, verification, validation, and model-improvement to assist computational modeling and analysis difficult and error-prone when performed manually.

1.1 Modeling, Simulations, and Inference

Most emphasis in computational modeling and analysis is on employing computers in building model specifications, verifying the code, and executing simulation. Indeed, the notion of computational modeling and analysis usually means quantitative models run on computers and inference/analysis done by human experts on the results of the computer runs. Much less emphasis is given to employing computers to help automate the inference, validation, model improvement, and experiment control. Figure 1 depicts this imbalance of automation, which this dissertation addresses. In the figure, the dash-lined box delineates the focus of this dissertation. Not shown is the possibility of automating simulation control and experiment design.

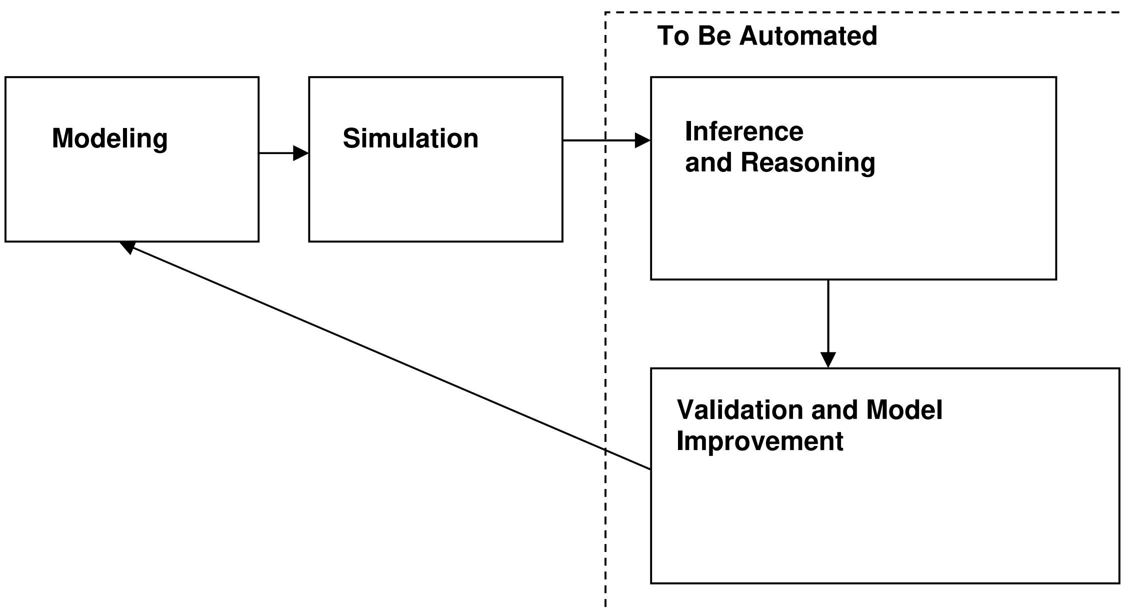

Figure 1. Automation of Inference, Validation, and Model Improvement

Improved data gathering and computational resources mean more detailed simulation models can be built and run, but deciding how best to use the simulation, which produces tremendous amount of data, is still being done manually. Indeed, we are in the period of data-rich, inference-poor environments. Typically, simulation results are designed solely for human analysis and validation is provided by subject matter experts judging that the model “feels right” (face validity). While this may be sufficient for small-scale simulations, it is inadequate for large high-fidelity simulations designed to inform decision-makers. Expert systems (Durkin 1994) exist to codify subject matter expert knowledge, but they are used separately outside the field of simulations (Kim 2005, National Research Council 2004). There is a knowledge acquisition bottleneck in expert systems. Augmenting knowledge acquisition with inference from data is an active area of research. A decade or so ago the computational intractability problems in reasoning with logic rendered knowledge-based approach unattractive. Recent research advances in logic however have started reversing this trend.

While granting that human experts can be efficient and effective, the lack of automated tools for analysis, validation, and model improvement – at least as the assistant to human experts – hinders speedier advancement in many fields, including the socio-technical and biomedical fields. Recent advances in data mining have started to make automated analysis common. A paradigm shift is needed: from focusing on design and specification toward validation and model-improvement. (Validation can be thought of as a bootstrap process for model-improvement.) Instead of focusing on the science of design[1], a more fruitful focus might be on the science of simulated experiments, which is to say, on the experimental approach (Edmonds and Bryson 2004).

Formal method (Dershowitz 2004, Etessami and Rajamani 2005) is an alternative to doing simulations or testing. A formal method provides a formal language for describing a software artifact (e.g. specifications, designs, source code) such that formal proofs are possible, in principle, about properties of the artifact. It is used for specification, development, verification, theorem proving, and model checking. Formal method has had successes in verification of software and hardware systems. The verification of the AMD-K5 floating point square root microcode is one example. While formal method has been successfully used to produce ultra-reliable safety-critical systems, it is not scalable to handle large and complex systems. Most importantly, due to its logical closed-world and mathematical/logical formality requirements, formal method cannot be used for validation.

---

1 http://www.cs.virginia.edu/~sullivan/sdsis

1.2 The Approach

This dissertation describes a knowledge-based and ontological approach for doing validation of simulation systems, implemented in a tool called WIZER (**W**hat-**I**f Analy**ZER**). The approach allows the modeling of knowledge, the control of simulation, the inferences based on knowledge and simulation, and systematic knowledge-based probes and adjustments of the parameter, model, and meta-model spaces for validation.

WIZER handles calibration, verification, and validation for simulation systems, with a side effect of facilitating a rudimentary model-improvement. Calibration is part of validation and validation forms a basis for model-improvement. Key features of WIZER are the simulation data descriptor, the data matcher (which matches simulation data descriptions against empirical data), the inference engine, the simulation knowledge space, and the empirical knowledge space. Included in the inference engine is a parameter value modifier. The data descriptor and data matcher form a component of WIZER called Alert WIZER, which produces symbolic/semantic categorizations of data and of data comparison. Statistical routines are employed in the data descriptor and data matcher. The inference engine employs rule-based, causal, and ontological reasoning.

WIZER is able to reduce the number of searches that need to be performed to calibrate a model, improve the focus of these searches, and thereby facilitate validation. Validation is achieved by performing knowledge-based search in parameter and model spaces. Model-improvement is achieved by performing search in meta-model space, after the comparison of simulation model and knowledge against target/empirical knowledge. Knowledge-based hypothesis building and testing is employed to help reduce the amount of search.

One of the currently active areas of research in Artificial Intelligence is in integrating deductive logic (including propositional logic and first-order logic) and probabilistic reasoning. The brittleness of first-order (symbolic) logic has caused the popularity of statistics – particularly Bayesian statistics – as the preferred Artificial Intelligence method. Indeed, Bayes rule forms the core of probabilistic algorithms (Thrun et al. 2005) behind the Stanley driverless car that traversed 132 miles of Southwest desert and won the 2005 DARPA Grand Challenge. The statistical approach, however, has an inherent weakness of being unable to support the structures of domain knowledge and the fertile inferences of logic. Behind the winning probabilistic algorithm of Stanley, there was a critical logical inference that the short range laser vision should be used to train the longer range camera vision. The belief driving logic and probabilistic integrative research (probabilistic logic) in Artificial Intelligence is that logic and probability are sufficient for representing the real world. The approach underlying WIZER indicates what is missing in this view: the importance of modeling and simulation, the significance of hypothesis building and testing, and the need to focus on natural processes instead of just pure logic. WIZER combines the power of logic, the expressiveness of model and simulation, and the robustness of statistics. In addition to mathematics, simulation is a tool capable for representing processes with high fidelity. Intertwining previously separate simulation and knowledge inference, the force behind WIZER, shows a way to have validated simulation that is capable for representing processes with high fidelity with knowledge inference (and explanation) capability.

Changing part of the structure of social and agent-based simulations may fit into the verification problem if we have either a complete logically-clean conceptual model or logically-clean conceptual models against which the simulation can be compared. (An incomplete model does not meet the closed world requirements of logical systems.) If we compare the simulation against the empirical/domain data and knowledge, however, changing the simulation becomes part of the validation process. This is an important distinction. Depending on the nature of data, changing the simulation model can be part of verification or validation. If the empirical data is logical and computational (this is rare in the real world, except for some engineering and scientific fields such as electronic engineering) such that logically-clean conceptual model can be constructed from and verified against it, the changing of simulation model is part of the verification process. Formal methods can be used for this verification process. If the empirical data is noncomputational or not logically-clean, which is the case for social sciences, the changing of simulation model becomes part of the validation process as it must be compared against empirical data and knowledge in addition to the conceptual model (the conceptual model itself must be empirical and not necessarily logical). The Alert WIZER can be used to pinpoint part of the simulation model that must be changed given empirical evidence. If the Alert WIZER cannot match the parameters without changing the model, it can show the mismatched parameters as the starting point for model change. For example, in the BioWar simulator (Carley et al. 2003), if the influenza incidence curve matches the empirical curve well, but the number of influenza strains greatly exceeds that of the empirical reality, then the Alert WIZER will show that there is a potential model error related to the number of influenza strains. This is part of validation and model improvement. WIZER can be used in many ways: for validation, for pinpointing model discrepancies, for semantic categorization of data, and for model improvement.

1.3 Contributions

This dissertation provides a new conceptualization for how to do automated validation of simulations particularly agent-based simulations, and then also implements a tool WIZER that is consistent with this conceptualization. The conceptualization is based on knowledge-based and ontological approach and it sheds light on the relationships between simulation code, process logic, causal logic, conceptual model, ontology, and empirical data and knowledge. The tool WIZER is implemented in four parts: the Alert WIZER, the Inference Engine, the Simulation Knowledge Space, and the Domain Knowledge Space. The Alert WIZER can do semantic categorizations of simulation data and of the comparisons between simulation and empirical data, with the support of statistical tools it semantically controls. Using the semantic categories produced by the Alert WIZER, the Inference Engine can perform causal, "if-then", and ontological reasoning, and determine new parameter values best judged to move the simulation closer to validity. This thesis has several knowledge-based measures of validity. The Simulation Knowledge Space and the Domain Knowledge Space support the explicit encoding and computer processing of simulation and domain knowledge, respectively, in the form of causal rules, "if-then" rules, and ontology. They also assist the determination of new parameter values by the Inference Engine. Several validation scenarios done on two simulation models, BioWar and CONSTRUCT, indicate the feasibility and applicability of WIZER for automated validation of simulations.

In a nutshell, the contributions of this dissertation are:

1. A novel approach for doing validation of simulations. This includes a knowledge-based and ontological method utilizing the inference engine and a new method to do a simple hypothesis formation and testing in simulations utilizing symbolic/ontological/knowledge-based information, instead of just doing permutation, parametric, and bootstrap tests (Good 2005).
2. WIZER, an automated validation tool implementing the above knowledge-based and ontological approach to validation. This includes the Alert WIZER which is

capable of symbolic categorizations of data and of semantic control of statistical routines.

3. Showing that WIZER can reduce the amount of search and focus the search, utilizing knowledge-based and ontological reasoning.
4. Partially validated the BioWar and CONSTRUCT simulators. Full validation is a major project in its own right.
5. A novel conceptualization combining modeling, simulation, statistics, and inference for a unified Artificial Intelligence reasoning construct. Until now, simulation was considered to be separate from Artificial Intelligence. Logic, simulation (and thus processes), and probability/statistics are intertwined in the conceptualization. This allows the brittleness of logic to be ameliorated by simulation-mediated statistical reasoning. Furthermore, this lets the knowledge-less statistical reasoning to be grounded in simulation model/structure.
6. A novel knowledge-based and ontology-based augmentation to simulation. This enables inference and control of simulation, including those of simulation statistical tools. Knowledge management and strategic planning in organizations and businesses can be enhanced by knowledge-augmented and validated simulations.
7. A novel description logic and ontology reasoning for simulations, which I call Simulation Description Logic (SDL). This is inspired by ontology and inference language DAML+OIL, RDF, and RuleML. SDL allows the descriptions of simulation models, simulation results, and statistical tools used to analyze the results. Based on the descriptions, the knowledge inference is performed. SDL paves a way toward the Simulation Web.

This dissertation touches upon a central problem in many fields of research and application – how to build models, do simulation, do model verification and validation, perform inferences, and improve on them. As a result, there are a number of audiences that can benefit from the work herein, including:

Simulation Modelers. The field of modeling and simulation conventionally regards the inference or analysis work as the domain of human experts with minimal assistance from computer tools. Normally only statistical packages and data mining tools are used to assist human experts. WIZER provides an automated tool to do knowledge-based and ontological reasoning for validation. As a side effect, model improvement is facilitated by WIZER through a simple knowledge-based and ontological hypothesis formation and testing. WIZER thus adds a reasoning capable tool to the repertoire of modeler tools. In short, WIZER adds the automated inference component to the modeling, simulation, and human analysis.

Policy Designers. The integration of simulation and inference advocated by this dissertation allows the simulation and inference of many policy problems. Current policy deliberations use math models (economic models are popular) and simple simulations. Most policy designs are based on meticulous examinations of the nature of the problem, issues, options, and cost/benefit of options by human policy experts. Validated simulations serving as an important tool of policy are uncommon. WIZER provides a means to automate simulation validation, thus making them more common. Validated simulations with coupled knowledge bases and inference would help greatly in the integrative treatment of the multiple aspects of a problem. By the virtue of its knowledge and ontological inferences, WIZER assists in this regard too.

Computer Scientists. The field of Computer Science is transitioning towards handling more real world problems. As a result, domain knowledge from other fields including physics, biology, sociology, and ecology is becoming more important. The reasoning algorithms in Computer Science and Artificial Intelligence must evolve as more interdisciplinary challenges are encountered. No longer is it sufficient to use simple Bayesian reasoning with its conditional dependence assumption of the known information. Now it is necessary to incorporate domain knowledge via more sophisticated reasoning algorithms. It is becoming crucial to be able to represent real world processes. Representing real world processes – and cause-effect relations – is doable by simulations, in addition to by mathematics. WIZER can validate such simulations and integrate knowledge inference and simulation. It makes domain knowledge and simulation knowledge explicit and operable, which is to say, suitable for automated or computer processing.

Epidemiologists. As the field of epidemiology considers spatial and sociological aspects of disease spreads, it is inevitable that more sophisticated and complex models upon which epidemiologists can rely on to compute and predict the spread of diseases will appear. Spatial epidemiology is now relatively mature field, but "social" epidemiology is not. This dissertation brings forward an automated validation of a multi-agent social-network model of disease spread called BioWar. A multi-agent social-network model is an appropriate tool for modeling social interactions and phenomena. In BioWar, it is shown that anthrax and smallpox can be simulated agent-by-agent and the resultant population behavior and disease manifestations mimic those of the conventional Susceptible-Infected-Recovered (SIR) model of disease spread. WIZER, the automated validation tool of simulations, allows epidemiologists to build, validate, and use more complex model of disease spread that takes into account social, geographical, financial, and other factors. It helps make prognosis, planning, and response more accurate, thus saving lives.

Social Scientists. Multi-agent modeling and simulation is becoming a preferred tool to examine social complexity. The software to do meaningful social inquiry is usually complex, due to the social interactions and the emergence of social patterns. WIZER provides the automation tool for the validation of social software, particularly the multi-agent social-network software.

1.4 Outline

This thesis research is presented in thirteen chapters, organized by three parts: 1) conceptualization and theoretical justification, 2) implementation, experiments, and results, and 3) discussion and future work.

Chapter 1 introduces the reader to the background, the rationale, the approach, and the contributions of this research.

Chapter 2 contains descriptions about related work in validation and model-improvement.

Chapter 3 contains descriptions about inference techniques in artificial intelligence and scientific method, shows the need for a new inference. Empirical reasoning and knowledge-based hypothesis building and testing are shown as a good choice for a new inference mechanism.

Chapter 4 contains the description of WIZER. This includes the description of Alert WIZER, the Inference Engine, and the knowledge spaces. It also describes in detail the reasoning mechanisms in the Inference Engine, which includes rule-based reasoning and hypothesis formation and testing. It describes the use of novel simulation description logic to describe the simulation results and the statistical tools.

Chapter 5 explains the evaluation criteria for validation and model-improvement along with the metrics.

Chapter 6 describes the BioWar testbed, the experimental setup for it, the runs, and the results. BioWar (Carley et al. 2003) is a city-scale spatial social agent network model capable of simulating the effects of weaponized biological attacks against the background of naturally-occurring diseases on a demographically-realistic population. Included is the description of empirical data used to validate BioWar.

Chapter 7 describes the CONSTRUCT testbed, its experimental setup, the runs, and the results. CONSTRUCT (Carley 1991, Schreiber and Carley 2004) is a multi-agent model of group and organizational behavior, capturing the co-evolution of cognition (knowledge) and structure. The empirical data used to validate CONSTRUCT is Kapferer's Zambia tailor shop data of workers and management interactions.

Chapter 8 describes the strengths and weaknesses of current WIZER and potential improvements. This includes a comparison between WIZER and Response Surface Methodology and a comparison between WIZER and the subject matter experts approach. The reasoning mechanisms in WIZER could be improved further. This chapter also describes how WIZER can work together with existing tools in COS such as AutoMap, ORA, and DyNet.

Chapter 9 positions WIZER and its contributions in Computer Science perspectives, with Computer Science and Artificial Intelligence terminology.

Chapter 10 describes the relationships between causality, simulation, and WIZER. It advances the use of validated simulations as a better way to examine causality and to perform causal inferences. This chapter also contains the construction of process logic and ontology to describe processes and mechanisms crucial for any causal relation.

Chapter 11 explores the potential extensions and implications of WIZER. First, it probes and describes potential extensions of the work. These include: 1) the work toward the realization of the Simulation Web, the potential next step of the Semantic Web, 2) the work toward super-simulations, and 3) the work toward creating knowledge assistant and knowledge assisted communication. Second, it explains the potential implications of WIZER in wider fields, including Policy Analysis and Design, Organization and Management, Biomedical Informatics, and Bioinformatics/Computational Biology. In particular, WIZER enhances knowledge management in many fields with validation simulation enabled by its validation automation capability.

Chapter 12 contains the description of WIZER code and a guide for the configuration and use of WIZER.

Chapter 13 summarizes the contributions, limitations, and potential extensions to this research.

Appendix A describes the field of modeling and simulation, conventional simulation approaches, and shows what and how WIZER contributes to the field. It also describes how simulation models can be learned from data.

Appendix B shows how WIZER can augment system dynamics by knowledge representation, inference, and control of system dynamics models.

Appendix C contains the ontology and knowledge base for the BioWar simulator.

Appendix D has the ontology and knowledge base for the CONSTRUCT simulator.

1.5 Definition of Terms

The following are the definition of terms related to this research.

Verification: a set of techniques for determining whether the programming implementation of the abstract or conceptual model is correct (Xiaorong 2005).

Validation: a set of techniques for determining whether the conceptual model is a reasonably accurate representation of the real world (Xiaorong 2005). Model validation is achieved through the calibration of the model until model accuracy is acceptable.

Calibration: an iterative process of adjusting unmeasured or poorly characterized model parameters or models to improve the agreement with empirical data (Xiaorong 2005).

Accreditation: a certification process by an independent/official agency (Balci 1998) which is partly subjective and often includes not only verification and validation but items such as management policy, documentation, and user interface.

Training: procedures for supplying data and feedback to computational learning models

Model improvement: a set of techniques to enhance the model relative to the epistemic and empirical knowledge of the problem of interest.

Unless mentioned otherwise, the term validation in this dissertation will denote calibration, validation, and model-improvement.

Chapter II: The Need for a New Approach

Validation has been addressed using different approaches from many fields. I elaborate on these below and point to a promising new approach to the problem of validation. Validation is not to be confused with verification. The latter deals with how to build a product right, while the former concerns itself with how to build a right product which is a far more important and difficult problem. Validation is also different from diagnosis, as the former concerns itself to ascertain if a model is correct, while the latter probes what causes a malfunction(s) in parts of a model given than the model is correct.

2.1 Related Work

Verification and validation can theoretically be performed by utilizing formal methods (Weiss 1999, Dershowitz 2004, Davies et al. 2004, Bertot and Castéran 2004, Hinchey et al. 2005, Fitzgerald et al. 2005) if a formal specification of validity exists. A formal method is a method that provides a formal language for describing specifications, designs, and source code such that, in principle, formal proofs are possible. Formal methods can be categorized into "traditional" formal methods which are used for design verification and algorithm/code verification, and "lightweight" formal methods which are used for requirements "validation" and conceptual model "validation", that is, analyzing assumption, logic, and structure. It is not yet applicable to "validation" at the run-time level and the empirical level. Formal methods depend on denotational, operational, and axiomatic semantics. The value of formal methods is that they provide a means to symbolically examine the entire state space and establish a correctness property that is true for all possible inputs. Formal methods can be used for specification, development and verification, and automated provers. Automated provers include:

- Automated theorem proving, which produces a formal proof from scratch, given a description of the system, a set of logical axioms, and a set of inference rules.
- Model checking, which verifies properties by means of an exhaustive search of all possible states that could be entered during execution.

Neither of these techniques works without human assistance. Automated theorem provers usually require human inputs as to which properties to pursue, while model checkers have the characteristic of getting into numerous uninteresting states if the model is sufficiently abstract. However, while formal methods have been applied to verify safety critical systems, they are currently not scalable to reasonably complex simulations. In addition to relying on logic and automata (finite state machines), formal methods rely on specified "truths", ignoring the empirical nature of reality. They also rely on a limited set of semantics, ignoring natural processes and causality. A formal proof of correctness, if attainable, would seem to be the most effective means of model verification and validation, but this impression is wrong. Indeed, formal methods can prove that an implementation satisfies a formal specification, but they cannot prove that a formal specification captures a user's intuitive informal expectation and/or empirical foundations for a system. Furthermore, non-computational data inherent in the validation process cannot be properly handled by formal methods, which requires strict logical representation. In other words, formal methods can be used to verify a system, but not to validate a system. The distinction is that validation shows that a product will satisfy its user-desired mission, while verification shows that each step in the development satisfies the requirements imposed by previous steps. Contrary to intuition, forcing formality on informal application knowledge may in fact hinder the development of good software. Successful projects are often successful because of the role of one or two key exceptional designers. These designers have a deep understanding of the application domain and can map the application requirements to software.

In software engineering (Pressman 2001), "validation" of multi-agent systems is done by code-"validation", which means the determination of the correctness of the software with respect to the user needs and requirements. In contrast, my concern is with empirical in addition to epistemic validation. In principle, if – this is a big if – the real-world problems could be specified formally, then formal methods could be applied.

However, formal methods (Dershowitz 2004, Davies et al. 2004, Bertot and Castéran 2004, Hinchey et al. 2005, Fitzgerald et al. 2005) used in software engineering for the control and understanding of complex multi-agent systems lack an effective means of determining if a program fulfills a given formal specification, particularly for very complex problems (Edmonds and Bryson 2004). Societal problems include complex communication patterns (Monge and Contractor 2003), messy interactions, dynamic processes, and emergent behaviors, and thus are so complex that applying requirements engineering and/or formal methods is currently problematic. Still, formal methods have value in requirements "validation", not least by its virtue of precise specification, which could reveal ambiguities and omissions and improve communications between software engineers and stakeholders.

Evolutionary verification and validation or EVV (Shervais et al. 2004, Shervais and Wakeland 2003) can be also applied to multi-agent social-network systems. EVV utilizes evolutionary algorithms, including genetic algorithms (Deb et al. 2004) and scatter search, for verification and validation. While EVV allows testing and exploitation of unusual combinations of parameter values via evolutionary processes, it employs knowledge-poor genetic and evolutionary operators rather than the scientific method, for doing experiments, forming and testing hypotheses, refining models, and inference, precluding non-evolutionary solutions and revolutionary search/inference steps.

Docking – the alignment of possibly-different simulation models – is another approach to validating multi-agent systems (Axtell et al. 1996). Alignment is used to determine whether two simulation models can produce the same results, which in turn is the basis for experiments and tests of whether one model can subsume another. The more models align, the more they are assumed to be valid, especially if one (or both) of them has been previously validated. The challenges in applying docking are the limited number of previously validated models, the implicit and diverse assumptions incorporated into models, and the differences in data and domains among models. Two successful examples of docking are the alignment of the anthrax simulation of BioWar against the Incubation-Prodromal-Fulminant (IPF) mathematical model, a variant of the well-known Susceptible-Infected-Recovered (SIR) epidemiological model (Chen et al. 2006), and the alignment of BioWar against an SIR model of smallpox (Chen et al. 2004). While aligning a multi-agent model with a mathematical model can show the differences and similarities between these two models, the validity it provides is limited by the type and granularity of data the mathematical model uses and by the fact that symbolic (non-numerical) knowledge is not usually taken into consideration.

Validating multi-agent social-network simulations by statistical methods alone (Jewell 2003) is problematic because the granularity required for the statistical methods to operate properly is at a sample population level and the sample has homogeneity assumptions. Much higher granularity and heterogeneity can be achieved using knowledge-based validation. Statistics averages over individuals. Individual importance and eccentricity hold little meanings for a population from the statistical point of view. Moreover, statistical methods cannot usually deal with symbolic – instead of numeric – data and cause-and-effect relationships.

Human subject matter experts (SMEs) can validate computational models by focusing on the most relevant part of the problem and thinking about the problem intuitively and creatively. Applying learned expertise and intuition, SMEs can exploit hunches and insights, form rules, judge patterns, analyze policies, and assess the extent to which the model and their judgments align. To deal with large-scale simulations, SMEs' effectiveness can be enhanced with computer help. Managed and administered properly, SMEs can be effective. The Archimedes model of diabetes is an example of successful validation by SMEs assisted by statistical tools (Eddy and Schlessinger 2003). However, human judgment based validation is subject to pitfalls such as bounded rationality, biases, implicit reasoning steps, and judgment errors. Moreover, the fact that validation knowledge is often not explicitly stated and encoded hinders the validation process. When SMEs evaluate the results of the changes they suggested earlier, some results may be wrong. Pinpointing exactly where in the process the error occurs is difficult due to the above implicit assumptions and sometimes ambiguous statements. Even if the validation knowledge is explicit, it is not structured and codified for automation by computer.

Another approach to validation is direct validation with real world data (empirical validation) and knowledge (epistemic validation). Validation can be viewed as experimentation with data and knowledge, and models as infrastructure or lab equipment for doing computational experiments or simulations (Bankes 2004). Simulation (Law and

Kelton 2000, Rasmussen and Barrett 1995) has an advantage over statistics and formal systems as it can model the world as closely as possible (e.g., modeling emergence), free of the artifacts of statistics and formal systems. Direct validation requires a number of virtual experiments be run using the simulator. The results from these experiments are then compared with the real data. Two techniques for this comparison are Response Surface Methodology (Myers and Montgomery 2002) and Monte Carlo simulations (Robert and Casella 1999). These two approaches, however, can only be used for numerical data and are limited to a small number of dimensions.

An interesting and somewhat related work is the extension of C++ language with a programming construct for rules called R++[2] (Crawford, et al. 1996). A rule is a statement composed of a condition, the left-hand side (LHS), and an action, the right-hand side (RHS), that specifies what to do when the condition becomes true. R++ rules are path-based, which means the rules are restricted to the existing object-oriented relationships, unlike data-driven rules. R++ however is not available in the public domain. On June 16, 1998, Patent Number 5768480 ("Integrating Rules into Object-Oriented Programming Systems") was issued to Lucent Technologies for R++, but the production version of R++ is owned by AT&T. The legal complications of figuring out who owns R++ and licensing issues due to AT&T and Lucent breakup has meant that R++ is not available commercially or through free distribution.

Diagnosis is another somewhat related technique. Diagnosis is concerned with ensuring a product works correctly. The frame of thought for diagnosis is finding the causes of symptoms in the model, assuming that the model is correct or several alternative candidate models are correct. Diagnosis does not deal with the validation of models. It mostly focuses on heuristic inference, except for model-based diagnosis. The models and the processes are not examined to see if they are valid empirically (they are assumed and given to be valid *a priori* in model-based diagnosis). Diagnosis is usually done for illness, mechanical malfunctions, and software failure. Tools used for diagnosis include expert systems (Jackson 1999) and Bayesian networks.

One of subject matter experts' approaches to validation, the Verification, Validation and Accreditation (VV&A) method, is a regimented process to ensure that

---

[2] http://www.research.att.com/sw/tools/r++

each model and simulation and its data are used appropriately for a specific purpose, usually for military systems development and acquisition. VV&A is conducted throughout the modeling and simulation life-cycle management (LCM) process. While VV&A has proven to be a successful approach for military systems, the task of VV&A is labor intensive and involves several organizations. VV&A is done mainly by human experts or trained personnel with the help of quantitative tools. Only organizations with deep resources and sufficient time can apply VV&A.

2.2 Why Validation of Multi-Agent Social-Network Simulations is Hard

All simulations are wrong, but some are useful. It is currently impractical to have simulations completely mirror the real world, except for the cases where real world processes are well understood. Validation is usually performed against a small part and/or an abstracted part of the real world which the policy question at hand is concerned with.

The task of validating a simulation – and the model behind it – against that portion of the reality that the simulation needs to address is hard due to the often-implicit assumptions, unclear correspondence, uncertainty, compounding, combinatorial explosion of the possible combinations of parameter values, the large amount of time needed, human cognitive limitations, changes in the real world, possibly chaotic system behavior, interaction, system dependence, the non-Markovian nature of the real world, and emergence of patterns or behaviors.

Validating multi-agent simulations is harder due to the magnitude of interactions between agents, the increased action choices of agents, knowledge dimension, and causal relations. While neither networks nor organizations are present, the combinatorial explosion of possible agent-to-agent interactions and actions makes validation difficult.

Multi-agent social-network simulations are even harder to validate because they have an additional social-network aspect to contend with. The social-network can have multiple attributes such as friendship, family relationships, work relationships, and others. The relationship between agents forms dyads and triads with differing tie strengths. The structure of the social network itself gives rise to cliques, coalitions, isolates, and others. The social networks constraint possible agent behaviors, while agent behaviors shape the social networks. These networks give rise to organizations.

Dynamic multi-agent social-network simulations are even harder to validate because they are dynamic – behaviors and agents change over time. Most multi-agent social-network simulations are dynamic.

In general, any model that deals with uncertainty and dynamics is complex and potentially hard to validate. The sources of uncertainty include ignorance, ambiguities, belief/disbelief, changing worlds, and incorrect and/or incomplete knowledge.

2.3 Special Challenges posed by Subject Areas

Subject areas may pose special challenges to the task of validation. Relevant subject areas for this dissertation which pose special challenges are biomedical informatics, epidemiology, and social science. There are several kinds of special challenges:

1. Data gathering: fields such as physics, chemistry, and mechanical engineering have a straightforward data gathering procedure. In social sciences and biomedical science, the data gathering process is more complicated, as it requires informed consent and almost always involves biases.
2. Data quality: in physics it is feasible and even routine to have data with high accuracy. In sociology, when data is gathered by using surveys, accuracy is not high. In medical science, some data have good accuracy (such as the genomic data), while others do not (such as the effectiveness of certain treatments).
3. Data quantity: it is generally harder and more costly to get data in large quantity for sociology and organization science than for physical sciences.

4. Process clarity: in physics the processes are usually precisely defined and understood. Many processes in social sciences, organizational science, and medical science are not as precisely defined and understood.

In this dissertation, validation of the BioWar testbed presents the following challenges:

1. Center of Disease Control (CDC) data about influenza is not precise. This is due to the fact that there is no precise knowledge about influenza manifestations. We know a lot about the influenza virus, but do not know the infection rate and the death rate of influenza for an individual. This is due to the complexity of the human body. We do not know when precisely influenza symptoms will manifest for each individual.
2. The incidence rate of influenza is not known to a high degree of accuracy. Due to the nature of influenza spread, the smaller the sample area, the less reliable are the statistics.

Validation of the CONSTRUCT testbed has the following challenges:

1. Limited data: the Kapferer's Zambia tailor shop data encodes the social and instrumental ties extensively, but not the deep knowledge dynamics of each individual. All facts are encoded to be the same, that is, if a fact is known it is encoded as 1, otherwise as 0. In reality, not all facts are the same. A fact may have correlation and compounding with another due to the semantics of the facts.
2. Imprecise data: the tailor shop data abstracts the societal and instrumental ties. How strong the ties really are is not given. While granting that social survey is not an easy task, this remains a hindrance to knowledge-based inference.
3. Non-repeatability of social situations: while WIZER can play out what-if scenarios, the hypothetical scenarios do not have the corresponding empirical data as there was no exactly corresponding social situation. In general, no social situations happen twice with exact precision, unlike physics experiments. History may repeat, but not exactly.

2.4 Validation and Policy Question

As the validity of simulation is measured against the policy question the simulation is designed for, the types and extent of knowledge need to be clarified, as follows:

- The reality of the universe. Part of the reality is basically known but the totality of it is still unknown despite the advancement of science since the Renaissance.
- The global knowledge of human beings. The knowledge may or may not be true with respect to the reality of the universe.
- The knowledge relevant to the policy question at hand. This knowledge is a subset of the global knowledge. This includes domain/epistemic knowledge and empirical knowledge.
- The knowledge embodied in simulation models and simulation executions. This knowledge may or may not be a subset of the policy question's knowledge.

Figure 2 illustrates the scopes of the knowledge spaces.

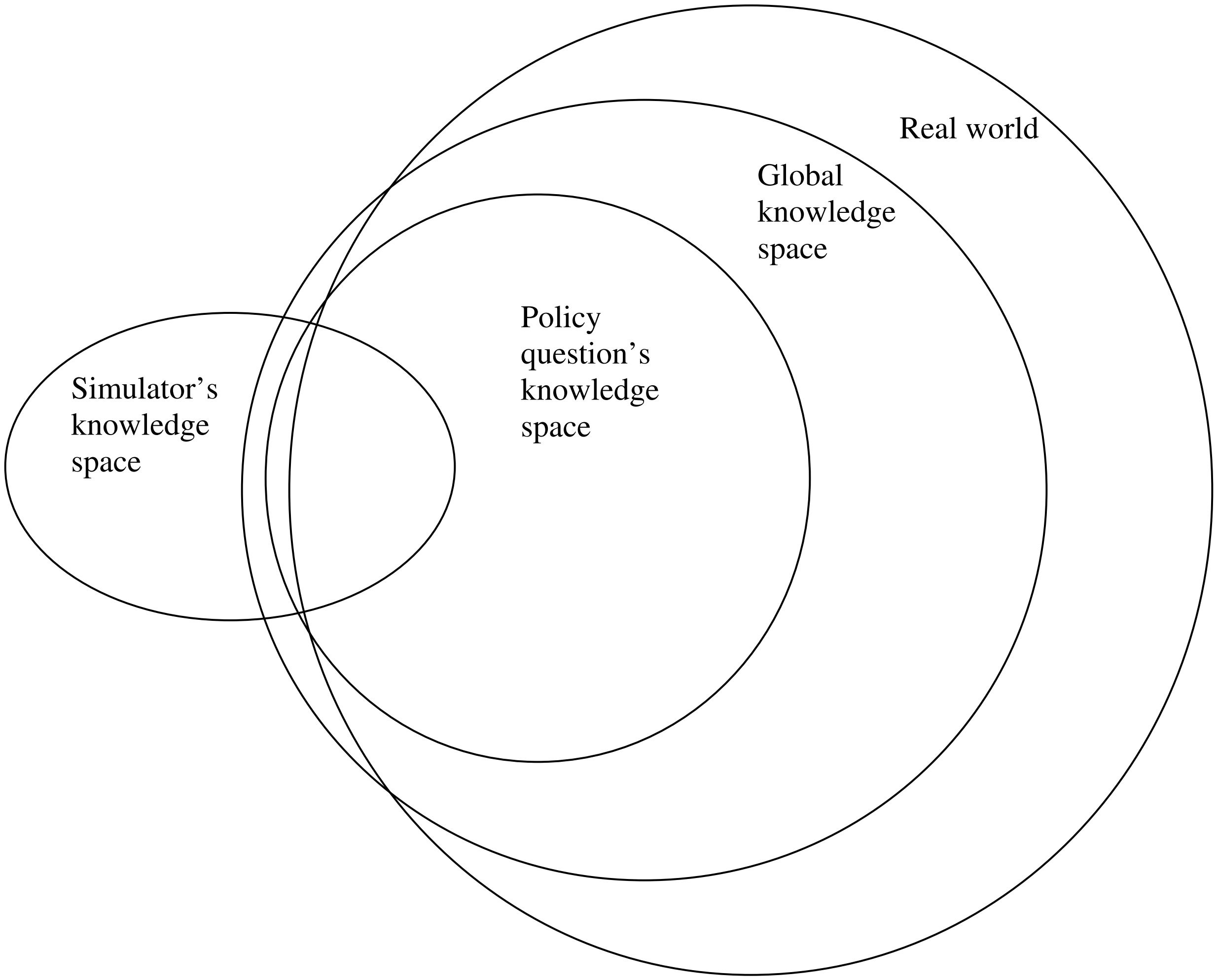

Figure 2. Knowledge Spaces

The task of validation is defined as the process of fitting the simulator's knowledge space into the policy question's knowledge space. The simulator's knowledge space includes the static knowledge behind model specification and implementation, and the dynamic knowledge arising from the execution of the simulator.

The policy question's knowledge space is defined as all relevant knowledge pertaining to the policy question. For example, in a biological attack, the policy question may be what the most effective response is, while the knowledge enabling this question to be answered forms the knowledge space of the policy question. Needless to say, this knowledge space is much larger than the semantics of the policy question.

The policy question's knowledge space is contained within the global knowledge space. The global knowledge space, which includes physical and psychological knowledge, may not necessarily capture the real world.

Empirical data is assumed to be part of the global knowledge space. In actuality, empirical data functions as an extender of the global knowledge space to be closer to reality. For clarity, I chose not to draw another circle denoting the empirical data space which intersects the global knowledge space.

As simulations are basically knowledge systems, the knowledge-based approach enables the control and validation of simulations directly with empirical knowledge and data. The knowledge-based approach has a representation in the form of knowledge space, a generalization of version space (Mitchell 1978).

A positive aspect of having to answer a policy question is that the policy question can be used to restrict what validation needs to be performed on a simulation system. If the policy question, for example, is concerned with school absenteeism, then the validation task is made to focus on school absenteeism, not on other data streams such as work absenteeism. While the two may be correlated, validation of school absenteeism does not need work absenteeism data, except in the case that we want to model what effects non-working parents have on children's school absenteeism rate. Having to answer a policy question reduces the search space and the amount of inferences that need to be done.

2.5 Mathematical Reasoning Automation

An integral part of any reasoning system is mathematical reasoning. Numerical computations are the domain of computers, which can perform them effortlessly and speedily. Symbolic math computation is also doable by software. Both numerical and symbolic computation is an achievement, but the most exciting area is automated reasoning. Automated reasoning for math and logic (Hutter and Stephan 2005), particularly for theorem proving and proof finding, has progressed significantly to the point that in 1996 artificial mathematicians EQP and Otter proved a conjecture of the so-called Robbins problem, a conjecture which was open for sixty years, unsettled by the best human mathematicians. The progress of math and logic automated reasoning provides hope for future realization of a program that understands math (and not just manipulating bits, numbers, and symbols mindlessly).

I mention the automated reasoning for math and logic here to illustrate the power and potential of automated reasoning. While this dissertation does not cover this aspect of automated reasoning, there is a potential for cross-fertilization in the future.

2.6 Causal Analysis, Logic, and Simulation

Statistical analysis is routinely employed to help experts perform validation. Statistical analysis can infer parameters of a distribution from samples. The associations among variables and the likelihood of events can be estimated. Dynamic environments however entail changing experimental conditions which make statistical analysis insufficient. For example, the joint distribution of symptoms and diseases cannot say that curing the former would or would not cure the latter. It cannot say how the distribution would differ if external conditions were to change.

In contrast to statistical analysis, causal analysis – which can infer aspects of data generation process – can deal with dynamic changes. Here simulation plays an important role, by quasi-experimenting the data generation process. This enables the deduction of not only the likelihood of events under static conditions, but also the dynamics of events under changing conditions. Causal analysis and simulation enables the estimation of how events which have not happened yet will play out (scenario analysis), of intervention outcomes, and of what events will most likely happen.

Associational assumptions – such as Bayesian conditional or prior – can be tested and estimated in principle given a sufficiently large sample. Causal assumptions, on the other hand, cannot be verified even in principle, unless we use experimental control. This is where carefully-designed simulation plays an important role. The simulation facilitates quasi-experiments which, given good enough empirical data, can reflect the consequences of causal assumptions in the real world.

In computer science and artificial intelligence, there has been work on integrating logical and probabilistic reasoning. Logical reasoning such as propositional and first-order logic is brittle especially if data is noisy. Real world data is usually noisy, especially in humanities and social sciences. Chaining logic inferences has the greater risk of irrelevant inferences the longer the logical chain of reasoning. Logical reasoning also has combinatorial explosion and scalability problems. In everyday life, people do not usually form a long chain of logical reasoning. Instead, it can be argued that people handle minimal logical reasoning but have superb knowledge representation. Modeling and simulation is one of the most accurate tools for knowledge representation. Statistical reasoning, on the other hand, lacks the structural knowledge of the world. Modeling and simulation can lend logical reasoning robustness and statistical reasoning structural knowledge of the world. To achieve this, it is important that the modeling and simulation focus on real world processes instead of just pure logic, thus the importance of validation.

While a rule-based system is sufficient if knowledge engineers are able to check the causal relations inherent in some rules, for large knowledge bases manual checks are cumbersome and prone to errors. Thus there is a need for automation through formal causality checking.

There are computer models for learning causal relations from data and for causal inference, a result of causal analysis research at Carnegie Mellon, UCLA, and Stanford (Pearl 2003, Spirtes et al. 2000). These models for causality however do not consider simulation as an important tool in causal analysis. Instead, they rely on graph analysis, Bayesian models, and mathematical analysis. Here, I will deal primarily with causal inference, not with the causal learning from data.

A state-of-the-art causal inference model is the Pearlian causal model (Pearl 2003, Pearl 2000). To account for the probability of causation, the Pearlian causal model requires the use of Bayesian priors to encode the probability of an event given another event. It is unable to model ignorance, ignores contradictions, and is incapable of expressing evidential knowledge without the use of the probability distribution format.

Different kinds of uncertainty (whether it is subjective or objective) are modeled the same using Bayesian distributions.

The Pearlian causal model is insufficient for validation of simulations for several reasons:

1. The aim is to do validation in uncertain and noisy environments.
2. Assumptions need to be managed explicitly, not through conditional probability.
3. There is a need to clearly delineate between subjective uncertainty (judgment) and objective uncertainty (frequency).
4. Bayesian priors are problematic for specification. Rather than using Bayesian priors and probabilistic variables, we can do detailed simulations. In addition to inference/reasoning mechanisms, the representation is important. Using graphs or Bayesian networks as representation hinders the accurate representation of reality. Simulations, on the other hand, can emulate real world entities, processes, and mechanisms closely.
5. There is no simulation component in the Pearlian causal model, forcing it to resort solely to graph and Markovian assumption to compute the effect of interventions. The addition of knowledge inference renders any finite state machine with it to be non-Markovian. An intervention in the causal network is represented by cutting the path to the intervened variable from all other variables (deleting certain mappings from the model), and setting the value of that variable to the intervention value, while keeping the rest of the model unchanged. In this dissertation, simulations are utilized to compute the effect of interventions. The intervention and the simulation are the reflections of physical intervention and reality. Identification – the determination of whether one can compute the post-intervention distribution from data governed by the pre-intervention distribution – is also possible by a direct estimation through simulation.

Simulation, however, is not without weaknesses. Any simulation model is only as good as its assumptions. Additional weaknesses include inherent difficulties in getting decision rule accuracy, soft variables, and model boundaries right. Decision rules for each agent are difficult to get and ascertain. Their accuracy is often uncertain. Software architects are often in need of the skills of domain experts for this decision rule determination. Some variables affecting the agent decision are soft in nature. For example, an agent may buy an automobile if it is inexpensive, reliable, stylish, and fun to drive. The variables – especially the last two – are soft, meaning they are hard to quantify and their meanings can differ greatly from agent to agent. In building a simulation model, software architects make decisions about which variables are exogenous and which are endogenous to the model. The decisions have a large effect on the model prediction. This dissertation provides a remedy to the weaknesses above, particularly in managing the model assumptions and managing the model boundaries. It also suggests an avenue to ameliorate the decision rule accuracy and soft variables problems by knowledge-based validation.

