

AFRL-IF-RS-TR-2006-176
Final Technical Report
May 2006



PLANNING, EXECUTION, AND ASSESSMENT OF EFFECTS-BASED OPERATIONS (EBO)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK**

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2006-176 has been reviewed and is approved for publication.

APPROVED: /s/

JOSEPH A. CAROLI
Project Engineer

FOR THE DIRECTOR: /s/

JAMES W. CUSACK
Chief, Information Systems Division
Information Directorate

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE MAY 2006	3. REPORT TYPE AND DATES COVERED Final May 01 – Sep 05	
4. TITLE AND SUBTITLE PLANNING, EXECUTION, AND ASSESSMENT OF EFFECTS-BASED OPERATIONS (EBO)			5. FUNDING NUMBERS C - F30602-01-C-0065 PE - 63789F PR - EBO0 TA - 00 WU - 05	
6. AUTHOR(S) Lee W. Wagenhals, Alex Levis, Sajjad Haider				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) George Mason University C3I Center 4400 University Drive Fairfax Virginia 22030-4444			8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/IFSA 525 Brooks Road Rome New York 13441-4505			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2006-176	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Joseph A. Caroli/IFSA Joseph.Caroli@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA # 06-355				12b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 Words) This is the final technical report of the contract entitled "Planning, Execution, and Assessment of Effects Based Operations" that was conducted by George Mason University between 1 May 2001 and 30 September 2005. The overall objective of the effort was to enhance and use a suite of modeling and simulation tools called Computer Aided Evaluation of System Architectures (CAESAR II/EB) to support the warfighter when planning, executing, and assessing Effects-Based Operations (EBO). At the completion of the effort, an integrated EBO tool suite called Pythia had been developed and tested. The tool operates in a single Windows-based environment. It allows analysts to develop and analyze EBO courses of action by creating Timed Influenced Nets that relate potential actions of an overall course of action (COA) to desired and undesired effects. This report documents the EBO concept that was used to develop Pythia, describes the tools, technologies and techniques that support that concept, and documents the key technical feature of Phthia.				
14. SUBJECT TERMS Effects Based Operations, Influence Nets, Timed Influenced Nets, Bayesian Nets, Temporal Logic, Colored Petri Nets, Course of Action Analysis				15. NUMBER OF PAGES 107
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

TABLE OF CONTENTS

1. SUMMARY	1
2. INTRODUCTION	4
2.1 EBO Concept	4
2.2 Technology Tools and Algorithms to Support EBO Activities	7
3. METHODS, ASSUMPTIONS, AND PROCEDURES	20
4. RESULTS AND DISCUSSION	25
4.1 Definitions	27
4.1.1 Influence Nets (INs)	27
4.1.2 Timed Influence Nets	29
4.1.3 Modeling of Time Varying Influences: Dynamic Influence Nets	31
4.2 Technical Features of Pythia	36
4.2.1 Probability Propagation in Static Influence Nets	39
4.2.2 Sensitivity Analyses	40
4.2.3 Incorporation of Temporal Information	41
4.2.4 Algorithms/Analyses Using Temporal Information	42
4.2.5 Integration of the Temporal Logic	43
4.2.6 Effective Courses of Action Determination	44
5. CONCLUSION	46
REFERENCES	48
LIST OF ACRONYMS	50
APPENDIX A: PYTHIA Version 1.0 USER MANUAL	51

LIST OF FIGURES

Figure 2.1. IDEF0 Process Model for Dynamic Effects Based Command	7
Figure 2.2 Effects Based Operations: Defining Effects.....	8
Figure 2.3 Development of the Influence net model	9
Figure 2.4 Analysis of Alternatives	16
Figure 2.5 Decision Support for COA	17
Figure 3.1 The Initial Suite of Tools.....	22
Figure 4.1 A Sample Influence Net	27
Figure 4.2 Probability Profiles of Event D	30
Figure 4.3 A TIN Having Time-Variant Influences	32
Figure 4.4 A TIN with Self-Loop and Time-Varying Influences.....	34
Figure 4.5 Comparison of Profiles Generated by a TIN and a DIN	35

1. SUMMARY

This is the final technical report of the contract entitled “Planning, Execution, and Assessment of Effects Based Operations” that was conducted by George Mason University between 1 May 2001 and 30 September 2005. The overall objective of the effort was to enhance and use a suite of modeling and simulation tools called Computer Aided Evaluation of System Architectures (CAESAR II/EB) to support the warfighter in planning execution and assessing effects-based Operations (EBO). Specifically, the tools support the efficient development and evaluation of alternative courses of action (COAs) that implement an Effects Based campaign strategy; they allow for the integration of lethal and non-lethal Information Operations (IO) with conventional operations; and they enable the tracking of effects to support decision making at the Commander level.

The underlying technical effort consisted of the enhancement and integration of a set of five applications that have been developed between 1996 and 2001 at George Mason University and at AFRL/IF. A goal was to sufficiently mature the technology so that it could be included in the Effects Based Operations (EBO) Advanced Technology Demonstration (ATD) that was targeted toward JEFX 04¹. The result was the development of a new tool called Pythia.

The suite of tools is based on four key technologies: (a) Bayesian nets and their variant, Influence nets; (b) Colored Petri Nets; (c) Temporal Logic; and (d) Modeling and Simulation. The strength of the approach was the innovative integration of these technologies for application to the planning, execution, and assessment of Effects Based Operations. Embedded in the conceptualization of the problem is the recognition that Information Operations are an integral component of EBO and need to be integrated in the planning and execution paradigm.

The five tools that were the bases for creating the new integrated EBO tool were: (1) The Campaign Assessment Tool (CAT) developed at AFRL/IF and enhanced at GMU; (2) COA/EB, a set of algorithms embedded in Design/CPN, a software package for the design, analysis, and simulation of Colored Petri Nets; (3) TEMPER 2, a temporal logic tool for the time phasing of

¹ It should be noted that the effort described in this report was not formally part of the EBO ATD that was being managed by AFRL/IF; rather it was a parallel effort that provided technology to the AFRL/IF EBO ATD.

tasks to meet temporal and resource constraints; (4) the Real-Time Execution Monitor (R-TEM) that receives as input the occurrence of events during execution and calculates the impact of these events on the desired effects, and (4) a Visualization Module (V-Mod) for the presentation of the results.

Specifically, the technical program consisted of the following efforts:

- Enhance further the Campaign Assessment Tool (CAT) developed by AFRL/IF to include alternative algorithms for the computation of the probabilities of achieving the desired effects. Include temporal information in the inputs to CAT and pass that information automatically to the CPN executable model.
- Integrate the capability to develop Courses of Action that include both Information Operations and conventional operations in the CAESAR II/COA/EB module. Develop an assessment module for comparing alternative COAs and send the results to the Visualization module (V-Mod).
- Integrate a version of the temporal logic module (TEMPER) into CAESAR II/COA/EB and develop a browser based user interface that is appropriate for the use of TEMPER in EBO.
- Implement the results of on-going research in the development of the Real-Time Execution Monitor (R-TEM) module. Develop the interface between this module and the Visualization module (V-Mod) as part of the Commander's Decision Support System.

At the start of the effort, all of the individual applications described above were at different levels of maturity; except for R-TEM, all others had been demonstrated and used singly or in combination in war games or technology demonstrations for the Air Force or the Navy. Thus the integration of all these tools into a collaborative environment capable of supporting Effects Based air operations was the specific objective of the effort.

At the completion of the effort, an integrated EBO tool suite called Pythia, had been developed and tested. The tool operates in a single Windows Based environment. It allows analysts to develop and analyze EBO courses of action by creating Timed Influence Nets that relate potential actions of a COA to desired and undesired effects. The analysts (1) create the actions and the causal or influencing relations to effects, (2) provide estimates of the strengths of those

relationships, and (3) introduce temporal information representing time delays in the propagation and processing of actions, causes, and effects, and the time of actions or events. Once the influence net has been created, Pythia provides a suite of analysis tools that assist the analyst in comparing and selecting COAs. The tool also supports the assessment of progress toward achieving desired effects during the execution of the COA by transforming the timed influence net into a time sliced Bayesian Net where evidence can be entered.

An installation CD with a User's Manual has been produced for Pythia, and it has been provided to several government organizations and contractors.

The rest of this report is organized as follows. Section 2 provides an introduction to the effort by describing the EBO concept, activities needed to support EBO, and the underlying technology and tools that were used to create the final Pythia tool. Section 3 outlines the methods, assumptions, and procedures that were used during the effort. It describes the tasking and the set of technologies that were available at the start of the effort and the process that was used to test and integrate the tools and technologies into the integrated Pythia tool. Sections 4, Results and Discussion, provides a more technical description of the effort including formal definitions of the modeling techniques and descriptions of the algorithms used to support the analysis of the models that were incorporated in the tool. The descriptions are in summary form. The details of the algorithm may be reviewed in the many published articles and papers that are referenced in this report.

2. INTRODUCTION

The problem of planning, executing and assessing Effects-Based Operations (EBO) requires the synthesis of a number of approaches that have been emerging in the last few years from basic research efforts by DOD and industry. The purpose of this development effort was to examine the operational concepts for Effects Based Operations, identify and select a set of tools, techniques, and algorithms, and integrate them into a tool suite that could support the complete EBO process. To understand the approach and the results of the effort, it is necessary to examine the EBO Concept that was the driver for the tool suite development, the activities that are conducted within that concept and the technology and tools that can support those activities. Given this understanding, then the tasks that were undertaken in this effort can be examined in Sections 3 and 4.

2.1 EBO Concept

Effects Based Operations (EBO), the notion of selecting actions that comprise a COA based on their collective contribution to desired and undesired effects, is not a new concept. In the past decade there has been an increased emphasis on modeling tools and techniques to support effects based planning and execution. The rapid advancement of technology, particularly information technology, has focused attention on EBO as an essential organizing principle for command and control of military operations across multiple echelons. New technology that allows precision attack with weapons of pinpoint accuracy, intelligence systems that provide accurate location of targets, and stealth technology that greatly reduces the requirement of defensive support systems to protect striking weapons has enabled selective components of adversary systems to be struck with precision to achieve desired effects with minimum risk and destruction. In addition, the complexity of coalition operations and the understanding that an important aspect of warfare is the actions that will take place after the combat operations have ceased, has led to the notion that we should consider alternatives to the concept of maximum destruction attrition warfare. By focusing on the overall effects needed to achieve objectives and considering a spectrum of lethal and non-lethal actions, COAs can be formulated that use precision intelligence and strike capabilities to inflict the minimum collateral damage while achieving objectives. To do this one

must understand and develop a set of effects that, if achieved, will result in the overall objectives and then determine the best set of actions to take, *along with their timing*, to achieve those effects. In modern coalition operations, such actions include not only traditional military attrition based operations, but a spectrum of actions across the instruments of national power employed by coalition partners to influence and persuade an adversary to change his behavior and at the same time maintaining cohesion within the coalition.

While these concepts have allowed us to go well beyond the construct of massive attrition-based warfare, they have tightened many of the traditional constraints: collateral damage must be minimized, our own casualties must be virtually nil, the long term impact on the well being of native populations should be limited (e.g., do not destroy beyond repair the infrastructure).

The concept of effects based operations is well suited to this problem. Instead of focusing on the servicing of a well defined a priori target list, we focus on the effects that we wish to achieve. The target list still exists and includes both hard and soft targets: from weapons systems, to C2 nodes, to leadership nodes, to infrastructure nodes, to the contents of communications. But the target list is only an intermediate construct, a means to an end that can change rapidly as the effects we wish on the adversary are being achieved or not. Indeed, the list of possible actions we can take is now much larger as it includes all instruments of national (or coalition) power: political, military, or humanitarian; physical or ideological. The availability of all instruments gives us much flexibility in trying to achieve the desired effects and to avoid undesirable ones. But it also makes the Course of Action (COA) problem and the subsequent planning problem much harder. There are now many alternatives, many choices. The choice of a set of actions, their sequencing, and their time phasing become problems in their own right.

It is useful to partition the overall problem into five inter-related ones. These problems are addressed in stages of an integrated process. Each problem requires models and algorithms that are specific to that stage.

1. *EBO Problem*: Relate effects to actionable events. In this problem, we need to define the set of desired and undesirable effects on the adversary. Then, working backwards, from effects to causes, arrive at the actions that we have at our disposal (the application of the instruments of national/coalition power) for achieving these effects.

2. *COA Problem*: Select from the set of all actions those subsets that will yield with high probability the effects we wish to achieve (including low probability for undesirable effects). Take into consideration constraints associated with specific actions or combinations of actions. Then sequence the actions in each subset and time-phase them. The result is a set of alternative COAs. This is done at the operational level. When the selected COA becomes a plan, the operational aspects are mapped to system (target) aspects and the actions are translated into tactical level tasks.
3. *ISR problem*: We need to identify those observables (phenomena that can be observed by our sensors) that either directly or indirectly indicate whether we are achieving the effects or not. This information provides the basis for assigning Intelligence, Surveillance, and Reconnaissance assets to monitor the execution of the operations. Furthermore, this means that by monitoring the progress we are making we will be able to adapt plans in a dynamic manner.
4. *Evaluation Problem*: We need metrics by which we can assess the effectiveness of different COAs. When we have such metrics, it then becomes possible to generate algorithmically the set of preferred COAs that either maximizes a measure of effectiveness (MOE) subject to constraints, or satisfies a set of constraints including MOE thresholds.
5. *Execution Assessment Problem*: Once the plans that constitute a selected COA begin to unfold, we need to be able to use the models created to address problems 1, 2, and 3 and the metrics of problem 4 to measure and calculate the degree to which the desired effects are being achieved and if necessary adjust the selected COA.

A process to address this set of problems can be expressed in terms of specific activities that need to be performed and the tools and techniques that support them. This is illustrated as an activity model in Fig. 2.1 using the IDEF0 formalism. The first activity is the analysis of the situation using several modeling techniques. This activity is carried out by situation analysts, both intelligence and operations analysts. The second activity is the development and selection of alternative Courses of Action. This includes the generation of a variety of contingency COAs and the ability to evaluate these COAs in terms of their likelihood of achieving desired effects. The third activity is to generate detailed plans (e.g., the ATO) from the selected Courses of

Action. The approved plan is disseminated to the units that carry out the tasks in the plan and to operational controllers who monitor the execution. In the fourth activity, the execution of the plan is controlled using the capability to exercise feedback from various unit and ISR reports. In the last activity, the results of the plan execution are monitored and assessed to determine whether how well the plan is working and whether adjustments need to be made to the Models, COA, Plan, or even the directives being generated by the controllers.

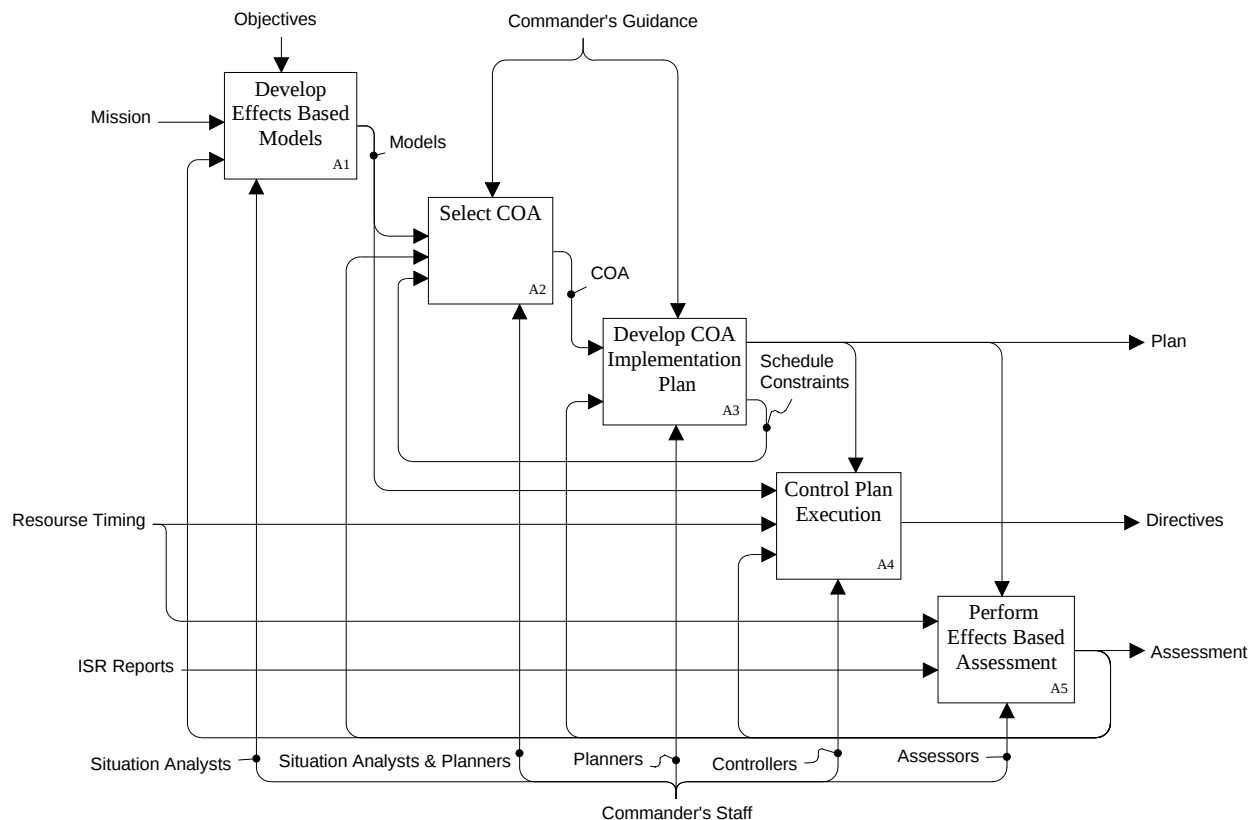


Figure 2.1. IDEF0 Process Model for Dynamic Effects Based Command

2.2 Technology Tools and Algorithms to Support EBO Activities

The first activity is supported by a set of models that are at the heart of providing the capability for Effects Based Operations. Figure 2.2 shows the basic process. The goals are set by the National Command Authority at the strategic level and by the Commander for the operational level through the development of the *commander's intent* which is a textual description of what

the commander wishes to be accomplished. It is then determined that, to reach the goals, certain effects must be achieved.