Human beings constantly fashion causal relations, even for complex systems. Many of these causal relations are spurious, but some people take them for granted. These need to be modeled in social simulation systems, since however misguided the causal relations and assumptions may be they guide manifested human behaviors.

2.7 Knowledge-based Approach

Knowledge-based representation can theoretically represent any other representation. A knowledge-based approach is a promising approach for automating validation and model-improvement of simulations, particularly multi-agent social-network simulations. Knowledge-based approaches denote the use of knowledge representation and inference, whose manifestations are in the form of knowledge base and inference engine in Artificial Intelligence (Russell and Norvig 2003). Systems utilizing knowledge-based approach are called knowledge-based systems. An example of knowledge-based systems is Cyc (Lenat and Guha 1990). Cyc is a very large, general, multi-contextual "common sense" knowledge base and inference engine. It is an attempt to do symbolic Artificial Intelligence on a massive scale by vacuuming facts, rules of thumb, heuristics about entities and events. Despite its massive knowledge bases, Cyc is still brittle due to its pure symbolic approach.

The breath and depth of knowledge needed for validation and model-improvement usually exists in databases and/or silos of expert knowledge. Capturing this knowledge into a form that can be processed by computers is a cornerstone of knowledge-based approach. The computerized knowledge processing is usually totally disconnected from the process of designing, validating, and improving a simulation model: conventionally it is the job of human experts without the aid of databases. Corporations and government agencies have large databases and separate simulation projects: combining the two effectively can provide new insights.

Simulations are basically knowledge systems. They can be viewed as a black box spewing out knowledge as output given certain pieces of knowledge as input. Testing a black box is done by giving a certain input and observing the outputs. This is similar to the cracking of the Enigma machine by utilizing the existing knowledge, not solely by statistical tests.

The knowledge-based approach is mostly symbolic, which supports the modeling of intelligence and reason. Indeed, the Physical Symbol System hypothesis (Simon 1996) proposes that a physical symbol system has the necessary and sufficient means for general intelligence, which is a Strong AI view. This dissertation subscribes to the view that a physical symbol system is important but the underlying non-symbolic physical processes are the true foundation. It attempts to capture the physical processes via simulations and the symbol system via knowledge-based and ontological reasoning. It is midway between the Strong AI and Weak AI views. The advantages of symbolic architectures are:

- much of human knowledge is symbolic, so encoding it in a computer is more straightforward.
- how the symbolic architecture reasons may be analogous to how humans do, making it easier for humans to understand.
- symbolic architecture may be made computationally complete (e.g. Turing Machines).

The knowledge-based approach allows the structure of the real world problem to be incorporated and reasoned about. This stands in contrast to the statistical approach which cannot handle well structural and componential knowledge and fertile inference of logic. The knowledge-based approach can capture a user's informal and intuitive understanding of a system. Thus, unlike formal methods knowledge-based approach (including rule-based and causal methods) is suitable for validation. The knowledge-based approach lends itself to hypothesis building and testing.

2.8 Knowledge Acquisition Bottleneck

For any knowledge-based system to have value, the knowledge bases need to be constructed. This is done by knowledge acquisition from human experts possibly in the form of heuristics. Knowledge acquisition takes time and is prone to errors. How to learn knowledge automatically from data is an active area of research. Causal learning from data is an example. Machine learning and data mining are two fields dealing with learning and extracting knowledge from data respectively.

This dissertation suggests a policy-based way to minimize the problems with knowledge acquisition bottleneck. It puts the knowledge acquisition on the shoulders of persons who are the likeliest to possess and have interests in entering the knowledge bases correctly. This means the simulation knowledge bases are acquired and input by the simulation developers, while the validation knowledge bases are acquired and input by the validators or the VV&A practitioners.

While the above may minimize the problems of knowledge acquisition, true knowledge acquisition should happen automatically. This dissertation suggests a simple but powerful method of hypothesis building and testing (first in the simulation proxy and then with the empirical data). The difference between this method and machine learning is that machine learning focuses on general algorithms and is knowledge poor, while our method of hypothesis building (e.g., constructing new causal relations in a causal networks) and testing is knowledge intensive and is not focused on any general (or specific) algorithm.

2.9 Models, Inference, and Hypothesis Building and Testing

All models are approximations. There are mechanistic models (models that have physical mechanisms related to them available) and empirical models (no underlying physical mechanisms are known so the model is purely empirical). When available, a mechanistic model has advantages because it may provide a physical understanding of the system and greatly accelerate the process of problem solving and discovery. A mechanistic model frequently requires fewer parameters and thus provides estimates of the fitted response with proportionately smaller variance. Sometimes however an empirical model can suggest a mechanism.

Mechanistic models are constructed with the structural knowledge of the relevant real world processes. As structural knowledge is formalized and put into knowledge bases, inferences from the knowledge bases can be made. The knowledge bases also show the extent and the uncertainty of the knowledge therein. Based on the existing knowledge and knowledge about the unknown and/or the uncertain, hypotheses can be constructed. A simplest hypothesis construction is done by searching and/or reasoning through the knowledge bases and ontology to look for implications that have not been explored. Simulation then allows the hypotheses to be tested in proxy.

2.10 Alert and Inference Engine

Knowledge-based systems operate on mostly symbolic data. They employ a symbolic inference engine. Simulation systems operate on mostly numerical data. In order to use numerical data from simulation systems, there is a need to convert them to symbolic information. This is accomplished by the Alert module, which tests numerical data using statistical routines against certain criteria to produce symbolic information. For example, the Alert module tests the simulated average yearly school absenteeism against the empirical minimum and maximum value of the annual absenteeism rate, and produces a "value-too-high" alert if the simulated average is higher than the empirical maximum. The criteria to test against are normally based on empirical data, but they do not have to be. Tests based on empirical knowledge can also be used. There are many other types of test that could be performed, including classification, peak detection, anomaly detection, etc. The statistical routines used vary from the simplest to the most sophisticated. If the best information that could be gathered is uncertain, it is represented in probabilistic form and written as symbolic information. For example, if we are interested in the probability of a child going to school on day *D*, and the outputs of the simulation show that probability being equal to 0.9, we could put it symbolically as "a child going to school on day *D* is uncertain with probability 0.9". This is then encoded as a propositional variable (in the form of a symbolic "alert") for the Inference Engine to process. Similar encoding is applied to the information about value ranges and curves.

After symbolic information is gathered, the information is fed into the Inference Engine. The Inference Engine is a production system. The inference proceeds in a forward-chaining fashion. The Inference Engine takes the alerts and the simulator's causal diagram, in addition to the empirical data and the domain knowledge and parameter constraints, to make a judgment on which parameters, causal links, and meta-models to change – or not to change – and how.

2.11 Summary

To validate multi-agent social-network systems designed to characterize complex social problems a new approach is needed. The new approach must be scalable to a large number of variables and a high number of interactions. The new approach must be sufficiently automated that it can be reapplied as new data comes to light and the model is changed. It must be flexible enough to handle data at different levels of granularity, missing data, and otherwise erroneous and/or messy data. Most importantly, it must be able, at least in principle, to capture a user's intuitive informal understanding of a system. Formal methods lack precisely this ability, rendering them not applicable for validation. Formal methods are restricted by their need to be a closed world and to be logically derivable. The only technique that can scale and fulfill the above requirements is the knowledge-based methods as knowledge can be as abstract or as detailed as needed.

Capturing existing knowledge in a form that can be processed by computers, the knowledge-based approach allows knowledge inference. Furthermore, the knowledge-based approach is able to focus the search and inference in parameter space. It is scalable due to its intelligent focus with the help of ontology. It can also process causal knowledge and deal with imperfect data.

One drawback of knowledge-based approach is that there is currently a bottleneck in knowledge acquisition from human experts. The data are plentiful, particularly in the bioinformatics, biomedical informatics, economics, and social sciences. Trends in data gathering point to a deluge of data in more fields as time progresses. Current research on data mining and causal learning from data shows that it is feasible to extract knowledge from data. A drawback of knowledge-based approach is that it also needs validation, that is, the validation of knowledge bases. The validation of knowledge bases is, however, easier for human experts than the validation of simulations, as the knowledge bases are largely in the form of human-level rules and relations. Besides, stakeholders who want to validate simulations have the domain knowledge required to validate it, and should provide one for the validation process.

Chapter III: Inference in Artificial Intelligence and the Scientific Method

Knowledge-based systems (Stefik 1995) include reasoning steps or inferences on knowledge bases. An Inference Engine is the part of knowledge-based systems that controls and performs the inferences. Ontology, a specification of conceptualizations, augments knowledge bases. The type and nature of inferences need to be designed to fit the problem domain.

This chapter explores existing inference techniques in artificial intelligence and in the scientific method, describes their strengths and weaknesses, and shows a new inference technique suitable for use in validation and model-improvement.

Inference is a part of learning. While learning in artificial intelligence (e.g., machine learning, data mining, explanation-based learning, causal learning from data, reinforcement learning, case-based learning, etc.) is an intriguing topic, I chose not to delve into learning – outside inference and a simple hypothesis building and testing – to limit the scope of the work.

3.1 Inference Techniques in Artificial Intelligence

Artificial intelligence is a study of how to make computers do things at which, at the present moment, people are better. It deals with how to make computers act in a human fashion, think like humans, act rationally, and/or think rationally (Russell and Norvig 2003).

Inference is the act or process of drawing a conclusion based solely on what one already knows. Inference is studied within several different disciplines. Human inference (i.e., how humans draw conclusions) is studied within the field of cognitive psychology

(Sternberg and Pretz 2005). The rules and processes of inference are some of the oldest subject matters in philosophy. Logic studies the laws of valid inference. Statisticians have developed formal "rules" for inference from quantitative data (Lehmann and Romano 2005). Artificial intelligence researchers develop automated inference systems.

3.1.1 Inference by Search

Along with representation, search is fundamental in artificial intelligence. Search, by its virtue of looking for and of testing possible solutions, can be thought of as inference. In search, the sequence of actions required for solving a problem cannot be known *a priori* but must be determined by a trial-and-error exploration of alternatives. Almost all artificial intelligence problems require some sort of search. There are different kinds of search:

- Search in search space (Russell and Norvig 2003)
    - The representation of search space usually takes a form of graph or cellular tessellation. The way the search is performed can be breadth-first, depth-first, best-first, or heuristic.
- Search in production/expert systems (Giarratano and Riley 2004)
    - In a backward chaining procedure, the search is performed for facts that make the premises of a rule eligible for firing the rule containing the goal as a result. In both forward and backward chaining procedures, search is carried out for facts matching the clauses of a rule. Forward chaining is very useful for a system to respond rapidly to changes in its knowledge and to be able to detect one of a large number of possible unusual events. On the other hand, backward chaining is more directed, and so is more appropriate for a system that knows what it is trying to do.
- Search in genetic/evolutionary algorithm (Goldberg 1989)
    - In genetic/evolutionary algorithm, search is a function of the fitness value and is carried out by genetic/evolutionary operators. That is to say, subpopulations with the best fitness scores are sought and then

recombined (by mutation, crossover, and other genetic/evolutionary operators) to produce offspring. The process of fitness selection is then repeated with this progeny population.

The strength of search is its generality and applicability. The weakness of search is the explosion of the number of items or states a search algorithm usually has to deal with. More specifically, for

- Search in search space (Russell and Norvig 2003)
    - o Strengths: its generality and mathematical soundness.
    - o Weaknesses: the large number of states, the need for heuristics, and the need for Markovian assumption.
- Search in production/expert systems (Giarratano and Riley 2004)
    - o Strengths: it operates in the symbol space, which is usually smaller in size than the state space. It does not need to have Markovian assumption.
    - o Weakness: it is inefficient for the straightforward implementation of expert systems – keep a list of the rules and continuously cycle through the list, checking each one's left-hand-side, LHS, against the knowledge base and executing the right-hand-side, RHS, of any rules that apply. It is inefficient because most of the tests made on each cycle will have the same results as on the previous iteration. Since the knowledge base is mostly stable, most of the tests will be repeated. The computational complexity is in the order of $O(RF^P)$, where R is the number of rules, P is the average number of patterns or clauses per rule LHS, and F is the number of facts on the knowledge base. This is alleviated by the Rete I algorithm (Forgy 1982). In the Rete I algorithm, only new facts are tested against any rule LHS. Additionally new facts are tested against only the rule LHS to which they are most likely to be relevant. As a result, the computational complexity per iteration drops to $O(RFP)$, or linear in the size of the fact base. Rete I has high memory space requirements. The Rete II and III algorithms are said to have fixed this memory problem, but the algorithms are a trade secret and thus not in public domain. Using

context or structured knowledge, we may be able to avoid high computational complexity.

- Search in genetic/evolutionary algorithm (Goldberg 1989)
    - Strengths: it mimics evolution in nature, a simple but powerful mechanism. It is relatively robust to environmental change and individual failures.
    - Weaknesses: it only changes in incremental fashion and it is slow to converge. It is prone to dead ends and suboptimal solutions. Unless there is an incentive for diversity, the populations tend to become homogeneous within one particular environment or niche. If the environment drastically changes, the previously fit populations could disappear in a short period of time. While it is robust, it does not have the methodological rigor of the scientific method. It is knowledge-poor. It does not directly support the accumulation of knowledge. It relies on knowledge-less evolutionary operators of mutation and crossover.

3.1.1.1 Is Search Unavoidable?

Human scientists reason by experiments and accumulation of knowledge, in addition to search. Grandmasters in chess reason by using carefully learned structured domain knowledge. Novice chess players do a lot of unsophisticated analyses or searches. The amount of analysis and search increases significantly for the intermediate level players. The surprise is that grandmasters do not perform more analyses or searches than the intermediate level players. They instead carefully construct highly sophisticated domain knowledge and use it effectively. So the answer to the question "is search unavoidable?" is "yes, the search is avoidable". The qualification to this answer is that the search is unavoidable if specialized knowledge cannot be constructed. If knowledge can be constructed effectively, then the search is avoidable. Thus much of the computational complexity hindering an effective use of algorithms could potentially be avoided if the right knowledge is effectively constructed and used.

3.1.2 Inference by Logic

One of the foundations of modern science is the logical inference (Russell and Norvig 2003). The logical inference is a systematic method of deriving logical conclusions from premises assumed or known to be true and from factual knowledge or evidence. Logic is the science of reasoning, proof, thinking, or inference. Logic allows the analysis of a piece of reasoning, and the determination of whether it is correct or not. In artificial intelligence, logical inference is formalized into several kinds of logic, including:

- Propositional logic: a mathematical model for reasoning about the truth of propositions. Propositions are logical expressions or sentences whose truth values can be determined.
- First order logic or predicate logic: a mathematical model for reasoning about the truth of sentences that contain variables, terms, and quantifiers.
- Second order logic: a mathematical model for reasoning about the truth of sentences that contain variables, terms, quantifiers, and functions. An example of this second order logic is situational calculus (Reiter 2001).
- Temporal logic: a mathematical model for reasoning about propositions qualified in terms of time.

The strengths of logic are:

- The statements of logic are concise and clear.
- If the facts are watertight, inferences drawn from them are also watertight, provided a proper inference mechanism is used.

The weaknesses of logic are:

- Whether logic governs the workings of the universe is debatable. Logic does not trump physical experiments. Quantum mechanics, while strange to normal human logic and experience about how the (macro) world should operate, has been shown to be valid experimentally.
- Logic requires statements to be either true or false. The probabilistic nature of events and processes means that logic cannot be used without modification. Furthermore, some statements are neither true nor false.
- Logic is only part of the mental processes governing social systems.

- Predicate logic or first-order logic requires the predicate to be cleanly defined. Predicate acts like a membership function. For example, the assertion *bird(penguin)* with the predicate *bird* and the instance *penguin* assumes that a clear definition exists for the predicate *bird*. If the definition for *bird* is that of an animal that can fly and has feathers, then the assertion of penguin being a bird is false, even though in reality it is true. In social sciences, the difficulty encountered in the attempt to cleanly delineate predicates is more pronounced. For example, clean definition is difficult for the predicates *family*, *marriage*, *friends*, *enemies*, *middle-class*, etc. Without the ability to precisely delineate predicates, first-order logic and second-order logic which are based in part on predicates will not be able to perform accurately. This means the logic and the result of the logical reasoning become fuzzy.
- Second-order logic requires both the predicate and function cleanly delineated.

The above describes deductive logic. In addition to deductive logic, there is inductive logic. An argument is deductive if it is thought that the premises provide a guarantee of the truth of the conclusion. An inductive argument, on the other hand, only attempts, successfully or unsuccessfully, to provide evidence for the likely truth of the conclusion, rather than outright proof. Deductive logic works from the more general to the more specific, while inductive logic works the other way. Inductive reasoning is more open-ended and exploratory, while deductive reasoning is narrower and is concerned with testing or conforming hypotheses. Both kinds of reasoning usually present in synergy in scientific experiments. Deductive reasoning exists to confirm hypotheses from theories, while inductive reasoning exists to build theories from observations.

3.1.3 Rule-Based Systems

Given a set of facts and assertions, a rule-based system (Durkin 1994, Jackson 1999) can be created by specifying a set of rules on how to act on the set of facts and assertions. A rule is a statement composed of a condition and an action that specifies what to do when the condition becomes true. This forms the basis for expert systems (Durkin 1994, Jackson 1999). The concept of an expert system is that the knowledge of an expert is encoded into the rule set. When exposed to the same data, the expert system will perform in a manner similar to the expert. When feasible, it is desirable to derive knowledge directly from data. Causal relation learning is one of the methods to do this.

Rule-based systems are feasible for problems for which most of the knowledge in the problem area can be represented in the form of rules and for which the problem area is not too large to manage. These systems however hide the data generation process.

3.1.4 Model-based Reasoning

Rule-based systems have disadvantages. They hide the data generation process and the model of the problem. It is very difficult to build a complete rule set. It is time-consuming and error-prone to elicit empirical associations or heuristics for rules from human experts. Adding new rules requires consideration of the whole rule set, given that the rules are frequently interdependent. Furthermore, even if a rule set is complete, there is a chance of it becoming obsolete. Rules are notoriously brittle. When faced with inputs that deviate slightly from the normally expected, symbolic rule-based systems are prone to fail. Model-based reasoning provides a way to ameliorate these weaknesses of rule-based systems. Model-based reasoning, however, works wells when there is a complete and accurate model and degenerates for less accurate and less comprehensive model. A good approximation to models however is causal relations, which do not require a complete model.

3.1.4.1 Assumptions-Based Truth Maintenance System

In diagnosis, rule-based expert systems represent diagnostic knowledge mainly in terms of heuristic rules, which perform a mapping between data abstractions (e.g., symptoms) and solution abstractions (e.g., diseases). This kind of knowledge representation is shallow, in the sense that it does not contain much information about the data generation process, the causal mechanisms, and the empirical (physical, chemical, and biological) models underlying the relationships between diseases and symptoms. In everyday life, operating exclusively based on rules is quite common without the understanding or appreciation how and for what purpose the rules are created. The rules typically reflect empirical associations or heuristics derived from experience, rather than a theory of how a device, organism, or system actually works. The latter is deep knowledge in the sense that it contains the understanding of the structure, functions, and components of the device or system.

Rather than assuming the existence of an expert experienced in diagnosing a problem, model-based approaches assume the existence of a system description: a consistent and complete theory of the correct behaviors of the system. Assumptions-Based Truth Maintenance System (ATMS) is one of the approaches. Given a data set about a malfunction(s), ATMS conjectures one or more minimum perturbations to the system description that would account for the malfunctions(s).

The advantages of this deep knowledge approach over heuristic rule-based systems are (Jackson 1999):

- Given a system description, the software architect is able to avoid the laborious process of eliciting empirical associations from a human expert.
- The reasoning method is system independent, so it is not necessary to tailor the inference machinery for different applications.
- Since only knowledge of correct system behavior is required, the method is capable of diagnosing faults that have never occurred before.

3.1.5 Causal Reasoning

When there are good reasons to believe that events of one sort, the causes, are systematically related to events of some other sort, the effects, it may become possible for us to alter our environment by producing (or by preventing) the occurrence of certain kinds of events. Causal reasoning refers to the use of knowledge about cause-effect relationships in the world to support plausible inferences about events. Example applications of automated causal reasoning systems include solving diagnostic problems, determining guilt/innocence in legal cases, and interpreting events in daily life. Causal reasoning has been treated mathematically as a formal causal model and graph (Pearl 2003, Pearl 2000).

Causal reasoning has had problems with figuring out how to handle compounding, happenstance, and chaos. Causations risk oversimplifying complex phenomena. People tend to use one-to-one cause-and-effect notion. We often read "Fed interest rate increase will tame inflation" when the reality is much more complex. Major causes of inflation reduction might be the existence of Walmart and the deflationary effects of the global pool of labor. There is a debate on whether causation is fundamental. Causality is probably an anthropomorphic notion. Once a mechanism of physical or social processes is known, causality becomes secondary, which is to say, it functions as simplified explanations. On the other hand, causal reasoning is prevalent in human beings. Ignoring it is unwarranted, especially in any social modeling.

3.1.5.1 Rule-Based versus Causal Reasoning

The "if-then" rule-based inference has an unfortunate artifact of producing incorrect inferences if knowledge engineers do not take special precautions in encoding the rules. This artifact is demonstrated by the following incorrect inference from two correct rules using a correct inference mechanism (chaining):

Rule 1:	*If the lawn is wet, then it rained*
Rule 2:	*If we break the water main, then the lawn gets wet*
Inference:	*If we break the water main, then it rained*

Thus there is a need to explicitly represent causality, which includes representing actions instead of just observations and addressing confounding. Incorporating causality would enable a proper adjustment to the above rules:

Cause 1: *Raining caused the lawn to be wet*
Cause 2: *Breaking the water main causes the lawn to be wet*
Inference: *None*

As shown, *Rule 1* was encoded erroneously were causal relations taken into account. While erroneous in its cause-effect relation, *Rule 1* can still be useful as a suggestion or hint. Causal reasoning is similar to the deductive reasoning process while rule-based reasoning is similar to the inductive reasoning process. Thus both the rule-based and the causal inferences are useful.

3.1.6 Probabilistic Reasoning

Uncertainty is inherent in many problems because the real world does not operate as a Boolean system. To handle uncertainty, probabilistic reasoning is employed in artificial intelligence. There are several ways to do probabilistic reasoning: certainty factor, Bayesian Networks, fuzzy logic, etc. Bayesian Networks is currently the most widely used model in artificial intelligence, robotics, and machine learning for probabilistic reasoning.

3.1.6.1 Certainty Factors

Certainty factors provide a simple way of updating probabilities given new evidence. A certainty factor is used to express how accurate, truthful, or reliable a rule is assessed to be. It is used in the MYCIN expert system (Buchanan and Shortliffe 1984). Mathematically, a certainty factor is a number in the range -1.0 to +1.0, which is associated a rule. A certainty factor of 1.0 means the rule or proposition is certainly true. A certainty factor of 0.0 means the rule is judged to be agnostic as there is no information available about whether the rule is true or not. A certainty factor of -1.0 means the rule is certainly false. A certainty factor of 0.7 means that the rule is quite likely to be true, and so on.

Certainty factors are inelegant theoretically, but in practice this tends not to matter too much. This is mainly because the error in dealing with uncertainties tends to lie as much in the certainty factors attached to the rules (or in conditional probabilities) as in how the rules with certainty factors are manipulated. These certainty factors are usually based on rough guesses of experts in the domain, rather than based on actual statistical estimations. These guesses tend not to be very good. Certainty factors are bound by the laws of probability.

3.1.6.2 Bayes Theorem and Bayesian Networks

The essence of the Bayesian approach (Neapolitan 2003) is a mathematical rule explaining how one should change one's existing beliefs in the light of new evidence. The Bayesian approach is founded on Bayes Theorem, an expression of correlations and conditional probabilities. Conditional probabilities represent the probability of an event occurring given evidence. Bayes Theorem can be derived from the joint probability of A and B (i.e., $p(A,B)$) as follows:

$$p(A,B) = p(B,A)$$

$$p(A|B)p(B) = p(B|A)p(A)$$

$$p(A|B) = (p(B|A)p(A)) / p(B)$$

where $P(A|B)$ is referred to as the posterior, $P(B|A)$ is known as the likelihood, $P(A)$ is the prior and $P(B)$ is generally the evidence.

A Bayesian or belief network represents the same information as a joint probability distribution, but in a more concise format. The graph of a network has nodes which represent variables and directed edges which represent conditional probabilities. This directed graph is prohibited to have directed cycles. The nodes are connected by arrows or directed edges which show the influence of the variables upon one another.

Each node has a conditional probability table that quantifies the effects of the other nodes that have influences on it.

Bayesian methods have been successfully applied to wide range of problems. They are easy to understand and elegant mathematically. They are based on classical probability, and thus are considered sound by most researchers and have the aura of scientific respectability even though the specification of priors does not have a rigorous treatment. The determination of conditional independence and Markovian property is partially based on judgment calls.

In certain circumstances, however, Bayesian methods are not appropriate. Let A represent the proposition *Kirsten Dunst is attractive*. The axioms of probability insist that

$$P(A) + P(\sim A) = 1$$

Now suppose that a person does not even know who Kirsten is. We cannot say that this person believes the proposition if he/she has no idea what it means. Moreover, what makes a person attractive varies across cultures and persons. Neither is it fair to say that he/she disbelieves the proposition. It should therefore be reasonable and meaningful to denote his/her belief of $bel(A)$ and $bel(\sim A)$ as both being 0.

Bayesian networks are a powerful method for representing and reasoning with uncertainty. Most of the applications of Bayesian networks, however, have been in academic exercises rather than industrial applications that real businesses rely on. The main reason why Bayesian networks have not yet deployed into many significant industrial-strength applications lies in its knowledge acquisition bottleneck. It is immensely hard to acquire conditional probability relations and priors correctly from human experts. This lack of industrial applications of Bayesian networks stands in contrast with the successful industrial applications of simulations and expert systems.

3.1.6.3 Inference in Artificial Neural Networks

An Artificial Neural Network (ANN) is an information processing method that is inspired by the way biological nervous systems, such as the cortex, process information (Dayhoff 1990, Anderson 1995). It is composed of a large number of highly interconnected processing elements (neurons) working in concert to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true for ANNs as well. ANNs can handle numerical pattern classifications well but not symbolic reasoning.

3.1.6.4 Fuzzy Logic

Fuzzy logic (Kosko 1996) allows partial set or fuzzy membership rather than crisp set membership. This gives birth to the name fuzzy logic. It is a variant of multi-value logic. Fuzzy logic commences with and builds on a set of linguistic rules provided by humans, usually containing soft or qualitative variables. The fuzzy systems convert these rules to their mathematical equivalents using membership functions. This makes the task of the software architect simpler and results in closer representations of the way systems behave in the real world, especially when soft or qualitative variables are involved. Additional benefits of fuzzy logic include its simplicity and its flexibility. Fuzzy logic can handle problems with imprecise and/or incomplete data, and it can model nonlinear functions of arbitrary complexity.

Weaknesses of fuzzy logic include the use of *ad-hoc* non-linear truncation and jagged interpolation in its membership functions and the fuzziness of the qualitative symbolic data – linguistic variables – such as “very tall”, “tall”, etc. The vagueness of fuzzy variables hinders the exact representation and reasoning needed for the rigor of sound science. Furthermore, multi-value logic such as fuzzy logic has a larger risk of losing its meaning as the number of multiple-logic-values increases. Even though in the end a fuzzy variable gets mapped into real numbers, exactness is crucial and cannot be guaranteed by fuzzy logic.

3.1.7 Evidential Reasoning

Instead of focusing on the truth value or probabilistic value of assertions and propositions, which may be abstract, evidential reasoning focuses on the evidence itself and the data generation processes. Evidential reasoning requires several conditions to operate, such as:

- Falsifiability: contrary evidence that would prove a claim false must be possible to conceive of.
- Comprehensiveness: the evidences offered in support of any claim must be exhaustive.
- Logic: any argument offered as evidence in support of any claim must be sound. An argument is said to be "valid" if its conclusion follows unavoidably from its premises. It is "sound" if it is valid and if all the premises are true.

3.1.7.1 Dempster-Shafer Theory of Evidence

The Dempster-Shafer Theory of Evidence (Russell and Norvig 2003) was introduced as a way of representing epistemic knowledge. In this formalism, the best representation of chance is a belief function rather than a Bayesian mass distribution. There are two measures of certainty: belief and plausibility. Belief denotes the support each conclusion has from the observations. Plausibility accounts for all observations that do not rule out a given conclusion. Dempster-Shafer Theory (DST) of Evidence’s appeal rests on the fact it more naturally encodes evidence instead of propositions. Bayesian theory is included in the theory of evidence as a special case, since Bayesian functions are belief functions, and Bayes' rule is a special case of Dempster's rule of combination.

The Dempster-Shafer Theory of Evidence is not as widely applied as the Bayesian Networks due to the following:

- It is not based on classical probability. Thus it is deprived of the aura of scientific respectability.

- In the late 1970s, Lofti Zadeh wrote a critique that states that Dempster's Rule of Combination has a fundamental discrepancy in that it may yield counterintuitive results when given conflicting information (Zadeh 1984).
- However subjective and arbitrary Bayesian priors are they are simple to understand. Furthermore, Bayesian Networks and Bayesian Statistics can be elegantly formulated mathematically.

Recent research shows that Zadeh's critique against Dempster's Rule of Combination is unjustified (Haenni 2005). A compelling but ultimately erroneous example based on Zadeh's critique is as follows. Suppose that a patient is seen by two doctors regarding the patient's neurological symptoms. The first doctor believes that the patient has either meningitis with a probability of 0.99 or brain tumor with a probability of 0.01. The second doctor judges the patient suffers from a concussion with a probability of 0.99 but admits the possibility of a brain tumor with a probability of 0.01. Using the values to calculate the *m*(brain tumor) with Dempster's rule, it is found that *m*(brain tumor) = *bel*(brain tumor) = 1.0, which means it is 100% believed that the brain tumor is the correct diagnosis. This result implies a complete support for a diagnosis that both doctors consider to be very unlikely. A common but mistaken explanation for this is that the possible conflicts between different pieces of evidence are mismanaged by Dempster's Rule of Combination. This is a very compelling example, which is why this contributed to the near demise of Dempster-Shafer Theory of Evidence research. Many researchers have used this example to completely reject DST or construct alternative combination rules (Sentz 2003).

The counterintuitive result turns out not to be caused by a problem within Dempster's Rule of Combination, but rather by a problem of misapplication. Zadeh's model does not, in fact, correspond to what people have in mind in such a case. There are two different ways to fix the problem (Haenni 2005).

One way is based on the observation that the diseases are not exclusive. The simple set θ = {meningitis, concussions, brain tumor} implies exactly one of these diseases is the true one. In reality, diseases are almost never exclusive, so Zadeh's choice for θ is in question. Switching from {meningitis, concussions, brain tumor} to its power set (that is, the combinations of diseases) would give the frame of reference $\theta = \{\phi, M,$

$C, T, MC, MT, CT, MCT\} = 2^{\{M,C,T\}}$, where M=meningitis, C=concussions, T=brain tumor. Using this power set, DRC behaves normally (Haenni 2005).

The other way is based on the observation that experts are not fully reliable (Haenni 2005).

Thus as long as the right model is utilized, DRC is a good method to combine evidence thus making DST useful. In fact, DRC behaves very well for high conflicts and especially for high conflicts. This suggests DST is as reliable as – if not more so – Bayesian Methods. The caveat here is that, of course, the model must be correct. In many applications, it is not trivial to derive the correct model. Furthermore, to use DST, we have to first have the Frame of Reference chosen over certain parameter space. The choice of the Frame of Reference for simulation systems is non-trivial and usually *ad-hoc*. Additionally, ignorance implies a contradiction in simulation systems (which we have God's eye view) or when validation knowledge is provided.

3.1.7.2 Data Fusion

Data Fusion (Hall and Llina 2001) is the process of combining multiple data for the purpose of producing information of better value than the individual processing of data alone. Data originate from many sources. Sources may be similar, such as multiple radars, or dissimilar, such as acoustic, electro-optic, electro-mechanical (e.g., haptic devices), or passive electronic emissions measurement. A key issue is the ability to deal with conflicting data and producing intermediate results revisable as more data becomes available.

Instead of using Dempster’s Rule of Combination, in certain domains it is more reasonable to directly use the domain knowledge and ontology to combine evidence. If domain knowledge and ontology can be adequately specified, it can become more straightforward to use it to combine evidences and do reasoning. Dempster’s Rule of Combination and Dempster-Shafer Theory of Evidence, while useful, are general formalisms. Specific domain knowledge, on the other hand, has specialized, precise, and effective application.

3.2 Inference Techniques in Scientific Method

Strangely the field of artificial intelligence (Russell and Norvig 2003) largely ignores how scientists carry out experiments, build models and hypotheses, accumulate knowledge, test hypothesis, construct theories, and draw conclusions. The exception is in the use of inductive logic for scientific discovery. But due to the inherent weaknesses of induction, the method is not part of the main artificial intelligence reasoning techniques. It can be given any other name, but the above scientific activities can be best described by the word and notion of inference. It is known more commonly as the scientific method, which is fundamental in advancing science. No other methods of inference have matched the effectiveness of the scientific method in understanding the real world.

3.2.1 Statistical Inference

Statistical inference deals with the problem of inferring properties of an unknown distribution from data generated by that distribution. The most common type of inference involves approximating an unknown distribution by choosing a distribution from a limited family of distributions. Generally this family of distributions is specified parametrically.

One of the weaknesses of statistical inference is that it is biased toward the majority of a sample population. If there is a person who has a very eccentric personality and lifestyle, this person will often be considered as an outlier, a noise, or an error. A sufficient number of eccentric individuals may cause Type I Error. While focusing on getting accurate statistics of sample populations is useful, in many cases eccentric individuals are the key to describing and predicting events. For example, international air travelers played a major role in the spread of Severe Acute Respiratory Syndrome (SARS) in 2003 before it was contained.

Moreover, statistical inference lacks a means to describe causal relations. It also lacks an effective means to handle symbolic knowledge. During SARS outbreak, the symbolic knowledge of how the disease spreads was a crucial clue of how to handle the outbreak. It assumes independence between samples: it does not take networks into account. Social networks play an important role in human society and, by extension, in multi-agent systems modeling human society.

The above weaknesses can be remedied using a combination of simulation and knowledge inference. Simulation allows high precision in modeling a sample point (e.g., as an agent and as networks). Knowledge inference allows the reasoning based on the structures of the real world problems to augment statistics.

3.2.2 Hypothesis Building and Testing

There are really only two ways to ascertain how the world works. One way is to talk and argue about it, but this is unreliable as arguments and words alone cannot determine if a statement is true. Logic, while helpful, is not without difficulty: the premises, the assumptions, and the application of logical entailments need to be correct. Logic does not exist in a vacuum. Furthermore, how the physical and social worlds operate does not conform to "logical" commonsense or even pure logic. Much of physics is counterintuitive. Thus proofs based on observations and experiments are required.

A better way is to perform careful observations and carry out experiments. The result of doing this is universal as it is reproducible by any skeptic. This forms the basis of the scientific method, which is the best way yet discovered for discerning the truth from delusions and untruths. The basic steps of the scientific method are:

1) Observe some part of the reality.
2) Introduce a tentative description – a hypothesis – that is consistent with the observation.
3) Use the hypothesis to make predictions.
4) Test the predictions by experiments or further observations and modify the hypothesis in the light of the results.
5) Repeat steps (3) and (4) until there are no discrepancies between the hypothesis and experiment and/or observation.

When there are no discrepancies left between the hypothesis and the experiment and/or observation, consistency is obtained. Consistency is crucial to establish validity. Based on this consistency within a class of phenomena, the hypothesis becomes a theory. A theory is a framework within which observations are explained and predictions are made. The above steps show that knowledge is physically constructed from the empirical primitives through hypothesis formation and testing. In other words, meanings are constructed grounds-up from facts. Systems can only be understood in terms of physical processes which manifest them and by which they are assembled. Semantics, ontology, language, and mathematics must be understood in the context of the physical reality.

What is lacking in artificial intelligence inference techniques is a careful application of the scientific method. Artificial intelligence focuses on specific inference, search, and/or learning algorithms. Machine learning focuses on general learning algorithms. The crucial task of constructing and testing hypotheses is left for human researchers to perform. Part of the reason is that observation which requires visual recognition is hard to automate. However, in the modeling and simulation field (in which visual recognition is not as hard) hypothesis building and testing which requires knowledge-intensive model and meta-model building is left without automation. Policy iteration in reinforcement learning is one technique analogous to hypothesis building in artificial intelligence.

Using social simulations grounded by empirical knowledge and data as a proxy of the real world, a novel inference technique can perform experiments in simulations and build hypotheses using inference engines by tweaking meta-models of the simulations. If a means presented itself in sensing and manipulation, then real world experimentation is enabled, supplementing simulation-based experimentation.

The question of search space and computational complexity may be addressed by performing careful hypothesis building and testing. The hypothesis building and testing requires deep knowledge and educated guesses. In other words, it requires deduction and induction. Achieving human intuition is hard, but intuition can be at least partially emulated using deduction and induction. Knowledge inference and meticulous virtual experimentation is the first step toward scientific method-capable artificial intelligence. Here, the simulation with its models and meta-models is the representation of real world knowledge. Instead of representing the real world as Bayesian Networks or other conventional artificial intelligence representations, it is represented more faithfully as simulations.

3.3 Knowledge-based Hypothesis Formation and Testing

Simulations serve as a proxy to the real world. If the simulation representation of the real world is good enough for the policy question at hand, then experiments performed in the simulation would likely remain valid in the real world and the simulated experimental results would mimic the results of real world experiments.

As knowledge inference, ontological reasoning, and simulation are combined, the hypotheses can be constructed by:

(1) Searching the knowledge-base for unknown and/or uncertain knowledge areas.
(2) Inference using the knowledge-base to detect the probable existence of new rules.
(3) Discovery of data patterns that do not fit into any of the knowledge-base rules.
(4) Opportunistic search and fitness-based search.
(5) Knowledge-based ontology search or ontological reasoning.
(6) Classification examination by ontological reasoning.

Hypotheses can be tested by proxy using simulations. Empirical data is used to validate the simulations for the policy question.

The above differs from inductive logic programming in that it is not solely reliant on logic. It uses deep knowledge and pattern analysis for induction. It also goes in the deductive direction, working from domain conceptual knowledge to suggest a probable existence of new rules or ontological categories. It perturbs existing knowledge and model to find a fit to new experimental results and/or observations.

Natural science provides examples of the power and insight of a proper classification, a kind of ontology. Two successful examples are the Darwinian evolutionary classification of life forms even before the arrival of DNA classification and the Periodic Table of Chemistry that is capable to predict the existence of yet discovered chemical elements. The strengths of these classification ontologies derive from the fact that they focus on natural processes. Scientific understanding of the natural processes underlies the classification ontologies. The Standard Model of particle physics is another example of successful classification ontology. All these indicate the utility of knowledge-based and ontological reasoning when properly used, especially with careful consideration of natural processes.

3.4 Summary

Current artificial intelligence reasoning techniques have strengths and weaknesses summarized in the following table.

Table 1. Reasoning Methods Comparison

Method	Strengths	Weaknesses
Search in search space	Generality	Computational complexity
Search in production systems	Operates in symbolic space, usually a smaller space than state space	Naïve implementation causes long processing time. Rete I fixed this but at the cost of memory. Rete II and Rete III improved upon Rete I, but are not in the public domain.
Search in genetic and evolutionary systems	Powerful and robust	Incremental change, hard to avoid suboptimal solutions, knowledge-less evolutionary operator of mutation and crossover
Logical inference	Clear, concise, and if the facts and the inference mechanism are watertight, the inference is watertight.	Not all natural and social phenomena are logical. Need to assign true or false values to every statement.
Rule-based inference	Ability to capture expert knowledge	The inference is only as good as the quality of expert heuristic knowledge. If the rules are derived from the conceptual model, however, this weakness disappears.
Causal inference	Ability to emulate causal reasoning	Causal reasoning thrives when the mechanisms are still unclear or undiscovered. It risks oversimplifying complex phenomena.
Certainty factors	Simple but workable	Bound by the laws of probability. Ignorance cannot be modeled.
Bayesian networks	Easy to understand, mathematically elegant, based on classical probability	Non-rigorous priors, bound by the laws of probability. Ignorance cannot be modeled.

Artificial Neural Networks	Mimics natural neural circuits to a degree. Adaptive learning, self-organization, real-time operation, and fault tolerance.	Cannot operate on symbolic information. In pattern recognition, it is subsumed by the Support Vector Machine (Vapnik 2000).
Fuzzy logic	Simple and flexible	Fuzzy. Brittle membership functions. Multi-value logic risks losing meanings when the number of logic-values increases.
Dempster-Shafer	General, robust, and reliable. Generalize Bayesian Methods. Ignorance can bee modeled.	Need to be careful in modeling to avoid errors in evidence combination of conflicting information. Ad-hoc Frame-of-Reference determination.
Data Fusion	Specialized methods, including ontology-based method	Not general. Strengths and weaknesses depend on chosen methods and application domain
Statistical inference	Mathematically sound.	Cannot operate on symbolic information. Causality cannot be modeled. Minority eccentric individuals smoothed over.
Hypothesis building and testing (scientific method)	General and powerful method. Not limited to numerical information.	More complex than statistical inference. Much more knowledge-intensive. Require "intelligence" to construct hypotheses

The most promising techniques are knowledge-based methods and hypothesis building and testing based on the scientific method. Knowledge-based methods work mostly on symbolic information. Hypothesis testing in statistics works mostly on numerical data. Hypothesis testing based on symbolic information is only used manually. Knowledge-based hypothesis building and testing allows the processing of both numerical and symbolic data.

Combining knowledge-based methods (including causal and rule-based systems) and hypothesis testing – both numerical and symbolic – is a good way to create a validation and model-improvement system for simulations. Instead of focusing on pure logic, causal logic (and thus our knowledge-based methods) allows the focus on the processes and/or mechanisms of the real world. As knowledge-based hypothesis building and testing, augmented by simulations and focusing on processes and mechanisms, is similar to what human scientists do in their scientific work, it might form an empirical path toward artificial intelligence.

Chapter IV: What-If Analyzer (WIZER)

This chapter describes a tool based on the knowledge-based and ontological approach for validation. First, I elaborate how the tool works conceptually. Next, I describe the detailed components of the tool. This includes the knowledge spaces, the Alert module, and the Inference Engine. The Alert module performs data description and matching resulting in symbolic information in addition to numeric one. The Inference Engine performs inferences on simulation events. In the knowledge space, I create and use a simulation description logic, which is inspired by ontology (Gomez-Perez et al. 2004) and the DAML+OIL inference language to describe the simulation model and results. This integration effort follows similar integration efforts on Logic Programs and Description Logic (Grosof et al. 2003) and on Logic Programs and Production Systems (Grosof 2005).

I call the tool WIZER, for What-If Analyzer. While WIZER enables validation, I also describe how it enables model-improvement. Next, I give an illustrated run of the tool. Finally, feature comparison between WIZER and other tools is provided.

As WIZER is a knowledge-based tool, the importance of knowledge – and the reasoning based on that knowledge – is emphasized in this chapter. While WIZER uses statistical tools, they are used in the context of knowledge bases and inferences. The simulation output curves have the knowledge components behind them and they can be described based on knowledge. All inference rules and descriptions about statistical tools are encoded declaratively first, with additional supporting routines encoded imperatively (in procedural manner).

The main obstacle in any knowledge-based tool is the knowledge acquisition bottleneck, which is the difficulty of extracting knowledge from human experts. WIZER partially avoids this knowledge acquisition bottleneck to the extent possible by distributing the knowledge acquisition responsibility to the corresponding stakeholders: simulation knowledge to the simulation developers and validation knowledge to the validation evaluators. Causal learning from data and machine learning techniques can be used to address the knowledge acquisition bottleneck. This dissertation only gives an example of knowledge-based search and hypothesis testing for acquiring new knowledge in the form of new causal relations.