Once the effects that need to occur to achieve the goals and objectives have been identified, the next step is to determine the set of actions that can be taken that will cause or influence the selected effects to occur. This determination can be accomplished using probabilistic modeling tools (e.g., Influence Net modeling or Bayesian Net modeling). At the start of this effort three such tools, SIAM², CAT³, and the GMU developed CASEAR/IIEB tool existed. The development effort that is the subject of this report created an integrated tool that can be used to do influence net modeling and analysis.

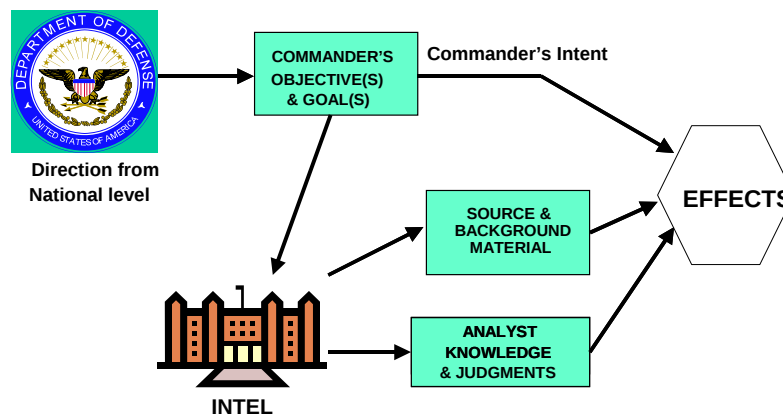


Figure 2.2 Effects Based Operations: Defining Effects

An Influence Net model allows the intelligence analyst to build complex models of probabilistic influences between causes and effects or actionable events and effects. This is shown in Fig. 2.3. The Influence net model is then used to carry out sensitivity analyses to determine which actionable events, alone and in combination, appear to produce the desired effects. The models also can indicate potential actions that may be observed by ISR assets to confirm events of importance during execution.

² SIAM is a COTS product developed by SAIC (Rosen and Smith, 1996) to support the intelligence community and is used as a module in the CAESAR II suite of tools. While earlier versions of the tool supported the conversion of the Influence net model to a Colored Petri net – a capability developed with AFRL/IF support – recent versions do not. Other probabilistic modeling tools include Hugin and Analytica, but do not have all the special features needed for Effects Based Operations modeling.

³ This is the Campaign Assessment Tool developed at AFRL/IF by Dr. John Lemmer. It has been enhanced by GMU with the cooperation of Dr. Lemmer to subsume the SIAM algorithms as well as include a variety of theories and algorithms for relating causes and effects.

At this point is important to understand the different modeling approaches that can be used to relate potential actions to effects. In the current practice, complex political, economic, and military situations are analyzed and evaluated using a combination of models and simulations. Many of the models deal with well known, physics based systems, where classic discrete event or continuous time dynamical models can be created to evaluate the behavior or performance of systems over a range of stimuli. Detailed models of integrated air defense systems that can be used to determine the expected attrition of air strikes, or define the best suppression techniques, are readily available as a case in point. But many aspects of situations involve phenomena that are difficult or impossible to model by precise, classic, physics-based models. Decision and policy making and command and control processes of nations or organizations and so called intelligent systems are examples of such phenomenon. As a result, the use of probabilistic models has been incorporated in the analysis of such processes and their role in political, economic, and military situations. In particular, Bayesian networks and variants called influence nets have been incorporated in the analysis of situations. These nets are graph theoretic, providing a powerful tool for visualizing dependencies between variables in the model of the situation. In a Bayesian net or influence net, the nodes of the graphical network represent hypotheses or propositions and the arcs represent direct dependency relationships between the hypotheses. Conditional probabilities are associated with the nodes of the net that encode the strengths of the dependencies. Algorithms have been developed that efficiently compute new values of all the variables whenever any variable value is specified.

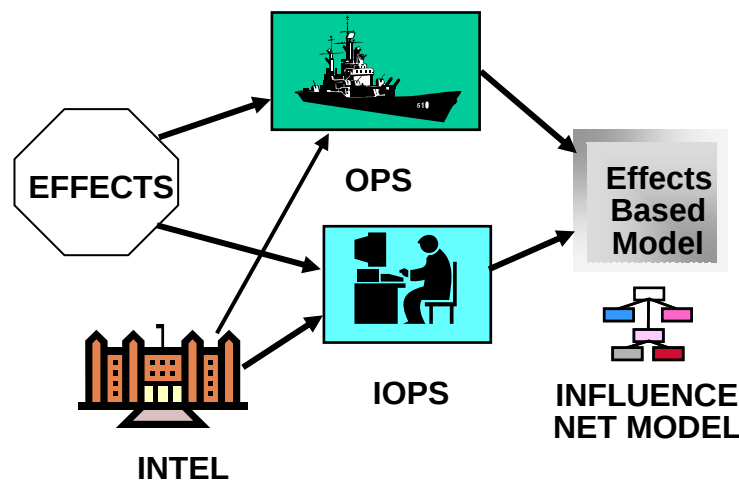


Figure 2.3 Development of the Influence net model

Bayesian nets (Jensen, 1996) have been used for a variety of applications. One of the most common uses is classification or diagnosis. A model of a situation is created with variables that represent causes and symptoms along with associations between those variables. Whenever certain symptoms are observed, the model is used to calculate the likelihood of the various potential causes of those symptoms. Once the most likely cause is identified, based on the set of symptoms, the “best” solution to the problem can be implemented.

One of the challenges in using Bayesian Nets is that the process of computing the inferences in Bayesian nets is NP hard. This means that the computational burden needed to “solve” a Bayesian net increases dramatically as the size of the network increases.

Influence nets (Rosen, 1996) are inspired by and are similar in form to Bayesian nets. Because they are simpler to construct and computationally less burdensome than the Bayesian net, they can support the development of models of situations by a group of subject matter experts who need not be experienced in Bayesian nets. A fundamental assumption used in their construction causes them to behave in a similar but not identical manner to Bayesian net of the same form. This assumption allows timing information to be introduced which is a fundamental enabler of the approach that was used to support EBO.

In general the approaches used by both CAT developed at AFRL and influence nets developed at GMU and SAIC are based on models of causality or causal models. CAT uses Bayesian nets to represent the models of causality. The Pythia tool developed under this effort uses Influence nets. While similar to Bayesian nets, influence net use a different, less computationally burdensome algorithm to “solve” the influence net that has the same characteristics as the Bayesian Net. In general, both methods of “solving” the net result in similar results.

While it has been clear for some time that Bayesian nets could be very useful in analyzing the reactive behavior of various actors to various actionable events, researchers discovered some limitations to their use. Rosen and Smith (1996), found that while the analyst responsible for analyzing politico-military situations understood that Bayesian nets could be useful in assessing cumulative impacts of complex situation, the majority did not have the experience required to use the Bayesian net software tools available. In addition, in most causes, the analyst did not have the information resources to fully specify all the conditional probabilities required in a Bayesian net. Specifying the conditional probabilities can be a reasonable task if the Bayesian

net is simple and contains nodes that have only one or two parents. In many probabilistic models of situations, a node can easily have four to eight parents. Because the size of the probability table grows exponentially with the number of parents, deciding on the requisite conditional probability values can be a daunting task for subject matter experts even if they are familiar with probability theory.

These problems can be further exacerbated by the need in many cases for a team of subject matter experts with diverse areas of expertise to collaborate on the creation and examination of the models of the situation.

To address these concerns, Chang, et al (1994). developed the Causal Strength Logic and Rosen and Smith (1996) developed a Unix based software implementation called the Situational Influence Assessment Module (SIAM) that is used to create influence nets. Three goals were considered in this development: (1) generate a user interface logic that requires a relatively small number of value (probability) assignments, which can be expanded to a full Bayesian model. (2) permit the users to modify the model with parameters that are meaningful to them and (3) provide a user-interface that follows consistent inference logic understandable to the user.

Thus, influence nets were developed as a variant of the traditional Bayesian net structure and allow the creation of useful models by analysts and subject matter experts who are unable to spend the time needed to fully specify a complete Bayesian net, if it were at all possible. This is accomplished by using independence of causal influence assumptions to simplify both knowledge elicitation and inferencing.

The Influence Net model captures an analyst's understanding of a causal system by a) a graphical depiction of the cause and effect relationship, and b) quantitative information regarding the strengths of these causal relationships. The approach captures this cause and effect information with the help of user-defined values for a set of parameters. Given this information, Bayesian nets for the causal model are generated.

Influence nets and Bayesian nets are static probabilistic models; they do not take into account temporal aspects in relating causes and effects. However, they serve an effective role in relating actions to events and in winnowing out the large number of possible combinations. The result of

the sensitivity analysis is the determination of a number of actionable events that appear to produce the desired effects and give an estimate of the extent to which the goal can be achieved.