4.1 How WIZER Works Conceptually

WIZER includes a knowledge space module, the Alert module, and the Inference module. The knowledge space module contains causation rules for the simulation model and the domain knowledge in the form of graph. The graph's nodes represent entities, while the edges represent relationships. The Alert module does two tasks: (1) describing the data, e.g., using statistical and pattern classification tools, (2) matching that data description with empirical data, producing symbolic alerts. Symbolic alerts here are defined to be symbolic characterizations of numerical data (not just alerts in the sense of imminent danger). These symbolic alerts allow WIZER's Inference Engine to process causation and IF-THEN rules. (The Inference Engine can also consider numerical data.) The principle of inference is a simple one: being able to derive new data from data that is already known. The Inference Engine module takes in the outputs of the Alert module, performs inferences on them, and produces recommendations on which variable to change and by how much. The inferences are aided by ontology. The ontology is defined as a specification of a conceptualization. Every knowledge-based system is committed to some conceptualization. Here I choose to make the conceptualization explicit, using ontology. The Alert and the Inference Engine modules can be used on their own given appropriate inputs. Figure 3 (below) shows the diagram of WIZER.

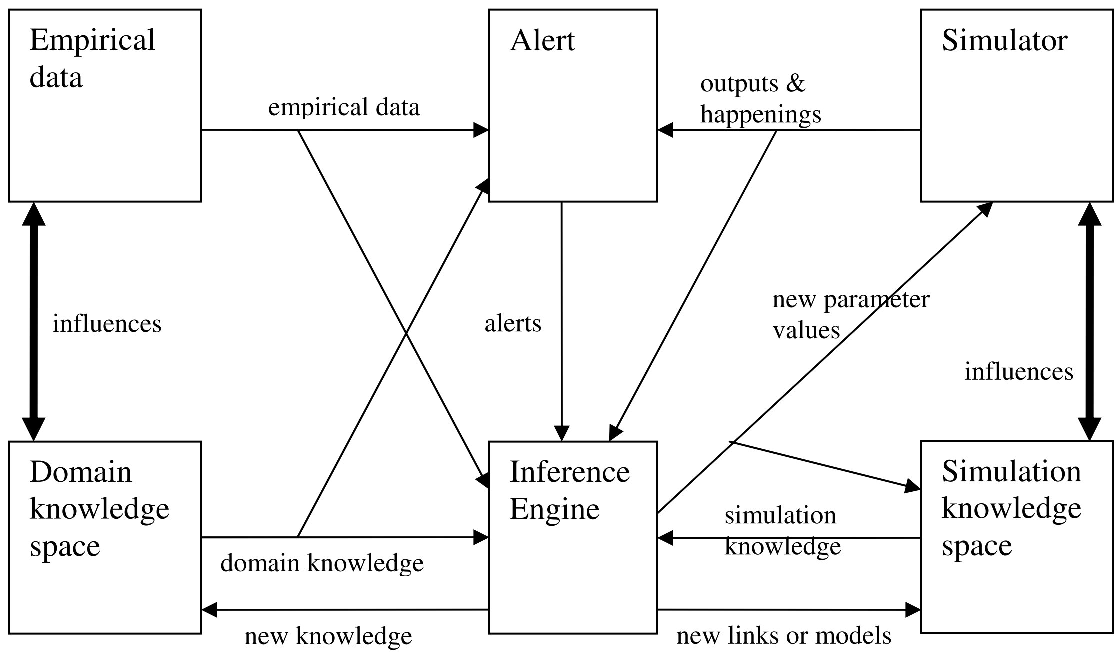

Figure 3. WIZER Diagram

The Domain Knowledge Space module provides domain knowledge to the Inference Engine. The knowledge is in the form of graphs or networks. The other name for domain knowledge space is domain ontology; they are assumed to be the same here. The empirical data could change the domain knowledge and the domain knowledge could restrict and influence what empirical data is acceptable. This depends on the strength of evidence supporting the knowledge and the data.

The Simulator Knowledge Space module provides the simulator with knowledge such as the causal network of the simulation model to the Inference Engine. The Inference Engine produces new parameter values and possibly new links for the Simulation Knowledge Space module. The simulator influences and is influenced by the Simulator Knowledge Space module. The parameter data used in the simulator is assumed to be contained in the Simulation Knowledge Space module. The parameter data is empirical, but this empirical data is used in the simulator. As the empirical data used in the simulator is not the same as the data used for validation, this separation makes the distinction conceptually clear.

Both domain and simulator knowledge spaces are represented by a graph. More significantly, I created a new derivation of description logic to describe the knowledge spaces, the simulation model, the simulation outputs, empirical data, and statistical test. This description logic was inspired by DAML+OIL and RDF and is called Simulation Description Logic. In the N3 notation for RDF, the basic syntax is a simple one: *<variable1> <relationship> <variable2>*, where *variable1* could be a subject, *relationship* could be a verb, and *variable2* could be an object.

The Alert module evaluates simulation output data with respect to corresponding empirical data. Before this evaluation, the Alert module computes the description of the output data, possibly using statistical tools. For example, the Alert module can symbolically describe the ups-and-downs of a school absenteeism curve taking into account other symbolic/contextual information such as the holidays and vacations. The evaluation produces symbolic alert information. The symbolic alert converts quantitative data into symbolic categories. Thus the Alert module converts quantitative data into alert symbolic categories. As noted before, the notion of alert here includes normal symbolic information, not just emergency/alert information. In other words, it is in essence the symbolic categorization or identification of numerical information. A measure of validity can be computed using special categories denoting that the outputs "match empirical data and/or knowledge". While not depicted in the figure to avoid unnecessary clutter, the Alert can semantically categorize input data and empirical data as well.

The Inference Engine takes the outputs from the Alert module and the simulator's causal diagram and possibly a meta-model (of the simulation's knowledge space), in addition to empirical data, domain knowledge, and parameter constraints (of the domain knowledge space), to make a judgment on which parameters, causal links, and model elements to change – or not to change – and how. How much a parameter value or a link should change is influenced by the simulation model. The inference engine calculates the minimal perturbations to the model to fit the outputs according to a model-based approach similar to the Assumptions-Based Truth Maintenance Systems, which keeps the assumptions about the model in an environment lattice. The model (including the causal diagram) and the potential alternate models are coded in ontology and rules using Simulation Description Logic. The perturbations are implemented as the effects of ontological and rule-based reasoning. The inference produces new parameters for the next simulation. This cycle repeats until a user-defined validity level is achieved. (The user interface module is not shown for clarity.) In short, the Inference Engine figures out which parameters, links, and models need to change to fit the simulation to empirical data and domain knowledge.

In addition to having rule and causal inference submodules, WIZER has submodules for simulator knowledge and domain knowledge operation, validation, and model-improvement. The validation submodule computes the degree of match between simulation outputs and knowledge against empirical data and knowledge. The model-improvement submodule determines the changes needed to make the simulator outputs and knowledge better match the empirical data and knowledge. Empirical knowledge here forms domain knowledge; while some domain knowledge may not be empirical, we use the terms interchangeably here for the notion of target knowledge. To compute the needed changes, hypothesis building is employed based on existing knowledge. The next simulation(s) would then test the hypothesis. An additional routine keeps track of whether there is an improvement in the simulation validity.

Figure 4 shows the inference or reasoning types in WIZER. The diagram in the figure is the same with that of Figure 3, but with the inner reasoning types shown and the data flow descriptions hidden for clarity.

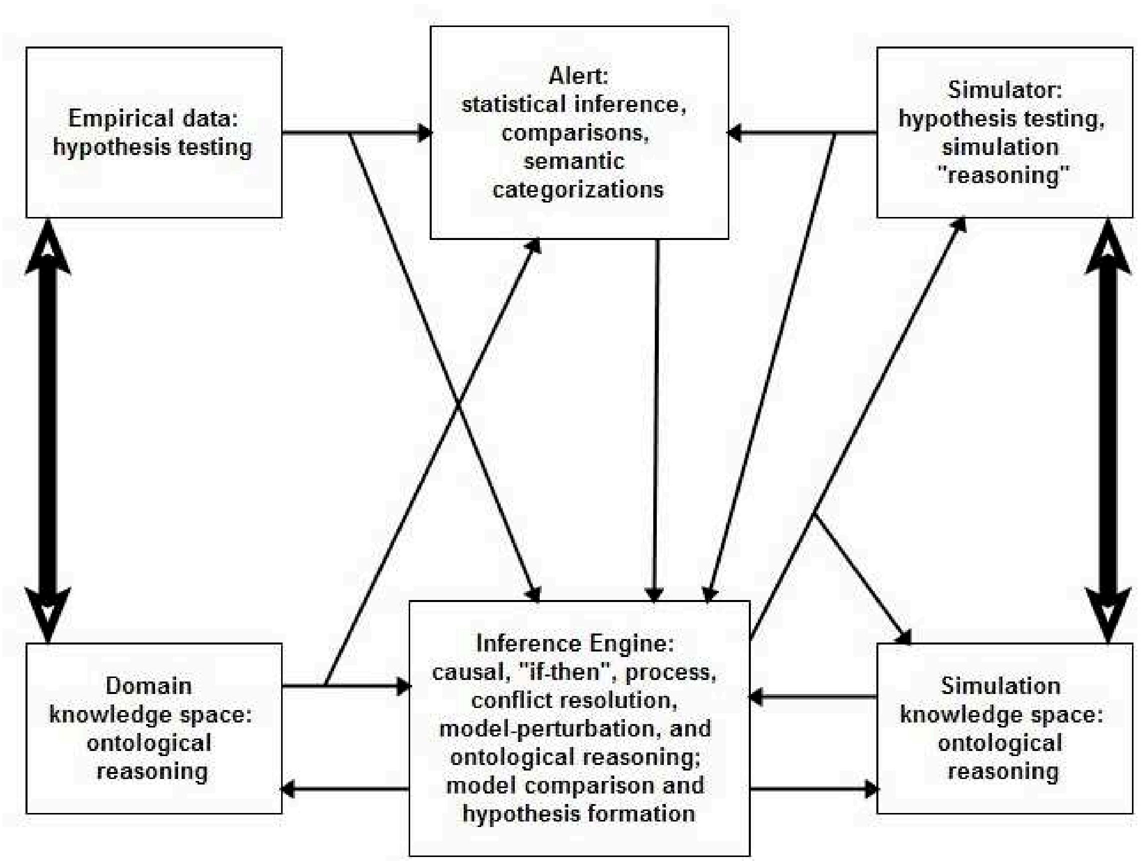

Figure 4. Types of Reasoning in WIZER

As shown, the Alert WIZER employs statistical inference, comparison, and semantic categorization aided with knowledge and ontology. The Inference Engine has reasoning mechanisms which form the core of WIZER: causal reasoning, "if-then" rule-based reasoning, conflict resolution, model perturbation, and ontological reasoning for validation and model improvement purposes. It also has model comparison and hypothesis formation for the purpose of model improvement. Both the domain knowledge space and the simulation knowledge space employ ontological reasoning. The simulator acts as if it has "simulation" reasoning, which plays a role at producing emergences, for example. The hypotheses are tested by proxy in simulator validated against empirical data and knowledge. They can also be tested directly against the empirical data. Data mining and machine learning tools, outside the current WIZER implementation, can be employed to extract information from the empirical data.

4.2 Definition of WIZER by the Computer Science Concepts

WIZER is a mix of knowledge-based and ontology-based system, tied to the simulation model. It includes model-based reasoning in the form of causal and ontological reasoning. It also has rule-based reasoning tied to the model. Additionally, it describes its statistical tools using ontology. Underlying the causal relations, WIZER has the process ontology and process logic based on the simulator conceptual model (and thus the code implementation of the conceptual model). WIZER model-based reasoning is similar to truth maintenance systems, but instead of using a dependency network (or an environmental lattice), it uses a causal network. WIZER rule-based reasoning is similar to a forward-chaining system but with rules tied to the simulation model and with ontology- and model-based conflict resolution. The knowledge-based and ontology-based routines are closely tied to the simulation models, simulations, and empirical data. This makes WIZER unique among and different from other knowledge-based systems.

Concisely, WIZER is defined as an ontological and knowledge-based simulation model reasoning system, with process, rules, causation, and statistical components.

The steps for preparing a simulation system for WIZER are:

1. Take or create the conceptual model of the simulation.
2. Create the causal model from the conceptual model. This causal model consists of the abstract influence/causal model and the concrete causal model. The abstract causal model represents which variable influences another variable. (This abstract causal model can be thought of as the influence model, but I use the notion causal model to emphasize causality.) The concrete causal model represents how a variable with a value causes another variable to have another value. These causal models allow expedited probing of the root cause of a problem. This is similar to

the environmental lattice which allows perturbations to the system descriptions in assumption truth maintenance systems.

3. Create the process logic for each causal relation in the causal model. This process logic is closely tied to implementation code.
4. For each relevant output variable of a causal relation, create a semantic/ontological description or potential classification of the possibly dynamic output/variable.
5. Create rules based on the causal model and the process logic.
6. Introduce conflict resolution rules based on the causal model.
7. For all the steps above, the relevant ontology is created and used as needed.

Generating causal models from simulation conceptual models may be ontologically and computationally feasible, but is not done here. Physically, mechanisms and/or processes form the foundation for causality. Causal relations are constructed by human beings based on perceived or inferred order in the seemingly chaotic world.

4.2.1 An Example of WIZER Setup

A small portion of the code of the BioWar simulator is presented below in pseudo-code to serve as an illustration. The pseudo-code represents a function which adds new symptoms and symptom severity values to an agent who contracts a disease.

```
function AddSymptoms in an agent who has a disease
for all new symptoms do
        let symp be a new symptom
        if symp already exists in the agent
                increase duplicate symptoms count
        else
                get the evoking strength value from the QMR table referenced by symp
                convert the evoking strength value to a severity value
                add the severity value to the total severity value
        end of if

        add the symptom symp to the agent
end of for
end of function, return the total severity value
note QMR stands for Quick Medical Reference, a table relating diseases and symptoms
```

The step-by-step procedure for the above routine is as follows:

1. The conceptual model for this routine is an agent with a disease having one or more symptoms manifested for this disease and these symptoms have severity values whose sum is sought.
2. The abstract causal model for the routine is simply "the existence of a symptom causes the realization of the symptom severity, which in turn causes the increase in the total severity for this agent". (Of course, a symptom and its severity are inseparable physically, but this is the causal model for the simulation, not for the empirical world.) The concrete causal model is not available for this example.
3. The process logic (and the process model) is the logic and semantic description of pseudocode, algorithm, and the code itself (for simulation models). It is augmented with the process ontology. For empirical or domain knowledge, the process logic represents the real world process.
4. We have three variables in the causal model: the existence of a symptom, the severity of this symptom, and the sum of the total severity of all symptoms. These are all described semantically. In more complex variables, curves or surfaces may be described semantically.
5. The rule for relating the severity of the symptom to the existence of the symptom is a simple "severity of the symptom implies existence of the symptom" for the

above routine. Moreover, another rule can say "if total severity value of symptoms is not zero then some symptoms must exist".

6. There is no conflict resolution for the rules, as the rules are simple and have no conflict.
7. Relevant ontologies are created for the conceptual model, causal model, process model, and rules.

4.3 Simulation Description Logic

Description logics are considered one of the most important knowledge representation schemes unifying and giving a logical basis to previous representations such as frame-based systems, semantic networks and KL-ONE-like languages, object-oriented representations, semantic data models, and type systems. Resource Description Framework (RDF) is a universal format for data on the Internet based on description logic. Using a simple relational model, it allows structured and semi-structured data to be mixed, exported, and shared across different applications. RDF data describes all sorts of things, and where XML schema just describes documents, RDF – and DAML+OIL – talks about actual things. In RDF, information is simply a collection of statements, each with a subject, verb and object – denoted as a triple. A human readable notation for RDF is known as Notation3 or N3. In N3, an RDF triple can be written as the following, with a period ending:

```
<#pat> <#knows> <#jo> .
```

DAML provides a method for stating properties such as inverses, unambiguous properties, unique properties, lists, restrictions, cardinalities, pairwise disjoint lists, datatypes, and others.

Following the spirit of description logic, here I create the Simulation Description Logic (SDL) adopting a somewhat modified tripled notation. The subject and object part of the triple can be any variable, instance, or concept. In WIZER, the subject and object part can consist of multiple variables, instances, or concepts. Thus this forms a modified N3. The verb part of the triple contains any relation of interest. In particular, the verbs "causes" and "is influenced by" denote causal relations and "if-then" denotes if-then rule relations. The real-world semantics can be encoded this way of what the simulation code is supposed to accomplish. This corresponds to facts and rules of the knowledge base. The real-world semantics can be delimited and influenced by the policy question at hand.

Additionally, to describe the simulation outputs which often come in the form of curves, in addition to the semantic description of the outputs using the above tripled notation, I use an augmented notation of:

<input> <computational logic> <output> .

to denote the computational logic or process. The process logic underlying causal relations adopts similar notation. For example, the process for the statistical computation of a mean is as follows:

```
Name: mean computation
Input: real numbers N1, N2, …, Nx of a sample population, or in N3,
               <x numbers> <is part of> <a sample population> .
Computational logic: add(N1, N2, …, Nx) divided by count(N1, N2, …, Nx)
       Mean describes a sample population, or in N3:
               <mean> <describes> <a sample population> .
               <mean> <is> <add numbers divided by number count>
Output: result of the computational logic in the form of a real number N
```

Using this logic description, the Inference Engine can be made to reason about the statistical tools it is using. The triple *<mean> <describes> <a sample population>* is the semantic description, while the "add…divided by count…" part is the computational or process logic. The former is declarative, the latter is procedural. Both are needed and related to each other.

4.4 Simulation and Knowledge Spaces

The steps of finding a match between the simulator outputs/model and the target knowledge can be viewed as a search in model and knowledge spaces. Figure 5 illustrates the relationship between the search in the model and knowledge spaces. In the model space, the search is through its parameters and links. In the knowledge space, the search is through rules and causations, in addition to ontological reasoning. The two searches influence each other. The results of search in knowledge space may reduce the scope for search in simulation model space; the range of allowed parameter values in simulation model space may restrict the scope of search in knowledge space. The perturbation of system description is described in the knowledge and simulation space using ontology and rules. This includes the consideration of variable compounding (e.g., variables that must be simultaneously on, off, or have certain values for a specific output response to occur).

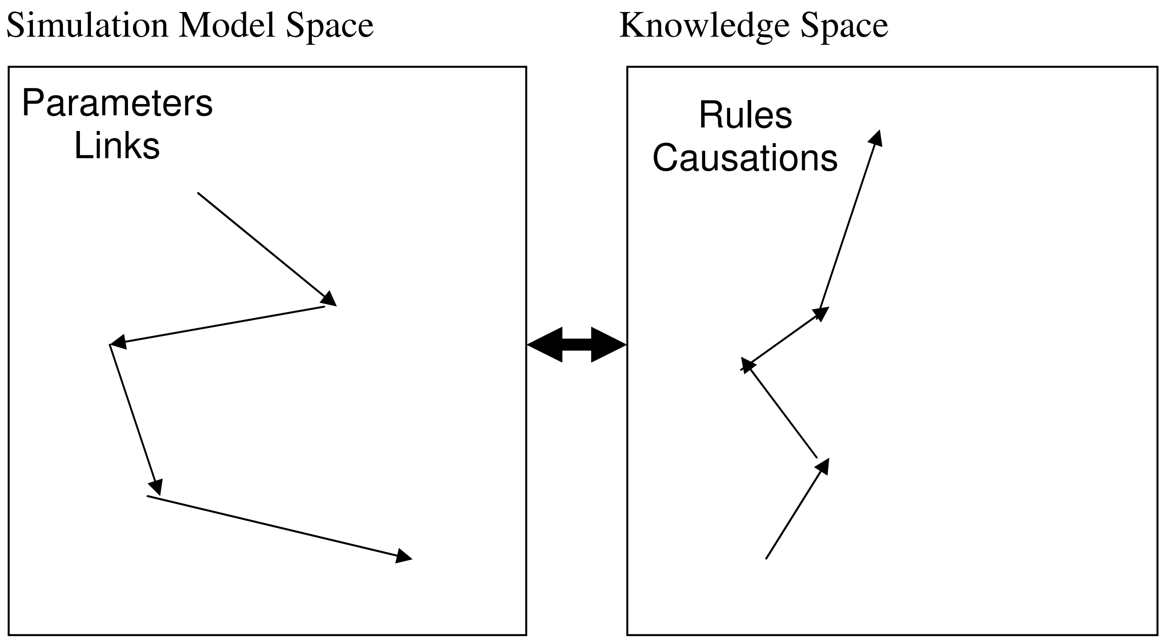

Figure 5. Searches through Simulation Model versus Knowledge Spaces

In WIZER, the simulation knowledge space is represented by a graph with its corresponding operations. This graph has nodes representing concepts and links/edges representing the relationships between concepts. Furthermore, it is augmented by knowledge-bases and routines denoting how the value of a node should change with respect to its neighboring node(s). In N3, this is written as

<node1> <relationship> <node2> .

The N3 requirements are relaxed somewhat in WIZER in that the nodes can contain multiple variables, instances, or concepts. For example, the type of the management of a firm can be either homogeneous or heterogeneous, and in a modified N3 this can be written as

<management> <has_type_of> <homogeneous, heterogeneous> .

4.5 Alert WIZER

The Alert module evaluates simulation output data with respect to corresponding empirical data. The evaluation can be in the form of alerts set when the simulated data is outside empirical data bounds. To generate alerts, the data must be described, possibly using statistical tools. After data description, comparison with empirical data yields symbolic information. The Alert module converts quantitative data into alert symbolic information. The notion of alert here includes normal non-emergency symbolic information. The Alert module acts as a symbolic or knowledge-based pattern classification and recognition system. Potential patterns can be enumerated in advance. Surprise patterns can be classified by their componential knowledge. Totally unexpected patterns can be marked as such and a special alert is issued for human examination of these patterns.

The Alert can also evaluate any simulation data, including input data, based on symbolic, rule-based, and/or ontological criteria. For example, the Alert can characterize the interaction network for management personnel as either homogeneous or heterogeneous based on the features of the interaction network.

The statistical routines that the Alert could use include:

1. Mean computation:

```
Input: real numbers N1, N2, …, Nx of a sample population, or in N3,
               <x numbers> <is part of> <a sample population> .
Computational logic: add(N1, N2, …, Nx) divided by count(N1, N2, …, Nx)
       Mean describes a sample population, or in N3:
               <mean> <describes> <a sample population> .
Output: result of the computational logic in the form of a real number N
```

2. Minimum and maximum computation

```
Input: real numbers N1, N2, …, Nx of a sample population, or in N3,
               <x numbers> <is part of> <a sample population> .
Computational logic: max(N1, N2, …, Nx)
       Max describes a sample population, or in N3:
               <max> <describes> <a sample population> .
Output: result of the computational logic in the form of a real number N.
(similar semantics for the minimum)
```

3. Variance and standard deviation computation

```
Input: real numbers N1, N2, …, Nx of a sample population, or in N3,
               <x numbers> <is part of> <a sample population> .
Computational logic: square root of ((sum of (N – mean of (N)) squared))
                      divided by (x – 1))
       Variance describes a sample population, or in N3:
               <variance> <describes> <a sample population> .
Output: result of the computational logic in the form of a real number N
(similar semantics for the standard deviation)
```

4. Curve classification/categorization. There are many curve classification types. A curve can be matched against a template. Alternatively, we can symbolically describe the curve trends. For example, for a monotonically increasing curve, the semantics is as follows.

```
Input: real numbers Y1, Y2, …, Yn of the Y-axis of a curve, or in N3,
               <n numbers> <is part of> <Y-axis of a curve> .
       real numbers X1, X2, …, Xn of the X-axis of a curve, or in N3,
               <n numbers> <is part of> <X-axis of a curve> .
Computational logic: for all (X2 >= X1), (Y2 >= Y1) must be true.
       "Monotonically increasing" describes a curve, or in N3:
               <monotonically increasing> <describes> <a curve> .
Output: result of the computational logic in the form of Boolean values of whether
       the curve is monotonically increasing.
```

5. Peak classification

```
Input: real numbers Y1, Y2, …, Yn of the Y-axis of a curve, or in N3,
               <n numbers> <is part of> <Y-axis of a curve> .
       real numbers X1, X2, …, Xn of the X-axis of a curve, or in N3,
               <n numbers> <is part of> <X-axis of a curve> .
Computational logic: find Ymax such that all X,
       (Ymax >= all Y and Ymax > most Y) must be true.
       (Ymax > average Y for Xmax-Delta < X < Xmax+Delta,
               where Delta is a number delineating the closest neighbors of Xmax
       "Peak" describes a curve, or in N3:
               <peak> <describes> <a curve> .
Output: result of the computational logic in the form of (X, Y) coordinate pairs
       denoting where the peak is. This assumes the outliers have been
       previously filtered out. The computation logic above checks the closest
       neighboring points to the peak to see if them are higher than average to
       guard against outliers.
```

6. Value range classification

```
Input: real numbers N1, N2, …, Nx sample population, or in N3,
               <x numbers> <is part of> <a sample population> .
```

two numbers denoting the range [A, B], where A is the lower range
Computational logic: all N >= A and all N <= B
Range describes a sample population, or in N3:
<range> <describes> <a sample population> .
Output: result of the computational logic in the form of a Boolean value denoting whether the numbers satisfy the range.

The Alert itself has many different types of alerts, including

1. Value-too-high and value-too-low alerts for bound checking

Input: real number N describing a sample population, or in N3,
<a number> <describes> <a sample population> .
two numbers denoting the bounds [A, B], where A is the lower bound
Computational logic: N >= A and N <= B, gives out the “Normal” alert
N < A gives out the “Value-too-low” alert
N > B gives out the “Value-too-high” alert
Value-too-high alert describes a number, or in N3, with respect to empirical bounds:
<value-too-high alert> <describes> <a number> .
<value-too-high alert> <measured against> <empirical bounds> .
(similarly for value-too-low and normal alerts)
Output: result of the computational logic in the form of alerts showing whether the number is above, below, or within the bounds.

2. Mean-different alerts for mean comparison

Input: real number N describing a sample population, or in N3,
<a number> <describes> <a sample population> .
the number M denoting the empirical mean
the tolerance E denoting the amount of difference that can be tolerated
Computational logic: M-E <= N and N <= M+E, gives out the “Same” alert
otherwise gives out “Different” alert
“Same” alert describes a number, or in N3:
<same alert> <describes> <a number> .
<same alert> <measured against> <empirical mean> .
(similarly for the “different” alert).
Output: result of the computational logic in the form of alerts showing whether the number matches the empirical mean or not.

3. Variance-different alerts for variance comparison

Input: real number N describing a sample population, or in N3,
<a number> <describes> <a sample population> .
the number V denoting the empirical variance
the tolerance E denoting the amount of difference that can be tolerated
Computational logic: V-E <= N and N <= V+E, gives out the “Same” alert
otherwise gives out “Different” alert

```
         "Same" alert describes a number, or in N3:
               <same alert> <describes> <a number> .
               <same alert> <measured against> <empirical variance> .
        (similarly for the "different" alert).
Output: result of the computational logic in the form of alerts showing whether
        the number matches the empirical variance or not.
```

4. Value range mismatch alerts for value range classification

```
Input: real numbers A, B describing a sample population, or in N3,
               <range numbers> <describes> <a sample population> .
        the numbers MIN, MAX denoting the empirical range
        the tolerance E denoting the amount of difference that can be tolerated
Computational logic: if MIN-E <= A and A <= MIN+E and MAX-E <= B <= MAX+E,
                      gives out the "Same" alert
                      otherwise gives out "Different" alert
         "Same" alert describes a range, or in N3:
               <same alert> <describes> <a range> .
               <same alert> <measured against> <empirical range> .
        (similarly for the "different" alert).
Output: result of the computational logic in the form of alerts showing whether
        the range numbers matched the empirical range or not.
```

5. Peak mismatch alerts for peak classification

```
Input: real number (X, Y) describing a curve peak, or in N3,
               <a coordinate> <describes> <a curve peak> .
        the coordinate (U, V) denoting the empirical peak
        the tolerances EX denoting the amount of difference in the X-axis, and
               EY denoting the difference in the Y-AXIS that can be tolerated
Computational logic: if U-EX <= X and X <= U+EX and
                      V-EY <= Y and Y <= V+EY, gives out the "Match" alert
                      otherwise gives out "Mismatch" alert
         "Match" alert describes a number, or in N3:
               <same alert> <describes> <a coordinate> .
               <same alert> <measured against> <empirical peak> .
        (similarly for the "mismatch" alert).
Output: result of the computational logic in the form of alerts showing whether
        the coordinate matches the empirical peak or not.
```

6. Numerous curve-type alerts for curve comparison. This is done by template matching. More sophisticated matching algorithms such as classifiers can be employed too. For example, a Support Vector Machine

(SVM) can be employed to learn how to classify a set of labeled data (Cristianini and Shawe-Taylor 2000).

7. Peak relative difference. For example, comparing the time difference between two peaks.

8. Two simulated means bracket an empirical mean.

9. Relative magnitude of curves.

10. Other specialized symbolic, rule-based, and/or ontological categorizations. For example, semantically describing an interaction network as either homogeneous or heterogeneous.

As shown above, the statistical routines and the alerts are encoded with an augmented description logic notation to allow their use in the Inference Engine reasoning. The augmented description logic adopts an approach similar to the Description Logic Program, which inter-operates rules and ontologies semantically and inferentially. The description logic is declarative, with the imperative routines tied to the declarations.

There are different types of simulation data. As WIZER is a knowledge-based tool, it is can flexibly handle the different types of simulation data. The data types include:

1. Single-simulation-run output data: in this case, WIZER just takes the output values, categorize them, and reason about the categories.
2. N-simulation-run (N>1) output data: in this case, WIZER computes the probability of the output values fall into a category. In other words, it counts the number of times the outputs values fall into a category divided by the total number of simulation runs. WIZER could also compute the statistics for the output data before putting it into categories. Doing so depends on the nature of the data, so care must be taken in which methods are applied. In general, curves should not be averaged but rates can be averaged. For example, curves of doctor

visits across N simulation runs in a 1 year simulation should not be averaged, but the rates of doctor visits across N simulation runs for 1 year intervals could be averaged.

3. Longitudinal data: an agent-based simulator could trace the history of an individual in the course of the simulation. In this case, WIZER could put the data in longitudinal categories and reason about them.

4.5.1 Alert WIZER as applied to Testbeds

Two testbeds are used to test validation automation capability of WIZER. The first is BioWar, a city-scale multi-agent social-network of weaponized disease spread in a demographically realistic population with naturally-occurring diseases. The second model is CONSTRUCT, a model for co-evolution of social and knowledge networks under diverse communication scenarios. Table 2 shows the features of the Alert WIZER as applied to the two testbeds. The features are applied to the validation scenarios of the two testbeds of BioWar and CONSTRUCT in Chapters 6 and 7 respectively.

Table 2. Alert WIZER as Applied to BioWar and CONSTRUCT

Features of Alert WIZER	BioWar	CONSTRUCT
Mean comparison	Yes	No
Curve peak determination	Yes	No
Relative timing of curve peaks	Yes	No
Threshold determination	No	Yes
Simulated means bracket the empirical mean	No	Yes
Curve magnitude comparison	No	Yes
Qualitative comparison of the curve magnitude comparisons	No	Yes
Interaction network categorization as homogeneous or heterogeneous	No	Yes

4.6 The Inference Engine

The WIZER Inference Engine is based on reasoning on facts, rules, and causations. Causations describe the simulation code. Rules are tied to the causal relations and to the simulation entities, so that the number of rules is constrained. This partially avoids the computational complexity of rule-based systems. The declarative causal and rule relations are in turn tied to the procedural simulation code. Thus the causal and rule relations can operate on the simulation code through inference. Ontological reasoning utilizing Simulation Description Logic augments the inference. Causations also describe empirical knowledge. The Inference Engine incorporates hypothesis building and testing as a way to explore knowledge space, in addition to rule-based/causation-based inferences. The simplest hypothesis building method is simply to search in the empirical knowledge's causal graphs.

Rule-based probabilistic argumentation systems (Haenni et al. 1999), causal analysis (Pearl 2003, Spirtes et al. 2000), and the Dempster-Shafer Theory of Evidence were early inspirations for the creation of inference mechanisms in WIZER. All have weaknesses which make them unsuitable for use in the validation of simulations. The probabilistic argumentation systems require a complete probabilistic space to function, which is hard to define for sociotechnical problems. Causal analysis makes the assumptions about conditional probability and conditional independence. It reduces the problem of causality to graph notation, when in the real world causality is much more complex. This dissertation suggests causality is best approached by simulating the mechanisms and processes closely.

The Inference Engine has components for rule and causal clause operation, simulation knowledge operation, domain knowledge operation, validation, and model-improvement. Implicit in this are the math and statistics routines employed to support all components. These support routines are semantically described and can be used by the Inference Engine for reasoning. The application of rules is weighted by their supporting evidences, within the context of existing knowledge and model. The building of hypotheses is based on the discrepancy between domain knowledge, empirical data, and simulation knowledge.

The causation clauses of the Inference Engine define the knowledge space of the simulator. The rule clauses of the Inference Engine encode what variables should be changed given the alerts. As noted above, the rules are tied to the causal relations. The firing of rules follows a forward-chaining method – a data-driven method. In the forward-chaining method, the system compares data against the conditions – the IF parts – of the rules and determines which rules to fire. Figure 6 shows the operations of the forward-chaining method. The forward-chaining method represents production systems. Production systems are Turing-equivalent, which means they are as powerful as a Turing machine. Thus production systems can do everything computable. WIZER augments the production systems with ontological reasoning, simulation descriptor, minimal model perturbation, and operations on the computational or process logic representing the procedural simulation code. The rules in WIZER have access to ontology, enabling them, for example to deduce rules such as “if a person is a child then he/she plays with another child”, as the ontology for children includes the attribute that they play with each other. Integration of ontology and rule-based systems lends to similar meta-rule capabilities of high performance expert systems.

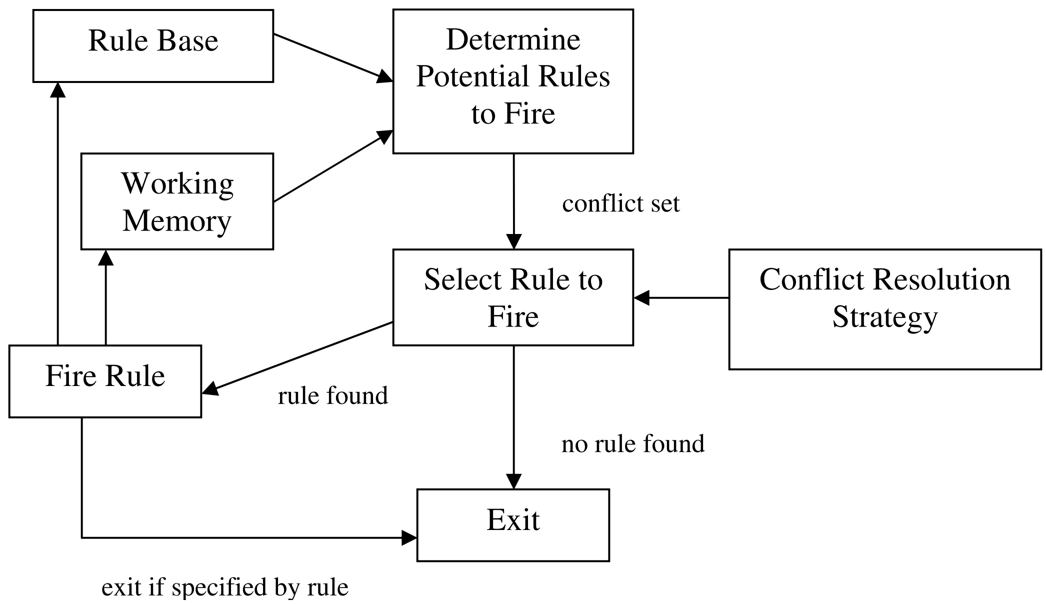

Figure 6. Forward-Chaining Method

As shown in Figure 6, using the rule base – the knowledge base containing the rules – and working memory – a place to store facts, the system determines possible rules to fire. These rules form a conflict set, where rules may conflict with each other. A conflict resolution strategy is then employed to select which rule to fire. After the firing of the rule, new inferred facts are added to the working memory. To prevent duplicate firing of the same rule, the triggering facts are removed from working memory. If there are no more rules to fire, the system stops. The rules in WIZER are not based on heuristics but on the model, as they are tied to the simulation model. The conflict resolution strategy in WIZER includes the task of selecting rules based on the result of forward-chaining inference and also the task of determining what value/rule to change and how much to change based on the minimal perturbations to the model to fit the simulation and inference results. The latter is a feature of model-based reasoning and is implemented using ontological and rule-based reasoning in WIZER. Thus in WIZER the rules have a supporting role of pinpointing areas to change, while the actual change to the value or rule, and the amount of this change, is determined by model perturbations using knowledge-based and ontological reasoning.

WIZER stores the history of conflict resolutions, model perturbations, and simulation trials. This history allows WIZER to avoid testing the same set of parameters twice. Avoiding the same simulation twice is a primitive form of simulation control.

The knowledge and simulation space operations are based on knowledge and ontological reasoning. The operations are similar to the forward-chaining method above but with a label added to the rules denoting the type of relationships/edges between entities.

Production systems have the following advantages:

- They are plausible psychologically.
- They are ideal for homogeneous knowledge representation in the form of rules.
- They can be highly modular as each rule is theoretically independent.
- They allow incremental rule growth if modularity is maintained.
- They allow well-defined and almost unrestricted communication between pieces of knowledge.
- They are a natural way to represent many kinds of knowledge.

Production systems are not without disadvantages however; WIZER ameliorates some of the disadvantages:

- Inefficient: production systems are inefficient due to the explosion in number of rules and inferences. WIZER however avoids the explosions of rules and assertions by tying them with the causal relations and thus the simulation model. Moreover, the match complexity of rule clauses is ameliorated by the organization of facts and rules using ontology and causal constraints. These constraints allow ontology-based modularization of rules and facts. This is similar to the RETE algorithm (Forgy 1982), but based on ontology.
- Opaqueness and unexpected interaction: It is very hard to figure out the effects of adding a rule and interdependencies. WIZER reduces this problem by using causal relations and ties to the simulation model. Furthermore, as the simulation system is tied to the rules, the resulting interaction of changed rules can be tested by running the simulation. This is asssited by using the ontology in the form of Simulation Description Logic.

- Difficult to design: WIZER simplifies design by tying the rule design to the causal relation design (which is easier) and to the simulation modeling (which should be conceptually clear). Thus it should be easier to use a sequence: first, designing the simulation model, then causal relations, and finally if-then rules.
- Difficult to debug: as rules are tied to causal relations, debugging the rules should be easier as they are modularized by their causal relations. The causal relations themselves should also be easier to debug as they are tied to the simulation model and to the mechanisms in the form of procedural simulation pseudocode.
- Is knowledge really rule-based? At the deepest level of human knowledge (wisdom, learning, and creativity), no. However, many forms of knowledge can be expressed as rules. Furthermore, however superficial the rules may be, they may be able to get the job done, as evidenced in the rule-based justice systems and some financial/business operations. This dissertation argues that knowledge is really process-based, and can thus be simulated closely. (Simulation is one of the best tools to mirror processes; another important tool is mathematics).

4.6.1 Variable, Rule, and Causation Definition

The variables in WIZER can have values which are Boolean, integer, real, curve (an array of real numbers), or symbolic. Additionally, they have the upper and lower limits of the value when applicable. Each variable also has fields for name and attributes such as belief, alert, and changeable. The variables correspond to the nodes in the graph of the simulation knowledge space or the empirical knowledge space. In essence, a variable has the following fields:

Variable = <name, value, alert, belief, changeable, priority, is_inferred_or_not>

The attributes of a variable have the following meanings:

- Belief: the probability of the variable value being correct.

- Alert: the symbolic value denoting qualitative “alerts” or classifications of data.
- Changeable: the degree to which a variable value can be changed.
- Priority: the experiment control value to prioritize probing of variables.
- Is_inferred_or_not: a Boolean value denoting whether the value of a variable is empirical or inferred.

The rules are defined as follows. Each statement of rules has the left-hand side (LHS) variables and the right-hand side (RHS) variables. It has an indicator for whether the variables are in conjunctive normal form (CNF) or disjunctive normal form (DNF). The default in WIZER is CNF. It has fields for name, label, and the attributes of belief, changeable, priority, and an indicator for whether or not this rule/causation is inferred. In essence,

Rule = <name, label, LHS, RHS, belief, changeable, priority, is_inferred_or_not>

The implication format of the rule is LHS → RHS, where the arrow sign denotes the implication. It also has the label “if-then” denoting that this rule belongs to the if-then rule type. Furthermore, each rule has ties to the ontology and to the causal relations and the simulation model. The attributes of the “if-then” rules are:

- Belief: the probability of the rule being correct.
- Label: the type of relations, in this case, “if-then”.
- Changeable: the degree to which a rule can be changed.
- Priority: the experiment control value to prioritize probing of rules.
- Is_inferred_or_not: a Boolean value denoting whether the rule is inferred. If the rule is not inferred, it is either empirical or based on a model.

The causal rules are defined similar to the definition of “if-then” rules. In essence, causal relations have the following fields:

Causation = <name, label, LHS, RHS, belief, changeable, priority, is_inferred_or_not>

The implication format of the causation is LHS → RHS, with the “causation” label denoting this implication as being the causation type. Other labels include “correlation”, “if-then”, and “convertible”. These correspond to the edges of the graph of simulation knowledge space and empirical knowledge space. The attributes of the causation rules are:

- Belief: the probability of the causal relation being correct.
- Label: the type of relations, in this case, "causation".
- Changeable: the degree to which a causal relation can be changed.
- Priority: the experiment control value to prioritize probing of causations.
- Is_inferred_or_not: a Boolean value denoting whether the rule is inferred. If the rule is not inferred, it is either empirical or based on a model.

The "belief" and "changeable" attributes are similar, but denote two different things. Even if a causal relation is 100% correct, it can still be changed. Inferred causal rules are weaker than either empirical or model-based causal ones, thus another attribute "is_inferred_or_not" is included to note the difference.

4.6.2 Conflict Resolution Strategy

WIZER utilizes the conflict resolution strategies as follows:

1. Semantics based conflict resolution when the information is available. This is based on the Simulation Description Logic and its inference.
2. Absent the ontological/semantics information, the conflict is resolved by numerical weighting of the <belief, changeable, priority, is_inferred_or_not> factors.
3. The combination of the above two, by reasoning about the numerical weighting factors.

Compounding variables (variables that must be simultaneously on, off, or have certain values) are resolved based on the perturbation of system description contained in the simulation knowledge space (and its ontology). This includes Boolean operations of AND, Inclusive-OR, Exclusive-OR, and NOT. For real values, AND corresponds to positive correlation, OR to choices, and NOT to negative causation or non-existence. Additionally, the value and link/model adjustment is considered during the perturbation of system description using knowledge-based and ontological reasoning. The policy question at hand can constraint both the conflict resolution strategy and the value and link/model adjustment. All these knowledge are encoded in knowledge bases and ontology.

As an example, suppose that the following causal model is defined for a simulation routine, written in N3:

<ailment-effective-radius> <causes> <infection-rate> .

<ailment-exchange-proximity-threshold> <causes> <infection-rate> .

<base-rate> <causes> <infection-rate> .

<vaccination-rate> <hinders> <infection-rate> .

<vaccination-rate> <hinders> <base-rate> .

<infection-rate> <is convertible with> <incidence-rate> .

where the verb "causes" means to influence positively, "hinders" means to influence negatively, "is convertible with" means the value can be transformed mathematically. The ailment-effective-radius variable denotes the radius within which people can get infected with the initial release of a disease agent. The ailment-exchange-proximity-threshold variable denotes the distance within which a person can infect another with a disease agent. For infectious and communicable diseases like influenza the two factors are closely correlated. The base-rate denotes the percentage of people who are susceptible to the disease agent.

The if-then rules related to the above causal model are (for the case of infection rate being above limit only for simplicity):

<infection-rate is above limit> <if-then> <op-lower aliment-effective-radius> .

<infection-rate is above limit> <if-then>

<op-lower ailment-exchange-proximity-threshold> .

<infection-rate is above limit> <if-then> <op-lower base-rate> .

<infection-rate is above limit> <if-then> <op-higher vaccination-rate> .