Influence net models are based on causality. Causality helps facilitate understanding and communication and generally reduces computational complexity by creating models with a minimum of connectivity within the directed acyclic graph. It is partly for these reasons that the influence nets require relationships between nodes to be causal. More importantly, the causality restriction on the design of influence nets, for the purpose of developing and evaluating COAs, makes it possible to incorporate time into these models.

In creating the influence net, the modeler uses the reasoning that if proposition A becomes true (or false), then it will cause proposition B to become true (or false) with some probability. The causal relationship implies a precedence relationship between the propositions. This means that proposition B should not be triggered before proposition A, if proposition A is the sole cause of proposition B. Furthermore, there is an implied but unspecified mechanism by which a cause can trigger an effect. The implied mechanisms impose functional relationships between a cause and its effects along with arbitrary disturbances. These disturbances are not directly known, but can be reflected by a probability distribution that is incorporated in the model.

It is a fundamental premise of this research that in creating an influence net of a situation, the causal influencing mechanisms are realized by a real world phenomenon to which a time delay may be associated. In many cases, influence nets model the effects of command and control or distributed decision making processes. In these models, the nodes are either actionable events or propositions about the results of a C2 process. The actionable events, the source nodes in the influence net, can be associated with a time stamp. They fall into two classes: actionable events that are caused by some underlying process over which the planners have control so that the time of the occurrence of each event can be controlled, and events about which planners have knowledge of the time when they will occur, e.g. moonrise will occur at 2212 hours on July 27. The nodes representing propositions about the results of a C2 process can be grouped into three categories. The first are propositions about sensors; they are either sensor events (a radar detects an aircraft) or the state of a sensor (the radar is operating). The second category contains propositions about decisions, (the leader decides to negotiate or issues the launch command). In Influence net modeling, the probability of a proposition about a decision changes when the

probability about propositions that influence that decision change. The third category of propositions concerns actions (a missile is launched, an aircraft is shot down, etc.). In the C2 system, the evidence of the truth or non-truth of a proposition is transferred from one process to another over some transfer mechanism such as a communications channel or courier system.

The locality principle of influence nets and Bayesian nets has important implications. At any instance, the probability of a proposition is dependent only on the known set of probabilities of its parents. This implies that the influence net is an abstracted model of a set of interconnected distributed processes. Details of the processes are not modeled, only the probability of the propositions about the processes. The distributed processes are governed by concurrent and asynchronous events. However, influence nets and other static models do not capture the dynamic behavior of these distributed processes. They only reveal the final result of any set of stimuli.

AFOSR/NM sponsored research by the GMU System Architectures Laboratory has shown that it is possible to enhance these models so that the impact of timing of the inputs on the outcomes/effects can be determined.⁴ This impact can be represented by the timed sequence of changes in the likelihood of the outcomes/effects determined by the timing of the actionable events. The sequence of changes in probability is called the *probability profile*. It is a key measure of the effectiveness of a COA that can be used to evaluate COAs during their development and to determine when and how to change the COA during execution. In addition, the timed probability profiles of observable events can indicate *time windows* when those events are most likely to occur to support the scheduling of ISR assets to observe those events.

The goal is to incorporate knowledge about the time delays of the mechanisms into the model based on the structure of the influence net that will reflect the concurrent and distributed nature of the underlying process. The resultant model will generate a timed sequence of probability changes of each proposition for a given set of timed initial causal events: a probability profile of the change in the likelihood of a proposition as a function of time. Thus a probability profile is composed of a set of time windows. In each time window there is a probability that the proposition about an event or state is true. The probability is based on the state of the evidence

⁴ AFOSR Grant to GMU on "Time Sensitive Control of Aerospace Operations, F49620-01-1-0008."

in the model during the time window, specifically the state of the probabilities of the set of parents of the proposition using the locality principle.

Because of the use of the assumption of independence of causal influences and the inherent locality principle, it is possible to associate time with the arcs of the influence net. These times represent the amount of time it takes for knowledge about a change in the status of any variable to be propagated by some real world phenomenon to node that is affected by that change. Influence nets with temporal information have been defined as a result of this project as Timed Influence Nets (TIN).

Once time has been added to the influence net, it represents a dynamic system composed of a set of distributed processes. This new model can generate a probability profile for each node in the net including nodes that represent to prime objectives in the situation. Each probability profile consists of a timed sequence of probability values for the node. Both the probability values and the timed sequence are dependent on not only the certainty of the actionable events, but also on the temporal relationships between those events. The final values in the sequences are the same values provided by the standard un-timed influence net. The intermediate values in the probability profile are interpreted as the probability of the proposition being true during the time interval of that value in the profile.

Because of the need to assess the impact of the sequence and timing of the actions to be taken, once the Influence net of the situation has been developed, the situation analyst converts it into a Timed Influence Net which is an executable model that allows the introduction of temporal aspects (Fig. 2.4). With TINs, three general classes of temporal information are added. The first captures time delays associated with the propagation and processing of reactions of causal or influencing actions or events and the second captures information about the time of occurrence of actions or events. Within the second class of temporal information one concept of persistence of actions in which actions can be allowed to start and stop at different time intervals is included. In particular, information is added to account for temporal and logical sequencing of actionable events. *A particular sequence of actionable events represents an alternative Course of Action.* Note that in a threat environment proper sequencing is critical; reversal of two operations can endanger lives and affect critical operations. Consider a trivial example: wear protective equipment then step into a hazardous environment vs. step into a hazardous environment and

then put on protective equipment. While this example is obvious, such reversals are not easily observed in a complex scenario with many concurrent tasks. The TIN model brings these issues to the fore and also created the need to extend the capability of the basic TIN.

As was noted above, the final probability values of a TIN execution are the same as those in the untimed influence net. As illustrated in the hazardous environment example, in real life situations the order of the execution of a set of actions may have very different impacts on the effects. The reason for these differences can be explained in part by concepts of persistence. One concept deals with situations in which the probability of an event at a particular time instance depends upon its probability in the previous time interval. The basic TIN fails to capture this phenomenon. A second persistence concept, not captured in TINs, is the notion of time-varying influences. They assume that the strength of an influence remains constant for all time. This results in memoryless TINs; no matter what the sequence or timing of actions is, the final probability of achieving the desired effects is always the same. In some real world cases, the intensity of an influence can change over time. In general they tend to decay. These types of persistence issues can mean that the sequence and timing of actions and time delays in the TIN will impact the probability profiles that are generated including the final value of the profile.

As part of this project, an enhancement to the TIN formalism was developed to incorporate these concepts of persistence. The result was the creation of Dynamic Influence Nets or DINs. The DIN concept has been incorporate in Pythia. This allows the modelers to include a third category of temporal information in the models, the concept of persistence of influence, in addition to the time of actions and the time delays.

The Timed Influence Net or its enhancement as a Dynamic Influence Net, when properly initialized with a scenario, can be used in execution mode to test the various COAs to determine their effectiveness by generating the timed probability profile for the particular COA. (Fig. 2.4) The results can be shown to the commander and the planning staff in which the situation is presented along with alternative Courses of Action and their assessment.⁵ A Commander can

⁵ In August 2000, CAESAR II/EB was used in the Naval War College's two-week long Global 2000 war game. The results of the CAESAR II/EB analyses were displayed successfully on the Knowledge Wall both at the War Gaming Center and at the CVN Coronado in San Diego.

then make an informed choice and direct the planning staff to prepare the detailed plan for the chosen COA.

Once the TIN or DIN has been created, the next step of the overall EBO process is for the operational planners and the situation analysts to use the influence net and executable models (TINs) to select candidate COAs. The concept for this procedure is shown in Fig. 2.5. The TIN and the probability profile it produces reveal dynamic changes in the likelihood of achieving the effect or objective of the COA. These dynamic changes are not indicated by the Influence Net model, which provides only a single equilibrium value. Traditionally, the Influence Net is used to identify a set of actionable events that provide the best chance of achieving a desired effect. The time when the actionable events are made to occur produces a sequence of changes in the probability of the overall effect that can vary significantly, in discrete steps over time. These variations mean that the likelihood of achieving the objectives of the actionable events depends on the timing of those events. Thus some COAs will be preferred over others. In the example of Figure 2.5, COA 1 is preferred over COA 2 because it has the higher probability values at all time points and reaches the highest probability the fastest.

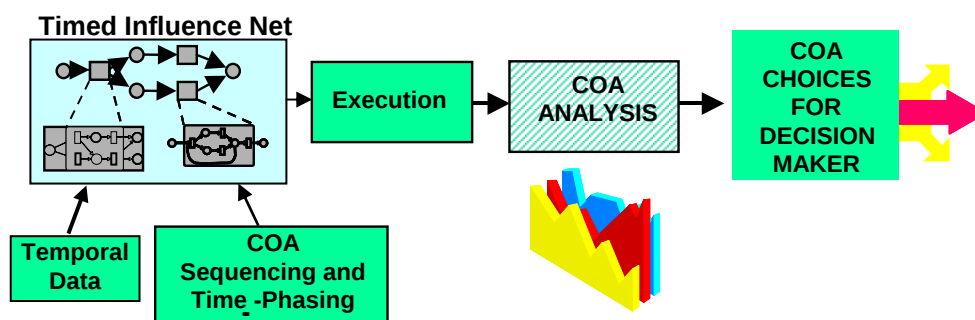


Figure 2.4 Analysis of Alternatives

An execution of the TIN can be used before, during, or after planning to generate and evaluate the timed probability profile for producing the desired effects. An iterative trial and error process can be used to generate probability profiles of COAs and corresponding plans for comparison and selection. If a probability profile has unacceptable features, adjustments must be made to the COA and the corresponding plan to change the timing of the actionable events to

eliminate those features. Unfortunately, it is not easy to determine what those changes should be by inspecting the TIN. Thus, a process of trial and evaluation is employed until an acceptable probability profile is found.