<incidence-rate is above limit> <if-then> <infection-rate is above limit> .

where the prefix "op" denotes the operation on parameter/node values. Suppose now that the Alert module gives out the fact that <incidence-rate is above limit> from a comparison between simulation outputs and the empirical data. The inference then gives a notice that <infection-rate is above limit> and all the four rules have their LHS to be true and are entered into the conflict set. Assume the following knowledge exists for the simulation:

<vaccination> <exists> <false> .

<vaccination> <causes> <vaccination-rate> .

<ailment-effective-radius> <positively correlates with>

<ailment-exchange-proximity-threshold> .

The Inference Engine deduces that

<vaccination-rate> <exists> <false> .

and thus the rule

<infection-rate is above limit> <if-then> <op-higher vaccination-rate> .

cannot fire (and not entered into the fire set, that is, the set of rules to fire). The other rules to fire include:

<infection-rate is above limit> <if-then> <op-lower aliment-effective-radius> .

<infection-rate is above limit> <if-then>

<op-lower ailment-exchange-proximity-threshold> .

<infection-rate is above limit> <if-then> <op-lower base-rate> .

The first two rules should fire together due to their correlation if it is so specified, while the last one can fire independently. But in this case, as there are no facts preventing them to fire, all will fire.

The amount of change to the value if the "op" prefixed clause is true is determined by existing knowledge if available, or else by a simple divide-and-conquer probe. The divide-and-conquer probe looks at the extreme values first than the midway values and the midway of the midway values and so on. It is similar to binary search. Suppose here we have the following knowledge:

<ailment-effective-radius> <has-values-of> <10 m, 50 m, 100 m> .

<ailment-exchange-proximity-threshold> <has-values-of> <10 m, 50 m, 100 m> .

<base-rate> <has-values-of> <0.10, 0.30, 0.50> .

Suppose again that the current values are 10 m for both ailment-effective-radius and ailment-exchange-proximity-radius and 10% for base-rate. The adjustments are decided by the Inference Engine to be: 50 m for both ailment-effective-radius and ailment-exchange-proximity-radius and 30% for base-rate. When more sophisticated knowledge for deciding whether and how to change the base-rate value is available (e.g., taking into account people who are in the time and place of the disease agent release and their immunity status), more detailed probes into how base-rate should change are made feasible. Using knowledge inference in this way, combinatorial explosion of parameter values can be somewhat tamed. This is similar to the approach taken in policy analysis in which all options are carefully thought over so that no brute force search is necessary.

4.6.3 Value and Link/Model Adjustment

The amount of change to a parameter value is determined by the alert type and the knowledge and ontological inference for that parameter. If a continuous and dynamic relationship is known, the adjustment amount may be derived from differential equations.

For a link or model adjustment, the model is perturbed minimally to get the change for the next simulation to fit the empirical data better. The perturbations proceed minimally from the least to the most perturbations: change in the parameter values, change in causation links, and change in the meta-models. How the model is perturbed is based on the knowledge-base and ontology about the model, including the assumptions about the model. The procedural code of the model is described by Simulation Description Logic and can be used to help determine the changes.

The conflict resolution and the value/rule adjustment are implemented in one module. This module has a supporting knowledge base for the purpose of model perturbations and conflict resolution.

The nature of the physical and social world can sometimes help in value adjustment. For example, in the physical world the road network restricts part of human mobility. Adjustments in the values of human mobility may only be needed along the transportation networks, for the first approximation. No explosive search in a two dimensional cellular space is needed. In the social world, the mobility of children is restricted during the school hours. Zoning laws affect the adult mobility patterns. These constraints of the physical and social world can be best captured by ontology and knowledge bases. As an example, a rough ontology for children can be written in N3 as

<children> <goes to> <school> .

<children> <rides> <a school bus> .

<children> <does not go to> <pharmacy> .

<children> <has> <curfew> .

<children> <has> <parents> .

<children> <lives with> <parents> .

<children> <plays with> <children> .

Of course, these relationships are not fixed. Specialized ontologies may be created for more detail descriptions of children with different backgrounds and age.

In addition to curves, surfaces, volumes, and higher dimensional manifolds must sometimes be probed. Aided by knowledge-bases and ontology, WIZER can probe the critical points, sample the area around the points, and use the results to guide further exploration in the manifolds. Absent knowledge, WIZER's performance degenerates to that of numerical techniques such as Monte-Carlo and divide-and-conquer search.

4.7 Domain Knowledge Operations

Domain knowledge is represented as a graph of knowledge structures along with its node value ranges. It encodes both the conceptual layer (the T-Box) and the instantiation layer (the A-Box) when data is available. The node has attributes denoting properties such as the attribute *has-type-of*, which defines the possible types of the node or the adjective of the node "noun". There are different kinds of edges in the graph or network:

- Causation: encoding what causes what.
- If-Then: encoding what will infer what. Note that inference is not the same as causation.
- Convertible: encoding what can be transformed mathematically to what.
- Correlation: encoding what correlates with what.

- Instance-of: encoding that this node is an instantiation of a conceptually higher node.
- Concept-of: encoding this node is the conceptual "parent" of another node.
- Other relationships or *"verbs"*.

The operations on domain knowledge are based on the Simulation Description Logic. The knowledge can be described in the modified N3 notation.

The operations are implemented in a graph search algorithm with edge labels denoting the graph/network attributes, which is to say, the values of the edges of the graph. The forward chaining inference rules correspond to the graph searches. The rules have access and control to the graph searches. They become different kinds of rules depending on the attribute label. If the label has the attribute of "causation", it becomes the causation-type production rule. If it has the attribute of "correlation", it becomes the correlation-type production rule. The rules mentioned here are to be understood in the context of production systems.

The queries that can be answered by the operations include "what properties a node has", "find what class a node belongs to", "find all nodes that are part of this node", "find all nodes that correlate with this node", "find all nodes that influence this node", etc. The answers are found by searching the graph and controlled by rules.

As an example, suppose that the management node has the type property value of either homogeneous or heterogeneous. The search performed to answer the query "find all types of the node *management*" includes the localization of the node and probing of the values of the node's type attribute. This corresponds to the conceptual definition (the T-Box) of management. If a specific data is available, the instantiation definition (the A-Box) of a specific management is feasible. This is an example of ontological reasoning in WIZER. The result of ontological reasoning is new values for Inference Engine rules. The new values can be used to perform what-if analyses as they are new hypotheses. Furthermore, a rule such as "if the management is homogeneous, then probe other types of management" can trigger ontological reasoning. Rules have access to the ontology for reasoning.

4.8 Simulation Knowledge Operations

Simulation knowledge is also structured as a graph similar to the domain knowledge graph. It contains simulated or inferred output values and clauses. The nodes of the graph denote variables. The edges of the graph denote relationships between variables. The nodes have slots denoting properties. The node is analogous with the noun in the English language, the edge analogous with the verb, the node attribute with the adjective. The description of an edge, the adverb, while feasible is not implemented. The relationships implemented are causal, if-then, and convertible relationships.

The operations on simulation knowledge are based on the Simulation Description Logic. They use a graph search algorithm like the one used in the domain knowledge operation. The graph search corresponds to a forward chaining inference. Simulation knowledge operations are tied to the simulation model, whereas the domain knowledge does not necessarily have direct ties to the simulation model.

The reason why there is a separation between simulation and domain knowledge is that the simulation knowledge is owned by the simulation developers whereas the domain knowledge belongs to the validators or VV&A practitioners which they use as a standard to judge simulators against.

4.9 Validation Submodule

The validation submodule computes the degrees of validity of a simulator. As validation depends on domain knowledge, validation is measured against a definite piece of knowledge. For example, a simulator may output valid behaviors in terms of school absenteeism, while not necessarily in terms of other pieces of knowledge. Thus every validation is measured based on a specified piece of knowledge underlying a data stream of simulation outputs and occurrences.

On the conceptual model level, it takes the domain knowledge graph and simulated knowledge graph as input to calculate the intersection between them. The extent of the intersection determines the degree of validity of the simulator model. This assumes domain knowledge and empirical data are correct.

Thus, there are two measures of validation: one for data/outputs/behaviors validity and the other for conceptual validity.

4.10 Model-Improvement Submodule

This submodule takes the intersection of domain and simulated knowledge graphs and calculates the changes needed in the simulated knowledge graph (and thus the simulation parameters and meta-models) needed to better align the two. Better alignment gives better validity. A simple method for alignment is to search in the domain knowledge space to find links that do not exist in the simulation knowledge space. Guided by ontology and semantics, these new links or modified ones are then added to the simulation knowledge space. The simulation links can also be deleted when warranted. The above alignment method is similar to morphing. This module performs hypothesis building for a more advanced morphing. After the addition of the links, simulations are run to test the links.

4.11 Ontological Reasoning in WIZER

Triggered by rules or scenario plans, WIZER performs ontological reasoning. The ontological reasoning is implemented in the form of a graph search with its corresponding semantic description. The graph has nodes, node attributes, and edges. The simulation knowledge space and the domain knowledge space have this graph representation. Rules in the Inference Engine have access to the graph and thus the ontology for reasoning. Graph searches correspond to rule inferences.

WIZER enables the Inference Engine to have rules such as “find all nodes that correlate with this node” and then “for all correlated nodes, run test *T*”. Thus the rule inference and ontological reasoning support each other.

4.12 Structural Changes and WIZER

The currently implemented WIZER deals primarily with parameter value changes, not the (simulation) structural changes. While there is a simple structural change handled by WIZER for the CONSTRUCT Validation III in Chapter 7 (the heterogeneous workers heterogeneous management case), this dissertation does not show more comprehensive results of structural changes enabled by WIZER. The simple structural change of the CONSTRUCT Validation III is in the form of ontological reasoning: changing the type of management from homogeneous to heterogeneous.

In the conflict resolution and the value/link adjustment process of WIZER, minimal model perturbations are performed by changing causal relations. How causal relations are changed depends on domain knowledge, simulation knowledge, empirical data, simulation results, and ontology.

From knowledge-based perspective, parameter values and links/edges are both a piece of knowledge. The distinction is that a parameter value directly influences one node, while a link/edge directly influences two nodes. But at the knowledge level, this distinction does not really matter. All that matter is that determining the effects of the change in a piece of knowledge on the simulation when such change is made. In the context of validation, if the change results in better validity, then that piece of knowledge (regardless of whether it is a parameter value or a link/edge) is good and the change is good. In the context of model improvement, if the change in the piece of knowledge results in a better model, that that change is good, regardless of whether it is changing simple parameter values, links/edges, or even more complex meta-models. The pieces of knowledge and their relationships are encoded in the causal relations, "if-then" rules, process model/logic, and other types of inference in the knowledge-based systems. Parameter values and links/edges/rules are both first-class objects (roughly speaking, they are equals). Based on the above knowledge level view, if WIZER can handle parameter value change well, it must be able to handle link/edge change well too. This is not a conceptual problem nor a software design problem, but simply a matter of programming.

4.13 An Example of Simple WIZER Runs

This example is based on four runs of BioWar simulator for 100%-scale Hampton city (population 146,437 persons) with no biological attacks (i.e., intentional disease releases).

FIRST ITERATION

- Alert outputs:
  - ER registration is above the empirical bound of 0.232 visits per person per year
    edregistration-yearly-avg,2.24856,0.056,0.232,above the bounds
  - Doctor visit is above the empirical bound of 1.611 visits per person per year
    insuranceclaim-yearly-avg,3.16572,0.415,1.611,above the bounds
  - School absenteeism is within the empirical bound of absence rate
    school-yearly-avg,3.62525,3.04,5.18,within bounds
- Inference Engine outputs:
  - Increase the behavior threshold for going to ER (by a constant amount)
  - Increase the behavior threshold for going to doctor office

- Leave alone the behavior threshold related to school going behavior
- 3 data streams with 1 data stream within bounds → 33% validity

SECOND INTERATION

- ER registration is within the empirical bounds
  edregistration-yearly-avg,0.0818865,0.056,0.232,within bounds.
  - 3 data streams with 2 data streams within bounds → 67% validity

The example covers WIZER submodules as follows.

The example corresponds to the situations when agents get sick and manifest symptoms of illness. The severity of symptoms governs where agents will go in the next time step. The thresholds of severity that trigger certain movement behaviors are named accordingly. These thresholds affect the visit rates of places. In N3,

<going to work threshold> <influences positively> <work presence rate> .
<going to work threshold> <influences negatively> <pharmacy visit rate> .
<pharmacy visit threshold> <influences positively> <pharmacy visit rate> .
<pharmacy visit threshold> <influences negatively> <doctor visit rate> .
<doctor visit threshold> <influences positively> <doctor visit rate> .
<doctor visit threshold> <influences negatively> <emergency room visit rate> .

The verbs denote the relationships between the behavioral thresholds of going to places with the places' visit rates. An agent's decision of going to work, pharmacy, or doctor is based on the thresholds and on the agent's health status, that is, whether an agent is sick, what disease(s) an agent has, and how severe the symptoms an agent has.

The domain knowledge space has the same format as the simulation knowledge space. In the simple example above, the domain knowledge space is the same as the simulation knowledge space. In general, however, the domain knowledge space is larger than the simulation knowledge space.

In the example, the alerts are in the form of value-too-high and value-too-low for bound checks of the simulation output value against empirical minimum and maximum values. WIZER compares the annual mean value of numerous data streams against the empirical data and gives out the alerts. As noted, 4 simulation runs of Hampton city with 100% scale are done. In this case, the output data is averaged over the 4 simulation runs, before the Alert does its symbolic categorization.

- ER registration is above the empirical bound of 0.232 visits per person per year edregistration-yearly-avg, 2.24856 (simulation yearly average), 0.056 (empirical min), 0.232 (empirical max), above the bounds

- Doctor visit is above the empirical bound of 1.611 visits per person per year

  insuranceclaim-yearly-avg, 3.16572,0.415,1.611,above the bounds

- School absenteeism is within the empirical bound of absence rate

  school-yearly-avg, 3.62525,3.04,5.18,within bounds

Of the four simulations in the trial, all give consistent results of ER registration being above the empirical bound, doctor visit being above the empirical bound, and school absenteeism being within the empirical bounds.

An example of the operation of the Inference Engine is as follows.

- ER visit rate is too high so increase the behavior threshold for going to ER (by a constant amount)
- Doctor visit rate is too high so increase the behavior threshold for going to doctor office
- School visit rate is within bounds so leave alone the behavior threshold related to school going behavior
- 3 data streams with 1 data stream within bounds: 33% validity

As the domain knowledge space and the domain knowledge space are assumed to be the same, it is the degree of match between simulation outputs and empirical data that is counted toward the (total) validation level. For the first iteration of the simulation, we have

- 3 data streams with 1 data stream within bounds → 33% validity

For the second iteration, we have

- 3 data streams with 2 data streams within bounds → 67% validity

4.14 Comparisons of WIZER to Other Tools

Few multi-agent simulations have exploited the depth and breadth of available knowledge and information for validation that resides in journals, textbooks, websites, human experts, and other sources. Typically, simulation results are designed solely for human analysis and validation is provided by subject matter experts employing the labor-intensive and tedious VV&A process.

WIZER is unique in that it utilizes ontological and knowledge-based inference for validation and model-improvement. It strives to use as much deep and profound knowledge as possible by making use of works in description logics and ontological reasoning. WIZER seeks to emulate scientists doing experiments and analyses via the scientific method, instead of simply providing a programming environment.

While other toolkits such as Swarm (http://wiki.swarm.org), TAEMS (O'Hare and Jennings 1995, Lesser et al. 2004), and Repast (http://repast.sourceforge.net) are designed with the goal of assisting the design and implementation of agent-based systems, WIZER is designed to help with scientific experimentation, validation, analysis, and model improvement. WIZER is conceptually able to run on top of any simulation system, including those constructed using Swarm and Repast toolkits provided that corresponding knowledge bases are provided. WIZER is basically a causal and logical reasoning, experimentation, and simulation control engine with statistical and pattern recognition capabilities. This is similar to techniques scientists employ when forming hypotheses and designing, executing, and analyzing experiments for hypothesis testing.

WIZER differs from evolutionary programming (Fogel 1999), evolutionary strategies, and genetic algorithms. WIZER does not need a population of mutation/crossover candidates nor does it need the mutation, crossover, and other evolutionary and genetic constructs. Instead, WIZER applies knowledge inference to simulations to design the next simulation run, based on scientific experimental method. If the result of inferences mandates a radical change, a revolution will occur.

The following table shows the comparison between WIZER and other tools.

Table 3. Features Comparison

	WIZER	**Swarm/TAEMS/Repast**	**Evolutionary Strategies**	**Data Farming**
Programming environment?	No	Yes	No	No
Unit of inference	Rule and causation	None	Evolutionary and genetic operators	Data growing heuristics
Object of operation	Simulation, data, knowledge	Code	Simulation and data	Data
Experimentation?	Yes, automated	Yes, human operated	Yes, automated (fitness)	No
Automated simulation control?	Yes	No	Yes	No
Knowledge operation?	Yes	No	No	No

4.15 Conclusion

WIZER is a knowledge-based and ontological tool for validation and model-improvement of simulation systems. It is capable of emulating the basic inferences that human experimenters perform to validate and improve simulation models. It can reduce the number of search needed to validate simulation models as it makes use of knowledge space search in addition to parameter space search. WIZER is powered by knowledge inference, so it is as powerful as the knowledge and the inference mechanisms contained in it. WIZER is unique as it focuses on the analysis, inference, and control of simulations instead of providing a verification and programming environment.

WIZER is limited by the knowledge inside its system and its reasoning mechanisms. If the majority of the knowledge is wrong, WIZER will output wrong inferences and wrong validations. An anchor to the empirical data may mitigate this, but how to change existing knowledge based on data remains a research question. Hypothesis building and testing using simulation proxies may be one answer, as this dissertation indicates. The reasoning mechanisms in WIZER currently consist of causal and IF-THEN forward-chaining mechanisms and the ontological/semantic reasoning. WIZER does not incorporate a learning mechanism, except for the simple hypothesis building using search in the ontologies and knowledge bases and for virtual experiments performed to test the hypothesis which may result in the acquisition of new facts and relations.

CHAPTER V: Evaluation Criteria

As a knowledge-based tool for validation and model-improvement, evaluation in WIZER derives in part from employing a knowledge-based system in the evaluation. Any knowledge-based system depends to a large extent on its knowledge bases, in addition to its inference mechanisms.

In this chapter, I present simple evaluation criteria for validation: value comparison, curve and pattern comparison, statistical comparison, and conceptual comparison. This is similar to statistically comparing two sample distributions (Box, Hunter, and Hunter 2005). Additionally, performance (defined as how quickly and effectively validation is completed) can be measured by comparing the search space in parameter, knowledge, and meta-model spaces before and after knowledge and simulation inference.

For validation, the outputs and occurrences of the simulation are converted into symbolic knowledge by using mathematical routines such as a bound checking, which examines how much the simulation outputs fit the empirical bounds. This is one evaluation criterion for validation. Another, more profound, criterion is whether the behaviors of the simulation model itself fit the empirical knowledge. This is measured by comparing model knowledge bases and links with the empirical ones.

For performance evaluation, a set of measures gauges the effects of knowledge, simulation, and inference on the amount and the focus of search in the parameter, meta-model, and knowledge spaces. This includes the comparison between knowledge-less and knowledge-based parameter search space.

For model-improvement evaluation, I describe a simple semantics-based comparison of validity before and after an attempt to improve the simulation model by the model-improvement module.

As WIZER is a knowledge-based system, the ontology (Gomez-Perez et al. 2004) and the need to include the knowledge in the inference engine facilitate a clear and succinct representation of subject matter expert's and policy analyst's knowledge and judgment. Due to its emphasis on precision and transparency, the process and the evaluation of validation and model-improvement are facilitated.

5.1 Statistical Sample Comparison

Statistics can compare samples parametrically and non-parametrically. To use parametric methods, samples must have a normal distribution and be independent. The sample size must be large enough, usually more than 30. Absent an assumption of normal distribution, non-parametric methods must be used.

Parametric methods have the advantage of being easy to use and understand. They make it easy to quantitatively describe the population or the actual difference between populations. The methods employ established statistical distributions (e.g., normal, Poisson, and Gamma distributions). The disadvantage of parametric methods is that they require the assumption of the underlying statistical distribution for the sample. A skewed distribution cannot be assumed away.

The advantages of non-parametric methods include:

1. They provide an aura of objectivity when there is no reliable underlying, universally recognized, scale for the original data and there is some concern that the results of standard parametric techniques would be criticized for their dependence on an artificial metric.
2. Non-parametric tests make less stringent demands of the data. It is not required, for example, that normality or equal standard deviation applies.
3. Non-parametric test can be used to get a quick answer with little calculation.
4. They can be employed when the data do not constitute a random sample from a larger population and standard parametric techniques based on sampling from larger populations are no longer appropriate.

The disadvantages of non-parametric methods include:

1. They still require random samples.

2. As they contain no parameters, it is difficult to make quantitative statements about the actual difference between populations.
3. They throw away information, for example, the sign test only uses the signs of the observations.

Parametric tests for comparing samples include *t* test to compare two independent samples. The equivalent non-parametric one is the Wilcoxon rank-sum test.

A simulation usually uses hypothesized families of distributions for its stochastic variables, estimating the statistical parameters, and determining how representative the fitted distributions are. The degree of fitness of a distribution against the data is determined by heuristic procedures and goodness-of-fit tests (Law and Kelton 2000). Heuristic procedures include density/histogram overplots and frequency comparisons, distribution function differences plots, and probability plots. Goodness-of-fit tests include chi-square tests, Kolmogorov-Smirnov tests, Anderson-Darling tests, and Poisson-process tests.

5.2 Validation Evaluation Criteria

The validation evaluation is based on a set of statistical tests of simulation events/outputs against empirical data. It consists of value comparison, curve comparison, pattern comparison, statistical comparison, and conceptual comparison.

5.2.1 Value Comparison

Value comparison simply compares the value of a simulation output stream against an empirical value (or a pair of empirical values in the form of the minimum and maximum bounds). One simulation trial suffices for some cases. However, given multiple simulation trials, the mean and standard deviation of the simulation output streams are calculated and compared against the empirical values of mean and, if available, standard deviation.

In data stream comparison, the semantics of the data cannot be neglected. For example, annual school absenteeism has the semantics of counting absenteeism when school is in session. This means summer vacation, holidays, Saturdays, and Sundays are not counted and have no meaning of absenteeism.

If the simulation and empirical values compare 100% with each other, then the validity is 100% for the data stream. The semantics is noted for this data stream, using N3 notation:

<simulated data stream S> <is 100% validated with> <empirical data E> .

When the values compare with *n* percent probability, it is noted as

<simulated data stream S> <is *n* percent probability validated with>
<empirical data E> .

When the mean and standard deviation of simulation output data are available – assuming a normal distribution, and that the empirical value as the value *V* is available to compare against, a parametric confidence interval can be computed and the probability can be assessed. In N3, the semantics is noted as

<simulated data streams S>
<is validated with *n* percent probability using 95% confidence interval>
<empirical value V> .

If the simulation output data has to be assumed to be non-parametric, then the non-parametric confidence interval is computed for the median. A non-parametric significant test can also be computed.

5.2.2 Curve Comparison

While value comparison is simple and useful, in some cases curves need to be compared. There are two ways to compare curves: semantics-based and mathematical. The mathematical approach assumes differentiable curves. It employs the methods of curvature matching, tangent angles, and template matching. As the curves for our purpose have meaningful parameterization (e.g., have a time axis), which is to say, they are not purely geometric, the semantics of the curves helps in the comparison. As a result, a mix of mathematical and semantic-based methods can be used.

The curves are compared by the following methods:

1. Magnitude comparison: whether a curve value is higher than a reference value or than that of the other curve within a certain interval.
2. Trend/gradient comparison: whether and how fast a curve increases or decreases.
3. Peak comparison: whether a curve peak is similar to that of another curve
4. Curve shape comparison. The tangents, curvatures, and semantics are matched and compared. (This is harder that the above.)

The result of curve comparison are validation values that say how valid a curve is compared with a reference value or curve. For example, in N3 the influenza peak comparison can be noted as:

<simulated influenza peak> <is similar to, with 95% validity> <empirical influenza peak>

5.2.3 Pattern Comparison

Patterns are more complex than curves. While curves are more or less continuous, patterns can change abruptly and are sometimes intermittent. As patterns for our purpose have meaningful parameterization, the semantics of the patterns is used to help with comparison.

The result of the comparison are validation values which in N3 can be noted as

<simulated pattern I> <is similar, with 95% validity> <empirical pattern *B*>

5.2.4 Statistical Comparison

When the empirical data samples are available, both the simulated data and the empirical data can be characterized statistically. If a normal distribution can be assumed, *t*-test is used to compare the two sets of data. (The data is assumed to be independent samples.) If no normal distribution can be assumed, the Wilcoxon rank-sum test is used.

The result of the statistical comparison are the validation values, which in N3 can be written as the followings:

<simulated data stream> <has the same population, with 100% validity>
<empirical data stream> .

<simulated data stream> <uses> <Wilcoxon rank-sum test> .

5.2.5 Conceptual Comparison

When a simulation has longitudinal data such as an agent's history, the longitudinal data can be compared conceptually or semantically with empirical data. For example, a typical pattern of a K-12 child includes taking the school bus and going to school from Monday to Friday during the school year. The simulation output for the average child behavior history should conceptually match the pattern. This can be thought as symbolic patterns, as opposed to numeric patterns. The patterns are compared conceptually or semantically by comparing the semantics description of the empirical data and the simulation output. In a modified N3, for example, the child activity pattern can be written as:

<a chronological pattern> <consists of> <four sequences> .

<a K-12 child> <takes> <a school bus> .

<a K-12 child> <goes to> <school> .

<a K-12 child> <takes> <a school bus> .

<a K-12 child> <goes to> <home> .

The comparison can be record-by-record, but smarter comparison utilizing ontological reasoning can be employed.

The result of the comparison is the validation values which give a notice of whether the symbolic patterns match. In N3,

<simulated child behavior history for Mondays> <matches>

<empirical child behavior history for Mondays> .

5.3 Performance Evaluation Criteria

The speed and effectiveness of a validation are measured against the search space and the knowledge space – including ontology.

5.3.1 Reduction of the Amount of Searching

The reduction of the amount of searching is simply measured by:

1. The size of the search space before the application of WIZER inference,
2. The size of the search space that need to actually be searched when WIZER is applied.

The division of (2) by (1) indicates the proportion of search reduction by WIZER.

5.3.2 Showing How Knowledge Can Help Focus the Search

How much knowledge can help focus the search is measured by

1. Knowledge bases and inferences,
2. The extent of search space before the application of knowledge inference,
3. The extent of search space after knowledge inference of WIZER.

The comparison of (3) with (2) in light of (1) produces the focus “quality” for a particular knowledge and inference by WIZER.

5.4 Model-Improvement Evaluation Criteria

Whether the simulation model is improved after the addition or deletion of simulation links by the model-improvement module is determined by comparing the validity of the simulation before and after the addition of said links. This comparison is guided by ontology or semantics. In other words, the many validation values for various data streams (which are described semantically) are examined for their relative importance by the ontology or semantics. Furthermore, while a single "total" validity value can be calculated, utilizing ontology and semantics to weigh and assess the true significance of various validation values is a more sensible method. When the model consists of causal relations, the comparison – aided by ontology – of models before and after adjustments indicates comparative causality.

5.5 Summary

This chapter describes knowledge-based evaluation criteria for validation, performance, and model-improvement. It describes how curves and other simulation outputs are compared. Knowledge and ontological inference allows WIZER to prune and focus the search space for validation and model-improvement.

Chapter VI: BioWar TestBed

WIZER is applied to partially validate a multi-agent social-network model called BioWar. BioWar (Carley et al. 2003) is a model capable of simulating the effects of weaponized biological attacks on a demographically-realistic population with a background of naturally-occurring diseases. It integrates principles of epidemiology (Anderson and May 1991, Lawson 2001, Bhopal 2002), health science (Cromley and McLafferty 2002), geography, demographics, sociology, social networks (Wasserman and Faust 1994), behavioral science, and organization science.

In this chapter, I show how WIZER is used to partially validate BioWar in two validation scenarios. The validity of the results is described and discussed.

6.1 Description of BioWar

BioWar simulates what persons do in daily lives before and after they are infected with diseases in a city. The following figure partially shows the causal relationships among the simulation entities in BioWar. Note that the arrow direction means “may cause or influence”. These relationships are put into the knowledge base for the WIZER Inference Engine.

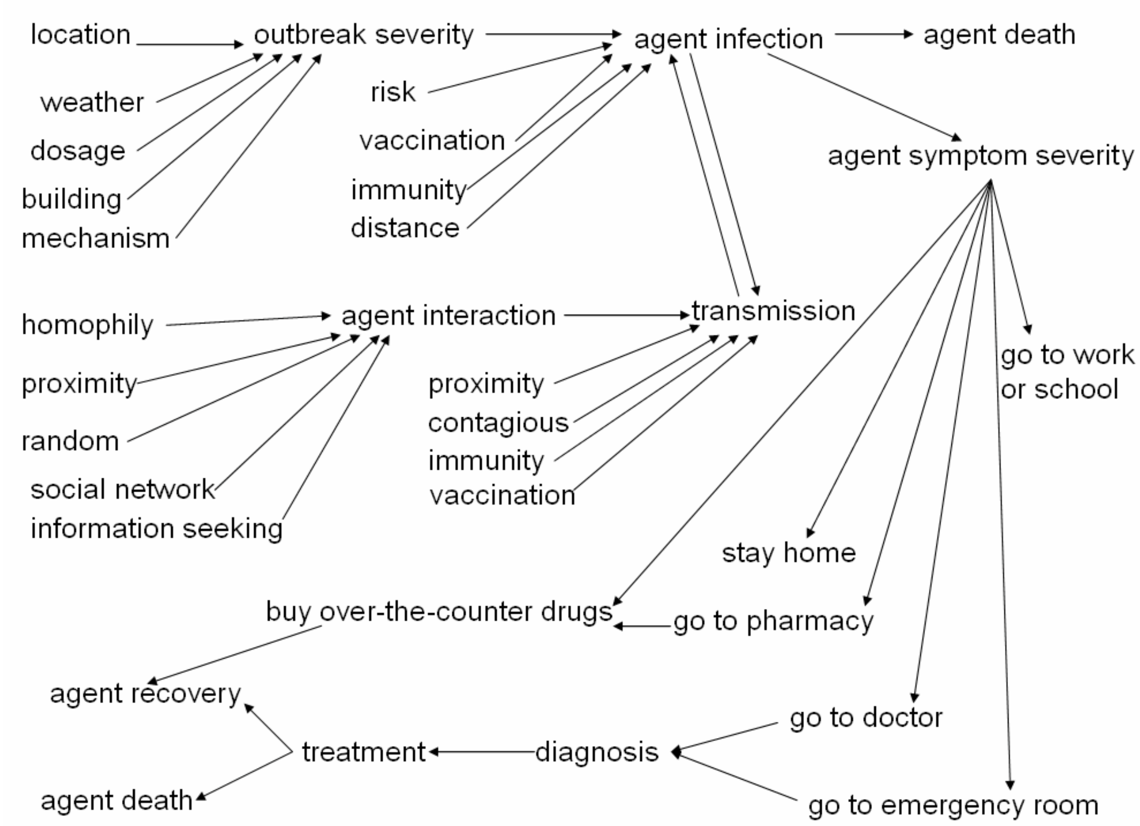

Figure 7. Partial Causal Diagram of BioWar

To facilitate descriptions without the need to draw graphs and for automation, however, we could use a simple syntax in the form of:

(setvalue predicate value): set the value of a variable/predicate to “value”.

(setstdev predicate value): set the standard deviation in the value of the predicate set previously by the setvalue operator to “value’.

(setbelief predicate value): set the probability of the value of the predicate being correct to “value”.

(setpriority predicate value): set the priority of the variable/predicate. This priority determines the order by which rules and/or entities are examined.

(setchangeable predicate value): set the degree by which a rule and/or an entity can change in value.

(causes predicate predicate): set the causation relationship relating two variables/entities.

(convertible predicate predicate): defining that the values of two variables/entities can be transformed mathematically to each other.

(if-then (predicate value) (predicate value)):

set the if-then relationship between two variables/nodes

where *predicate* is a node/variable/entity, and *value* is any Boolean, integer, real, qualitative, or enumerated value where applicable. The special/predetermined predicates include "causes", "if-then", and "convertible", where *causes* denotes causal relations, *if-then* denotes if-then relations, and *convertible* denotes that the values can be mathematically converted to each other. A prefix of "op-" in the predicates means the predicates modify values in the working memory of the forward-chaining mechanism.

The figure below shows one process model related to the causal model of BioWar shown above in Figure 7. This process model elucidates the causal relation of agent's infection and agent's symptom severity in the causal diagram. It is applied to one individual, instead of a population sample. It is general enough to capture the processes of most infectious diseases.

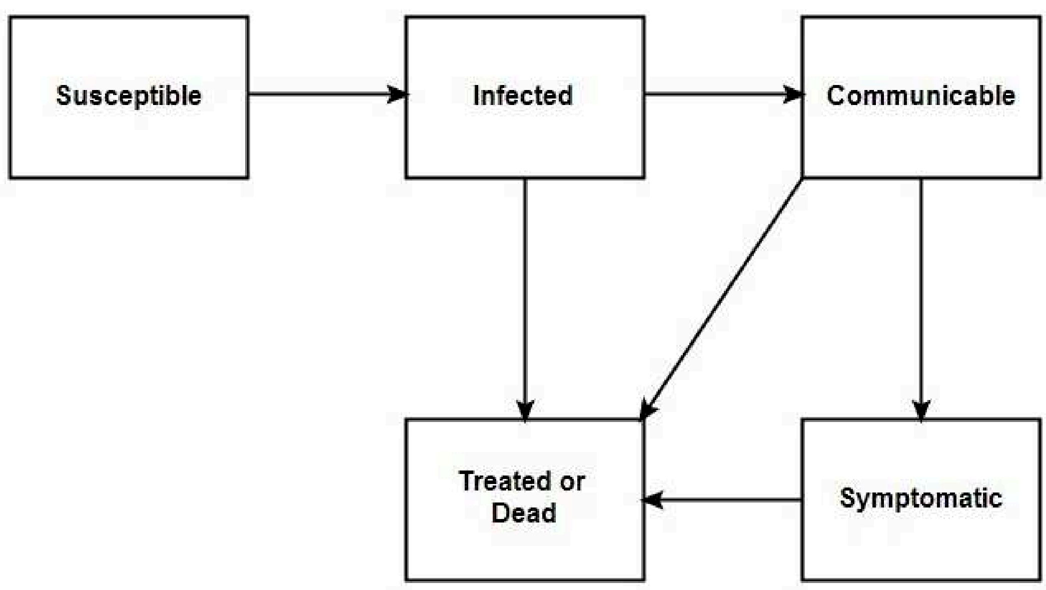

Figure 8. A Process Model for Infectious Diseases

As shown, the process starts in the state/phase of susceptible, then transitions to infected state, communicable/contagious (a period when a person can infect others) state, and symptomatic state. These phases exit to the state of either treated or dead. Note that while the rectangles seem to suggest the states are distinct, here they do not mean so. (Thus the process diagram is not the same as the finite state machine, as in finite state machine states do not overlap.) The infected state or phase brackets both the communicable and symptomatic phases. The communicable and symptomatic phases can overlap. As an example, for influenza, an adult person can be in the communicable phase 1 day before being in the symptomatic phase and continue to be in the communicable phase until 5 days after the initial infection. Treatment for influenza only minimizes the symptoms, and does not cure the disease. For smallpox, the incubation period last 7 to 17 days during which a person is not communicable/contagious, followed by the symptomatic and communicable phases at the same time. This symptomatic phase is further divided into initial symptoms (prodome) which lasts 2-4 days, early rash for about 4 days, pustular rash for about 5 days, pustules and scabs for about 5 days, resolving scabs for about 6 days, and then finally resolved scabs. (The subphases are modeled in a manner similar to the process model for phases.) The early rash is the most contagious subphase, while other subphases are contagious except for the resolved scabs phase. All this symbolic and semantic information is critically important to the process model. Augmenting the process model with symbolic and semantic information produces the process logic. Process logic is defined as the sequenced or ordered events based on the process model augmented by semantic information and ontology.

6.2 The Need for Automated Validation

BioWar has a large number of model variables. The interactions between variables influence the outcome.

The Spiral Development model (Boehm 2000) of BioWar means that the previous validation of model predictions may no longer hold. Furthermore, the assumptions behind large scale multi-agent models are complex, often not stated, and not operable (not suitable for computer processing and automation). Changing parameter values without understanding the assumptions and reasoning behind them can lead to unforeseen results.

To address the above problems, automated validation with the assumption tracking is needed. So here WIZER plays a vital role, by explicitly stating the assumptions and reasoning behind changes in parameter and model values in relation to the validation process.

6.3 WIZER as Applied to BioWar

WIZER takes the BioWar outputs and occurrences as input (e.g., school absenteeism, work absenteeism, doctor visits, emergency room visits, pharmacy visits, drug purchases, and agent health statistics) along with the context of the simulation (e.g., city layout, demographics distribution, school district boundaries, calendar of events, and agent occupations) and the corresponding empirical data. After comparing the simulation outputs with the empirical data, WIZER performs inferences to decide what parameters need to change so that the next simulation would be the most reasonable search step toward increased validity. As more diverse types of empirical data are compared, the chance of different parameter values fitting all empirical data gets smaller.

6.4 Data Sources for Validation

Getting access to data for validating BioWar is non-trivial. In this dissertation, limited data are used. The following are the data streams that can be used by WIZER. Note that both the input and output data sources for BioWar are listed, because both can be used in WIZER. Note that not all validation scenarios require all the data sources.

- Hospital and park locations from GNIS database, http://geonames.usgs.gov
- Demographics from Census Bureau's Summary File 1, http://factfinder.census.gov/home/en/sf1.html
- Work, medical, recreation location counts from Census Bureau's Economic Census, http://www.census.gov/econ/www/econ_cen.html
- Cartographic boundaries from Census Bureau, http://www.census.gov/geo/www/cob
- School demographics and locations from NCES' CCD database, http://nces.ed.gov/ccd
- Student absenteeism statistics, http://nces.ed.gov/pubsearch
- Social network characteristics from GSS, http://www.icpsr.umich.edu:8080/GSS/homepage.htm
- Climate and wind data from NCDC at NOAA, http://www.ncdc.noaa.gov/oa/ncdc.html
- Disease symptoms and diagnosis model from Internist 1 (Miller et al. 1982)

- Medical visit, mortality and morbidity statistics from CDC's NCHS surveys, http://www.cdc.gov/nchs
- Disease timing and symptoms from CDC, http://www.cdc.gov/publications.htm
- CDC weekly report for influenza
- Influenza data from http://www.wrongdiagnosis.com/f/flu/stats.htm and NIAID, the National Institute of Allergy and Infectious Disease
- SDI (Surveillance Data Inc) emergency room registration data
- DARPA SDI hospital and clinics visit data for five cities
- DARPA ADS hospital and clinics visit data for five cities
- DARPA PDTS hospital and clinics visit data for five cities

6.5 Validation Scenarios

I examine two validation scenarios. Validation Scenario I examines the influenza effects of incidence (and thus prevalence and death rate) in relation to several input parameters such as ailment exchange proximity threshold. Validation Scenario II examines the relative timing of the peaks of the children absenteeism curve, the over-the-counter drug purchase curve, and the incidence curve. Empirical data is gathered from http://www.wrongdiagnosis.com/f/flu/stats.htm and the National Institute of Allergy and Infectious Disease (NIAID).

6.5.1 Validation Scenario I: Incidence Factors

This scenario examines the simulated incidence compared to the empirical observed incidence for influenza in relation to the input parameters of initial rate of spread, ailment effective radius, and ailment exchange proximity threshold.

Prevalence and incidence are measures of a disease's occurrence. The *prevalence* of a condition denotes the number of people who currently have the condition, while the *incidence* refers to the annual number of people who have a case of the condition. A cumulative sum of incidence yields prevalence, on the condition that other factors are assumed to have no effects. Influenza has a high incidence but low prevalence, while diabetes – a chronic incurable disease – has a high prevalence but low incidence.

In the simulation, we can have the God's eye-view of incidence and prevalence, so we can have the actual numbers of incidence and prevalence, thus actual incidence and actual prevalence can be known. In the real world, only observed incidence and prevalence are known (in simulation these are mirrored by the simulated observed incidence and observed prevalence variables).

The variables and output values for this scenario are as follows.

(1) Outputs for empirical matching: I choose the simulated actual incidence to match with the empirical data, given that other measures are more or less the same for the purpose of inference. Which is to say, I can use the simulated observed incidence, but this adds another factor (the rate by which incidence is "observed" in simulation) which is non-critical. I could also use prevalence, but this also adds more factors such as disease recovery rate. As the empirical data I have from NIAID, the National Institute of Allergy and Infectious Disease, is the observed incidence data, I do not compare it with the simulated observed incidence data to keep things simple (there is no duplicate "observation"). Instead, I compare it with the simulated actual incidence data.

(2) Variables: base-rate (the rate of infections among susceptible persons exposed by a disease release), ailment effective radius (the radius from the center of disease agent release that persons can get infected initially), and ailment exchange proximity threshold (the distance over which the probability of ailment transmission decreases significantly).

The knowledge base is as follows.

The causal conceptual diagram:

(causes ailment-effective-radius infection-rate)

(causes ailment-exchange-proximity-threshold infection-rate)

(causes base-rate infection-rate)

(convertible infection-rate actual-incidence)

Note that the optional mechanisms/processes underlying causal relations are not used in this scenario. The mechanisms can be represented as a table, a function, or a pseudocode.

The rules related to the causal relations are as follows. The “op-” prefix in a predicate denotes the operand predicate which changes the value of the variable.

(if-then (toolow actual-incidence) (op-higher ailment-effective-radius))

(if-then (toohigh actual-incidence) (op-lower ailment-effective-radius))

(if-then (toolow actual-incidence)

(op-higher ailment-exchange-proximity-threshold))

(if-then (toohigh actual-incidence)

(op-lower ailment-exchange-proximity-threshold))

(if-then (toolow actual-incidence) (op-higher base-rate))

(if-then (toohigh actual-incidence) (op-lower base-rate))

The simulation instantiations of variables are as follows.

(setvalue base-rate 0.2)

(setbelief base-rate 0.5)

(setpriority base-rate 3)

(setvalue ailment-effective-radius 1000)

(setbelief ailment-effective-radius 0.1)

(setpriority ailment-effective-radius 1)

(setvalue ailment-exchange-proximity-threshold 1000)

(setbelief ailment-exchange-proximity-threshold 0.2)

(setpriority ailment-exchange-proximity-threshold 3)

The simulation instantiations of outputs are as follows. The BioWar simulator is run for 10 trials for 100% scale Hampton city. Part of the Alert WIZER module computes the statistical descriptions of simulated actual-incidence from the 10 simulation trials. It gives out the mean of 0.0821 and the standard deviation of 0.0015 for simulated actual-incidence.

(setvalue actual-incidence 0.0821)

(setstdev actual-incidence 0.0015)

(setbelief actual-incidence 1.0)

The empirical data is as follows:

(setvalue emp-observed-incidence-lowval 0.10)

(setvalue emp-observed-incidence-highval 0.20)

This scenario is based on the comparison of simulated actual-incidence of influenza with the empirical data from NIAID. The empirical data: 10% (the lower bound) to 20% (the higher bound) of people have flu incidence yearly. The simulated average actual incidence of 10 runs of 100% Hampton (population 142,561 persons) is 8.21% of people have flu incidence yearly.

The Alert WIZER module compares the simulation instantiation of the output actual-incidence with the empirical observed incidence. The Inference Engine performs rule inferences based on the symbolic results of the comparison. After conflict resolutions based on the priority value (here other weighting factors are not considered), it gives the inference of:

(toolow actual-incidence)

(op-higher ailment-effective-radius)

The inference is that the ailment effective radius should be increased. How much the increase should be is determined by domain knowledge, ontology, and experiment design. Absent these, the value is simply determined by a simple divide-and-conquer mathematical routine, assuming that the parameter is more or less monotonic. If it is not monotonic, the routine degenerates to random search like the Monte Carlo method.

In the next simulation cycle, the ailment effective radius is increased from 1000 meter to 1500 meter, based on a rough estimate of the extent of the area in which the ailment would affect people, as encoded in the knowledge base for the value/link adjustment routine. This represents a change of 50%. BioWar is re-run for 10 trials for the same 100%-scale Hampton city and then WIZER is re-run, and the results are:

(setvalue actual-incidence 0.1482)

(setstdev actual-incidence 0.0156)

The empirical data is again as follows:

(setvalue emp-observed-incidence-lowval 0.10)

(setvalue emp-observed-incidence-highval 0.20)

The Inference Engine responds with the notice:

(op-valid actual-incidence)

The above means that BioWar is now generating simulated incidence levels that are within the empirical observed incidence bounds. This indicates WIZER can be used to increase a model's validity, such as BioWar's validity based on its inferences.

The following table summarizes the simulated incidence rate before and after parameter value change as compared to the empirical bounds of observed incidence rate.

Table 4. Simulated Incidence Rate before and after Change

	Empirical lower bound	**Empirical higher bound**	**Simulated rate before change**	**Simulated rate after change**
Incidence rate	0.10	0.20	0.08	0.15

As shown in the table, the simulated incidence rate is moved to within the empirical bounds by WIZER.

6.5.2 Validation Scenario II: Absenteeism and Drug Purchase Curves

This scenario examines the relative timing of peaks of the children absenteeism and the drug purchase curves against the peak of the incidence curve.

The variables and output values for this scenario are as follows.

(1) Outputs for empirical matching: I choose the simulated actual incidence, the school absenteeism, and the influenza drug purchase curves. The Alert WIZER finds the peaks of the curves and computes the time-differences between the peaks.

(2) Variables: as the onset of absenteeism is influenced by symptom onset and symptom severity, these two factors form the variables. In addition to being influenced by the two factors, the onset of influenza drug purchase is influenced by the going-to-pharmacy behavioral threshold. Thus, the total variables for this scenario (with some simplifications) are symptom-onset, symptom-severity, and going-to-pharmacy-threshold.

The knowledge base is as follows.

The causal conceptual diagram:

(causes symptom-onset absenteeism-onset)

(causes symptom-severity absenteeism-onset)

(causes symptom-onset drug-purchase-onset)

(causes symptom-severity drug-purchase-onset)

(causes going-to-pharmacy-threshold drug-purchase-onset)

(convertible infection-rate incidence-rate)

The onsets are computed against the time of infection. Note that the optional mechanisms underlying causal relations are not used in this scenario. The mechanisms can be represented as a table, a function, or a pseudocode.

The rules related to the causal relations are as follows. The “op” prefix denotes the operand predicate which changes the value of the variable.

(if-then (toosoon absenteeism-onset) (op-lengthen symptom-onset))

(if-then (toolate absenteeism-onset) (op-shorten symptom-onset))

```
(if-then (toosoon absenteeism-onset) (op-lower symptom-severity))
(if-then (toolate absenteeism-onset) (op-higher symptom-severity))
(if-then (toosoon drug-purchase-onset) (op-lengthen symptom-onset))
(if-then (toolate drug-purchase-onset) (op-shorten symptom-onset))
(if-then (toosoon drug-purchase-onset) (op-lower symptom-severity))
(if-then (toolate drug-purchase-onset) (op-higher symptom-severity))
(if-then (toosoon drug-purchase-onset) (op-higher going-to-pharmacy-threshold))
(if-then (toolate drug-purchase-onset) (op-lower going-to-pharmacy-threshold))

(if-then (tooshort absenteeism-vs-actual-incidence)
        (op-toosoon absenteeism-onset))
(if-then (toolong absenteeism-vs-actual-incidence)
        (op-toolate absenteeism-onset))
```

The simulation instantiations of variables are as follows.

```
(setvalue symptom-onset 2)
(setbelief symptom-onset 0.5)
(setpriority symptom-onset 3)
(setvalue symptom-severity 3)
(setbelief symptom-severity 0.1)
(setpriority symptom-severity 1)
(setvalue going-to-pharmacy-threshold 100)
(setbelief going-to-pharmacy-threshold 0.2)
(setpriority going-to-pharmacy-threshold 2)
```

The simulation instantiations of outputs are as follows. One simulation of Hampton city with 100% scale is run. The Alert WIZER computes the peaks of actual-incidence, school absenteeism, and drug purchase curves. It produces the relative timing of the peaks with respect to the actual-incidence peak. The following figure shows the actual-incidence curve.

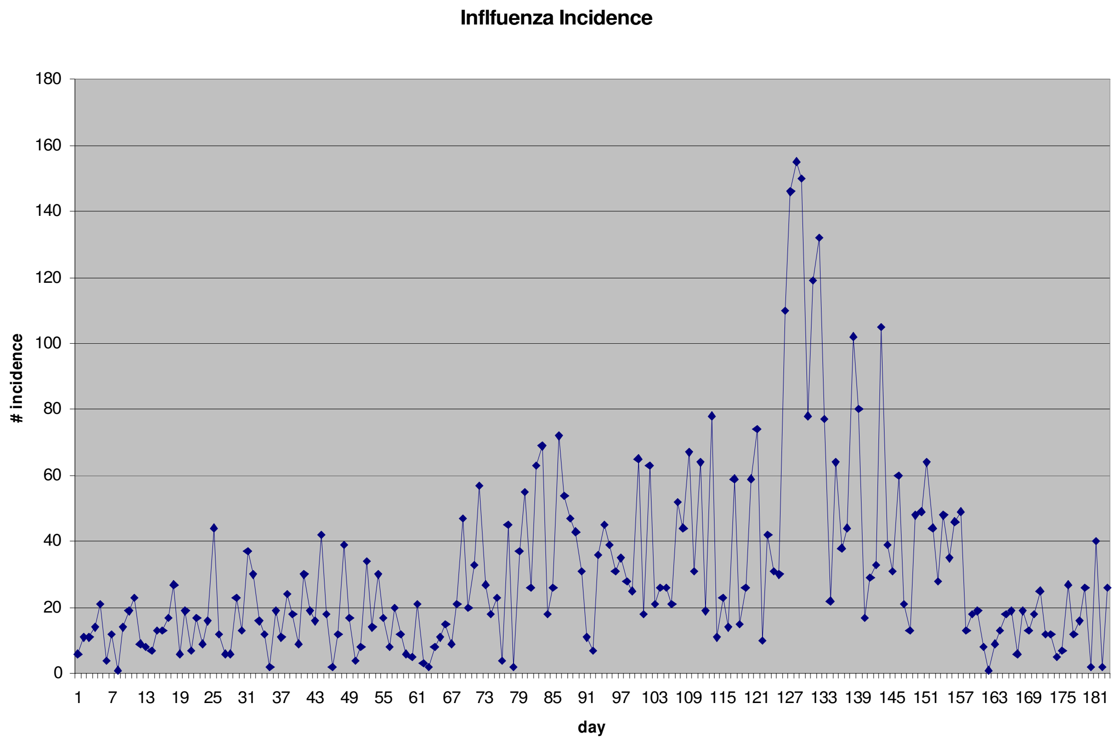

Figure 9. The Peak of Incidence Occurs on Day 128

As shown, the peak of incidence occurs on Day 128. Day 1 is the start of the simulation, corresponding to September 1, 2002. The peak is computed by finding the maximum point and averaging the data points within a fixed time interval around the maximum point time. This assumes no outliers.

The Relative Timing of School Absenteeism Peak

In the simulation trial, the relative time difference between simulated absenteeism and simulated actual-incidence peaks is 10 days.

(setvalue absenteeism-vs-actual-incidence 10)

(setbelief absenteeism-vs-actual-incidence 1.0)

The CDC data says the incubation period for influenza is 1-4 days. Absenteeism occurs a day after the end of incubation. Thus, the empirical data is as follows:

(setvalue emp-absenteeism-vs-actual-incidence-lowval 2)

(setvalue emp-absenteeism-vs-actual-incidence-highval 5)

The following figure shows the school absenteeism curve.

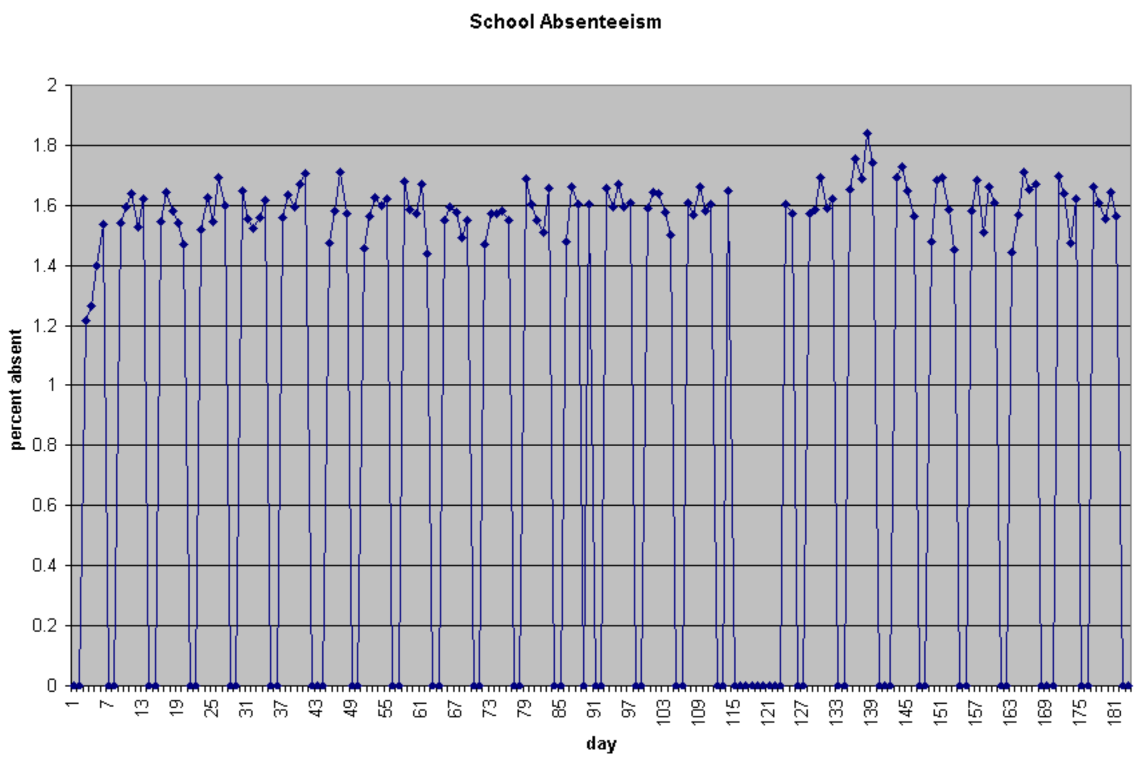

Figure 10. The Peak of School Absenteeism Occurs on Day 138

As shown, the peak of school absenteeism occurs on Day 138. The curve is broken on Saturdays and Sundays as the schools are closed. Days 115-121 are holidays. The peak is computed by finding the maximum average of weekly data and the averaging a few data points within set time intervals around the maximum point. This of course assumes that there are no outliers.

The inference engine compares the relative timing of absenteeism and incidence peaks with the empirical relative timing. After conflict resolutions based on the priority value (here other weighting factors are not considered), it produces the inference of:

(toolong absenteeism-vs-actual-incidence)

(op-higher symptom-severity)

because the absenteeism peak lags 10 days behind the incidence peak; twice as long as the empirical maximum of 5 days.

The inference is that the symptom-severity (the relative timing and magnitude of manifested symptoms) should be increased. How much the increase should be is determined by domain knowledge, ontology, and experiment design. Absent these, the value is simply determined by a simple divide-and-conquer algorithm.

For the next cycle of simulation, the symptom severity is increased by 100% by the value/link adjustment routine using an encoded rule about critical point heuristics. BioWar is re-run and then WIZER is re-run. The following figure shows the resulting curve of school absenteeism.

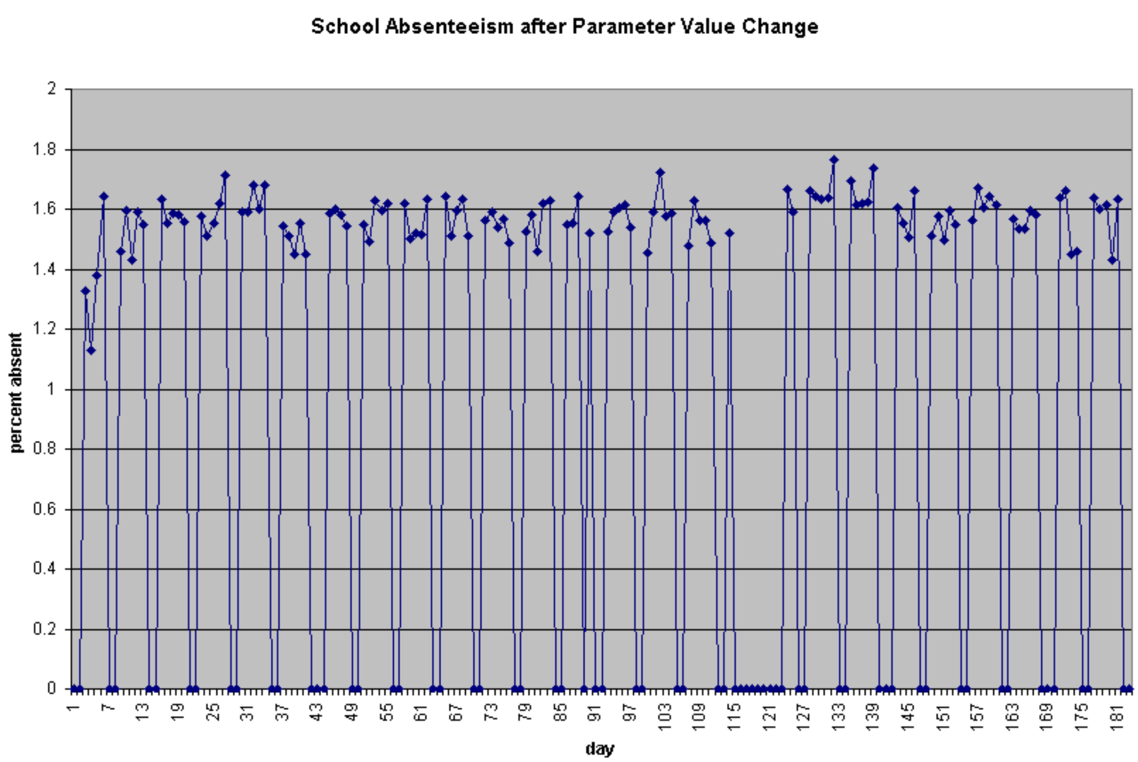

Figure 11. The Peak of School Absenteeism after Change Occurs on Day 132

As shown, the peak of school absenteeism now occurs on Day 132. The Inference Engine compares the relative timing of absenteeism and incidence peaks with the maximum empirical relative timing. After conflict resolutions are performed, it now produces the inference of:

(within-range absenteeism-vs-actual-incidence)

(op-valid)

The relative time difference between absenteeism and actual-incidence peaks is now 4 days, which is less than the previous cycle's relative time difference of 10 days. It is now one day shorter than the maximum empirical time difference. Thus the peak of school absenteeism is moved to the valid range within the empirical bound of 2-5 days. So the Inference Engine produces a notice that the simulated absenteeism curve peak is now valid.

The following figure shows the school absenteeism curves before and after parameter value change.

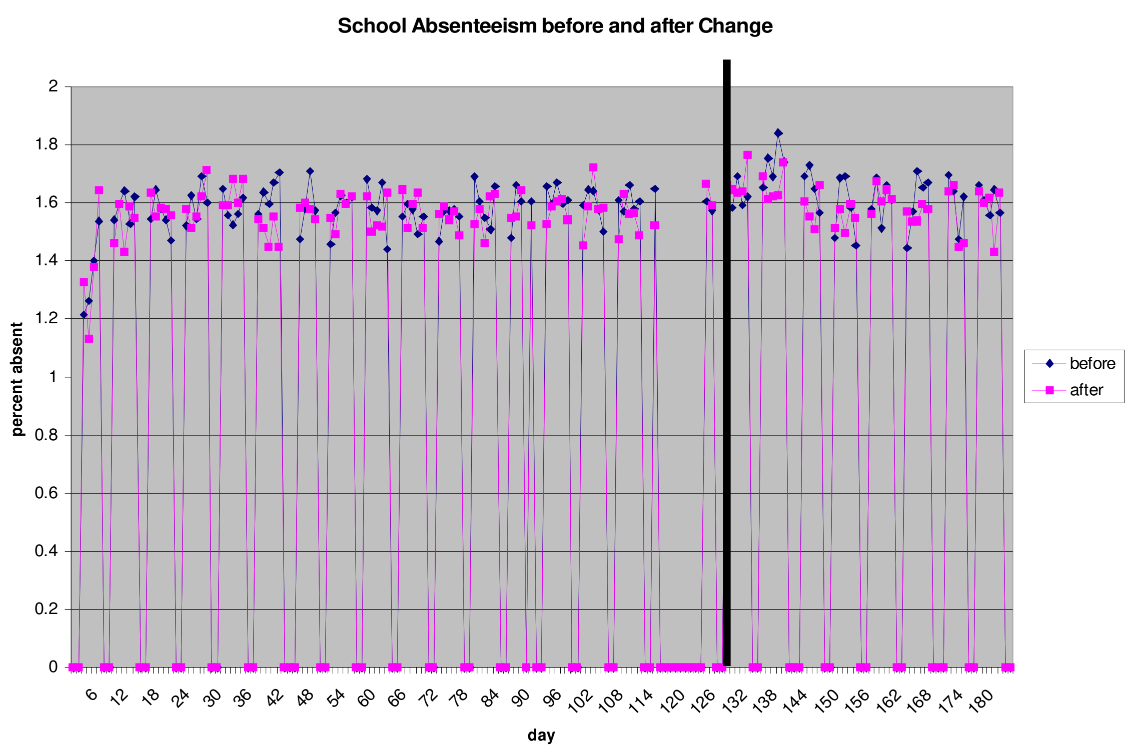

Figure 12. School Absenteeism Curves before and after Parameter Value Change

As shown, the absenteeism peak after parameter value change moves closer to the time at which the incidence peaks (as shown by the black vertical line) than the before-change absenteeism peak.

The Relative Timing of Drug Purchase Peak

The next comparison of curve peaks is between the drug purchase curve and the incidence curve. I show first the virgin case, before parameter value change. The following figure shows the drug purchase curve for cold/cough medication of influenza before change.

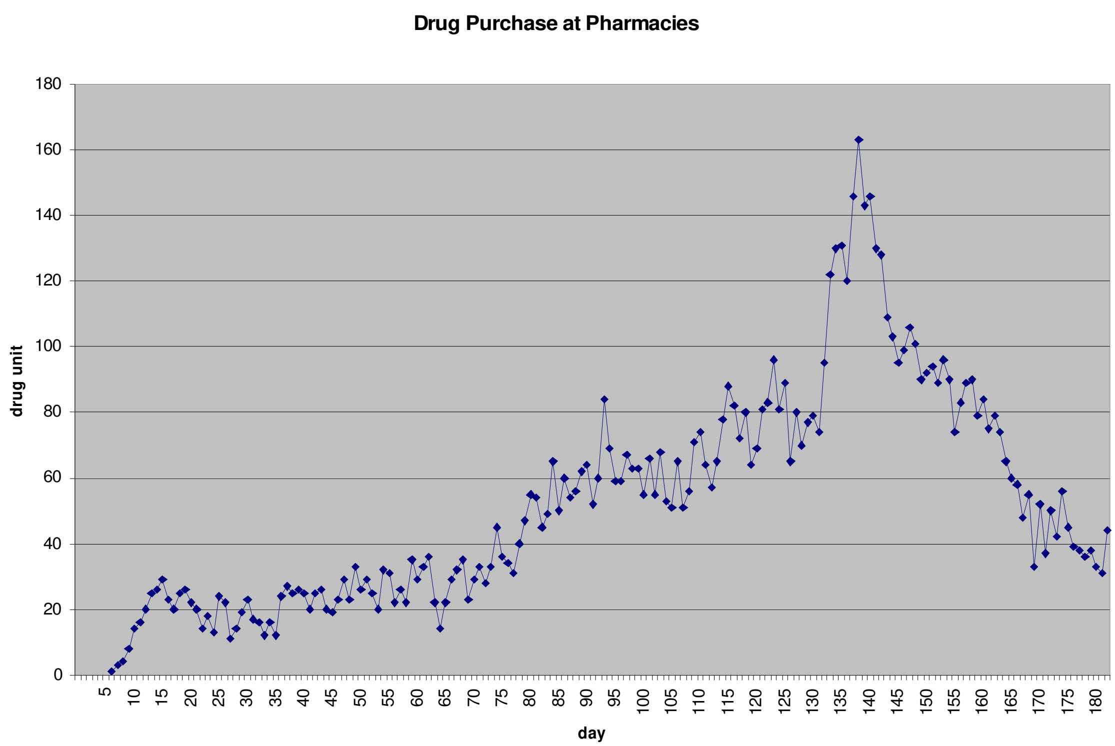

Figure 13. The Peak of Drug Purchase for Influenza Occurs on Day 139

As shown, the peak of drug purchase for influenza occurs on Day 139. The peak occurs 11 days after the incidence peak.

The incubation period for influenza is 1-4 days, and the illness typically resolves after 3-7 days for the majority of persons according to CDC http://www.cdc.gov/flu/professionals/diagnosis. The maximum days for typical influenza are 11 days.

Assume that on the day after (Day 6) the symptom shows up (Day 5) the parents go to pharmacies and buy the influenza medication for their children. This means the peak of drug purchase must be 6 days after the incidence peak, which is to say, the peak of drug purchase must occur on Day 134. This means the peak of drug purchase above is too late, by 5 days.

The WIZER Inference Engine yields:

(toolate drug-purchase-onset)

(op-shorten symptom-onset)

(op-higher symptom-severity)

(op-lower going-to-pharmacy-threshold)

After conflict resolution by the priority measure, it yields:

(op-higher symptom-severity)

Again, how much the symptom-severity should increase is determined by knowledge base, ontology, and experiment design.

For the next cycle of simulation, the symptom severity is increased by 100% by the value/link adjustment routine using a simple heuristic knowledge encoded in its rules. BioWar is re-run and then WIZER is re-run. The result for drug purchase curve is shown in the following figure.

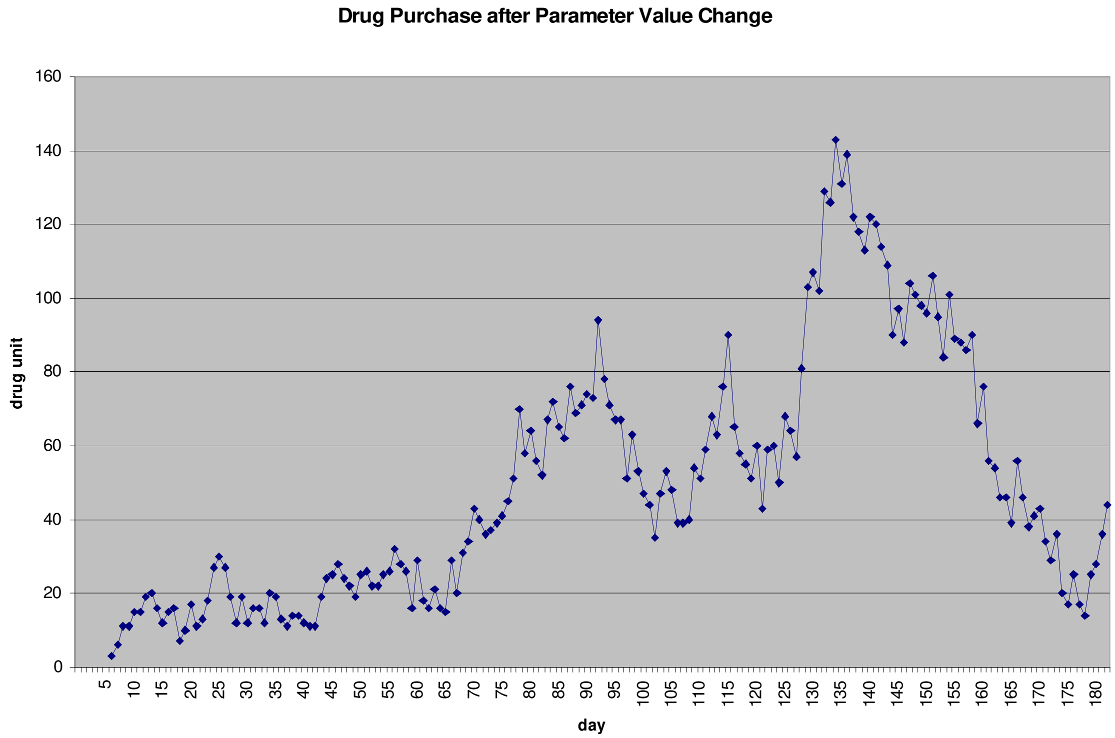

Figure 14. The Peak of Influenza Drug Purchase after Change Occurs on Day 135

As shown, the peak of drug purchase for influenza after parameter value change now occurs on Day 135. This is 3 days after the peak of the after-change school absenteeism and 7 days after the incidence peak. This means the peak of drug purchase above is one day longer than the maximum empirical length of time between the drug purchase peak and the incidence peak.

The WIZER Inference Engine yields:

(toolate drug-purchase-onset)

and after conflict resolution, it produces:

(op-higher symptom-severity)

Thus the drug purchase peak has been moved to be within one day of the maximum empirical range of the time difference between the incidence and the drug purchase peaks. The following figure shows the drug purchase curves and peaks before and after parameter value change. Also shown is the incidence curve and peak. The Y-axis unit denotes either the drug purchase unit for drug purchase curves or the number of incidences for the incidence curve.

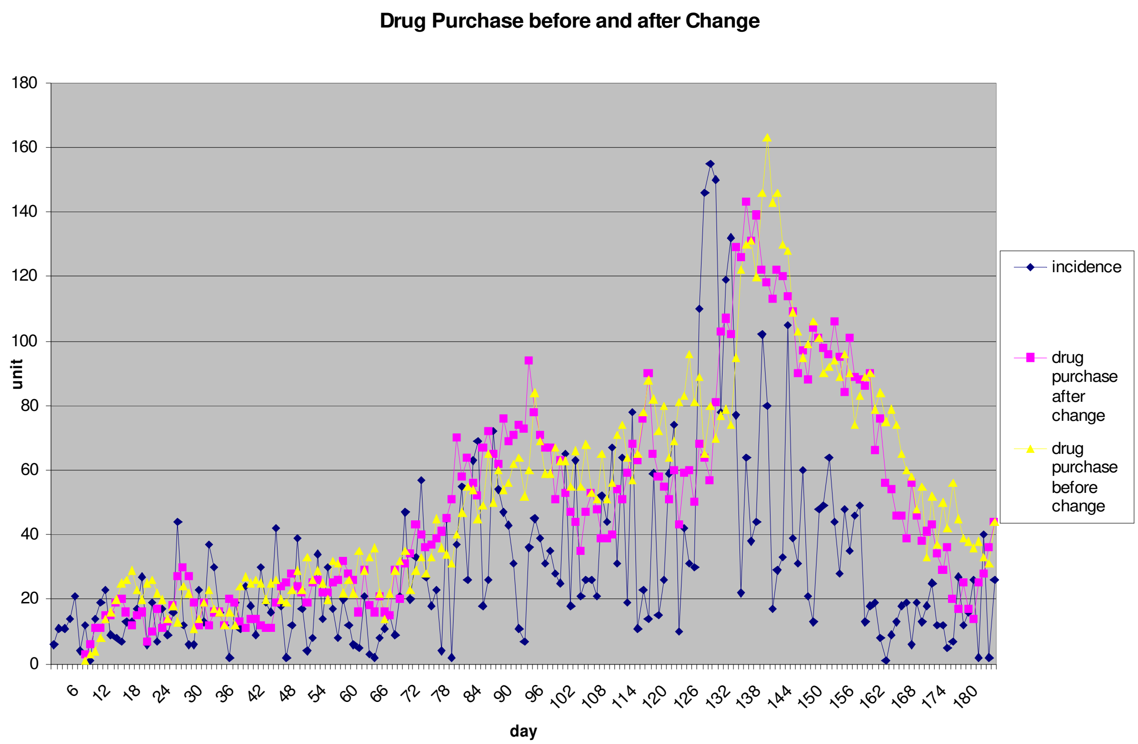

Figure 15. Drug Purchase Curves before and after Parameter Value Change

As shown, the drug purchase peak is moved closer to the incidence peak after the parameter value change. It is moved closer to the empirical time at which the drug purchase should peak, with the time difference of only one day. For the next value adjustment, WIZER makes a slight change to the symptom severity value as the time difference between the peaks of the simulated drug purchase and incidence curves lags its maximum empirical range by only one day, instead of the previous cycle's 5 days.

6.6 Validation Measures

Validation is measured based on a piece of knowledge that corresponds to a data stream. For the results above:

1. Incidence Factors: the simulated actual incidence rate is lower than the lower bound of the empirical observed incidence rate. Strictly speaking, the data stream output is not valid. After value change by WIZER, the simulated incidence rate is moved to be within the empirical range, achieving validity.
2. School Absenteeism: the simulated school absenteeism peak occurs later than it should be. Thus this data stream has zero validity, strictly speaking. But the comparison of the shape/trend of the curves seems to indicate that the validity level is much higher than zero. Furthermore, after value change by WIZER, the simulated absenteeism peak is moved to be within the empirical range, achieving validity.
3. Drug Purchase: the simulated drug purchase peak also occurs later than it should be. Strictly speaking, this data stream is invalid. Again, if the shape of the curves is compared, it looks like the validity level for this data stream is much higher than zero. After one cycle of value change by WIZER, the drug purchase peak is moved to be within one day of its maximum empirical range.

The results indicate the iterative process of doing validation. In addition to the need to perform multiple runs, there can be multiple measures of validity.

6.7 WIZER versus Response Surface Methodology for BioWar Validation

BioWar has hundreds of parameters. The resulting parameter space is gigantic. Suppose that the Response Surface Methodology or RSM (Myers and Montgomery 2002, Carley, Kamneva, and Reminga 2004) is used to completely characterize BioWar for validation. Given that it is estimated that BioWar has 200 parameters (a conservative number) and assuming that each parameter can have 3 different values (3 levels), the parameter space is *3^200* cells, which is unmanageable by the current technology. As BioWar is stochastic, each cell requires 40 virtual experiments to get statistically significant results, incurring 40 times increase in the parameter space. Quantum computers might someday make the execution of *40 x 3 ^ 200* simulations feasible but not today.

Experimenters, of course, can divide the system into modules and validate a module by module, assuming all other modules have reasonable parameter values and the existence of some modularity in the system. If this is done for BioWar, experimenters can probe the relationships between incidence rate and infection factors such as ailment_effective_radius, ailment_exchange_proximity_threshold, and base_rate. Assuming each of these factors has 3 levels (3 possible values), the following table shows the number of cells required.

Table 5. Number of Cells for Validation of Incidence Factors

Parameter	Categories	Size
Ailment effective radius	500, 1000, 1500	3
Ailment exchange proximity radius	500, 1000, 1500	3
Base rate	10%, 30%, 50%, 70%	4

As shown, the total number of cells required is *3 x 3 x 4 = 36* for non-stochastic program. Being stochastic, BioWar requires *36 x 40 = 1,440* virtual experiments, which is perfectly manageable. The way experimenters decide to choose the parameters and the parameter levels, however, is totally *ad-hoc*, implicit, and unusable for computer operation.

WIZER enhances the way experimenters decide which parameters and what parameter levels to choose by codifying the knowledge in a form that is clear, explicit, and operable by computers. It is codified in the form of knowledge bases and ontology. With its inference engine, WIZER can reason about parameters and simulation results producing new inferences, that is, inferences that no human experimenters have input or thought of before. Furthermore, utilizing its knowledge inference, WIZER can further reduce the number of virtual experiments needed. The above number of virtual experiments for RSM of *1,440* is the upper limit of what WIZER needs. Typically, WIZER needs fewer than that due to its inferences about simulation results after each simulation cycle.

6.8 Summary

WIZER is shown to be able to partially validate the BioWar simulation model. The incidence factors and the relative timing of school absenteeism peak are validated using WIZER. The relative timing of drug purchase peak is almost validated: it falls within one day of the maximum empirical relative timing. The results show the use of Alert WIZER to describe the output data streams/curves and compare them to produce symbolic or semantic alerts.

The results show that while WIZER is capable of doing validation, validation depends critically on the provided knowledge. This brings up the issues of knowledge engineering and knowledge acquisition. Fortunately, as part of the process of simulation model development and of validation, the task of both knowledge engineering and knowledge acquisition can be handed over to the respective stakeholders: the task to create simulation knowledge space to the simulation developers and the task to create domain knowledge to the validators or the VV&A practitioners.

Chapter VII: CONSTRUCT Testbed

CONSTRUCT (Carley 1990, Carley 1991, Schreiber and Carley 2004) is a multi-agent model of group and organizational behavior in the form of networks, capturing the co-evolution of cognition (knowledge) and structure of said networks.

This chapter explains why automated validation for CONSTRUCT is desirable. It also provides several partial validations of CONSTRUCT using WIZER. It shows one simple hypothesis building and testing case: asking the question of what if the management is not homogeneous as it is assumed before.

7.1 Description of CONSTRUCT

CONSTRUCT adopts the constructural theory for the formation of social structure and knowledge. This means when a person interacts with another, he/she exchanges knowledge. This exchange modifies both persons' store of knowledge. The change in the store of knowledge in turn affects with whom a person will interact next. Thus both the cognition (knowledge) and structure of the interaction network change.

As an example, suppose that person A interacts with person B, and learns that person B knows for a fact that there is a sale going on at Macy's on a particular Sunday. Once person A learns this fact, this person A may go to Macy's on Sunday and inadvertently meets person C who also knows the same fact (from someone or somewhere). Once in proximity, person A and person C may interact and exchange another bit of knowledge. Notice that once person A and person C interact with each other, their social network changes. This shows how social networks can change due to change in knowledge. The piece of knowledge person A and person C exchange with each other may affect their decisions about who to interact next, where to go, what to do, and so on. The change in social networks in turn affects which piece(s) of knowledge gets exchanged next.

The above determination on who to interact based on common bits of knowledge is known as homophily. A wealth of social science studies has shown that people consciously and unconsciously prefer to interact with people who look like themselves (e.g., have the similar age, hobbies, socioeconomic status, etc.). This kind of interactions is also known as social or emotional ties. Another mode of interaction is geared toward seeking expertise or information one does not have. A patient seeking a doctor is a perfect example. This is known as instrumental or information seeking ties. The social ties are usually symmetrical or reciprocal, while the instrumental ties are asymmetrical.

All of the above has been turned into mathematical formulas in CONSTRUCT. CONSTRUCT mimics how people interact and how social ties are formed and dissolved. Augmented by how friendships and enmities are estimated from interaction probabilities, CONSTRUCT is able to predict the formation and dissolution of friendships and enmities, and in the case of Kapferer's Zambia tailor shop, the possibility of a successful strike. An augmented version of CONSTRUCT can encode that a person knows another person knows something. In other words, transactive memory can be handled. The mathematical details of CONSTRUCT are provided in (Carley 1990, Carley 1991, Schreiber and Carley 2004).

7.2 The Need for Automated Validation

In the original CONSTRUCT paper (Carley 1990), CONSTRUCT was validated using Kapferer's Zambia tailor shop worker and management interaction network data. The transactive memory version of CONSTRUCT, the CONSTRUCT-TM, has also been validated several times, with the latest validation finding a significant correlation between communication patterns in real-world organizations and agent interactions in the CONSTRUCT model (Schreiber and Carley 2004). CONSTRUCT has a derivative called DyNet (Carley, Reminga, and Kamneva 2003) which includes an agent removal feature.

The validations above were done semi-automatically, with minimal assistance from computer tools. CONSTRUCT has a large parameter and model space partially due to its knowledge vector, interaction modes, network data, and information and application contexts, so it is desirable to have the validation knowledge managed and to have the validation automated.

7.3 Validation Scenarios

Australian anthropologist Bruce Kapferer observed people interactions in a tailor shop in Zambia (then Northern Rhodesia) over a period of ten months in 1972. He collected two sets of data. The first set of data was collected just before an abortive strike. This instant of time is denoted *Time1*. The data is collected over a period of a month. After seven months, Kapferer collected a second set of data. Shortly after this second data collection, denoted *Time2*, a successful strike took place. This data collection also took a month. The data sets consist of both the "instrumental" (work- and assistance-related) interactions and the "sociational" (friendship, socioemotional) interactions. The data collections occurred during extended negotiations for higher wages. The data is in the form of matrices of interactions, which can be transformed into matrices of person-knowledge.

Here I create three validation scenarios for CONSTRUCT. Validation Scenario I demonstrates how WIZER can be used to facilitate the validation of CONSTRUCT against Kapferer's data of empirical average interaction probability among workers. There are 39 maximally heterogeneous workers data that I use here. “Maximally heterogeneous” means the workers have the most diverse background and knowledge. They are significantly different from one another. Here the behavior of workers is examined by probing the change of the average probability of interaction among workers before the successful strike. Validation Scenario II shows how WIZER facilitates the validation of CONSTRUCT against Kapferer’s tailor shop data, but with the examination of two group behaviors and one intergroup behavior. The groups are the maximally heterogeneous workers and homogeneous management. Validation Scenario III plays out a "what-if" scenario of the management being not homogeneous.

7.3.1 Validation Scenario I: Interaction Probability around the Time of the Successful Strike

Based on the empirical network data he gathered, Kapferer calculated that the workers had the interaction probability of 0.005502 just before the successful strike at *Time2*. In this validation scenario, the CONSTRUCT model is initialized by the network data at *Time1* (the time just before the abortive strike) for the start of simulation. It is run for 30 simulation trials with the maximum of 45 time-steps per simulation trial.

The knowledge base is as follows.
The causal conceptual diagram:

(causes interaction knowledge-exchange)

(causes knowledge-exchange shared-knowledge)

(causes shared-knowledge interaction)

This causal diagram is cyclic but it has a time delay between the causes and effects. The unary operand "op-valid" means the simulation is valid for the case.

The rules related to the causal relations:

(if-then (toolow interaction) (op-higher shared-knowledge))

(if-then (toohigh interaction) (op-lower shared-knowledge))

(if-then (lower shared-knowledge) (op-lower knowledge-exchange))

(if-then (higher shared-knowledge) (op-higher knowledge-exchange))

(if-then (lower knowledge-exchange) (op-lower interaction))

(if-then (higher knowledge-exchange) (op-higher interaction))

(if-then (lessthan interaction-probability-after-strike emp-avg-int-prob-strike)
    (op-toolow interaction))

(if-then (morethan interaction-probability-before-strike emp-avg-int-prob-strike)
    (op-toohigh interaction))

(if-then (and

(morethan interaction-probability-after-strike emp-avg-int-prob-strike)
(lessthan interaction-probability-before-strike emp-avg-int-prob-strike))
(op-valid))

The empirical data is declared as follows.

(setvalue emp-avg-int-prob-strike 0.005502)
(setbelief emp-avg-int-prob-strike 1.0)

The simulation instantiations of variables are as follows.

(setvalue workers-matrix Kapferer-Time1-workers-data)

which initiates the simulation starting state with the workers data at *Time1*.

(setmode construct-interaction-mode homophily)

CONSTRUCT is run for 30 trials and the Alert WIZER takes the average probability of interaction output of CONSTRUCT and looks for the number before and after the successful strike. The following figure shows the average probability of interaction among workers as a function of time. The timestep "unit" correlates loosely with the actual period of time, which is to say, I do not perform time-validation here. The Alert WIZER spots the transition (before and after the strike) at Timestep 30, which has the average interaction probability of 0.005188 (which is less than 0.005502). The next timestep, Timestep 31, sees the average interaction probability of 0.008612 (which is more than 0.005502). The two probabilities bracket the empirical probability of interaction, thus the simulated average interaction probabilities and the simulation model for this case are correct. This indicates that CONSTRUCT is valid for this case.

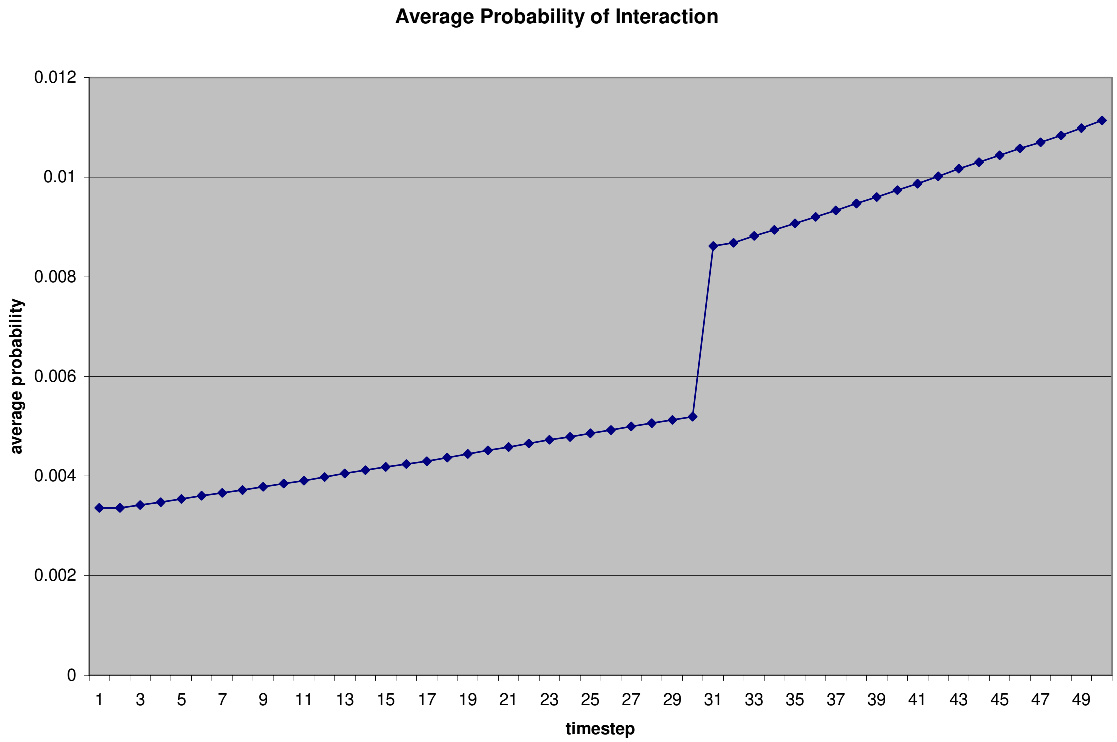

Figure 16. The Average Probability of Interactions among Workers

The average probability of interactions among workers is shown: the transition representing the successful strike happens between Timesteps 30 and 31. Kapferer's empirical average interaction probability of 0.005502 lies between the average interaction probabilities of Timestep 30 and 31.

In WIZER inference traces,

The simulation instantiations of outputs:

(setvalue interaction-probability-before-strike 0.005188)

(setvalue interaction-probability-after-strike 0.008612)

The inference engine has the following step.

```
(if-then (and (morethan 0.008612 0.005502)
          (lessthan 0.005188 0.005502))
          (op-valid))
(op-valid)
```

One interpretation of the above result is that as the interactions and shared-knowledge increase, the workers are being primed for a leap of faith of increased unionization (and thus homogenization and radicalization). Increased unionization increases the risks of confrontation with the management. However, to really explain why the successful strike happened it is necessary to account for friendships and enmities. The treatment of friendships and enmities and the explanation of why the successful strike occurred were given in (Carley 1990).

7.3.2 Validation Scenario II: Maximally Heterogeneous Workers and Homogeneous Management

In the Zambia tailor shop, there was a management consisting of 4 Indians among the mostly African workers. One Indian of these four, Patel, serves as the actual factory manager. Most of the interactions between workers and management occurred between the workers and Patel. The management is homogeneous. They interact with each other most of the time and have the same culture and economic status.

During the period between the abortive strike and the successful strike, the interactions among workers and between workers and management increased. In this validation scenario, WIZER is setup to allow detection and comparison of the trends of the change of interaction probabilities within and between groups. The knowledge base is as follows.