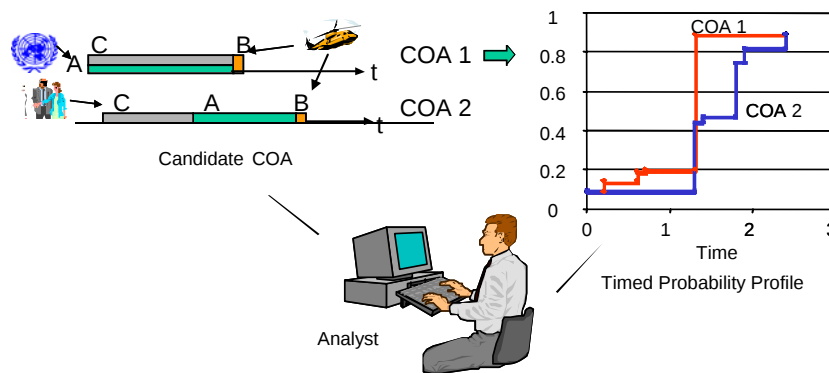


Figure 2.5 Decision Support for COA

Part of the objective of this effort was to develop procedures and algorithms that mitigated the trial and error nature of determining an acceptable COA. A two step procedure has been created that improves the development and evaluation of Courses of Action and enhances the collaboration between analysts that create COAs and planners that develop the detailed plans that instantiate them. The first step uses sensitivity analysis to identify and rank order the impact that each potential action has on the desired effects. This algorithm has been expanded to identify the combinations of actions that cause the probability of a desired effect above a certain threshold. These capabilities enable an analyst to determine the combination of actions that are likely to give the best chance of eventually achieving the desired effects. However, this analysis does not consider the temporal aspects of the problem. To examine the temporal considerations, one must consider the probability profiles that are associated with the timing of actions and the resultant “trajectory” of the probability profile. To provide a deeper insight into the trajectory taken by a probability profile, algorithms that use a temporal logic that answer several types of temporal queries is also used to perform ‘What-If’ analysis that provides the potential temporal relationship which should exist between two actionable events in order to achieve a desired effect at a particular point in time. In addition, an Evolutionary Algorithm (EA) based approach has been developed under this contract to automate the process of effective COAs determination.

The approach generates several alternative courses of action that have a *similar* likelihood of producing the desired effect.

Having selected a COA, a detailed executable plan or COA Implementation Plan is developed in the third activity of Figure 2.1. The detailed plan determines and schedules the resources that will be used to cause the actionable events and tasks the ISR assets against the observable actions indicated in the Influence net or TIN model. The existence of the TIN model gives the opportunity to test the plans and *also to monitor their execution by inserting actual events as they are observed*.

Once the detailed executable plan has been developed, evaluated, and approved, it is used by controllers who provide directions and instructions to the operators who are carrying out the plan (Activity 4 of Figure 2.1).

The fifth activity involves the continual assessment of the detailed plan execution including assessment of the actionable events, the assessment of their effects, and the impact they have on achieving the goal. This assessment activity is called “Perform Effects Based Assessment” in Figure 2.1. The assessments correspond, very approximately, to Measures of Performance (MOPs), Measures of Effectiveness (MOEs), and Measures of Force Effectiveness (MOFEs). The set of models created during the COA development activity can be used to support these assessments.

During the execution of a plan, there are two major factors that can impact the expected effectiveness of that plan. First, the timing of the actionable events may change as the resources or assets perform the tasks in the plan. The impact of these timing changes in terms of the timed probability profile can be quickly examined using the TIN model. If anticipated timing changes (such as delays) have an adverse effect on the probability profile, adjustments to the timing can be determined that will bring the profile within acceptable levels. The second type of changes involves the occurrence or non-occurrence of anticipated events in the Influence net and the corresponding TIN. These events can be the ones for which the ISR assets have been tasked. In the planning mode, events were assumed to occur with some probability; in the assessment mode, events occur with probability one or zero – depending on whether they occurred or not. The impact of these changes on the timed probability profiles can be observed by updating the elements of the models. To do this, two techniques were developed. The first is based on a

heuristic algorithm that updates the marginal probability values in the influence net when new probability values are provided for nodes that do not represent the actionable events. The second uses the standard Time Sliced Bayesian Network algorithm to do the calculations.

The overall development effort was driven by the combination of the EBO Concept and the supporting processes and technology. The effort consisted of developing and testing new and existing algorithms, described in detail in Section 4 and in the references, to support the various steps in the process designed to address the five elements of the basic EBO problem described in Section 2.1. Once the algorithms were vetted, the effort focused on creating an integrated tool suite that could support the entire EBO concept and process.

3. METHODS, ASSUMPTIONS, AND PROCEDURES

The essence of the GMU technical effort was to enhance and integrate the suite of modeling and simulation tools that had been under development from 1996 to 2001 with support from ONR, AFOSR, AFRL/IF and SPAWARSYSCEN. This integration and synthesis provided specific capabilities required for the EBO. These tools have been designed to allow analysts to produce and analyze the models of the EBO Planning Problem.

At the start of the effort, five tools that were in various states of development:

Campaign Assessment Tool (CAT). This was an application initially developed at AFRL/IF by Dr. John Lemmer. He had provided the source code of this Windows based application to GMU in 1999. GMU extended the source code to include influence net technology in 2000, in collaboration with Dr. Lemmer. This GMU version of CAT incorporated a variety of algorithms for calculating the influence of actionable events (potential actions that can be taken in a COA) on the probability of achieving the desired effect. The enhancement to CAT, seen as a general purpose Influence net tool, was that it subsumed the Influence net modeling approach used by SIAM, a commercial Influence net modeling tool produced by SAIC and used by the Intelligence community. The original GMU version of CAT could export a file that contained information about the structure of the influence net. The exported file was the interface to the second tool (COA/EB) that converted the influence net to a discrete event model that could be executed and allowed temporal information to be added.

COA/EB. This Unix based tool is an integrated set of algorithms built using Design/CPN that reads an Influence net model file produced by CAT (or version 2 of SIAM) and convert it automatically into an executable model that can be analyzed as well as used in simulation mode. Design/CPN is an industrial-strength implementation of Hierarchical Colored Petri Nets that was developed and maintained by the CPN Group at the University of Aarhus⁶ in Denmark and available for free to qualified users⁷. It contains an Editor for the creation of Hierarchical Colored Petri Net models, a Simulator for carrying out simulations, as well as

⁶ The CPN Group at Aarhus and the System Architecture Lab at GMU have had a collaborative agreement that includes exchange of students and researchers. The CPN group, as subcontractors to GMU, developed the web browser interface to Design/CPN.

⁷ Design/CPN has been replaced with a tool called CPN Tools.

an expanding set of analysis tools such as the Occurrence Graph Analyzer. The set of algorithms developed by GMU within Design/CPN allows the generation of alternative Courses of Action (COA) designed to achieve, in a probabilistic sense, the desired effects. It contains new algorithms for the selection of COAs that meet temporal constraints. Another version of this tool had been developed with AFIWC support through AFRL to enable the integration of lethal and non-lethal Information Operations. This was demonstrated in the Spring of 2000; the project was completed in September 2000. The user interface to this tool was through a web browser that provided a set of forms for the user to fill out to input the temporal information. The user was not shown the colored Petri net but instead was provided with a file that could be opened in Power Point that showed the probability profile generated by the course of action that had been input by the user through the browser interface. Those two developments were merged into one tool.

TEMPER. This was a Unix based tool built with AFOSR/NM support that implemented an extension of Temporal Logic called Point-Interval Temporal Logic (PITL). It analyzes automatically temporal constraints associated with the development of Courses of Action or of detailed plans and allows for “what if” questions associated with Dynamic Battle Control. This application also uses Design/CPN as its platform. It was integrated with COA/EB in the sense that it ran in the same environment.

Real-Time Execution Monitor (R-TEM). This was a new tool concept, not fully developed at the time, that exploits the executable models developed for the Course of Action development and uses them for the monitoring of the execution as the COA evolves. The concept supports the assessment of the COA execution by calculating the changing probabilities of achieving the desired effects. When execution starts, planned events may or may not occur and, if they occur, they may not do so at the planned or assumed time. In addition, ISR may indicate that some effect contained in the model has or has not occurred during some time window. The objective was to develop new algorithms that can use as input the occurrence of an event (probability 1), either an action or an effect, and the time it occurred and assess the impact on the desired effect. TEMPER is also integrated in this algorithm because the time of occurrence of an event may be such that it can cause dynamic restructuring of the remaining cause and affect relationships.