The causal conceptual diagram:

(causes (homogeneous management) (increasing intergroup-interaction-change))

(causes (homogeneous management) (increasing workers-interaction-change))

(causes (homogeneous management)

    (higherthan intergroup-interaction-change workers-interaction-change))

(causes (homogeneous management)

    (higherthan workers-interaction-change management-interaction-change))

(causes (wage negotiations) (increasing intergroup-interaction-change))

The rules related to the causal relations relevant here are:

(if-then (higherthan workers-interaction-change management-interaction-change)
    (op-valid))

(if-then (higherthan intergroup-interaction-change workers-interaction-change)
    (op-valid))

The simulation instantiations of variables are as follows.

(setvalue management homogeneous)

(setvalue workers-matrix Kapferer-Time1-workers-data)

(setvalue construct-interaction-mode homophily)

CONSTRUCT is run for 30 Monte-Carlo trials. Alert WIZER processes the output interaction curves, and produces the curve trend comparisons as symbolic or semantic information. The simulation instantiation of outputs are:

(higherthan workers-interaction-change management-interaction-change)

(higherthan intergroup-interaction-change workers-interaction-change)

The Inference Engine then produces:

(op-valid)

This means that the CONSTRUCT model for the case of homogeneous management validly reproduces the empirical trends of the change of interaction probabilities for the workers-group, the management-group, and the workers-management intergroup as observed by Kapferer during the period between the abortive strike at *Time1* and the successful strike at *Time2*.

The following figure shows the interaction curves as measured by the percent change of probability of interactions.

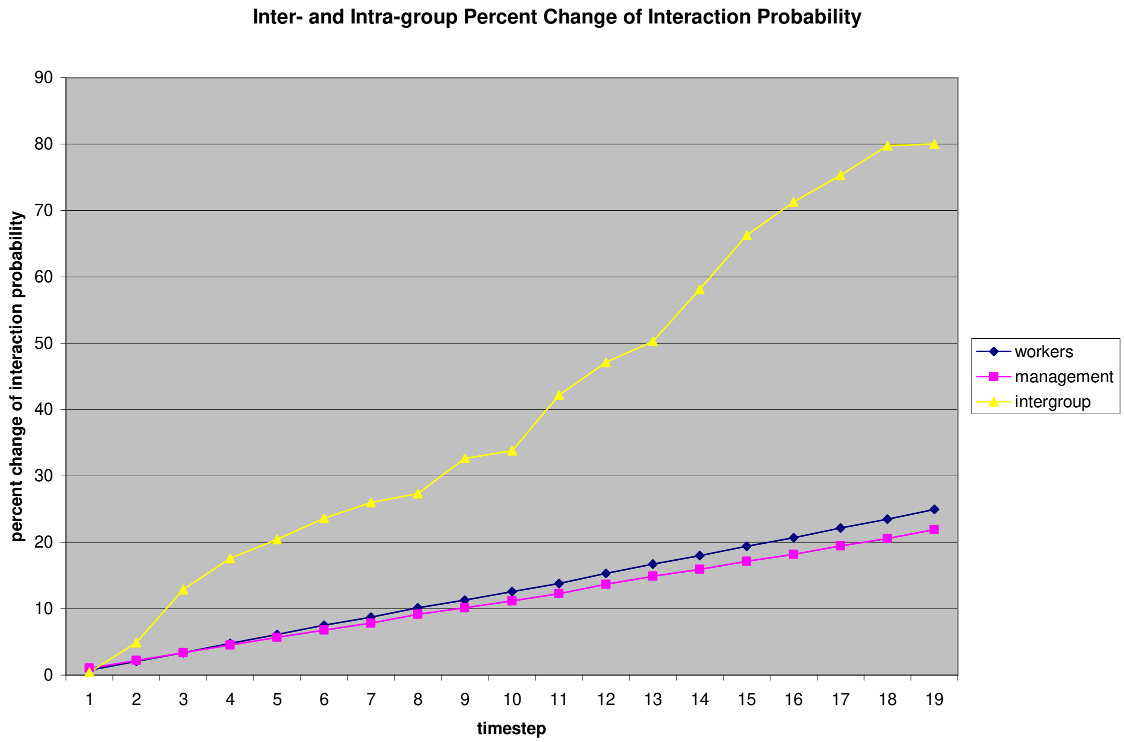

Figure 17. Percent Change of Interaction Probabilities for the Workers Group, the Management Group, and the Intergroup

The significantly increased workers-management interaction change can be a catalyst for the strike is one interpretation of the results. How exactly this plays out, however, depends on how enmities and friendships are formed. Increased workers intragroup interactions could lead to more integration among workers, forming an almost-homogeneous group challenging the homogeneous management group, resulting in a successful strike. This almost homogeneous worker state stands in contrast to the initial maximally heterogeneous state that the workers were in.

7.3.3 Validation Scenario III: Maximally Heterogeneous Workers and Heterogeneous Management

Previously WIZER has shown that the CONSTRUCT model for homogeneous management case is valid in light of the empirical trends of interaction probability change. It also indicates that homogeneous management could be a factor contributing to the successful strike. Now we are curious of what would transpire if the management is not homogeneous. This curiosity is encoded in ontology. WIZER can handle the "what-if the management is heterogeneous question" by doing a search in the ontology, forming a new causal conceptual diagram, and then doing hypothesis testing. In the extended N3 notation, the ontology for the management is written as having the attributes:

<management> <has-type-of> <homogeneous, heterogeneous> .

The domain knowledge's inference engine executes the what-if statement of: if homogeneous type of management has been probed then probe other types of management. As shown the probing of other types of management is assisted by ontology. This is similar to model perturbations in the model-based reasoning. The exact mapping of the management attributes of homogeneous or heterogeneous to the interaction and person-knowledge matrices is declared by Alert WIZER's symbolic or semantic characterization of inputs, instead of outputs, with the help from ontology. As described earlier, Alert WIZER is capable of doing symbolic and semantic categorization of numeric and network data.

The causal conceptual diagram for this what-if case becomes:

```
(causes (heterogeneous management) (higher intergroup-interaction-change))
(causes (heterogeneous management) (higher workers-interaction-change))
(causes (heterogeneous management)
        (higherthan intergroup-interaction-change workers-interaction-change))
(causes (heterogeneous management)
        (higherthan workers-interaction-change management-interaction-change))
```

The rules related to the causal relations are:

```
(if-then (higherthan workers-interaction-change management-interaction-change)
        (heterogeneous management))
(if-then (higherthan intergroup-interaction-change workers-interaction-change)
        (heterogeneous management))
```

The simulation instantiations of variables are as follows.

(setvalue management heterogeneous)

(setvalue workers-matrix Kapferer-Time1-workers-data)

(setvalue construct-interaction-mode homophily)

CONSTRUCT is run for 30 Monte-Carlo trials. Alert WIZER processes the output interaction curves, and gives out the comparisons. The following figure shows the percent change of interaction probability for the heterogeneous management group, the workers-management intergroup, and the maximally heterogeneous workers group.

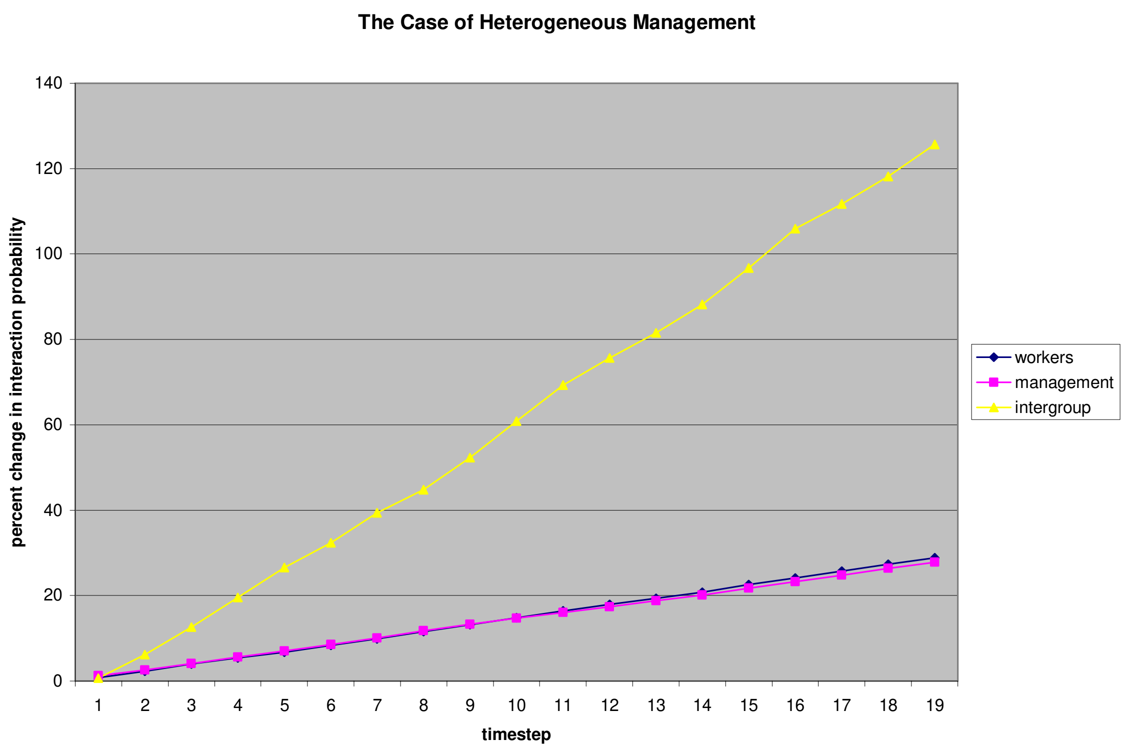

Figure 18. Percent Change of Interaction Probability for Heterogeneous Management Group, Heterogeneous Workers Group, and the Intergroup

The simulation instantiation of outputs are then:

(higherthan intergroup-interaction-change workers-interaction-change)

(equal workers-interaction-change management-interaction-change)

This means the initial assertion of (higherthan workers-interaction-change management-interaction-change) is false. Thus the initial rule of:

(if-then (higherthan workers-interaction-change management-interaction-change)

(heterogeneous management))

is false too. Moreover the initial causal relation of:

(causes (heterogeneous management)

(higherthan workers-interaction-change management-interaction-change))

is correspondingly false. This results in WIZER purging one causation and one rule from the CONSTRUCT model declaration for the heterogeneous case. The causation and rule are replaced by:

(causes (heterogeneous management)

(equal workers-interaction-change management-interaction-change))

(if-then (equal workers-interaction-change management-interaction-change)

(heterogeneous management))

In this simple way, WIZER "learns" by inference after the model outputs are compared against the empirical data.

It turns out that the changed data indicates the interaction probability trends for heterogeneous management for the CONSTRUCT model. The result also shows that WIZER can reduce the amount of simulation parameter search by doing the search or inference in the conceptual space or in ontology. The homogeneous or heterogeneous conceptual symbol has multiple manifestations in the interaction (and the person-knowledge) matrices. We do not need, however, to examine each and every combination of the matrix values as the values can be categorized at the symbolic level by the semantic label of homogeneous and heterogeneous in ontology. If robustness is desired, we can take a statistical sample of several different matrix values, but nothing approaching brute-force or Monte Carlo sampling is needed.

Additionally, to get to the difference between homogeneous and heterogeneous management, WIZER is set up to compare the results between two scenarios above. Alert WIZER does the comparison of same curves and of the differences between curves. It produces:

(morethan intergroup-interaction-probability-change-heterogeneous

intergroup-interaction-probability-change-homogeneous)

This new rule can be used for further inference in WIZER Inference Engine.

The above results can be interpreted as:

(1) When the management is heterogeneous, they are practically very similar to workers, thus their percent change of interactions is almost the same.

(2) When the management is heterogeneous, their intergroup interaction change is higher that that of the case of homogeneous management due to the increased management-workers interactions as management and workers are similar. The management does not form a cohesive/homogeneous group.

Whether the heterogeneous management could prevent a successful strike, however, cannot be explained by the interaction probability change alone, as the measures of friendship and enmity are needed. The increased interaction probability change could lead to both increased unification and friendship (for workers) and increased strife (for workers-management intergroup). How the heterogeneity of management affects enmities and friendships, which in turn affects the change of a successful strike taking place, depends on how friendships and enmities are determined.

7.4 Validation Measures

As validation is dependent on a specific knowledge, the validity of the results is as follows:

1. Average Interaction Probabilities around the Successful Strike: the average interaction probabilities bracket the empirical interaction probability around the successful strike. This means the CONSTRUCT model is valid as measured by the average interaction probability knowledge for the workers-group.
2. Maximally Heterogeneous Workers and Homogeneous Management: the intergroup's increased change of interaction probability is much more than the workers' change of interaction probability, which in turn is more than the management's change of interaction probability. This fits the trends of what empirically transpired between the period of time after the abortive strike and before the successful strike, during which wage negotiations occurred.

3. Maximally Heterogeneous Workers and Heterogeneous Management: this is a hypothesis building and testing scenario, so there is no validity value is assigned, as there is no corresponding empirical case. However, as workers' and heterogeneous management's changes of interaction probability are more-or-less equal, it indicates that heterogeneity makes workers and management behave more like each other. Also, as the increase in the change of the interaction probability between workers and management – the intergroup – is higher that that of the homogeneous management, it indicates that heterogeneity contributes to the increased interaction between workers and the non-homogeneous management. It seems that diversity has resulted in more interactions between different groups. How these increased interactions contribute to friendships and enmities however depends on how friendships and enmities are formed.

7.5 WIZER versus Response Surface Methodology for CONSTRUCT Validation

CONSTRUCT has many parameters: the size of the knowledge vector, number of agents, type of communication mode (homophily, information seeking, etc.), type of exchange, interaction matrix, number of groups, knowledge matrix (or the percentage of known facts), proximity matrix, and others. The task of completely characterize CONSTRUCT using Response Surface Methodology or RSM (Myers and Montgomery 2002, Carley, Kamneva, and Reminga 2004) becomes unmanageable due to combinatorial explosion. Suppose, for the best case, that we have 3 levels (3 different values) for each parameter in a CONSTRUCT program having a total of 8 parameters. This gives rise to *3^8 = 6,561* cells or cases. If the cells all correspond to non-stochastic variables, then the number of virtual experiments needed is *6,561* which is huge. Let's assume each stochastic cell needs 40 trials to get statistically significant results. If all the above cells correspond to stochastic variables, then that number increases to *262,440* which is gigantic. Doing *262,440* virtual experiments is difficult using the current state of computer technology.

In reality, experimenters think through and choose a few parameters and parameter values that correspond to policy questions and "common sense". The following table displays the number of cells corresponding to a typical CONSTRUCT setup.

Table 6. Number of Cells for a Typical CONSTRUCT Experiment

Parameter	Categories	Size
Number of groups	1	1 (fixed)
Number of agents	100	1 (fixed)
Knowledge size	100	1 (fixed)
Percent of known facts in the knowledge matrix	10%, 30%, 50%, 70%	4
Communication mode	Homophily, information seeking, 50/50	3
Proximity levels	20%, 50%, 70%	3

The above gives rise to *4 x 3 x 3 = 36* cells. As CONSTRUCT is stochastic, each cell needs 40 virtual experiments to get statistically significant results. Thus, the total number of virtual experiments required is *1,440* simulation trials, which is large but manageable.

The validation cases of determining the effects of homogeneous management versus heterogeneous one with the performance measure of the relative magnitude of change in average interaction probability curves are more complicated. The following table shows the number of virtual experiments needed, assuming that the interaction only has 2 levels (binary).

Table 7. Heterogeneous vs Homogeneous Management Cell Count

Parameter	Categories	Size
Number of groups	Workers, management, intergroups	1 (fixed)
Number of agents	43	1 (fixed)
Knowledge size	3045, a function of the initial interaction matrix	1 (fixed)
Percent of known facts in the knowledge matrix	Initiated by the interaction matrix at time *Time1*	1 (fixed)
Communication mode	Homophily	1 (fixed)
Initial interaction matrix, assuming binary elements, assuming no self interactions	*2 ^ (43 x 42 / 2) = 2 ^ 903*	*6.7622 e 271*

Thus probing the effects of heterogeneity or homogeneity of the interaction matrix on the relative magnitude of the change in the average interaction probability curves takes a gigantic number of virtual experiments if the interaction matrix elements are binary and the program is non-stochastic. If the element is not binary, but say can have an integer value from 0 to 20 (21 levels), and/or the program is stochastic (which it is) then the number of virtual experiments needed becomes impossible.

Experimenters, however, think through the above problem of huge number of needed virtual experiments. One solution is to focus on the change on the management part of the interaction matrix, instead of the total management and workers interaction matrix. This results in the following table. The management consists of only 4 people.

Table 8. Revised Heterogeneous vs Homogeneous Management Cell Count

Parameter	**Categories**	**Size**
Number of groups	Workers, management, intergroups	1 (fixed)
Number of agents	43	1 (fixed)
Knowledge size	3045, a function of the initial interaction matrix	1 (fixed)
Percent of known facts in the knowledge matrix	Initiated by the interaction matrix at time *Time1*	1 (fixed)
Communication mode	Homophily	1 (fixed)
Initial interaction matrix, assuming binary elements, assuming no self interactions	*2 ^ (4 x 3 / 2) = 2 ^ 6*	*64*

The above table shows that it takes 64 cells or virtual experiments to probe the effects of initial interaction matrix for the heterogeneous versus homogeneous management case. This assumes that the interaction elements are binary and the program is non-stochastic. As CONSTRUCT is stochastic, it requires *64 x 40 = 2,560* virtual experiments for the binary element case, which is perfectly manageable. However, the interaction matrix (based on the empirical Kapferer's data) can contain integer levels of interaction up to 21 levels (counting level 0). This incurs the total required virtual experiments to be *21 ^ 6 = 85,766,121* for the non-stochastic case, which is gigantic. Of course, experiments may reduce the levels to symbolic levels of “low, medium, or high” which reduced the total required cells to *3 ^ 6 = 729* cells for the non-stochastic case. This corresponds to *29,160* cells for the stochastic case, which is large, but still manageable.

The above only considers the obstacles to RSM validation caused by the large number of cells or virtual experiments needed. Another equally – if not more so – hard problem is devising a function relating the independent variables to the performance measure. As the performance measure for the above example are in the form or relative magnitude between curves (the curves are an emergent property which changes little for small changes in the interaction matrix), the direction of changes may be diluted by random noise in the system. Indeed, for the heterogeneous management case, the curves of management and of workers are judged to be the same even though they differ by a non-zero but small percentage. It is the relative magnitude that matters. The matter is made complicated by the difficulty determining in which direction to descent on the response surface, due to the fact that homogeneity is an abstract property of the interaction matrix elements.

Of course, experimenters may reduce the needed processing to the extreme by inferring that only interaction matrices representative of homogeneity and heterogeneity need to be probed. This is exactly what WIZER does. WIZER enhances the thinking through and the use of "common sense" that experimenters employ further by adding knowledge representation and knowledge-based and ontological reasoning. It codifies the symbolic thinking and converts "common sense" into computer operable rules. This codification makes computer inferences possible. No all parameters and/or parameter values combination should be probed. Extreme points may have to be probed to check the robustness of the model, but not all immediate points have to be probed. Without knowledge, WIZER degenerates to having to deal with the same number of virtual experiments or cells as RSM does. With knowledge, WIZER only needs 2 virtual experiments for the non-stochastic case and $2 * 40 = 80$ virtual experiments for the stochastic case which is the CONSTRUCT program. Sampling the surrounding area around the two cells for statistical robustness is an option, but not a requirement. WIZER makes the probing of the effects of homogeneity and heterogeneity of the initial interaction matrix perfectly manageable.

7.6 Summary

WIZER has partially validated CONSTRUCT. It shows that CONSTRUCT is valid with respect to the average interaction probability knowledge, using Kapferer's empirical average probability of interaction data. It also shows that CONSTRUCT is valid with respect to the general trend and the relative size of the change in the probability of interactions among workers, among management, and between workers and management. Finally, WIZER is shown to be able to construct a simple hypothesis (what if the management is heterogeneous) from its ontology using ontological reasoning, and test it successfully. In the process, WIZER gains new chunks of knowledge. Here, WIZER is also shown to be able to reduce the search space significantly, by simply examining the "heterogeneous" variable value in knowledge space, which has many manifestations in the management interaction matrix. Instead of the brute-force examination of all the manifestations of heterogeneity in the management interaction matrix, an examination of one or at most several samples (not all) of them is sufficient.

Chapter VIII: Strengths and Weaknesses of WIZER

This chapter talks about the strengths and weaknesses of the current WIZER implementation. This includes the comparisons of WIZER against the Subject Matter Expert approach and Response Surface Methodology.

8.1 The Strengths of WIZER

WIZER is a general knowledge-based and ontological simulation validation and model-improvement tool. It has the following advantages:

1. Unlike formal methods, WIZER can validate simulations against empirical data and knowledge. The results from several validation scenarios indicate that WIZER can be used to improve simulation models by perturbations in the model description. The perturbations are guided by ontological and knowledge inference.
2. The models and rules in WIZER are relatively easy to specify and use. The difficulty is at the programmer level, not at the expert level. The technical difficulty requiring an expertise at the computer scientist level and the resulting high cost in time and resources hinder the adoption of formal methods for software verification.
3. Unlike statistics, simulations validated by WIZER are more precise and require fewer assumptions. Instead of assuming the abstract notion of "sample", simulations can represent entities closely, in detail, with symbolic information. Moreover, they do not assume a normal distribution and random sample.

4. WIZER can understand simulation outputs (e.g., curves) semantically and ontologically. It can also understand simulation inputs, occurrences, and empirical data semantically and ontologically.
5. WIZER can reduce the amount of search needed for validation.
6. WIZER can focus the search to the relevant area of the search space.
7. WIZER can assist in closing the loop of modeling, simulation, inference, and experiment design.
8. WIZER does model perturbations avoiding pure rule-based systems. WIZER's rules are derived from and tied with the model. While heuristics can be used, the rules can encode deep knowledge.

8.2 The Weaknesses of WIZER

As a tool, the currently implemented WIZER has the following weaknesses:

1. It has no experiment design module. The experiment design module can be constructed utilizing ontology/semantics and causal rules. It is an extension of the model-improvement module, with hypothesis building and experiment design construction using ontology/semantics and causal rules added.
2. It has a limited, if powerful, mode of inference in the form of forward-chaining and ontological reasoning. There is a need for the research into more sophisticated reasoning, cognitive, and/or machine learning techniques to enhance WIZER.
3. It has minimal control of statistical tools. What is needed is the extensive ontology or semantics that understands statistical and mathematical tools (and concepts) and facilitates the use of them. WIZER currently implements only a rudimentary understanding of some statistical routines. OpenMath and OWL Full are a good starting point for the creation of extensive ontology for calculus, statistics, geometry, and other mathematical concepts and tools.
4. It has no simulation control. What is needed is a simulation control module which is capable of halting the simulation once the result is obtained, i.e., an interactive

module based on simulation, knowledge inference, and human input. This module should also be capable of interactive simulation mode.

5. It does not learn, except in the sense of search and hypothesis building. Machine learning and causal learning from data can be added.
6. It still requires the validation of its knowledge bases. A tool to validate knowledge bases automatically with empirical data is needed.
7. Related to (6) is the issue of how precisely to weigh and assess knowledge against data, if the two are in conflict with each other. An ontology or semantic construct to do this is needed.

Except for the last three points (points 5, 6, and 7), the above weaknesses are not conceptual. They are implementation issues.

8.3 WIZER and Subject Matter Expert Approach

In VV&A, subject matter experts evaluate the validity of the simulations. Subject matter experts have the expert insights, experience, and knowledge for the task. They are however prone to the pitfalls such as cognitive limitation (especially with respect to complex large simulations), judgment biases, and implicit decision making. WIZER promotes clarity, transparency, and reproducibility. The following table summarizes the capabilities or features of subject matter experts versus WIZER.

Table 9. Subject Matter Experts versus WIZER

Feature	Subject matter experts	WIZER
Learning	Yes	No, except search and hypothesis building & testing
Large problem handling	With difficulty	Facilitated
Multiple domain integration	Difficult, by Delphi method	Facilitated
Intuition and insight	Yes	No
Transparency	With difficulty	Yes, with grounded semantics and empirical underpinnings
Clarity	Difficult for large problems	Yes
Implicit biases	Yes	No
Knowledge level	Expert level (deep knowledge)	Intermediate (ontological reasoning and rules)

Instead of working in isolation, subject matter experts and WIZER can work in synergy. This results in better and deeper knowledge, encoding of intuition, and learning for WIZER, and in transparency, clarity, and large problem solving capabilities for subject matter experts. The trend of computational and inferential help is evident in science, where the use of computational resources in the form of cyber-environments and packaged data mining/machine learning modules for scientists has increased.

8.4 WIZER and Response Surface Methodology

Response Surface Methodology (RSM) is a set of statistical and mathematical techniques for developing, improving, and optimizing processes (Myers and Montgomery 2002). The applications of RSM are in situations where several input variables potentially influence some performance measure or quality characteristic of the process. As a simulation model can be thought of as a mechanism that turns input parameters into outputs, it can be approximated with RSM. The performance measure or quality characteristic is called the response or the yield. The input/process variables are known as independent variables. The response surface methodology includes (Carley, Kamneva, and Reminga 2004):

1. Experimental strategy for exploring the space of the independent variables,
2. Empirical statistical modeling to develop an appropriate approximating relationship between the independent variables and the yield,
3. Optimization methods to find independent variable values that produce desirable values of the yield.

RSM can be used for validation but the resulting state space is large, which is then explored using Monte-Carlo, simulated annealing, and steepest ascent methods. RSM is a mathematical method, in contrast to WIZER which is a knowledge-based method. It screens what independent variables are important, builds a first-order model to get close to the optimum, and then builds a second-order model (or a higher-order polynomial one) near the optimum to get an accurate response surface. More details on how RSM is used for validation can be found in (Carley, Kamneva, and Reminga 2004). The following table contrasts RSM with WIZER.

Table 10. Response Surface Methodology versus WIZER

Feature	Response Surface Methodology	WIZER
Operation	Mathematical	Knowledge-based
Search or optimization	Simulated annealing and steepest ascent	Knowledge inference
Large problem handling	Not able to	Facilitated
Local minima	Can get trapped with no means of escape	Depends on knowledge inference. Knowledge inference can lead to escape from local minima
Smoothness of surface	Requires some smoothness of response surface	No requirement for smoothness of response surface. It can be jagged.
Computational burden	High, most states must be probed	Intermediate, knowledge-inference allows focus of search
Semantics correspondence of search steps	Very low (e.g., what a steepest ascent step means semantically is often not clear)	High, as it is knowledge-based
Causal processing	No	Yes
Critical parameters	Must be known *a priori*	Can be inferred
Parameter variation	Varies continuously throughout the experimental range tested	Varies non-continuously or continuously, according to knowledge inferences
Use of good statistical principles	Deficient	Yes
Handling of time-variant and dynamic response	With difficulty	Facilitated

In RSM, the surface of response represents the search space to find optimum solutions. WIZER adds to the surface constraints and information based on knowledge and ontology of the problem. Due to this additional knowledge, numerical gradient ascent on the surface is assisted with knowledge about the local surface area. The sampling strategy/choice of local points on the surface helps to determine the gradient and is guided by knowledge inference. Absent smooth surface, WIZER helps the gradient ascent to fly over to other "hills". Local maxima (or minima) can be avoided or tunneled through (or bridged over) by knowledge and ontological inference. In effect, WIZER acts as if it is a symbolic "response surface" method.

8.5 WIZER and Sensitivity Analysis

One of the simulation goals is to determine how changes in the input parameters and simulation variables affect the output variables, in other words, how robust the output is with respect to changes or even violations in input variables. Sensitivity analysis (Clement and Reilly 2000, Breierova and Choudhari 2001) is a procedure to determine the sensitivity of the outputs to changes in input parameters. If a small change in a parameter results in relatively large changes in the outputs, the outputs are said to be sensitive to that parameter. This may mean that the parameter has to be determined very accurately or that an alternative has to be sought to get low sensitivity. Sensitivity analysis is numerical. WIZER does what can be viewed as symbolic sensitivity analysis or knowledge sensitivity analysis, as it probes the changes in the knowledge space in addition to the simulation/parameter/numeric space.

8.6 WIZER and Influence Diagram

An influence diagram (Clement and Reilly 2000) is a simple visual representation of a decision problem. Influence diagrams offer an intuitive way to identify and display the essential elements, including decisions, uncertainties, and objectives, and how they influence each other. It includes the decision node, the chance node, the objective node, and the compute (general variable) node. Influence diagrams offer visual aids for humans to construct a correct model. WIZER, on the other hand, offers causal diagrams in the form of rules and ontologies for computers to process automatically to aid humans in the validation and improvement of a model.

8.7 WIZER and Simulation Systems

Input of WIZER includes simulation model and knowledge bases and ontologies tied to the model. Each simulation should be accompanied by knowledge bases and inference, and validated. This knowledge-integrated simulation facilitated by WIZER allows us to reason with simulation aid (to reason via simulations and virtual experiments), instead of just reasoning logically or probabilistically (statistically). Simulation-based inference is made feasible through WIZER. Moreover, once the validated simulations are used to construct and test hypotheses against empirical data, the knowledge bases and ontologies can be updated or learned. Instead of Bayesian Artificial Intelligence, simulation-based Artificial Intelligence is clearer and more accurate. Instead of integrating symbolic and subsymbolic/connectionist systems like what ACT-R model does (Anderson et al. 1997), here symbolic (knowledge-based) and simulation systems are integrated.

8.8 WIZER and Knowledge-based Systems

WIZER grounds knowledge-based systems through validated simulation against empirical data. The validated simulation emulates processes and mechanisms of the real world. Inference, ontology, knowledge bases, and simulation are tied with each other. This differentiates WIZER from conventional knowledge-based systems such as Cyc (Lenat and Guha 1990). Cyc has the brittleness of knowledge-based systems due to its pure logic foundation even though it has been fed a massive amount of facts and rules.

8.9 Quantitative Metrics

In order to show the differences between WIZER and RSM, quantitative metrics are devised. These metrics include the size of the search space and the focus on the relevant portion of the search space. The values for these two metrics are determined for each validation case. The following table shows the quantitative comparison of WIZER and RSM for the CONSTRUCT Validation Scenario III.

Table 11. Quantitative Comparisons of WIZER and RSM

	WIZER	**RSM**
Size of search space	*2 x 40 = 80*	At least *2 ^ (4 x 4 / 2) x 40 = 256 x 40 = 10,240*
Focus quality	100%	At most *2 / 256*

As shown, the size of search space for RSM is at least *2 ^ (4 x 4 / 2) = 256*. This is because there are 4 persons in management and they interact with other symmetrically including with self, assuming the interaction is binary. (If they do not interact with self, then the number of connections becomes *4 x 3 / 2*.) The reason why it is "at least" *256* is that for the minimum case the interaction matrix is assumed to be binary. The number of possible states or cells or virtual experiments is $2^8 = 256$. In reality, the interaction matrix can contain any non-negative integer elements. Thus the size of search space for RSM is usually much larger. The focus quality for RSM displays the ratio between the necessary search (two for WIZER, because one can take just one sample for each symbolic category) and the size of search space of RSM. As CONSTRUCT is stochastic, the size of search space in the table was multiplied by 40 to get statistically significant results.

The reason why WIZER is able to reduce the size of the search space is that the ontological reasoning shows that the type of management can be either homogeneous or heterogeneous. It is also because the fact that it does not really matter what permutation is in the interaction relationships amongst management, as they can be characterized as either homogeneous or heterogeneous for the "what-if" scenario question. If we would like to examine deeper questions such as what output a particular configuration of interaction matrices would predict, then the size of search space changes.

For complete validation of BioWar and CONSTRUCT, naïve RSM implementation is intractable, as shown in the following table. This table gives estimates based on a best-case estimate of the number of parameters of complete BioWar and CONSTRUCT. In this estimate, BioWar has 200 parameters, while CONSTRUCT has 10. It is also assumed that each parameter has 3 value levels, for the optimistic case.

Table 12. Number of Cells for Naïve RSM

Simulation Engine	**#Cells for WIZER**	**#Cells for Naïve RSM**
BioWar	*O(200 N) = O(N)*	3^200 = 2.6561 e 95
CONSTRUCT	*O(10 N) = O(N)*	3^10 = 59,049

WIZER does not perform brute-force search on all parameter values. Its search steps are guided by inferences on parameters. They go from a parameter value to another. Furthermore, the change in parameter value can be discontinuous when the inference dictates so. If each parameter is probed *N* times by WIZER and the number of parameters is *P*, then the total number of search is in the order of *O(NP)*. As BioWar and CONSTRUCT are stochastic, each cell needs 40 simulation trials to achieve statistical significance. Thus the numbers of total simulations are 40 times higher than the numbers of cells as shown in the above table.

Of course, in reality no one does Naïve RSM except for small problems. Experimenters reduce the number of "core" parameters to consider based on sensitivity analysis, policy consideration, and judgment calls. Section 6.7 (particularly Table 4) and Section 7.5 (particularly Table 7) describe typical and non-naïve RSM validations of BioWar and CONSTRUCT. The following table summarizes the number of cells needed for the typical validations.

Table 13. Number of Cells for Typical RSM

Simulation Engine	**#Cells for WIZER**	**#Cells for Non-Naïve RSM**
BioWar (incidence factors case)	*O(3 N) = O(N)*	*36*
CONSTRUCT (heterogeneous management case)	*O(N)*	*64*

As shown, the number of cells for WIZER depends on the number of parameters considered: for BioWar it is 3 parameters (ailment effective radius, ailment exchange proximity threshold, and base rate), for CONSTRUCT it is 1 parameter (initial interaction matrix). The number of parameters is smaller for a typical case (submodule) of validation (the above table, Table 13) than for a complete validation (Table 12) as only subsets of parameters space and of model are considered. Due to the experimenter's pruning of the total number of “core” variables, doing RSM is feasible while tedious for parts of BioWar and CONSTRUCT. This is a divide-and-conquer approach. Because of the stochasticity of BioWar and CONSTRUCT each cell requires 40 simulation trials to get good statistical significance. This means the number of simulations using RSM for the above typical BioWar validation is *1,440* simulation trials. For CONSTRUCT, the number is *2,560* simulation trials. WIZER encodes the knowledge about how and why “core” variables should be chosen in a format that computers understand and can process automatically.

8.10 WIZER among Other Network Tools

As a knowledge-based and ontological reasoning tool, WIZER can be used to augment other simulation and analysis tools. Existing network tools for dynamic network analysis include AutoMap, ORA (Organizational Risk Analysis), and DyNet. The tools function as follows:

- AutoMap: performs network relationships extraction from textual data.
- ORA: performs statistical analysis on dynamic networks data.
- DyNet: performs simulation of dynamic networks.

WIZER can interface with DyNet to add knowledge-based and ontological reasoning to the simulation of dynamic networks. Through Alert WIZER, WIZER can augment the ORA statistical analysis with ontological reasoning. The following figure shows the interconnections between tools.

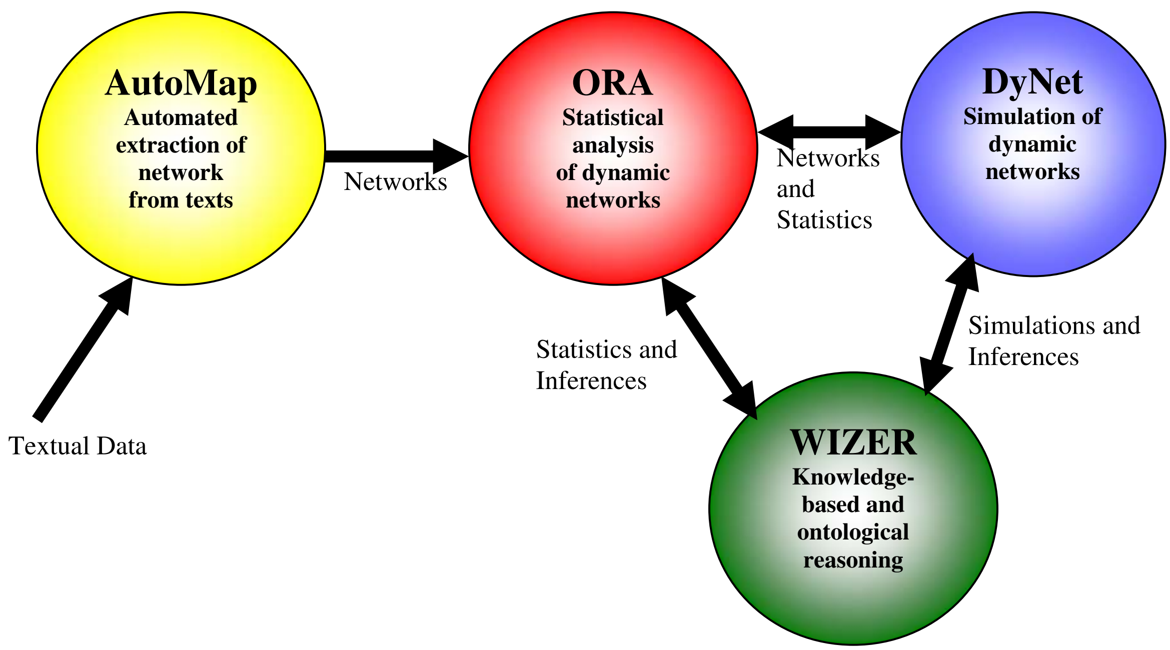

Figure 19. WIZER Working Together with ORA and DyNet

As shown, WIZER performs inferences on DyNet simulations. The inferences can be for validation and model-improvement purposes or for scenario analysis purpose. The inferences are used to guide DyNet simulations. WIZER symbolically and ontologically characterizes the statistical analyses of ORA through Alert WIZER. The resulting symbolic knowledge is then used for reasoning by WIZER. The inferences that result from this reasoning can be used to guide ORA statistical analysis.

8.11 What WIZER Gains

The following table shows what WIZER gains when used for BioWar and CONSTRUCT. The gain is compared against what normally transpires when humans do the validation. The numbers are estimates based on simulation and validation experience. The time it takes for WIZER (and the speed of WIZER) depends on computer speed, memory, and storage capacity. Being a piece of software, everything in WIZER is obviously limited by computer capabilities.

Table 14. WIZER versus Human Validation Gains

Aspect of Validation	**BioWar by human**	**BioWar by WIZER**	**CONSTRUCT by human**	**CONSTRUCT by WIZER**
Time to generate input data	Days if not weeks, due to the data access rights, usage policy, non-disclosure rules, privacy concerns, data ownership rights, and other problems.	Days if not weeks, and longer than what it takes if done by human, as the data needs to be formatted and prepared for computer processing	Days	Days and longer that what it takes if done by human, as the data needs to be prepared for computer processing
Number of points in response surface that can be estimated	1 per 10 minutes	20 per 10 minutes	1 per 10 minutes	20 per 10 minutes
Ability to handle qualitative data	Poor	Good, by mapping it to numerical range with added semantics	Poor	Good, by mapping it to numerical range with added semantics
Ability to compare means	10 comparisons a minute	Many more comparisons (>600) a minute, limited only by computer speed	20 comparisons a minute	Many more comparisons (>1200) a minute, limited only by computer speed

Ability to compare standard deviations	5 comparisons a minute	Many more comparisons (>300) a minute, limited only by computer speed	10 comparisons a minute	Many more comparisons (>600) a minute, limited only by computer speed
Number of data streams	One data stream examination per 15 minute	Many more data stream examinations (>15) per 15 minutes, limited only by computer speed	One data stream examination per 15 minute	Many more data stream examinations (>50) per 15 minutes, limited only by computer speed
Knowledge management	Difficult	Facilitated	Difficult	Facilitated
Number of rules processed	One per 5 minutes	300 per 5 minutes	One per 5 minutes	300 per 5 minutes
Number of causal relations considered	One per 5 minutes	300 per 5 minutes	One per 5 minutes	300 per 5 minutes
Common sense in selecting core variables	Implicit but good, depending on experience	Explicit and computer operable	Implicit but good, depending on experience	Explicit and computer operable
Use of statistical tools	Depends on experience	Encoded in the inference	Depends on experience	Encoded in the inference
Documentation of inference and experiment steps	Need extract work	Included in the inference trace	Need extra work	Included in the inference trace
Ability to explain simulation results	Depending on experience	Part of inference trace	Depending on experience	Part of inference trace
Enforced precision	No	Yes	No	Yes
Enforced clarity	No	Yes	No	Yes
Intuition	Yes	No	Yes	No
Learning	Yes	No, except for a rudimentary hypothesis building and testing	Yes	No, except for a rudimentary hypothesis building and testing
Model building capability	Depending on experience	No, only a basic model improvement ability	Depending on experience	No, only a basic model improvement ability
Thinking outside	Depending on	No	Depending on	No

the box?	intelligence		intelligence	
Man-hours	Large	Medium-to-Large	Large	Medium
Retention of knowledge	Depends on personnel	Facilitated	Depends on personnel	Facilitated
Large problem solving	Possible, e.g., by careful analysis	Facilitated	Possible	Facilitated
Policy scope taken into account?	Yes, written	Yes, encoded and processable by computers	Yes, written	Yes, encoded and processable by computers
Ability to handle quantitative data	Good, assisted by computers especially for large numbers, complex equations, and extensive networks	Yes	Good, assisted by computers	Yes
Visualization of the data	Need computer assistance	Not implemented yet, but feasible	Need computer assistance	Not implemented yet, but feasible
Exception handling	Good, depending on experience	Must be and can be encoded	Good, depending on experience	Must be and can be encoded

8.12 Summary

This chapter talks about the strengths of WIZER which include the capability to reduce and narrow the search for the purpose of validation. It also talks about WIZER weaknesses which include the lack of model/causal learning from empirical data. It gives comparisons of WIZER against the RSM and against subject matter experts approaches. The usability of WIZER among the existing social networks tools of ORA and DyNet is outlined.

Chapter IX: WIZER from a Computer Science Perspective

This chapter talks about WIZER from a Computer Science and Artificial Intelligence perspective. WIZER is a knowledge-based and ontological reasoning system for the validation and model-improvement of simulations.

WIZER advocates the centrality of hypothesis formation and testing in reasoning systems. In Computer Science and Artificial Intelligence, the task of mimicking scientist's work is relegated to a subfield of scientific discovery. The hypothesis formation and testing is not recognized as the one of the most important reasoning methods. (Bayesian networks have hypothesis formation but only in the sense of Bayesian conditionals.) Additionally, causal and ontological reasoning is important. Underlying causal and ontological reasoning is process reasoning/logic.

If the history of science could be a guide, the scientific progress depends on hypothesis formation and testing – in addition to observation. While reinforcement learning, case-based reasoning, genetic algorithm, first-order logic, second-order logic, Bayesian networks, and other reasoning methods in Artificial Intelligence are useful, they are not employed in scientific work as the primary method. Inductive reasoning employed in scientific discovery is one exception. In order to reason more effectively however, deductive logic and probability theory are more definitive than inductive reasoning.

9.1 Process-based Logic

Logic is an attempt to describe normative or correct reasoning. It includes propositional logic, predicate (first-order) logic, second-order logic (e.g., situation calculus), and causal logic. Logic depends on the correctness of the premise and the entailment operator to derive a correct conclusion. Any error in the assessment of the premise and the entailment results in an incorrect conclusion. Compounding of premise variables also complicates the derivation of a correct conclusion.

In the real world, logic must be based on reality. People do not reason in a vacuum. There are always entities with properties and behaviors, relationships, and processes. Without real contexts, the logical inference can be made to deduce anything. Causal logic, the part of the logic, is a logical formalism closest to reality. Underlying causal logic is descriptions about processes and mechanisms. There is a chasm between Computer Science and natural sciences like physics. In physics, researchers focus on underlying processes and mechanisms. In Computer Science, researchers focus on logic, representation, and algorithms (including control and vision algorithms in robotics).

A new kind of logic provides a foundation for propositional, first-order, and second-order logic. This logic is called process-based logic or process logic, as it describes processes and mechanisms instead of just truth values, predicates, functions, and causality. This logic augments the premises, the entailment operator, and the conclusions with process and entity descriptions. In creating process logic, processes are modeled and then augmented with semantic information and ontology. Modeling processes and entities with properties and behaviors can be effectively done with simulations. Thus simulations capture the structures of the real world for logical reasoning. Augmented with ontology and knowledge base (which is to say, process ontology), the process logic is reflected in simulation. The following figure illustrates the relationships between conceptual model, implementation, process logic describing processes/mechanisms, causal logic describing causal relations, and if-then rules describing process-based and thus conceptual-based changes to the model and parameter values. The arrows in the picture represent the notion of “is derived from”.