Visualization Module (V-Mod). The initial capability of this module was developed with Office of Naval Research (ONR Code 342) support and in collaboration with SPAWAR System Center - San Diego. Its purpose is to display the results of the COA/EB tool. The initial version of this capability was tested successfully in the collaborative environment of Global 2000 where the visualization of the results were presented on the Knowledge Wall at

the Naval War College in Newport, RI and the command center of the aircraft carrier Coronado in San Diego. This capability was developed further to provide the data in the format needed for the EBO concept.

At the start of the contract, four of the five tools were loosely integrated as shown in Figure 3.1. The influence net was created using CAT and Static Analysis was performed to identify the best set of actions for a COA. This was done in a Windows environment. A text file containing a description of the resulting Influence net exported by CAT. This file was then accessed by COA/EB that operated in the Unix environment. Design CPN was the discrete event executable model that was used to allow temporal information to be added and probability profiles generated. The interface for the user was through a web browser (shown as the small windows that the stick-figure “Analyst” is working with in Figure 3.1). The analyst uses the interface to determine the best sequence and timing of actions, determine the time windows when the desired effects are likely to occur, assess time windows of risk, and time window when ISR could be tasked to look for indicators.

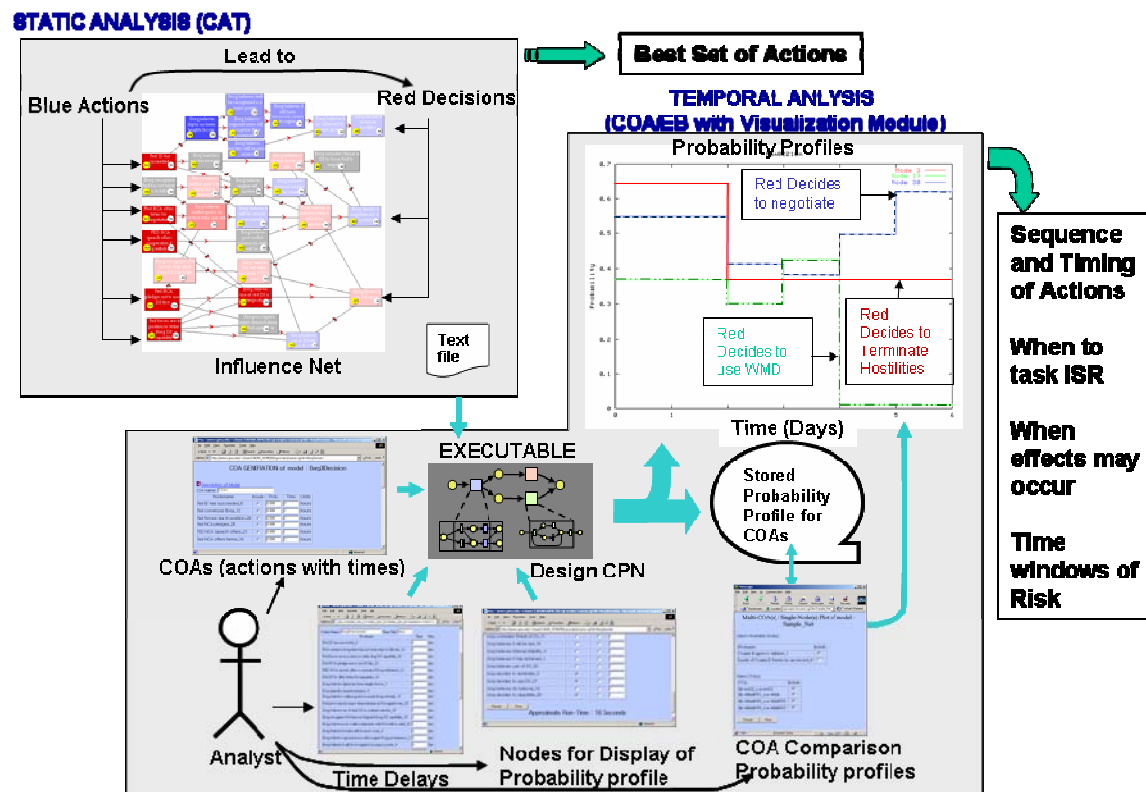


Figure 3.1 The Initial Suite of Tools

The suite of tools was contained in a single PC or laptop by using VM Ware, a COTS product that could be installed in the Windows environment and in which the Linux Unix based

operating system could be installed with the Design CPN tool. Thus both Windows and Unix could be run concurrently on the same machine.

Initially effort was composed of five interlocking tasks: Enhancement and Integration, Program Participation, Demonstration and Evaluation, Software Delivery, and Program Management. During the course of the effort, these tasks were expanded with a series of Grant Addendums.

The first task (SOW 4.1) was one of the major development efforts. It involved the enhancement of the Campaign Assessment Tool (CAT) developed by AFRL/IF and provided to GMU for research and development purposes. GMU assimilated the code that comprises CAT into the suite of tools it has developed over that past 15 years called CAESAR II. The resultant tool was called CAESAR II/EB and became the test bed environment for the development and testing of the algorithms and tools that would comprise the integrated tool suite to support EBO. The enhancements to CAESAR II/EB include adding alternative algorithms for the computation of the probabilities of achieving the desired effects, the incorporation of temporal information in the inputs to CAT, and the passing of that information automatically to the colored Petri net (CPN) executable model of CAESAR II/EB. In addition, the task specifies the integrations of several modules into the tool. The modules include an assessment module for analyzing and comparing several courses of action, the TEMPER logic module, the R-TEM module to support assessment, and a V-module to support visualization.

Task 4.2 required participation in program reviews, workshops, and demonstrations, and collaboration with other government and contractor entities, as required and feasible, on the development of the tools and techniques for EBO.

Task 4.3 required a series of demonstrations and evaluations of the improvement in the capability to support EBO using the tools and techniques developed in the program.

Tasks 4.4 and 4.5 included the delivery of the software developed under this program and the program management.

In August 2003 AFRL/IF granted an engineering change proposal that added an additional task under task 4.2 to develop an architecture description of the Joint Air Estimate Process to promote a better understand of the impact of an Effects Based Approach on that process. In January 2004 an additional task was added for support to the AFRL/IF participation in the Effects Based Operations element of the Air Force JEFX 04. In September 2004 two final tasks were added to

refine the technology development. One task was added to task 4.1 and the other to Task 4.2. The addition to Task 4.1 required the extension of the TEMPER module that was being integrated under a grant from AFOSR into the CAESAR II/EB tool suite to include newly developed point-interval logic and its application to model and plan time-sensitive aspects of a mission course of action. The addition to Task 4.2 was to investigate the impact of making the effects based planning, execution, and assessment process more dynamic. Special emphasis was to be applied to the dynamic effects based execution orders and near real time combat campaign assessments.

A four step procedure was used for the execution of the contract:

1. Use the GMU version of CAT as the platform and environment to develop new algorithms to enhance the EBO analysis capability of the existing suite of tools. The development of the algorithms to handle concepts of persistence and the development of the R-TEM module were included in this step.
2. Convert the discrete event model that was run in the Unix environment into an algorithm that would run in CAT. Leverage work sponsored by AFOSR to convert the TEMPER application to an API that could be incorporated in CAT.
3. Participate in various war games and exercises using the tools being developed to determine their potential usefulness and to refine the operational concept for EBO.
4. Once a satisfactory set of algorithms had been developed in the CAT environment, re-code all algorithms in a new Windows based environment. Test the new tool and develop a User's Manual for it.

4. RESULTS AND DISCUSSION

The first task of this development effort was to enhance the Campaign Assessment Tool (CAT), which was originally developed by the AFRL (Rome) as a Bayesian Network (BN) based tool to capture static uncertain situations. The plan was to incorporate the various modules into the CAT code. CAT's knowledge elicitation interface was based on Recursive Noisy-Or [Lemmer and Gossink 2004]. GMU modified the source code of CAT to make it Influence Net (IN) compliant. New graphical interfaces were designed to capture the Causal Strength (CAST) logic [Chang et al. 1994; Rosen and Smith 1996] parameters which are required to specify an IN. Algorithms were added to perform *static probability propagation* and *sensitivity analysis*. At this point the enhanced CAT had the same features that were used in SIAM for influence net modeling, and the enhanced CAT could export a file that was read by the CAESAR II/EB code written in Design/CPN. Analyst entered the temporal information in the Design/CPN environment which then was run as a simulator to generate the probability profiles. The Design/CPN code was further enhanced with a new and more compact translation from Timed Influence Nets (described in Section 4.1.2) with logic into colored Petri nets. The original CAESAR II/EB translation has the property that the net structure of the colored Petri net will be the same for all translated timed influence nets – only the initial marking changes depending on the actual timed influence net. This more compact translation avoids the generation of simulation code for each timed influence net.

At this point a decision was made to try and perform the executable model simulations directly in the CAT code rather than using the Design/CPN tool. This would provide the advantage of having all of the analysis capabilities in one environment rather than the two environments required by the original CAESAR II/EB.