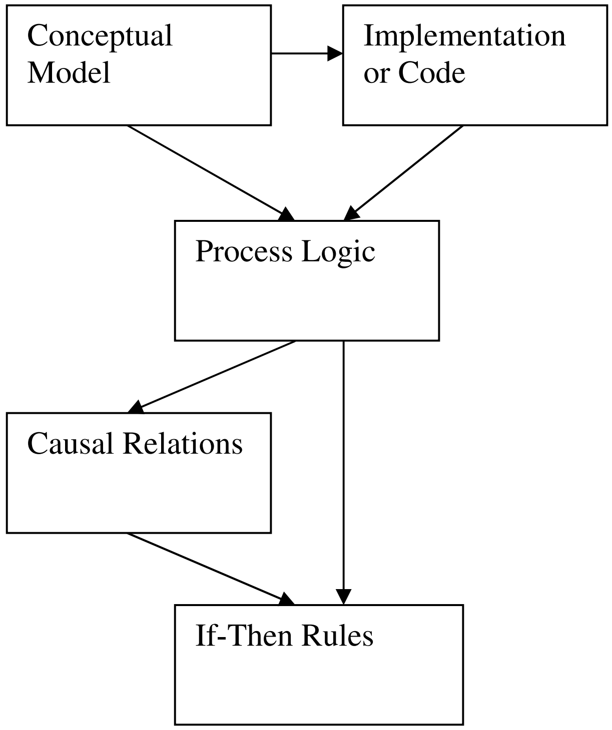

Figure 20. Process Logic and Its Derivation

Process logic denotes the change and the processes of change from one entity (or one entity value) to another in the conceptual model and/or in the implementation. It starts out with process model. Augmenting the process model with symbolic and semantic information relevant to the model produces the process logic. By definition, process logic is the sequences or ordered events based on the process model augmented by semantic information and ontology.

Abstracting the process logic using human-friendly causal language is causal relations. Causal relations abstract the thoughtless change to a meaningful semantics of causes and effects. The if-then rules describe the adjustments of the values of the variables in the causal relations and/or process logic based on empirical data. The rules are tied to the causal relations and/or process logic. The code/implementation can be in the form of simulations.

As an example, let us examine the process model and the process logic for smallpox. Smallpox has the incubation period, the initial symptom (prodome) which lasts 2-4 days, early rash for about 4 days, pustular rash for about 5 days, pustules and scabs for about 5 days, resolving scabs for about 6 days, and then finally resolved scabs. The process model for smallpox is illustrated in the following figure.

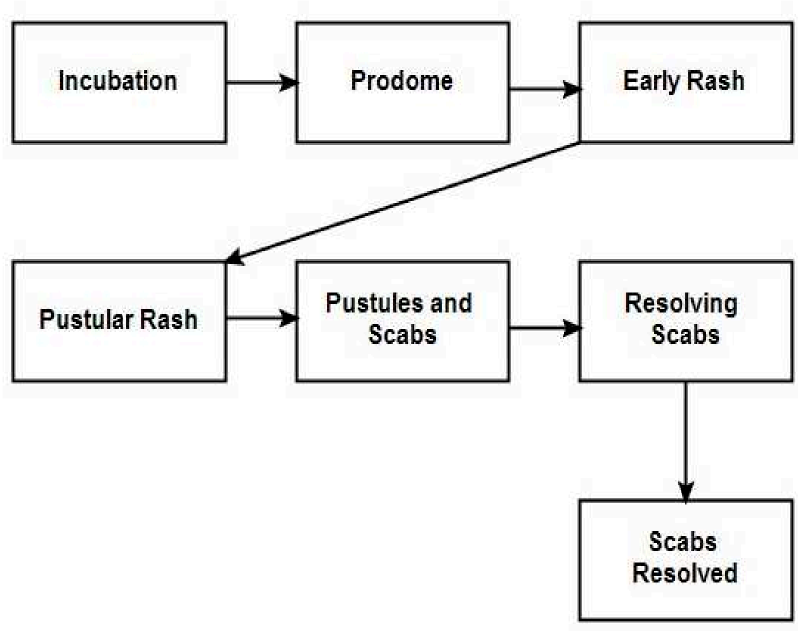

Figure 21. Process Model for Smallpox

As shown, smallpox progresses in roughly an orderly sequence of events.

The process logic based on the process model can be written by using the modified N3 notation (with the addition of the sequence primitive) as follows.

<sequence> <begins> <null> .

<sequence> <based on> <periods> .

<period> <numbers> <1> .

<period> <is> <incubation> .

<period> <has length of> <7 to 17 days> .

<incubation> <is> <non contagious> .

<incubation> <has symptoms of> <null> .

<period> <numbers> <2> .

<period> <is> <prodome> .

<period> <has length of> <2 to 4 days> .

<prodome> <is> <contagious> .

<prodome> <has symptoms of> <fever, malaise, headache, body ache,

vomiting> .

<period> <numbers> <3> .

<period> <is> <early rash> .

<period> <has length of> <about 4 days> .

<early rash> <is> <most contagious> .

<early rash> <has symptoms of> <red spots on the tongue, red spots in the mouth, rash everywhere on the body, reduced fever, rash becoming bumps, bumps filled with a thick opaque fluid with bellybutton-like depression in the center, fever rising again> .

<period> <numbers> <4> .

<period> <is> <pustular rash> .

<period> <has length of> <about 5 days> .

<pustular rash> <is> <contagious> .

<pustular rash> <has symptoms of> <bumps becoming pustules> .

<period> <numbers> <5> .

<period> <is> <pustules and scabs> .

<period> <has length of> <about 5 days> .

<pustules and scabs> <is> <contagious> .

<pustules and scabs> <has symptoms of> <pustules starting to form a crust, scabs> .

<period> <numbers> <6> .

<period> <is> <resolving scabs> .

<period> <has length of> <about 6 days> .

<resolving scabs> <is> <contagious> .

<resolving scabs> <has symptoms of> <falling scabs> .

<period> <numbers> <7> .

<period> <is> <resolved scabs> .

<period> <has length of> <an instant> .

<resolved scabs> <is> <noncontagious> .

<resolved scabs> <has symptoms of> <all scabs gone> .

<sequence> <ends> <null> .

While not shown in the above example, the sequence also allows the specification of the decision flow in the form of “if-then-else”. The sequence for the process logic is implemented as an ordered traverse in (the semantic networks of) simulation knowledge space and domain knowledge space.

9.2 Probabilistic Logic

Probabilistic logic, the intersection of probabilistic reasoning and logical representation, has become an active research area in Artificial Intelligence. The research in probabilistic logic pursues the integration of deductive logic and probabilistic reasoning. The brittleness of symbolic logic (e.g., first-order logic) lends to the choice of statistics – particularly Bayesian statistics – to tackle Artificial Intelligence problems. The statistical paradigm, however, has an inherent weakness of being unable to support the domain and/or structured knowledge and the wealth of inferences in logic. The view behind the probabilistic logic research in Artificial Intelligence is that logic and probability are enough for representing the real world. (A related subarea called probabilistic logic learning looks at how to learn the logical and probabilistic formalisms and values using machine learning.)

WIZER points to what is missing in this view: the importance of modeling and simulation, the need to focus on natural processes instead of just pure logic, and the significance of hypothesis formation and testing. Augmented by causal, process, and ontological reasoning, WIZER supplies knowledge structure for statistics through simulation models and validated simulations. It provides robustness for logical reasoning in the form of statistical calculations constrained by simulations (after simulation validation with empirical data).

9.3 Logic, Probability, and Structure of the World

The majority of work in Artificial Intelligence focuses on devising smart representations and algorithms to mimic part of human intelligence. Simulations are not considered an essential part of this endeavor. Simulations have great successes in mimicking complex systems. Consequently, simulation – and simulation modeling – is a great way to represent systems. Expert systems, while being part of Artificial Intelligence, are also researched separately from simulations.

Artificial Intelligence research went through several phases throughout several decades: symbolic logic phase in the 70s and 80s, connectionist phase in the 90s, genetic algorithm phase in the 90s, and probabilistic/statistical phase during this decade (the 2010s). As the time of this writing however, there is a revival of the trend toward knowledge-based methods especially for the Semantic Web.

Current work in logic is addressing problems such as the brittleness of first-order symbolic logic. Recent statistical/probabilistic phase – especially Bayesian statistics – is the evident of the unfulfilled promise of symbolic logic. The failed Japanese Fifth Generation Computer Systems project and the lukewarm Cyc project illustrated the difficulty of scaling up symbolic logic and of making logic not brittle. Probability and statistics however cannot handle well the structures of knowledge and the inferences of logic.

Logic, while powerful, derives its power from accurate representations of the world. As an example, while biologically a cat is a mammal, the correct first-order logic declaration in the context of society is that a cat is a pet. Statistics, while powerful and robust, does not form an accurate representation of the world and cannot handle symbolic information well. The structure of the world and the structure of the knowledge about the world cannot be represented by statistics. For this, we need simulations. Modeling and simulation can mimic the real world closely. It can mimic complex processes. This indicates that to be successful in achieving real-world logical reasoning, it is necessary to have simulation as an essential component in addition to logic and statistics. The empirical view of the world suggests that it is the – empirical – process that is fundamental, rather than logic. Validated simulations mimic real world processes. WIZER thus facilitates the connection between statistics and logic through validated simulations.

Instead of logical reasoning, the simple but profound scientific process of hypothesis building and testing – the scientific method – is fundamental. While logic is utilized in hypothesis building, knowledge accumulation of science is achieved by carefully constructing and testing hypotheses. If logic is used without the empirical check of hypothesis testing, the inference may look valid but it is empirically wrong. Both the premise and the inference rule must be empirically correct to allow empirically valid inference. Logic also depends on propositions being true or false. Attempts at multi-value logic and fuzzy logic have not produced sound reasoning formalisms. Here WIZER also facilitates the construction of hypotheses and testing of hypotheses in simulations as a proxy to the real world. It provides an empirical foundation through validated simulations on which logical reasoning is based.

9.4 Empirical Path toward Artificial Intelligence

The field of Artificial Intelligence has attempted to mimic human intelligence for at least five decades. The approaches to achieve artificial intelligence include logical, connectionist, and statistical (Bayesian) approaches. Outside scientific discovery, however, little attention is paid to the fact that human scientists gather knowledge by hypothesis generation and testing, which is to say, by the scientific method. Without the concept of falsifiable and testable hypotheses of the scientific method, the acquisition of new knowledge has been slow and error-prone. Science focuses on elucidating entities and processes/mechanisms, not just logical entailments. Thus, it may make sense to focus on processes/mechanisms to achieve artificial intelligence. I call this the empirical path toward artificial intelligence. Simulation is one of the most appropriate tools to mimic processes/mechanisms (the other being mathematics). It may take validated simulations with the capability of building and testing hypothesis for simulation model improvement to achieve artificial intelligence. While logic can represent other formalisms, simulations have the virtue of being able to add robustness through its statistical computations tied to the simulation model.

We live in the era of data rich and knowledge/inference poor in many scientific fields, especially in economics, business, and bioinformatics/computational biology. Data are inexpensive. From data, causal model can be constructed by causal learning/discovery algorithms. Simulation/process models can be improved by hypothesis building and testing.

9.5 Summary

This chapter shows that validated simulations, the result of WIZER, can function as the connector between statistics and logic as validated simulations represent the structures of the real world closely and add robustness to logical reasoning through the statistical computations tied to the simulation model. Structured knowledge and statistics can be captured in simulations and be made operable. This allows robust logical reasoning (including causal and process reasoning). As this era is blessed with rich data, high-fidelity simulations are feasible (validated with rich data and knowledge). Using machine learning and data mining techniques, knowledge can be learned and/or extracted from data.

Chapter X: Causality, Simulation, and WIZER

Causality is an important concept for humans and other living beings. Whether the real world is causal is debatable (quantum mechanics is an excellent example of non-causality). Underlying causality are physical processes and mechanisms. In physics, the fundamental laws of nature are expressed in continuous systems of partial differential equations. Yet the words and concepts that are used to talk and reason about causes and effects are expressed in discrete terms that have no direct relationship to theories of physics. This chapter describes the state of the art of causal modeling. It advances validated simulations through WIZER as a better method to do causal modeling, inference, and analysis.

10.1 Causal Modeling and Analysis

Causality is an approximation of orderliness in the macro-level universe even though the micro-level universe underpinning it is a causation-defying quantum universe. Squirrels bury nuts for the winter. People plan daily trips to work or shop. The success of these activities does not directly depend on theories of physics, but it indicates that the world is sufficiently orderly that a rough rule of thumb can be a useful guide. Causal relations represent one of such rule of thumb.

Being able to make causal predictions about the world is beneficial, so much so that causality has become an integral part of human worldview and language. Causal relationships are even sometimes assumed as facts without any conscious thought. People form causal relationships based on perception or estimation of order or regularity in the random world. Causal relationships are not without pitfalls. People believe in many spurious causal relationships and the effect is considerable. Empirical elucidating of processes or mechanisms behind a causal relationship is needed to ascertain its correctness. In addition to causal reasoning, process-based, and empirical reasoning is crucial.

Causal relationships are modeled by directed graphs (Greenland and Pearl 2006). Causal models have been known as structural-equations models (Kline 2004) in economics, behavioral science, and social sciences, which are used for effect analysis. The causal diagrams in form of directed graphs depict causal relationships. The following figure shows an example of causal diagrams. The arrow denotes the causal dependency.

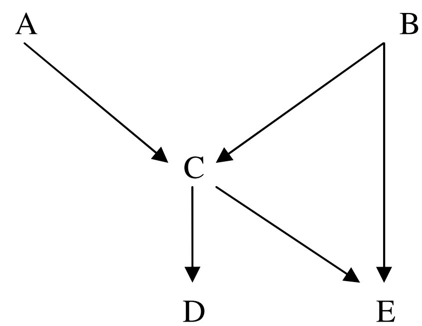

Figure 22. Simple Causal Diagram

As shown, *A* and *B* are independence, while *C* is directly dependent on *B*. *E* is directly dependent on *C* and *B*. *D* is directly dependent on *C*. *E* is indirectly dependent on *A*. The causal relations depicted above are assumed to be deterministic. But then the causal diagrams such as the above can be reinterpreted formally as probabilistic models or Bayesian network models to account for uncertainty. This is the first major advance of causal inference: from deterministic causality to probabilistic causality. The causal diagrams can further be reinterpreted as a formal tool for causal inference. This represents the second major advance of causal inference: from descriptive diagram of causality to actually use the diagram as a means to do causal reasoning.

Causal diagrams are assumed to be Markovian. Causal analysis, which deals with what inference one can draw from several causal statements, is based on directed graph, the notion of d-separation, and Markovian assumption (Pearl 2000). Causal analysis makes the initial assumptions of which variables are endogenous (to be examined in causal reasoning) and which ones are exogenous (to be assumed away as the environment or noise). Causal relations can be extracted from data by using causal Bayesian networks learning.

10.2 Causality and Process

Causal relations are constructed by humans to estimate some kind of order from physical processes. They are sometimes wrongly constructed. For example, it was wrongly believed that severe illness is caused by depression and/or anger. Without clear underlying mechanisms or processes, causality can still be useful (e.g., if causes of certain diseases are known but not the disease mechanisms inside a human body, a remedy can still be given by addressing the causes) but is risky. It is better, of course, if the underlying processes are elucidated. If the underlying processes are clear, causality is still needed to facilitate human understanding and use. This is similar to what higher-level computer language does, which is encapsulating the machine-level binary code.

10.3 Causality and WIZER

Instead of relying on directed graphs, Bayesian networks, and Markovian assumption to elucidate causality, WIZER utilizes validated simulations. Bayesian networks used to model causality in the form of causal Bayesian networks fundamentally suffer from the prior specification problem, the conditional dependence correlations, the inability to take into account the excluded middle, the disconnect with what human scientists normally do in their scientific work, the lack of knowledge and ontological inference, and the requirement for large enough samples to be meaningful. Validated simulations can depict more accurately the many variables and their potential interactions that could compound causal and/or Bayesian reasoning. They are also able to model individual-based causations and see the cumulative effects (or the emergence) on the sample populations. Validated simulations represent real world processes. Causality can be thought of as a simple search for regularity in the real world processes, resulting in an approximation or a simple rule of cause-and-effect "regularity". WIZER allows the grounding of causal relationships on processes and mechanisms as emulated by the validated simulation and on empirical data. As all causal relations are empirical, this capability of grounding inferred or conceptual cause-effect relations is important. The following table shows the comparison between graph-based and validated-simulation causality representation.

Table 15. Causality by Graph versus by Validated-Simulation

	Graph-based	**Validated-Simulation-based**
Causal relation representation	An edge in the graph	Simulated processes underlying the causal relation
Uncertainty assessment	Conditional probability with Markovian assumption	Detailed process simulation
Allow symbolic information?	No	Yes
Structured knowledge taken into consideration, other than the causal structures	Not in the probability assessment of a causal relation	Yes, including in the assessment of a causal relation
Abstract away minor factors?	Yes	Yes, but much less so
Knowledge inference?	No	Yes
Realism/believability?	Not good	Good
Exception handling	Difficult	Incorporated
Individual to population causality "emergence"	Cannot be modeled	Modeled in detail
Determination of exogenous factors	Determined *a priori*	All factors (as many as feasible) modeled and the exogenous factors are shown as having the minimal or no impacts to the causal relation

10.4 Summary

This chapter talks about causality and its graph-based modeling. It also talks how validated simulations and WIZER can supply better fidelity causal relations than causal analysis using directed graph alone.

Chapter XI: Potential Extensions and Implications of WIZER

This chapter talks about the potential extensions of WIZER. By potential extensions I mean the technological and conceptual extensions. The latter part of this chapter talks about the implications and applications of WIZER in diverse fields.

11.1 Toward a Simulation and Knowledge Web

The Semantic Web (Davies et al. 2003) is currently the next generation web. Unlike the current World Wide Web, the information in the Semantic Web is engineered in such a way to be easily processed by computers on a global scale.

As validated simulations and their semantic descriptions are made feasible by WIZER, it is now possible to use the semantic descriptions – and some additional resource-allocation ontology – to create a Simulation Web. Instead of focusing on the structures of knowledge, the Simulation Web allows the organic real world dynamics to be captured. As validated simulations imply validation knowledge, the Simulation Web produces the Knowledge Web. The Simulation Web and the Knowledge Web should be able to:

1. Ground any ontology or semantics on validated simulations based on empirical data. Ontological engineering deals with the issue of ontology construction and conflicts in ontologies. What ontology really means can be made empirical by validated simulations. This facilitates the resolution of ontological conflicts and provides an essential context and foundation on which ontologies are built on.
2. Examine any data critically through validated simulations.
3. Intelligently extract knowledge from validated simulations.
4. Distribute simulation tasks over the Internet based on semantics or ontology.

5. Perform not only logical inference but process-based and empirical-based inference.
6. Produce in-depth knowledge or knowledge grounded in empirical reality.

The modified N3 notation adopted for Simulation Description Logic of WIZER incidentally shows it is not conceptually difficult to interface simulations with the Internet. The simulation only needs to be ontologically described with appropriate knowledge bases and inference mechanisms. Once the ontology is tied with the simulation, the N3-like description of simulations and of simulation results can be shared through the Internet. More sophisticated simulation sharing includes distributing simulations by their components throughout the Internet. This would turn the Internet into one hypercomputer. The distribution of simulations is more appropriate for social systems where components are relatively loosely coupled than for fluid mechanics, for example. This is because the Internet connections incur delays which are substantial for vector or tightly-coupled applications. Issues of access rights, privacy, load-balancing, and others form intriguing research subjects.

11.2 Component-based Multi-scale Super-simulations

The integrated circuits and the automobile are the epitome of the success of component-based system building. Similar approaches may prove to be fruitful for building realistic simulations of many systems. Additionally, multi-scale components combine components from various physical scales (e.g., diabetes expression simulation with public health simulation). Related to this component-based simulation building is docking, which validates a simulation with another previously validated simulation.

In the modeling and simulation field, this composable system of systems approach for interoperability of diverse simulators is called Federated Simulation Systems. Federated Simulation Systems have the following concepts:

- A federation comprising of a collection of simulators (federates).

- Time-stamped event-based interactions between federates.
- Standardization for common objects and events.
- Scalability via parallel and distributed simulation techniques.

Current bottlenecks in integrating many simulation systems and in using simulations as components lie in the difficulty of getting the semantics and assumptions of the components to match. This spurred the work on simulation interchange/format standardization. Time-stamped events provide a primitive way for interactions between federates. The difficulty of getting the semantics right is partly caused by not making all assumptions explicit and operable. In addition, the results have not been put in consistent knowledge bases that could be automatically reasoned with. WIZER can remedy the above two issues. It can also provide a more sophisticated interaction method for federates in lieu of the time-stamped events. The simulations or simulation components will all have symbolic or semantic descriptions of them. WIZER can pave a way to the realization of component-based multi-scale super-simulations. If these super-simulations are valid and detailed enough, they may form the foundation for software and robotic systems that understand the real world. Needless to say, these systems will have immense utility.

11.3 WIZER and Knowledge Assistant

As WIZER facilitates validated simulations, the knowledge behind the simulations becomes clearer and more interactive for human users. Currently, when someone tries to find a specialized knowledge to understand what kind of materials he/she should choose to use for building his/her home, for example, he/she is forced to delve into technical papers describing the materials if he/she refuses to be guided by the commercial and advertising information alone. Reading and understanding technical papers written in technical terms for professionals is hard. Here WIZER, with its validated simulations, comes to help. Instead of simple technical papers and commercial information, “knowledge startups” will build validated simulations complete with symbolic (knowledge-based) and graphical interfaces for the end users. These validated simulators will take the form of software packages much like tax-preparation software today. In the future, when someone tries to find how best to build a home, he/she will purchase this knowledge-assistant package and use its intertwined validation simulation and knowledge inference for speedier understanding of difficult subjects. WIZER provides the foundation for such knowledge assistants.

Validated simulations can also be used as a means to communicate, augmenting video and human speech. Effective communication depends a lot on the context. Validated simulation can capture such a context. Today, communication is limited by language and cultural barriers. If one wants to communicate what it is like to be living in the real and current Costa Rica, for example, one can get some rough sense of it by reading (here is the language barrier), talking to people (language and contextual barriers), seeing pictures (limited knowledge-based explanation for them), or watching tourist movies (this kind of movie is limited and movies are non-interactive or have limited interactivity – changing several scenarios at most). One, however, cannot tailor the movies to his/her specific circumstance nor can one really get the feeling of living in Costa Rica. Simulations have been used for combat training purposes to give trainees the feel of various combat situations. WIZER through validated simulations enables more effective and detailed communication among people and across cultures.

11.4 WIZER and Ontological Engineering

People construct different and often conflicting ontologies, partly due the fact that the ontology does not exist in a vacuum (it is constrained by social and cultural contexts, for example). One way to fix this is to have empirical grounding of the defined meanings in ontologies. WIZER provides this empirical grounding through validated simulations.

11.5 WIZER and Policy Analysis

Policy Analysis uses numerous computational models, particularly economic models. Most policies and their driving politics are now governed using human languages, which are inadequate for objective, transparent, and accurate discourse and analysis, as the languages contain ambiguities and are loaded with historical, cultural, and emotional elements. This is not to say that historical, cultural, and emotional elements are not important, but they need to be explicitly noted to facilitate clear reasoning and understanding. The law witnesses the tailoring or formalization of a portion of human languages to try to eliminate ambiguities and misunderstanding, but it requires professionals to interpret them thus still leaving room for ambiguities, misunderstanding, and misapplication.

Imagine people being able to discern the policies and laws through realistic movie-like simulations based on validated models. If we read through the 396-page US bird flu plan, we are left with a sense of a good plan with nothing to worry about, but no clear idea of what would really transpire, especially on the all important questions of "What will happen to my family and me? How, where, when, from whom exactly could we get help?" Imagine people being able to walk through and play around with the bird flu plan just like playing games. This is possible through validated models and simulations, which WIZER facilitates.

Human-language plans leave too much uncertainty and ambiguity; both of which are fundamentally detrimental to the success of plans, especially ones whose success depends on individual behaviors. Plan writers consciously or unconsciously incur a positive-image bias in the plan. Imagine authorities providing people with not just written plans, but validated simulators. Besides, nobody wants, has time, or is able to read through the hundreds or thousands of pages of documents, but almost everybody likes to watch movies and play games. Validated simulations thus provide a more natural user interface (combined with 3D movie interactive presentation) to understand, analyze, and design policies and regulations.

The messy response to Hurricane Katrina in 2005 indicates that all the written texts on policies and regulations have never been validated (to see how all work with each other, for example). Validated simulations of all the policies and regulations in the context of a disaster would have made clear all the deficiencies. Thus WIZER facilitates the improvement of regulations, policies, and legislations through validated simulations.

11.6 Localization and Instantiation of Large Simulations for Decision Making

In large simulations such as BioWar, the simulations are constructed with a general set of parameters. They are developed with one or two test cases. In BioWar, for example, the simulation is developed with respect to five seed cities. By instantiation and localization, I mean the deployment of simulation to other cases: in case of BioWar, to other cities. WIZER can facilitate the parameter adjustments and the validation of simulations to instantiate and localize the simulations.

11.7 WIZER for Organization and Management

The way companies and societal systems are currently managed is based on case studies and management lessons based on human languages, with only necessitated support of computational tools. With the advent of Computational Organizational Theory and Computational Management Science, almost every aspect of organizations and management can now be modeled computationally and inferentially. For example, the management knowledge can be computationally and inferentially modeled. Business process design, operations management, and decision making do not happen in a vacuum, but within a context of organizational, legal, media, financial, societal, and technological background. In this era of globalization, electronic-commerce, and mobile-commerce, the background becomes much more a determinant of success for any business and management plan.

Organizational modeling and simulation is mostly quantitative. To improve upon the quantitative organizational modeling and simulation, WIZER contributes (symbolic) knowledge inference and validated simulation to the organizational modeling and simulation. Closely related to organizations are networks, including social networks, of which WIZER could facilitate the validation too.

On the other hand, knowledge management focuses exclusively on ontology and knowledge bases. Here WIZER contributes validated simulations to ground business/management rules on empirical data. Knowledge management includes the management of knowledge capital. WIZER facilitates knowledge management by the nature of its ability to handle symbols, numbers, and simulations.

As an organization is a knowledge entity, focusing on the nature, structure, and dynamics of knowledge in organization may shed light on organization performance problems. WIZER can assist in analyzing organization performance by looking into what knowledge resides where and how it is transformed and exchanged in organization, instead of just looking at the organizational structures, tasks, leadership, etc. The case of Enron is a good example. Enron has the same organizational structure and tasks as many other companies. Even the accounting seems to be similar to other organizations in terms of the system and the numbers. Only by carefully examining what is unusual about the knowledge in Enron and about Enron can one ascertain whether Enron is a company in a good standing or not. As an example, the knowledge about the multiplication of Special Purpose Vehicles should have triggered an alert among analysts.

11.8 WIZER and Biomedical Informatics

Biomedical informatics deals with all aspects of understanding and developing the effective organization, analysis, management, and use of information in health care. Hospital organization and care administration is complex, so much so that it is currently labor-intensive. While using standard protocols has its merits, in some cases they break down. Validated simulations can provide insights and possibly remedies to problems in the organization and management of care. Here WIZER facilitates the validation of simulations. It improves the confidence in the use of simulations, the ease with which simulations are validated and improved, and the ease with which simulation, model, and domain/empirical knowledge are managed.

Particularly urgent in biomedical informatics is finding a solution to the pervasive and persistent problem of medical errors. While training and use of standard protocols help, they are insufficient as medical errors still occur with a significant frequency. This dissertation shows an alternative way to address medical errors: by using validated simulations for systems of interest. The Agency for Healthcare Research and Quality (AHRQ) states that the single most important way to prevent errors is for the patient to become an active member of his/her health care team. This is a good advice, provided that the patient is knowledgeable and not gullible. A patient experience of having learned much information about a sports surgery before deciding whether or not to have one demonstrates that confusion still ruled and in the end the decision was made by weighting the factors such as the strength of a doctor's persuasion, trust, and a doctor's reputation. There were no clear reasoning steps before the decision; a simple random leap of faith might have played a big role. The surgery decision was partially informed, but to say it was an informed decision is an overstatement. Closely related to informed decision is informed consent. Validated simulations through WIZER with the corresponding knowledge bases and ontology can assist the patient to be knowledgeable and capable to make informed decision. More sophisticated way is to have a replica procedure using validated simulations. A departure in an actual procedure from the validated-simulation procedure should trigger a question or an alarm. A replica hospital in its entirety by validated simulations facilitated by WIZER is also possible.

11.9 WIZER and Bioinformatics/Computational Biology/Systems Biology

Recent advances in bioinformatics (Keedwell and Narayanan 2005), computational biology (Haubold and Wiehe 2006, Fall et al. 2005), and systems biology (Szallasi et al. 2006) open up exciting collaborative efforts intersecting biology, medical science, and computer science. Biology is an experimentally driven science as evolutionary processes are not understood well enough to allow theoretical inferences like what is done in physics. Quantitatively the biological systems are extremely challenging as they have large range of spatial and temporal scales, wide range of sensitivities to perturbations, incomplete evolutionary records, multiple functionalities, multiple levels of signal processing, and no separation between responses to external stimuli versus internal programs.

The computational challenge in bioinformatics, systems biology, and computational biology is immense: the complexity of biological systems includes the molecular underpinnings, the data from experimental investigations need extensive quantitative analysis, and it is not computationally feasible to analyze the data without incorporating all knowledge about the biology in question. This reinforces the sense that knowledge-based approach is needed to tame the computational complexity. Synergistic use of experimental data, computation, and domain knowledge is essential.

For simulations to be useful, they need to be validated. Conventionally, validation is done with minimal computational help. A recent successful simulation model is Archimedes, a diabetes model, which was validated semi-manually. WIZER can play a small part in bioinformatics/systems biology/computational biology by facilitating validation and knowledge management of biological simulations. A knowledge-based and ontological approach as implemented in WIZER can reduce the amount of search and the computational complexity in biological simulations.

11.10 Summary

This chapter talks about the potential extensions of WIZER to realize super-simulations, Simulation/Knowledge Web, and others. It also talks about the applications of WIZER on policy analysis, knowledge management and organization modeling, biomedical informatics, and others.

Chapter XII: WIZER Implementation and User Guide

This chapter describes the implementation of WIZER, provides information on knowledge and ontology preparation and a guide for the use of WIZER.

12.1 Code Structure

WIZER is implemented in C++, primarily because that it is intended to be runnable on a supercomputer. It does not yet have a shell similar to expert system shells. The planned shell will include both the inference and the simulation access. Based on the CLIPS[3] model, an expert system shell coded in C, it should be feasible to structure this shell to be runnable on a supercomputer.

The C++ code for WIZER follows the structure of a forward-chaining production system. Variables are encoded in a C++ structure, rules are implemented in another C++ structure with clauses containing nodes having the structure for variables.

As is currently implemented, Alert WIZER and the WIZER Inference Engine are separate programs. They can be linked, but Alert WIZER and the WIZER Inference Engine are intended to be usable in their own right.

12.2 Knowledge Configuration for WIZER

As a knowledge-based and ontological reasoning system, WIZER needs careful preparation of its knowledge bases and ontology. The inference mechanism, in the form of forward-chaining production system, is in place inside WIZER, as well as the

[3] http://www.ghg.net/clips/CLIPS.html

mechanism for conflict resolution. Knowledge, however, needs to be input into WIZER to allow useful inference and conflict resolution. Without proper knowledge, WIZER's performance degenerates. In this Appendix, I outline the steps to prepare knowledge and use WIZER.

Steps to prepare knowledge in the form of ontology and knowledge bases for WIZER include:

1. Take or create the conceptual model of the simulation.
2. Acquire the conceptual and causal models of the domain knowledge, that is to say, the empirical knowledge for validation and model-improvement. Also acquire the empirical data.
3. Create the abstract causal model from the conceptual model. This abstract causal model defines which variable influences another variable. (This abstract causal model can be thought of as the influence model, but I use the term causal model to emphasize causality.)
4. Create the concrete causal model from the abstract causal model. This concrete causal model represents how a variable with a value causes another variable having another value. The abstract and concrete causal models expedite getting to the root cause of a problem. This is similar to the use of an environmental lattice in assumption truth maintenance systems which allows perturbations to the system descriptions.
5. Create the process logic/model for each causal relation in the causal model. This process logic is closely tied to implementation code.
6. For each relevant output variable of a causal relation, create a semantic/ontological description or potential classification of the possibly dynamic output/variable.
7. Create rules based on the causal model and the process logic.
8. Create conflict resolution rules based on the causal model and ontology. The conflict is resolved by rule-based and ontological reasoning.
9. Introduce minimal model perturbation rules based on ontology and knowledge bases to describe how the value/link adjustments are to be determined. If process logic is available, it is also used to help determine how values/links should be

adjusted. The minimal model perturbation is closely related to the previous conflict resolution step.

10. For all the steps above, relevant ontologies are created and used as needed.

Once these steps are completed, WIZER is ready to run the simulation validation.

Table 16, below, lists the time it took for me to configure the model and to run WIZER for the BioWar and CONSTRUCT validation scenarios in Chapters 6 and 7. Being a program, the speed of WIZER depends on computer speed, memory, and storage capacity.

Table 16. Time for Knowledge Configuration of Testbed Scenarios

Configuration Step	BioWar (2 scenarios)	CONSTRUCT (3 scenarios)
Create a conceptual model, if it does not already exist	1 hour	1 hour
Acquire domain knowledge and data (including reformating the knowledge and data)	14 days	40 days
Create an abstract causal model (or influence model) from the conceptual model	1 hour	1 hour
Create a concrete causal model from the causal model	N/A	N/A
Create a process model for the causal models	N/A	N/A
Create semantic/ontological categorizations for potentially dynamic causal variables	4 hours	3 hours
Create rules based on causal models	7 days	4 days
Create conflict resolution rules	0.1 hour	0.1 hour
Create minimal perturbation rules for value/link adjustments	0.1 hour	0.1 hour
Create relevant ontologies for the steps above	1 hour	0.5 hour
Run the simulations	21 days	7 days
Run WIZER	1 day	1 day

Table 17 provides an estimate for the time required to perform the knowledge configuration steps and to run WIZER for BioWar and CONSTRUCT for their complete validation. The differences in the lengths of time are due to the fact that the testbeds have different conceptual structure, size, and complexity. The time is assumed to be for one person "team" and for the use of a computer server with quad-processors.

Table 17. Estimated Time for Knowledge Configuration for Complete Validation

Configuration Step	BioWar	CONSTRUCT
Create a conceptual model, if it does not already exist	7 days	2 days
Acquire domain knowledge and data	14 days	7 days
Create an abstract causal model (or influence model) from the conceptual model	7 days	3 days
Create a concrete causal model from the causal model	14 days	7 days
Create a process model for the causal models	14 days	7 days
Create semantic/ontological categorizations for potentially dynamic causal variables	7 days	7 days
Create rules based on causal models	7 days	4 days
Create conflict resolution rules	14 days	7 days
Create minimal perturbation rules for value/link adjustments	14 days	7 days
Create relevant ontologies for the steps above	30 days	14 days
Run the simulations	60 days	10 days
Run WIZER	3 days	1 day

The following table shows the level of expertise each step needs.

Table 18. Expertise Level for Each Configuration Step

Configuration Step	Expertise Level
Create a conceptual model	Knowledge modeling and domain knowledge
Acquire domain knowledge and data	Data entry
Create an abstract causal model (or influence model) from the conceptual model	Program design or software architect, with knowledge about the difference between causation and correlation
Create a concrete causal model from the causal model	Program design or software architect, with knowledge about the difference between causation and correlation
Create a process model for the causal models	Program design or software architect, with knowledge about algorithms and processes
Create semantic/ontological categorizations for potentially dynamic causal variables	Data classification and domain knowledge, with knowledge about ontology
Create rules based on causal models	Program design or software architect, with knowledge about rule-based systems
Create conflict resolution rules	Program design or software architect
Create minimal perturbation rules for value/link adjustments	Program design or software architect
Create relevant ontologies for the steps above	Program design or software architect, with knowledge about ontology
Run the simulations	Programmer
Run WIZER	Programmer

Thus, at the minimum, to configure the knowledge for and to run WIZER, four people are needed: one domain expert, one knowledge engineer, one program designer/software architect, and one programmer who can handle data entry. An advanced programmer can become a program designer and software architect with training in software modeling techniques. If the conceptual model already exists, which should be the case for most simulators, the number of persons needed reduces to two: one software architect/program designer and one programmer. If speed is essential, another person can be added whose tasks solely deal with the acquisition, preparation, and formatting of empirical knowledge and data.

To create the conceptual model, one process scenario would be for the domain expert and the knowledge engineer or software engineer to talk to each other. The talk should proceed informally first. After an informal understanding between the two is reached, the knowledge engineer extracts the knowledge from the domain experts step-by-step formally.

For the rest of the knowledge configuration and WIZER run, a process scenario would be for a program designer or an advanced programmer to prepare causal model, rules, semantic categorization of data, conflict resolution rules, model perturbation rules, and ontology. This person also leads in the running of WIZER and the interpretations of the results. They are assisted by a basic-level programmer or data entry person in the running of WIZER and in the acquisition of empirical data and knowledge.

12.3 An Example of Knowledge Configuration

A small portion of the code of the BioWar simulator is presented below in pseudo-code to serve as an example of the knowledge configuration steps. The pseudo-code represents a procedure which determines whether an agent gets infected with a disease in an outbreak.

```
procedure Outbreak

let outbreak = the outbreak
let agent = the agent the outbreak may cause infection

if agent has the outbreak (by strain) already
        do not reinfect the agent
end of if

dist = distance between this agent position and the location of the outbreak

if dist > ailment_effective_radius
        disease_contact_probability = a decaying function of ailment_effective_radius
else
        disease_contact_probability = 1.0
end of if

person_risk = risk of getting this disease based on age, disease type
adjust person_risk by risk multiplier and risk cap

base_rate = initial rate of getting an infection from a susceptible state
        for this outbreak
adjust base_rate by base_rate cap

infection_modifier_prophylaxis = the effect of an intervention or prophylaxis

total_risk = disease_contact_probability * base_rate * person_risk *
        infection_modifier_prophylaxis

if a random dice throw < total_risk
        infect this agent by this outbreak
end of if

end of procedure
```

The step-by-step procedure for the knowledge configuration related to the above routine is as follows.

1. The conceptual model of the above routine is a simple diagram depicting the relationship between an outbreak and an agent.
2. The empirical data is gathered for age risk factors.
3. The abstract causal model for the above routine is as follows, written in N3.
   <infection of an agent> <is caused by> <total_risk> .
   <total risk> <consists of> <disease_contact_probability, base_rate,

person_risk, infection_modifier_prophylaxis> .

<base_rate> <is influenced by> <base_rate_cap> .

<person_risk> <is influenced by> <age, disease type, risk multiplier, risk cap> .

<disease_contact_probability> <is influenced by>

<dist, ailment_effective_radius> .

4. Part of the concrete causal model is as follows.

   <disease_contact_probability = 1.0> <is caused by>

   <dist less or equal than ailment_effective_radius> .

   <disease_contact_probability = a decay function> <is caused by>

   <dist greater than ailment_effective_radius> .

   <agent previous infection> <prevents> <reinfection> .
5. The process logic is the pseudocode augmented by semantics (knowledge base) and ontology.
6. For each variable of disease_contact_probability, base_rate, person_risk, and infection_modifier_prophylaxis, semantic categories are created for their dynamic values. This requires domain knowledge. As an example, the determination of semantic categories depends on the medical knowledge about the infectiousness of a disease.
7. Rules related to the causal relations are created.
8. Conflict resolution rules for the variables are created.
9. Minimal model perturbation rules for the variables are created.
10. The relevant ontology is created.

12.4 Summary

This chapter describes the code structure of WIZER and outlines the steps for knowledge bases creation and ontology preparation for WIZER. It provides a guide for the use of WIZER.

Chapter XIII: Discussion

This chapter summarizes the contributions, limitations, and potential extensions of this dissertation research. The contributions are both conceptual/theoretical and practical. The limitations show themselves in the current WIZER implementation and also in the lack of learning capability, for example. Potential extensions including adding a learning capability are described.

13.1 Contributions

The contributions of this thesis are threefold. First, I developed a novel conceptual approach for the automated validation of multi-agent simulation tools. Second, I implemented and tested an automated tool for validation (WIZER), based on this conceptualization. Third, I examined the added-value of using this tool for validation using two distinct data sets and multi-agent simulations. The results indicate that WIZER speeds up the rate of validation, reduces the amount of necessary search, and focuses the search based on knowledge. The conceptual contributions include shedding light on the knowledge, logic types, and structures of simulation (the relationships between simulation code, process logic, causal logic, conceptual model, ontology, and empirical data and knowledge), a novel knowledge-based and ontological approach to validation automation, and the integration of simulation and knowledge management. Previously, knowledge inference and simulation were considered to be separate, as are knowledge management and simulation. This thesis indicates that these fields are closely intertwined and that they should inform each other closely.

As noted before, this thesis described the implementation of the conceptualization of knowledge-based and ontological validation in a tool called WIZER that is consistent with the conceptualization. The tool WIZER was implemented in four parts: the Alert

WIZER, the Inference Engine, the Simulation Knowledge Space, and the Domain Knowledge Space. This thesis demonstrated that:

1. Semantic categorizations of data and semantic control of statistical routines were made feasible by the Alert WIZER.
2. Knowledge-based and ontological reasoning and parameter value adjustments were made feasible by the Inference Engine in WIZER.
3. Explicit encoding and computer processing of simulation and domain knowledge were made feasible by the Simulation Knowledge Space and the Domain Knowledge Space as used in WIZER.
4. WIZER is a general tool, as evidenced by validation done on two simulation models, BioWar and CONSTRUCT.
5. WIZER speeds up the rate of validation, reduces the amount of necessary search in parameter space for validation, and increases the focus of the search to the most relevant area for validation.

By augmenting simulations with WIZER, simulation validation can be automated and simulation knowledge can be made clear and operable. Tools such as WIZER are important as they help clarify and speed up the validation process, in addition to helping automate the process.

13.2 Limitations

The dissertation does not implement all the conceptual potentials of WIZER. These include experiment design (which can be implemented via appropriate ontology and causal rules), simulation control (again, this can be implemented through ontology and causal rules), and a more sophisticated version of hypothesis building.

In summary, the limitations include:

1. No learning capability, except inference and a simple search for hypothesis building,

2. No causal and/or model learning from data implemented,
3. No causal relation derivations from simulation model or code,
4. Experiment design is not implemented,
5. Simulation control is not implemented,
6. Data limitations prevent more extensive and comprehensive validation and model-improvement trials,
7. Simple reasoning mechanisms via forward chaining and ontological reasoning,
8. The validation of knowledge bases is not covered. Knowledge bases should be validated against textbook knowledge, expert knowledge, and empirical data. How exactly to do this is a research topic in its own right.

13.3 Discussion

This dissertation provides a knowledge-based and ontological approach to validation with a side effect of model-improvement. Readers should be able to use the approach and the tool to do extensive validation of simulations and a simple model-improvement of them.

The work in this dissertation can be extended in many different ways. The immediate extensions are probing the structure and parameter trade-offs (in the conflict resolution and value/link adjustment routines), probing how to automatically get/derive causal models from the conceptual model, how to automatically get/derive process models from the causal model, how to automatically construct conceptual model from empirical data and knowledge (a data mining and machine learning problem), and how to automatically infer process models from code, causal models from the process model, and conceptual models from the causal model. The derivations and inferences may be aided by ontology and higher-level knowledge.

For agent-based simulations, a graphical user interface is needed for displaying and/or editing code, pseudocode, process models, causal models, conceptual model, and empirical data and knowledge. This graphical user interface should provide commands to run simulations, view the results, and see/trace the effect of knowledge bases and ontology.