To accomplish this CAT was further modified by GMU to capture certain types of temporal information. These enhancements allowed a system analyst to model a dynamic uncertain situation using Timed Influence Nets (TIN) [Wagenhals 2000]. Probability propagation algorithms that make use of this temporal information were added to the software. This approach replicated the capability afforded by the Design CPN colored Petri Net tool that was needed to provide the executable model of the influence net while eliminating the need to run a separate

tool in a separate environment. A heuristic algorithm was also developed to perform *belief updating* [Haider and Levis 2004]. The complete suite of tools retained the CAESAR II/EB name. It was tested and used in several war games, such as Global 2000 and 2001 and Millennium Challenge 02. The successful use of this single environment version of CAESAR II/EB in these war games resulted in the decision to continue the tool development in a single environment and not continue with the colored Petri Net implementation. The results of those experiments were published in several conference papers [Wagenhals and Levis 2000, 2001, 2002; Wagenhals et al. 2001; Wentz and Wagenhals 2004].

While the version of the CAT tool that was used provided a viable test bed to support the development of the new algorithms, it had several limitations that prohibited the full development of the EBO capability. The graphical user interface (GUI) of CAESAR II/EB was not very user-friendly. Furthermore, the architecture of the tool did not support the incorporation of many desired GUI features, and it also made it difficult to merge some of the related modules (such as TEMPER, R-TEM) into an integrated environment. These issues along with the lessons learned from the war games encouraged the development of a new software environment from scratch. The software was subsequently named Pythia and it contains all the algorithms that were available in CAESAR II/EB plus many other algorithms to support the modeling and analysis of an uncertain situation using a TIN based framework. Its architecture allowed the integration of assessment module, real time execution monitoring (R-TEM) module, the TEMPER logic module, and Visualization module (V-Mod) into a single environment.

As the development proceeded, it was apparent that formal definitions were needed to describe the modeling constructs that were being used. Thus formal definitions were created for Influence Nets and Timed Influence Nets. The definition of the TINs was extended to include concepts of persistence, and this extension was called Dynamic Influence Nets (DINs).

The rest of this section summarize the results of the integrated EBO tool suite development: Section 4.1 defines and describes Influence Nets, Timed Influence Nets, and Dynamic Influence Nets. Section 4.2 describes the set of algorithms that were developed to support the overall EBO concept and process. These algorithms are available in Pythia. Appendix A provides the Pythia manual and tutorial. It contains a full description of Pythia graphical interface features and provides examples of using Pythia to support the EBO concept activities.

4.1 Definitions

4.1.1 Influence Nets (INs)

Influence Nets (INs), an instance of the Bayesian framework, were proposed a decade ago to overcome the intractability issues present in BNs. They employ an approximation inference algorithm, termed as loopy belief propagation [Kschischang and Frey, 1998; McEliece et al., 1998; Murphy et al., 1999], and non-probabilistic knowledge acquisition interface, termed as the CAST logic [Chang et al., 1994; Rosen and Smith, 1996].

The modeling of the causal relationships using an IN is accomplished by connecting a set of actionable events and a set of desired effects through chains of cause and effect relationships. The strength of such relationships is specified using the CAST logic parameters instead of the probabilities. The required probabilities are internally generated by the CAST logic with the help of user-defined parameters. The Influence Nets are therefore appropriate for the following situations: i) for modeling situations in which it is difficult to fully specify all conditional probability values ii) and/or the estimates of conditional probabilities are subjective, and iii) estimates for the conditional probabilities cannot be obtained from empirical data, e.g., when modeling potential human reactions and beliefs.

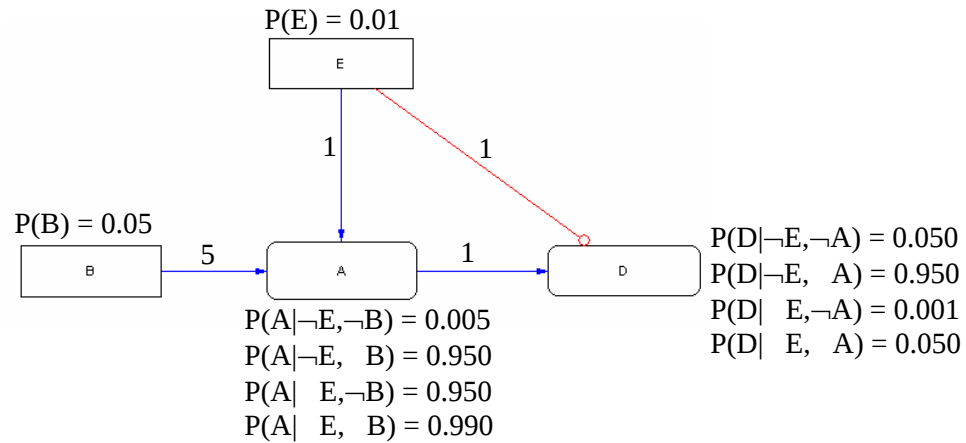


Figure 4.1 A Sample Influence Net

The actionable events in an IN are drawn as root nodes (nodes without incoming edges). A desired effect, or an objective the decision maker is interested in, is modeled as a leaf node (node without outgoing edges). Typically, the root nodes are drawn as rectangles, while the non-root

nodes are drawn as rounded rectangles. Consider the IN of Figure 4.1. The text associated with the non-root nodes represents the corresponding conditional probability values obtained from the CAST logic parameters (not shown in the figure) while the text associated with the root nodes represents the prior probabilities. The texts associated with arcs are time delays and are explained in Section 4.2. The belief propagation scheme used in INs is based on independence of parents assumptions. Thus, the marginal probability of a non-root node is computed with the help of its Conditional Probability Table (CPT) and the prior probabilities of its parents. For instance, the marginal probability of variable A in Figure 4.1 is computed as

$$\begin{aligned}
 P(A) &= P(A \mid \neg B, \neg E)P(\neg B)P(\neg E) + P(A \mid \neg B, E)P(\neg B)P(E) + P(A \mid B, \neg E)P(B)P(\neg E) \\
 &\quad + P(A \mid B, E)P(B)P(E) \\
 &= 0.005 \times 0.95 \times 0.99 + 0.95 \times 0.95 \times 0.01 + 0.95 \times 0.05 \times 0.99 + 0.99 \times 0.05 \times 0.01 \\
 &= 0.06
 \end{aligned}$$

The probability of D is then computed by using its CPT and the marginal probabilities of A (computed above) and E. Thus

$$\begin{aligned}
 P(D) &= P(D \mid \neg E, \neg A)P(\neg E)P(\neg A) + P(D \mid \neg E, A)P(\neg E)P(A) + P(D \mid E, \neg A)P(E)P(\neg A) \\
 &\quad + P(D \mid E, A)P(E)P(A) \\
 &= 0.05 \times 0.99 \times 0.94 + 0.95 \times 0.99 \times 0.06 + 0.001 \times 0.01 \times 0.94 + 0.05 \times 0.01 \times 0.06 \\
 &= 0.11
 \end{aligned}$$

Formally, Influence Nets are Directed Acyclic Graphs (DAGs) where nodes in the graph represent random variables, while the edges between pairs of variables represent causal relationships. The following items characterize an IN:

1. A set of random variables that makes up the nodes of an IN. All the variables in the IN have binary states.
2. A set of directed links that connect pairs of nodes.
3. Associated with each link is a pair of CAST Logic parameters that shows the causal strength of the link (usually denoted as g and h values).
4. Each non-root node has an associated CAST Logic parameter (denoted as the baseline probability), while a prior probability is associated with each root node.

Definition 1

An *Influence Net* is a tuple (V, E, C, B) where

V : set of Nodes,

E : set of Edges,

C represents causal strengths:

$$E \rightarrow \{ (h, g) \text{ such that } -1 < h, g < 1 \},$$

B represents a Baseline or Prior probability:

$$V \rightarrow [0,1]$$

4.1.2 Timed Influence Nets

Influence Nets were designed to capture *static* interdependencies among variables in a system. However, a situation where the impact of a variable takes some time to reach the affected variable(s) cannot be modeled by either of the two approaches. Wagenhals et al. [1998] have added a special set of temporal constructs to the basic formalism of Influence Nets. The Influence Nets with these additional temporal constructs are called Timed Influence Nets (TINs) [Haider and Levis, 2004; Haider and Zaidi, 2004]. The temporal constructs allow a system modeler to specify delays associated with nodes and arcs. These delays may represent the information processing and communication delays present in a given situation. For example, in Figure 4.1, the inscription associated with each arc shows the corresponding time delay it takes for a parent node to influence a child node. For instance, event B influences the occurrence of event A in 5 time units.

TINs have been experimentally used in the area of Effects Based Operations (EBOs) for evaluating alternate courses of actions and their effectiveness to mission objectives. [Wagenhals and Levis, 2000; Wagenhals and Levis, 2001; Wagenhals et al., 2003] The purpose of building a TIN is to evaluate and compare the performance of alternative courses of action. The impact of a selected course of action on the desired effect is analyzed with the help of a *probability profile*. Consider the net shown in Figure 4.1. Suppose it is decided that actions B and E are taken at time 1 and 7, respectively. Because of the propagation delay associated with each arc, the influences of these actions impact event D over a period of time. As a result, the probability of D

changes at a different time instants. A probability profile draws these probabilities against the corresponding time line. The probability profile of event D is shown in Figure 4.2.