Other extensions include:

1. Adding a experiment design module,
2. Adding a simulation control module,
3. Adding a mathematical ontology module,
4. Making the inference engine more sophisticated,
5. Enhancing conflict resolutions,
6. Enhancing value/link adjustment,
7. Adding a better model-improvement,
8. Adding a module for the validation of knowledge bases,
9. Validating the BioWar and CONSTRUCT testbeds comprehensively,
10. Examining more testbeds, including the Archimedes simulator.

Knowledge-based systems have been successfully embedded in many successful software applications, including tax preparation, business rules, and grammar/spelling checker applications. Integrating knowledge-based and ontology with simulation should ease the validation and model-improvement process, in addition to facilitating knowledge management of simulations.

Data mining tools can extract rules or relationships from empirical data. They can also extract rules or relationships from simulation data, by running the simulations and analyzing the resulting wealth of data. Their capabilities, however, are limited to classification and correlation. Neither causal relation nor conceptual model can be garnered by data mining tools. Thus data mining tools are the most useful in providing inputs for Alert WIZER in the form of symbolic categories and statistical measures of data. They can be useful for characterizing the simulation knowledge space (as rules extracted from simulated data) or the domain/empirical knowledge space (as rules extracted from empirical data) in the form of correlation between variables. No work yet is done in data mining or machine learning on extracting process model/logic from simulation code, extracting causal model from the process model/logic, and inferring conceptual model from the causal model. Inference in the other direction is not yet automated either: deriving conceptual models from human knowledge, causal models from conceptual models, process models/logic from causal logic and domain knowledge, and code or rules from process models/logic. Real world causality in particular is a major research problem: it is searching for cause-and-effect regularities in a not-so-orderly and uncertain world and it depends on, is inferred from, and should infer the (almost certain) predictable regularities to work. The above inference problems form an exciting future research for data mining and machine learning. Furthermore, this dissertation indicates that it is feasible to advance research on the fundamental and theoretical foundations for WIZER based on natural science (e.g., the physical nature of causality), computer science (including machine learning), and mathematics.

13.4 Summary

This dissertation demonstrates the utility of knowledge-based and ontological approach for validation and model-improvement of simulations, particularly social simulations. The validation scenario results of two testbeds show that the tool WIZER is useful for validation and model-improvement.

The contributions of this dissertation include a new conceptualization based on knowledge and ontology for validation and model-improvement of simulations, a tool implementing this conceptualization, partial validation results of the BioWar and CONSTRUCT simulators, a new simulation description logic, and knowledge-based hypothesis building and testing. This dissertation indicates that validated simulations are essential for the examination of causal relations to achieve better causal relation validity.

While many things about WIZER can be improved, this dissertation indicates there is a good conceptual foundation for future improvements of WIZER, including for the experiment design and simulation control enhancement of WIZER.

Appendix A. Modeling and Simulation

Modeling and simulation (Law and Kelton 2000) is an approach for developing a level of understanding of the interaction of the parts of a system, and of the system as a whole. It is one of the most widely used operations-research and management science techniques (two others are mathematical programming and statistics). The level of understanding which may be developed using modeling and simulation is seldom achievable otherwise, except possibly for system dynamics. A system is an entity which maintains its existence through the interaction of its parts. A model is a simplified representation of the actual system intended to elicit understanding.

This appendix talks about simulation types and where WIZER stands among them. It also indicates a way for learning simulation models from data, utilizing WIZER capabilites. The simulation types include discrete event simulation, continuous simulation, and agent-based simulation.

A.1 Simulation Model Classification

Simulation models can be classified along several dimensions:

1. Time: a simulation model can be static or dynamic. A static simulation model is one in which time plays no role or one that represents a snapshot of a system at a particular time. A dynamic simulation model represents a system as it evolves over time.
2. Randomness: a simulation model can be deterministic or stochastic. A simulation is deterministic if the model underlying this simulation does not contain any random or probabilistic components. Otherwise, a simulation model is stochastic.
3. Continuity: a simulation model can be continuous or discrete. A continuous simulation model represents a continuously evolving system often describable by

differential equations. A discrete simulation model represents changes of a system as separate events.

Most simulations are stochastic and dynamic. So a more cogent way of classifying simulation is dividing them into: discrete event simulation model, continuous simulation model, and agent-based simulation model. The latter is widely used as a versatile technique to model social and heterogeneous population systems. Of course, these models can be deterministic and/or static, but most are not, thus the new classification.

A.2 Discrete Event Simulation

Discrete event simulation concerns the modeling of a system as it evolves over time by representing the changes as separate events. This is the opposite of continuous simulation where the system evolves as a continuous function.

Among the discrete event simulation formalisms is the finite state machine. A finite state machine is a programming construct which proceeds in separate and discrete steps from one to another of a finite number of configurations or states.

A.3 Continuous Simulation

In continuous simulation, the system evolves in a continuous fashion, which can be described by differential equations. System dynamics is an approach to simulate continuous simulation. Continuous simulations are something that can only really be accomplished with an analog computer. Using a digital computer one can approximate or emulate a continuous simulation by making the time step of the simulation sufficiently small.

A.4 Agent-based Simulation

Agent-based simulation differs from traditional kinds of simulation in that some or all of the simulated entities are modeled in the form of agents. An agent is an abstraction of an individual. As it explicitly attempts to model specific behaviors of specific individuals, it is in contrast to methods where the characteristics of a population are averaged over, which is to say the model attempts to simulate changes in these averaged characteristics for the whole population.

A.5 Simulation Acceptance

While simulations have been used successfully for many tasks, wider acceptance of simulation is still impeded by the impression that simulation is just a toy, not a serious tool for decision analysis and making. Specifically, simulation acceptance is hindered by several factors including:

1. Models for large-scale systems tend to be very complex and coding them is an arduous task.
2. A large amount of computing time is required.
3. Simulation is thought of as just an exercise in computer programming. This neglects the important issue of how a properly coded model should be used to make inferences about the system of interest.
4. Understanding of a simulation depends on judgment calls of how much details of a problem should be modeled.
5. The development of simulation systems is often an iterative process by necessity. This sometimes invites a query of "if you say the simulation or program is valid, why is it that you have another version?"

This dissertation ameliorates the above factors by augmenting simulation (and modeling) with a symbolic, knowledge-based, and ontological representation and reasoning system. Specifically, this symbolic reasoning system has the following capabilities:

1. It grounds the complex models of large-scale systems on ontologies and knowledge bases enabling them to be reasoned about.
2. It reduces the amount of computing by knowledge and ontological inference.
3. It allows symbolic validation of the simulation. Symbolic validation, ontologies, and knowledge bases facilitate the use of the simulation model to make inferences about the system of interest.
4. How much detail a simulation should model is guided by symbolic, knowledge-based, and ontological reasoning of the policy question at hand.
5. It allows explanations of the simulation occurrences, particularly what variables or combinations of variables cause simulation outcomes and why.
6. Explicit and symbolic representation of the simulation model (which allows reasoning) enables an improved iterative process of simulation development cycles. This is aided by WIZER for the validation in each development cycle.

A.6 Simulation and WIZER

The BioWar simulation has the components of agent-based simulation and discrete event simulation. The outputs of the BioWar simulation are often in the form of (conceptually) continuous curves. WIZER can handle the validation of BioWar, which includes the handling and understanding of agents, discrete events, and continuous curves. This is because WIZER is a tool implementing the knowledge-based and ontological approach. WIZER performs inference on simulations. In database systems analogy, WIZER acts like SQL (structured-query-language) to the simulation. The following figure illustrates this point.

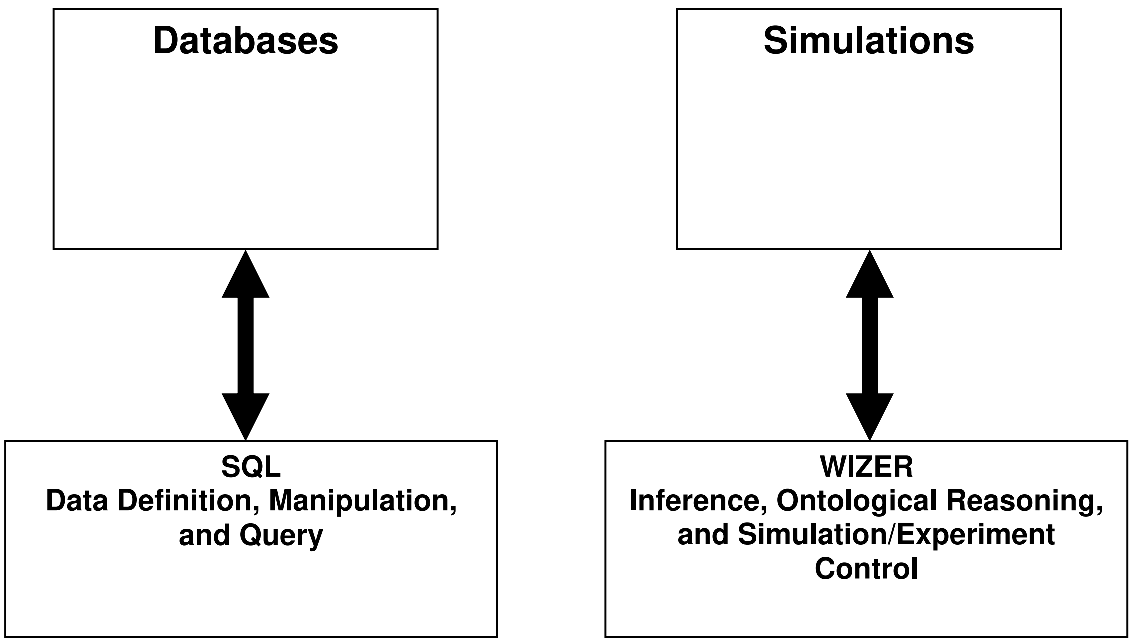

Figure A.1. Analogy of WIZER's Role to Simulations as SQL's Role to Databases

Simulations as computer programs are unconstrained, which means that anything can be simulated including processes and things that are unreal and erroneous. The constraints of physical reality are not built into simulations by default. They must be carefully designed and implemented. From the validation point of view, the physical constraints on simulation reduce the size of the search space for validation. Thus the physical constraints encoded by ontologies and knowledge bases are critical for validation. Another constraint that is important and useful for validation is the policy question constraint. Similarly encoded in ontologies and knowledge bases, the policy question restricts the scope and quantity of search for validation. The policy design and analysis field requires the application of careful and clear thinking in defining problems and finding solutions without needing to do extensive search. Only humans can perform policy design and analysis competently at the time of this writing. WIZER facilitates the use of both types of constraints for validation and model-improvement.

Statistics have the assumptions of sample independence, normality, and randomness for the parametric methods of statistics to work. Absent sample independence and normality, non-parametric methods of statistics can work given that the samples are random. Simulations, on the other hand, do not need to have the above assumptions. Instead of samples, entities and relationships can be modeled in great detail – as detailed as needed to answer a policy/research question. WIZER adds to this capability of simulations by providing the symbolic or semantic companion for simulation. This means all events that happen in simulation can be explained, used for reasoning, and controlled symbolically or semantically. Simulations can model networks, game-theoretic systems, and any other system. WIZER can describe the simulations of these systems semantically and perform suitable inferences. In the spectrum of increasing realism (and also the need for better data), statistics (parametric, non-parametric, and Bayesian statistics) is at the start of the spectrum while the physically-realistic simulation is at the end of the spectrum. Near the end of the spectrum is the agent-based simulation. The utility of WIZER increases as the simulation traverses toward the end of the spectrum. The principle of KISS (Keep It Simple Stupid) usually restricts the simulation to be simple, partly due to the difficulty of validation. WIZER helps the validation of more complex simulations, and thus helps push the realization of more complex and realistic simulations toward the end of the spectrum, especially of the simulations designed for use in policy making.

A.7 Simulation Model Learning from Data

Learning simulation models, or any models, from data is non-trivial. Recent research, however, points to a possible way to learn simulation models. It is done through causal learning from data. Once the causal relationships are learned (the relationships may have a probability of causation assigned to them), they are put together in a simulation model. The simulation is run and the strengths of causal relationships are adjusted based on the match against empirical data. In other words, WIZER allows validated simulations to verify the strengths of learned causal relationships. Depending on the data, detailed causal relationships can be extracted. More detailed causal relationships may approach the underlying processes and mechanisms.

To get to the underlying processes and mechanisms, a simulation model via WIZER can perform an exploratory search for possible processes/mechanisms (based on physical knowledge encoded in ontology, for example) for a given causal relationship. This is similar to the generate-and-test procedure. A more sophisticated hypothesis formation can also be employed. After a new process/mechanism is hypothesized for a given causal relationship, the simulation is run, the results of this simulation are validated using WIZER, and whether the hypothesized process/mechanism is valid can be assessed.

The above indicates that it is conceptually possible to learn a simulation model from data. This allows for model learning from data, an improvement over just learning Bayesian networks or causal networks. Using validated simulations and WIZER to uncover true causal relations is a novel concept. Previously, causality is assumed to be graph-based (Pearl 2000) with many “exogenous” assumptions abstracted away.

A.8 Summary

This appendix positions WIZER with respect to simulation types. It shows that WIZER acts as a symbolic manager and explainer for simulations. It is capable of inference and hypothesis building and testing. How WIZER can conceptually facilitate the learning of simulation models from data is also explained.

Appendix B. Augmenting System Dynamics

System dynamics is one of the most successful methods for modeling and understanding systems as a whole, not just as parts. Its core concept is the understanding of how all the objects and people in a system interact with one another. The objects and people can interact through feedback loops, where a change in one variable affects other variables over time, which in turn affects the original variable, and so on.

This appendix describes the state of the art of system dynamics. It then explains how WIZER could be employed to augment system dynamics.

B.1 Description of System Dynamics

System dynamics is concerned with the behavior of a system over time or the dynamic behavior of the system. Identifying its key patterns of behavior, known as "time paths" or "curves", is crucial. The time paths include the linear, exponential, goal-seeking, oscillation, and S-shape families of time paths or curves.

In system dynamics modeling, all dynamic behaviors in the world are assumed to occur in flows which accumulate in stocks. The following figure shows the stock and flow diagram for credit card inflow (Ratha 2001).

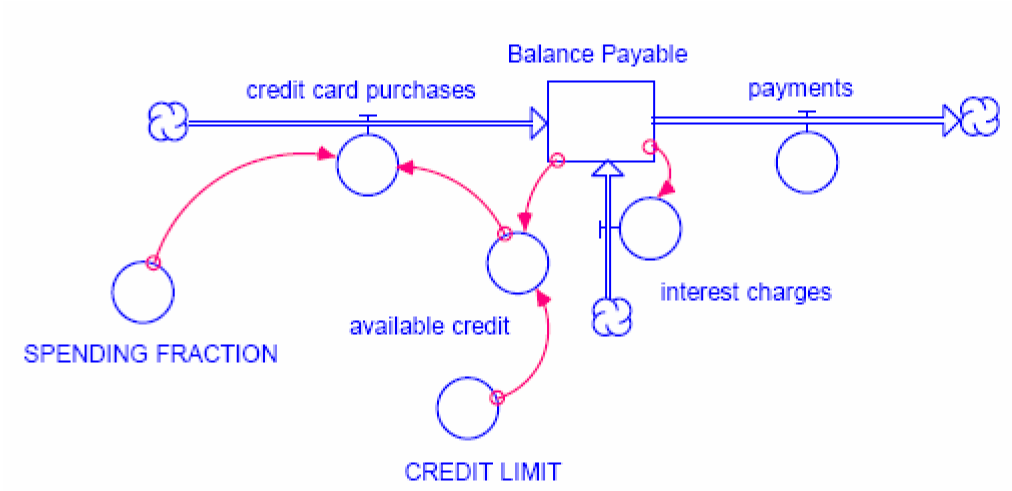

Figure B.1. Stock and Flow Diagram for Credit Card Inflow

As shown, the box indicates a stock "Balance Payable", where elements/amounts accumulate and then increase and decrease over time. The circle with a small T on the top represents a flow control, similar to a valve. The circles with extending lines and arrows indicate influences, which can be a positive or negative feedback. Finally the clouds indicate indeterminate supply. System dynamics uses mathematical equations including differential/difference equations to relate stock with flow.

In the example for Credit Card Inflow, the equations are as follows (where *t* denotes time and *dt* denotes the time difference).

```
Balance_Payable(t) = Balance_Payable(t - dt) + (credit_card_purchases + interest_charges
- payments) * dt
INITIAL VALUE = 0
SPENDING_FRACTION = 0.1
credit_card_purchases = SPENDING_FRACTION * AVAILABLE_CREDIT
CREDIT_LIMIT = 10000
available_credit = CREDIT_LIMIT - Balance_Payable
```

The constants for SPENDING_FRACTION and CREDIT_LIMIT are an instantiation for a particular credit card holder. The Balance_Payable is computed using a difference equation between time *t* and time *(t – dt)*.

B.2 WIZER and System Dynamics

As Alert WIZER can characterize curves or time paths (e.g., the S-shape of a curve) semantically, it can be used to characterize system dynamics' time paths into a knowledge format ready for knowledge inference.

Moreover, the level of stock and the speed of flow can be characterized semantically by WIZER using its knowledge bases and ontology. A specific type of stock and/or flow can have different effects and this can be characterized by WIZER using ontology. As WIZER does not require continuous stock and flow, jagged or abrupt transitions can be handled properly. What this abrupt transition means can be captured by WIZER. Using mathematical ontology, WIZER can describe the differential equations in system dynamics.

WIZER can validate the results of system dynamics simulation against empirical data. The system dynamics model of stocks, flows, and feedbacks is encoded in knowledge bases and causal/process ontology in WIZER. Based on the results of empirical data comparison against system dynamics simulation outputs, WIZER can suggest which stocks, flows, and/or feedbacks need change. WIZER outputs symbolic or semantics information about the system dynamics simulation runs.

As an example, Figure B.2 shows a system dynamics diagram on Factory (Albin 1997):

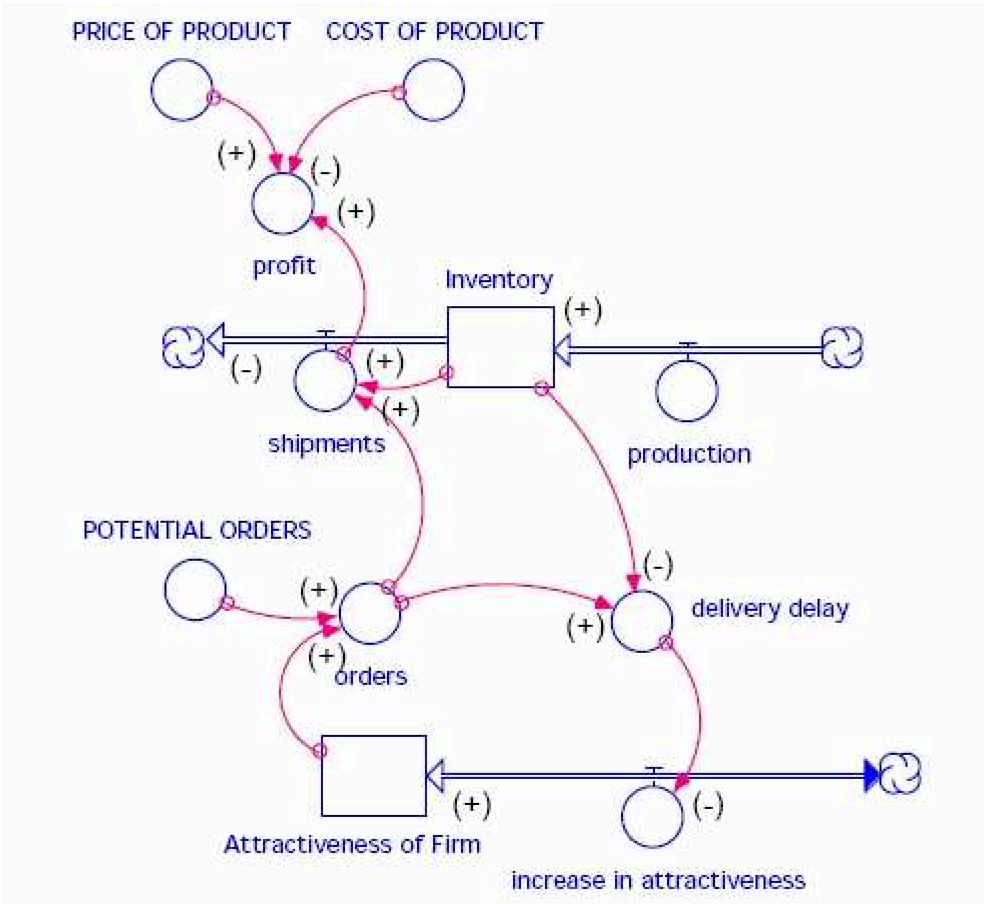

Figure B.2. Stock and Flow Diagram of a Company

As shown, the Inventory depends on production and shipments, while the Attractiveness of the Firm depends on the increase in attractiveness. The Attractiveness of the Firm then influences positively the amount of orders. In WIZER knowledge base, the above diagram can be represented as follows. "Negative" means negatively influencing or decreasing, while "positive" means positively influencing or increasing.

(causes (positive production) (positive Inventory))

(causes (and (positive Inventory) (positive orders)) (positive shipments))

(causes (positive increase_in_attractiveness) (positive Attractiveness_of_Firm))

(causes (positive delivery_delay) (negative increase_in_attractiveness))

(causes ((positive orders) and (negative Inventory)) (positive delivery_delay))

(causes (and (positive POTENTIAL_ORDERS) (positive Attractiveness_of_Firm)) (positive orders)) (causes (and (positive PRICE_OF_PRODUCT) (positive shipments) (negative COST_OF_PRODUCT) (positive profit))

The if-then rules are as follows.

(if-then (positive Inventory) (positive production))

(if-then (positive delivery_delay) (and (negative Inventory) (positive orders))

(if-then (negative delivery_delay) (and (positive Inventory) (negative orders))

(if-then (negative delivery_delay) (positive (minus (Inventory orders))))

(if-then (positive increase_in_attractiveness) (negative delivery_delay))

(if-then (positive orders) (and (positive INITIAL_ORDERS) (positive Attractiveness_of_Firm)))

(if-then (positive shipments) (and (positive Inventory) (positive orders)))

(if-then (positive profit) (and (positive PRICE_OF_PRODUCT) (positive shipments) (negative COST_OF_PRODUCT)))

(if-then (positive profit) (positive (minus (add PRICE_OF_PRODUCT positive shipments) COST_OF_PRODUCT)))

The processes/mechanisms underlying the above causal diagram are implemented in mathematical equations, similar to what system dynamics does but with added semantics via ontology and knowledge bases. The curves resulted from simulating the system are characterized by Alert WIZER symbolically. These symbolic characterizations allow the labeling of transition points, which is important for understanding the system.

Furthermore, when the underlying instantiated individual company data is available, WIZER can ground the abstract working of the system dynamics diagram on the empirical data. It can facilitate a more detailed symbolic and quantitative simulation of the Company (than the system dynamics simulation). As not all orders are the same, for a computer direct order company, for example, the order of a motherboard is more important financially than the order of a mouse. Some orders are linked to each other: the order of a graphics card is linked with the order of a flat screen. All these symbolic facts can be processed by WIZER, but not by system dynamics.

B.3 Summary

This appendix describes the fundamentals of system dynamics. It also explains how WIZER can augment system dynamics so that it can have improved knowledge management, improved simulation management, and better validation. System dynamics handles abstract systems while WIZER can handle concrete, individually instantiated systems including their semantics or knowledge aspect.

Appendix C. BioWar Ontology and Knowledge Base

The BioWar simulator is complex and evolving. The ontology and knowledge bases below represent the BioWar version 2.0 knowledge as of June 2006. For simplicity, all is written in the modified N3 notation.

Person:

<person> <has relationships of> <parent, spouse, child, sibling, neighbor> .

<person> <is located in> <a GPS coordinate> .

<person> <lives in> <a school district> .

<person> <has> <demographics> .

<demographics> <has type of> <age, gender, race> .

<age> <consists of> <0-4, 5-16, 17-25, 26-44, 45-55, 56 and more> .

<gender> <consists of> <male, female> .

<race> <consists of> <white, black, hispanics, asian> .

<person> <has> <knowledge vector> .

<knowledge vector> <causes> <interaction> .

<two persons> <has> <proximity> .

<proximity> <causes> <interaction> .

<person> <has> <disease vector> .

<interaction AND disease vector> <causes> <disease transmission> .

<person> <is part of> <ego network> .

<ego network> <causes> <interaction> .

<person> <has health status of>
<healthy, susceptible, infected, recovered, immune, deceased> .

<symptom severity> <causes> <agent sick behaviors> .

<agent daily behaviors> <has type of> <sleep, at school, at work, in transit,
at restaurant, at the mall, at cinema, outdoor recreation> .

<agent sick behaviors> <has type of>

<normal, rest at home, going to pharmacy, going to doctor office, going to emergency room> .

Disease:

<disease> <consists of> <attack disease, background disease>

<disease> <causes> <symptom> .

<disease> <has type of>
<communicable, noncommunicable, contagious, noncontagious> .

<symptom> <has phase of>
<infected, communicable, early symptomatic, late symptomatic> .

<symptom> <causes> <symptom severity> .

<symptom severity> <causes>
<going to pharmacy, going to doctor, going to emergency room> .

<going to pharmacy> <causes> <drug purchase> .

<drug purchase> <has type of> <analgesics, stomach, cough/cold> .

<symptom> <has type of> <fever, sneeze, cough, cold, headache, diarrhea> .

<symptom type> <causes> <specific drug purchase> .

<going to doctor, going to emergency room> <causes> <diagnosis> .

<diagnosis> <causes> <treatment, dismissal> .

<treatment> <causes> <recovery, hospital stay, death> .

<disease> <has instance of> <influenza, anthrax, smallpox, gastroenteritis, ...> .

<anthrax spread> <is influenced by> <wind, sunlight, terrain> .

<wind> <has property of> <speed, direction, atmospheric stability class> .

<initial infection number> <is influenced by>
<disease type, ailment effective radius, base-rate, weather, number of people> .

<disease exchange> <is influenced by>
<proximity, ailment exchange proximity radius, ego network, randomness> .

<ego network> <is caused by>
<initial ego network, homophily, information seeking, knowledge vector> .

Chemical ailment:

<chemical attack> <has type of> <noncontagious, noncommunicable> .

<chemical infliction> <has phase of> <inflicted, symptomatic, post-symptomatic> .

<symptomatic phase> <causes> <treatment> .

<treatment> <causes> <recovery, death> .

Response:

<response> <has type of> <medical response, physical response> .

<medical response> <has type of> <prophylaxis, autoimmune> .

<physical response> <has type of> <isolate, quarantine, evacuate, shelter> .

<physical response> <hinders> <people mobility> .

<people mobility> <causes> <contacts> .

<evacuate> <requires> <cooperation> .

<quarantine> <does not require> <cooperation> .

<shelter> <requires> <cooperation> .

<isolate> <does not require> <cooperation> .

<cooperation> <causes> <success of physical response> .

<physical response> <causes> <infection rate> .

Appendix D. CONSTRUCT Ontology and Knowledge Base

CONSTRUCT is a complex model of co-evolution of cognition (knowledge) and structure of social networks. It has many derivatives including CONSTRUCT-TM and DyNet. The following are the ontology and knowledge bases of the original CONSTRUCT, written in the modified N3.

<interaction> <causes> <knowledge exchange> .

<knowledge exchange> <causes> <shared knowledge> .

<shared knowledge, homophily, information seeking, randomness> <causes>
<interaction> .

<interaction> <influences> <friendship, enmity> .

<person> <has attributes of> <age, gender, race, economic status, social status> .

<person> <is embedded in> <social/associational network, instrumental network> .

<homophily> <influences> <social/associational network> .

<information seeking> <influences> <instrumental network> .

<interaction> <causes> <unionization> .

REFERENCES

Albin, S. 1997. Building a System Dynamics Model, Part 1: Conceptualization, MIT System Dynamics in Education Project, Report D-4597.

Anderson, J. R., Matessa, M., and Lebiere, C. 1997. ACT-R: A Theory of Higher Level Cognition and Its Relation to Visual Attention. Human Computer Interaction, 12(4), 439-462.

Anderson, J.A. 1995. An Introduction to Neural Networks. Cambridge, MA: MIT Press.

Axtell, R., Axelrod, R., Epstein, J., and Cohen, M. 1996. Aligning Simulation Models: A Case Study and Results. Computational and Mathematical Organization Theory, 1(2): 123-141.

Axelrod, R. 1997. Advancing the Art of Simulation. In Proceedings of the International Conference on Computer Simulation and the Social Sciences, Cortona, Italy.

Balci, O. 1998. Verification, Validation, and Accreditation. Proceedings of the 1998 Winter Simulation Conference, Winter Simulation Conference Foundation.

Bankes, S. C. 2004. Models as Lab Equipment: Science from Computational Experiments. In Proceedings of North American Association for Computational Social and Organizational Science (NAACSOS) Conference 2004. ISRI, CASOS, CMU, Pittsburgh, PA.

Berton, Y. and Castéran, P. 2004. Interactive Theorem Proving and Program Development, Berlin, Germany: Springer Verlag.

Breierova, L. and Choudhari, M. 2001. An Introduction to Sensitivity Analysis, MIT System Dynamics in Education Project Technical Report D-4526-2.

Buchanan, B.G. and Shortliffe, E.H. 1984. Rule-Based Expert Systems: The MYCIN Experiments at the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley.

Box, G.E.P, Hunter, J.S., and Hunter W.G. 2005. Statistics for Experimenters: Design, Innovation, and Discovery, 2nd ed., Hoboken, NJ: John Wiley & Sons.

Capcarrere, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., and Timmis, J. (eds). 2005. Advances in Artificial Life, Lecture Notes in Artificial Intelligence 3630. Berlin, Germany: Springer Verlag.

Carley, K. M., 1990, Group Stability: A Socio-Cognitive Approach. pp. 1-44 in Lawler E., Markovsky B., Ridgeway C. and Walker H. (Eds.) Advances in Group Processes: Theory and Research . Vol. VII. Greenwhich, CN: JAI Press.

Carley, K. M., 1991, A Theory of Group Stability. American Sociological Review, 56(3): 331-354.

Carley, K. M. and Prietula, M., eds. 1999. Computational Organizational Theory. Mahwah, NJ: LEA.

Carley, K. M., Fridsma, D., Casman, E., Altman, N., Chang, J., Kaminsky, B., Nave, D., and Yahja, A. 2003. BioWar: Scalable Multi-Agent Social and Epidemiological Simulation of Bioterrorism Events. In Proceedings of North American Association for Computational Social and Organizational Science (NAACSOS) Conference 2004, Pittsburgh, PA, http://www.casos.ece.cmu.edu/casos_working_paper/carley_2003_biowar.pdf.

Carley, K. M., Altman, N., Kaminsky, B., Nave, D., and Yahja, A. 2004. BioWar: A City-Scale Multi-Agent Model of Weaponized Biological Attacks. CMU-ISRI-04-101 Technical Report, Institute for Software Research International, Carnegie Mellon University, http://reports-archive.adm.cs.cmu.edu/anon/isri2004/CMU-ISRI-04-101.pdf

Carley, K.M, Kamneva, N.Y., and Reminga, J. 2004. Response Surface Methodology, CASOS Technical Report, CMU-ISRI-04-136. Pittsburgh, PA.

Chen, L-C, Carley, K.M., Fridsma, D., Kaminsky, B., and Yahja, A. 2006. Model Alignment of Anthrax Attack Simulations, Decision Support Systems, Volume 41, Issue 3, March 2006, pp. 654-668.

Chen, L-C., Kaminsky, B., Tummino, T., Carley, K. M., Casman, E., Fridsma, D., and Yahja, A. 2004. Aligning Simulation Models of Smallpox Outbreaks. In Proceedings of the Second Symposium on Intelligence and Security Informatics, Tucson, AZ, June 10-11, 2004. Also in Springer-Verlag Lecture Notes in Computer Science Vol. 3073.

Clement, R.T. and Reilly T. 2000. Making Hard Decisions with DecisionTools Suite. Duxbury Resource Center.

Crawford, J., Dvorak, D., Litman, D., Mishra, A., and Patel-Schneider, P. 1996. Path-Based Rules in Object-Oriented Programming. Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96).

Cristianini, N., and Shawe-Taylor, J. 2000. An Introduction to Support Vector Machines. Cambridge University Press.

Dayhoff, J.E. 1990. Neural Network Architectures: An Introduction. New York, NY: Van Nostrand Reinhold.

Dastani, M., Dix, J., and Seghrouchni, A.E.F., eds. 2004. Programming Multi-Agent Systems, Lecture Notes on Artificial Intelligence 3067, Berlin, Germany: Springer Verlag.

Davies, J., Schulte, W., and Barnett, M. eds. 2004. Formal Methods and Software Engineering, Lecture Notes on Computer Science 3308. Berlin, Germany: Springer Verlag.

Davies, J., Fensel, D., and van Harmel, F. eds. 2003. Towards the Semantic Web: Ontology-Driven Knowledge Management. West Sussex, England: John Wiley & Sons.

Deb, K., Poli, R., Banzhaf, W., Beyer, H-G., Burke, E., Dawren, P., Dasgupta, D., Floreano, D., Foster, J., Harman, M., Holland, O., Lanzi, P.L., Spector, L., Tettamanzi, A., Thierens, D., Tyrrell, A., eds. 2004. Genetic and Evolutionary Computation – GECCO 2004, Lecture Notes on Computer Science 3103, Berlin, Germany: Springer Verlag.

Dershowitz, N., eds. 2004. Verification: Theory and Practice. New York, NY: Springer Verlag.

Durkin, J. 1994. Expert Systems: Design and Development. Englewood Cliffs, NJ: Prentice Hall.

Eddy, D.M. and Schlessinger, L. 2003. Validation of the Archimedes Diabetes Model, American Diabetes Association.

Edmonds, B. and Bryson, J.J. 2004. The Insufficiency of Formal Design Methods: the Necessity of an Experimental Approach for the Understanding and Control of Complex MAS. In Proceedings of AAMAS 2004, ACM.

Epstein, J.M. and Axtell, R.L. 1996. Growing Artificial Societies. Cambridge, Mass.: MIT Press.

Epstein, J.M., Cummings, D.A.T, Chakravarty, S., Singa, R.M., and Burke, D.S. 2004. Toward a Containment Strategy for Smallpox Bioterror: An Individual-Based Computational Approach. Washington, DC: Brookings Institution Press. http://www.brook.edu/press/books/towardacontainmentstrategyforsmallpoxbioterror.htm

Etessami, K. and Rajamani, S.K. (eds.) 2005. Computer Aided Verification. Lecture Notes in Computer Science 3576, Springer Verlag.

Fall, C., Marland, E., Wagner, J., and Tyson, J. (eds.) 2005. Computational Cell Biology. Springer Verlag.

Fitzgerald, J., Larsen, P.G, Mukherjee, P., Plat, N., and Verhoef, M. 2005. Validated Designs for Object-oriented Systems. London, UK: Springer Verlag.

Fogel LJ. 1999. Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming. Wiley Series on Intelligent Systems, New York, NY.

Forgy, C.L. 1982. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. Artificial Intelligence, 19(1982) 17-37

Frith, C. and Wolpert, D. 2004. The Neuroscience of Social Interaction: Decoding, Imitating, and Influencing the Actions of Others. Oxford, United Kingdom: the Royal Society and the Oxford University Press.

Giarratano, J. and Riley, G. 2004. Expert Systems: Principles and Programming, 4th Ed. Course Technology.

Greenland, S. and Pearl, J. 2006. Causal Diagrams, UCLA Cognitive Systems Laboratory, Technical Report R-332, prepared for the Encyclopedia of Epidemiology

Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional.

Gomez-Perez, A., Fernandez-Lopez, M., and Corcho, O. 2004. Ontological Engineering with Examples from the Areas of Knowledge Management, e-Commerce, and the Semantic Web. London, UK: Springer Verlag.

Good, P. 2005. Permutation, Parametric, and Bootstrap Tests of Hypotheses, 3rd Ed. New York, NY: Springer Science+Business Media, Inc.

Grosof, B.N., Volz, R., Horrocks, I., and Decker, S., 2003. Description Logic Programs: Combining Logic Programs with Description Logic. WWW 2003, Budapest, Hungary.

Grosof, B.N. 2005, The Production Logic Programs Approach, in a Nutshell: Foundations for Semantically Interoperable Web Rules. Working Paper of Dec. 19, 2005.

Hall, D.L. and Llina, J. 2001. Handbook of Multisensor Data Fusion. CRC Press.

Haenni, R. 2005. Shedding New Light on Zadeh's Criticism of Dempster's Rule of Combination. FUSION'05, 8th International Conference on Information Fusion, Contribution No. C8-1, Philadelphia

Haenni, R., Kohlas, J., Lehmann, N. 1999. Probabilistic Argumentation Systems. Technical Report 99-9, Institute of Informatics, University of Fribourg, Fribourg, Switzerland.

Haubold, B. and Wiehe, T. 2006. Introduction to Computational Biology: An Evolutionary Approach. Basel, Switzerland: Birkhauser.

Hinchey, M.G., Rash, J.L., Truszkowski, W.F., and Rouff, C.A. (eds). 2005. Formal Approaches to Agent-Based Systems. Lecture Notes in Artificial Intelligence 3228. Berlin, Germany: Springer Verlag.

Huang, C-Y, Sun, C-T, Hsieh, J-L, and Liu, H. 2004. Simulating SARS: Small-World Epidemiological Modeling and Public Health Policy Assessments. Journal of Artificial Societies and Social Simulation, 7(4). http://jasss.soc.surrey.ac.uk/7/4/2.html

Hutter, D. and Stephan, W. (eds.) 2005. Mechanizing Mathematical Reasoning. Lecture Notes in Artificial Intelligence 2605, Springer Verlag, Berlin, Germany.

Jewell, N.P. 2003. Statistics for Epidemiology. Boca Raton, FL: Chapman & Hall/CRC.

Jackson, P. 1999. Introduction to Expert Systems, 3rd ed., Reading, Mass.: Addison-Wesley

Kapferer B. 1972. Strategy and Transaction in an African Factory. Manchester, UK: Manchester University Press.

Keedwell, E. and Narayanan, A. 2005. Intelligent Bioinformatics: The Application of Artificial Intelligence Techniques to Bioinformatics Problems. John Wiley & Sons.

Kim, T.G. 2005. Artificial Intelligence and Simulation. Lecture Notes in Artificial Intelligence 3397. Berlin, Germany: Springer Verlag.

Kline, R.B. 2004. Principles and Practice of Structural Equation Modeling, 2nd ed., New York: The Guilford Press.

Kosko, B. 1996. Fuzzy Engineering, 1st ed. Upper Saddle Riever, NJ: Prentice Hall

Law, A.M. and Kelton, W.D. 2000. Simulation Modeling and Analysis, 3rd Ed. New York, NY: McGraw-Hill.

Lehmann, E.L. and Romano, J.P. 2005. Testing Statistical Hypotheses, 3rd Ed. New York, NY: Springer Verlag.

Lenat, D. and Guha, R.V. 1990. Building Large Knowledge-based Systems: Representation and Inference in the Cyc Project, Addison-Wesledy.

Lesser V, Decker K, Wagner T, Carver N, Garvey A, Horling B, Neiman D, Podorozhny R, NagendraPrasad M, Raja A, Vincent R, Xuan P, and Zhang XQ. 2004. Evolution of the GPGP/TAEMS Domain-Independent Coordination Framework. Autonomous Agents and Multi-Agent Systems, Volume 9, Number 1, Kluwer Academic Publishers, pp. 87-143.

Lucena, C., Carcia, A., Romanovsky, A., Castro, J., and Alencar, P., eds. 2004. Software Engineering for Multi-Agent Systems II. Lecture Notes in Computer Science 2940. New York, NY: Springer Verlag.

Mitchell, T. M. 1978. Version Spaces: An Approach to Concept Learning. PhD Thesis. Electrical Engineering Dept. Stanford University, Stanford, CA.

Monge, P.R. and Contractor, N.S. 2003. Theories of Communication Networks. Oxford University Press.

Myers, R.H. and Montgomery, D.C. 2002. Response Surface Methodology: Process and Product Optimization using Designed Experiments, 2nd ed. New York, NY: John Wiley.

National Research Council. 2004. Computer Science: Reflections on the Field, Reflections from the Field, Washington, DC: National Academies Press.

Neapolitan, R.E. 2003. Learning Bayesian Networks. Prentice Hall.

Nickles, M., Rovatsos, M., and Weiss, G., eds. Agents and Computational Autonomy, Lecture Notes in Computer Science 2969. Berlin, Germany: Springer Verlag.

O'Hare, G. and Jennings, N. 1995. Foundations of Distributed Artificial Intelligence, Wiley Inter-Science, pp. 429-448.

Pearl, J. 2000. Causality: Models, Reasoning, and Inference. Cambridge, UK: Cambridge University Press.

Pearl, J. 2003. Statistics and Causal Inference: A Review. Test Journal 12 no. 2 (December): 281-345.

Pressman, R. S. 2001. Software Engineering., New York, NY: McGraw-Hill.

Prietula, M.J., Carley, K.M., and Gasser L. 1998. Simulating Organizations. Menlo Park, Calif.: AAAI Press/The MIT Press.

Rasmussen, S. and Barrett, C.L. 1995. Elements of a Theory of Simulation. ECAL 95, Lecture Notes in Computer Science. Berlin, Germany: Springer Verlag.

Ratha, M. 2001. The Credit Card Model. MIT System Dynamics in Education Project, Report D-4683-2.

Reifman J., Gilbert, G., Parker, M., and Lam, D. 2004. Challenges of Electronic Medical Surveillance Systems. RTO HFM Symposium on NATO Medical Surveillance and

Response: Research and Technology Opportunities and Options, Budapest, Hungary. http://www.rta.nato.int-/Pubs/RDP.asp?RDP=RTO-MP-HFM-108

Reiter, R. 2001. Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. Cambridge, MA: The MIT Press.
Robert, C. P. and Casella, G. 1999. Monte Carlo Statistical Methods. New York, NY: Springer Verlag.

Russell, P. and Norvig, P. 2003. Artificial Intelligence: A Modern Approach, 2nd Ed., Prentice Hall.

Schreiber, C. and Carley, K. M. 2004. Going Beyond the Data: Empirical Validation Leading to Grounded Theory. Computational and Mathematical Organization Theory, 10, 155-164.

Sentz, K. and Ferson, S. 2003. Combination of Evidence in Dempster-Shafer Theory. Technical Report, Sandia National Laboratories. http://www.sandia.gov/epistemic/Reports/SAND2002-0835.pdf

Shervais, S., Wakeland, W., and Raffo, D. 2004. Evolutionary Verification and Validation of Software Process Simulation Models. In the 5th International Workshop on Software Process Simulation and Modeling, extended abstract.

Shervais, S. and Wakeland, W. 2003. Evolutionary Strategies as a Verification and Validation Tool. Portland State University paper, http://www.sysc.pdx.edu/faculty/Wakeland/papers/EvoVandVRevD.pdf

Simon, H. 1996. The Sciences of the Artificial, 3rd Ed. Cambridge, MA: MIT Press

Spirtes, P., Glymour, C., and Scheines, R. 2000. Causation, Prediction, and Search. Cambridge, MA: MIT Press.

Stefik, M. 1995. Introduction to Knowledge Systems. San Francisco, Calif.: Morgan Kaufmann Publishers.

Sternberg, R.J. and Pretz, J.E. 2005. Cognition and Intelligence: Identifying the Mechanisms of the Mind. Cambridge, UK: Cambridge University Press.

Szallasi, Z., Stelling, J. and Periwal, V., eds. 2006. System Modeling in Cellular Biology: From Concepts to Nuts and Bolts. Cambridge, MA: MIT Press.

Thrun, S., Burgard, W. and Fox, D. 2005. Probabilistic Robotics. Cambridge, MA: MIT Press.

Vapnik, V.N. 2000. The Nature of Statistical Learning Theory 2nd Ed. New York, NY: Springer Verlag.

Wasserman, S. and Faust, K. 1994. Social Network Analysis: Methods and Applications. Cambridge, UK: Cambridge University Press.

Weiss, G. (Ed.) 1999. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press.

Xiaorong, X., Kennedy, R., and Madey, G. 2005. Verification and Validation of Agent-based Scientific Simulation Models, Agent-Directed Simulation Conference, San Diego, CA