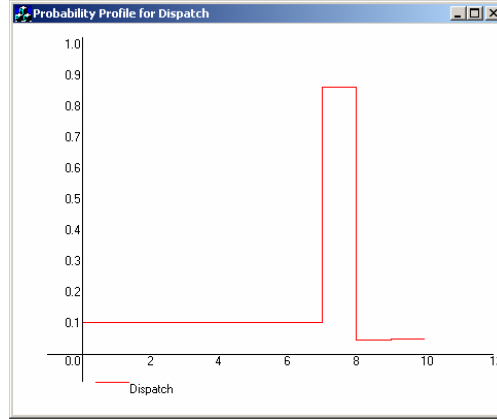


Figure 4.2 Probability Profiles of Event D

The following items characterize a TIN:

1. A set of random variables that makes up the nodes of a TIN. All the variables in the TIN have binary states.
2. A set of directed links that connect pairs of nodes.
3. Each link has associated with it a pair of parameters that shows the causal strength of the link (usually denoted as g and h values).
4. Each non-root node has an associated baseline probability, while a prior probability is associated with each root node.
5. Each link has a corresponding delay d (where $d \geq 0$) that represents the communication delay.
6. Each node has a corresponding delay e (where $e \geq 0$) that represents the information processing delay.
7. A pair (p, t) for each root node, where p is a list of real numbers representing probability values. For each probability value, a corresponding time interval is defined in t . In general, (p, t) is defined as

$$([p_1, p_2, \dots, p_n], [[t_{11}, t_{12}], [t_{21}, t_{22}], \dots, [t_{n1}, t_{n2}]]),$$

$$\text{where } t_{i1} < t_{i2} \text{ and } t_{ij} > 0 \forall i = 1, 2, \dots, n \text{ and } j = 1, 2$$

The last item in the above list is referred to as input scenario, or sometimes (informally) as course of action. Formally, a TIN is described by the following definition.

Definition 2 Timed Influence Net (TIN)

A TIN is a tuple $(\mathbf{V}, \mathbf{E}, \mathbf{C}, \mathbf{B}, \mathbf{D}_E, \mathbf{D}_V, \mathbf{A})$ where

$\mathbf{V}$: set of Nodes,

$\mathbf{E}$: set of Edges,

$\mathbf{C}$ represents causal strengths:

$$\mathbf{E} \rightarrow \{ (\mathbf{h}, \mathbf{g}) \text{ such that } -1 < \mathbf{h}, \mathbf{g} < 1 \},$$

$\mathbf{B}$ represents Baseline / Prior probability: $\mathbf{V} \rightarrow [0,1]$,

$\mathbf{D}_E$ represents Delays on Edges: $\mathbf{E} \rightarrow \mathbf{Z}^+$, (where $\mathbf{Z}^+$ represent the set of positive integers)

$\mathbf{D}_V$ represents Delays on Nodes: $\mathbf{V} \rightarrow \mathbf{Z}^+$, and

$\mathbf{A}$ (input scenario) represents the probabilities associated with the state of actions and the time associated with them.

$$\mathbf{A}: \mathbf{R} \rightarrow \{ ([p_1, p_2, \dots, p_n], [[t_{11}, t_{12}], [t_{21}, t_{22}], \dots, [t_{n1}, t_{n2}]] \}$$

$$\text{such that } p_i = [0, 1], t_{ij} \rightarrow \mathbf{Z}^* \text{ and } t_{i1} \leq t_{i2}, \forall i = 1, 2, \dots, n \text{ and } j = 1, 2 \text{ where } \mathbf{R} \subset \mathbf{V} \}$$

(where $\mathbf{Z}^*$ represent the set of nonzero positive integers)

4.1.3 Modeling of Time Varying Influences: Dynamic Influence Nets

In our daily life, events happen and they influence other relevant events. In many cases the intensity of their influences decays over time. Thus, an event having a very strong influence at the time of its occurrence on another event might have an insignificant influence after a certain period of time. In other words, the influence of an event is *time-variant*. The *time-variant* property also holds true for the state of an action. An action may be in two different states during two different time intervals. In TINs terminology, these two types of time-variant properties are referred to as *persistence*. The one related to the time dependent influence of an action is called *persistence of influence*, while the one related to the time-dependent state of an action is called *persistence of action*. Among these two types of persistence, a TIN models the latter one only. It assumes that the causal strength of the influences does not change over time, i.e., the underlying stochastic model is *stationary*. Thus, it lacks the ability to model persistence of influence.

To handle this limitation, an extension has been made to the TIN definition. The approach enables a system modeler to model *non-stationary* influences. Instead of asking a modeler to specify single-valued influences, the proposed approach would allow the modeler to specify various strengths of influences and their corresponding window of effectiveness as shown in Figure 4.3.

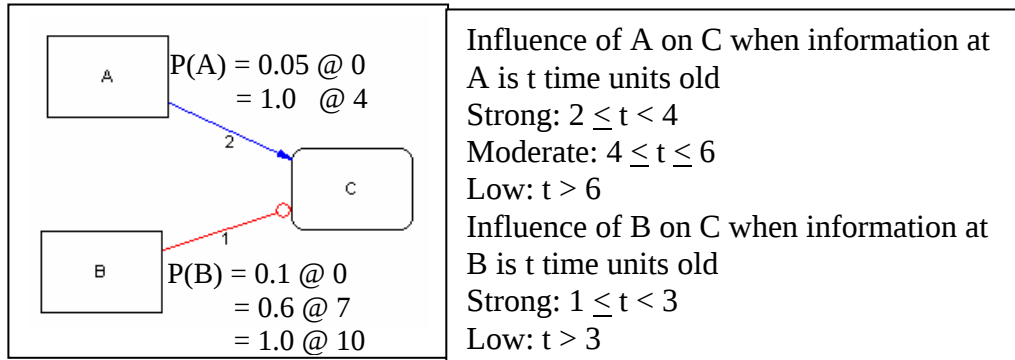


Figure 4.3 A TIN Having Time-Variant Influences

To incorporate these persistence concepts in the TIN requires structural and parametric changes in TINs. This enables a system modeler to observe the impact of repeated actions. For instance, an air-strike on a bridge makes it inoperable for several days. The current implementation of TINs would assume that the influence of air-strike remains the same throughout the campaign. It is obvious that the assumption is unrealistic. Furthermore, in the event of a new air-strike a TIN would discard the impact of the previous air-strike as the events in a TIN are assumed to be memory-less. The enhanced approach, which allows time-variant influences and incorporation of memory through self-loops, models this situation in a more intuitive manner. It should be noted that like other arcs in a TIN, a self-loop also represents influence – from the previous state of a node to its current state. Thus, time-variant parameters can be associated with a self-loop, too. For the air-strike example, the presence of self-loop would combine the influences of both (or many) air-strikes while the strength of the self-loop accounts for the time delay between the two air-strikes. If the timing of two air-strikes is far apart then there is almost no influence of the first strike on the operability of the bridge (provided that the bridge has been repaired), but if the two strikes occurred very close in time then their impact would be more destructive. In other words the impact of two actions on the effect convolves. The issue is further explained with the

help of the following example. Suppose the variables in the model of Figure 4.3 have the following descriptions:

A - Regional Countries Opposes Sanctions against Country R

B – Country G Threatens to Take Unilateral Actions against Country R

C – Leader of Country R Decides to Accept UN Demands

The model in Figure 4.3 shows that the effect C will be positively influenced by Action A, but negatively influenced by Action B. The time delay between Action A and effect C is two time units and the time delay between Action B and effect C is one time unit. The model also shows that the initial probabilities of A and B are 0.05 and 0.1, respectively. At time 4 the probability of A changes to 1.0. At time 7 the probability of B changes to 0.6, and at time 10 it changes again to 1.0. The information in the right side of Figure 4.3 shows the time varying influences of A and B on C. For example, when C receives an update on the probability of B at time 8, the influence of the information about A will be 2 units old, so C will use the “strong” influence factors for A. This is done by selecting the CAST Logic parameters for g and h (0.90, -0.90). But when it receives the next update from B at time 11, the influence from B will be 5 units old, so the influence of A will only be moderate for A; g and h (0.66, -0.66). If the TIN were being used, the strength of the influence of A as reflected in the CAST parameters would remain fixed at (0.90, -0.90) and thus the probability profile would be different.

Now let us further assume that the belief of event C at a particular time depends upon its own belief at a previous time instance though not very strongly. We further assume that this self dependency remains constant (does not vary with time), although it could be made to vary. This behavior is modeled by adding a self-loop having a low influence on event C. The resultant model is shown in Figure 4.4.

It is interesting to compare the probability profiles that result with the different models of persistence of influence. Figure 4.5 shows the comparison of the three profiles that could be generated based on the different models of persistence. Figure 4.5 (a) shows the probability profile for the TIN version of the model with no time varying influences. Figure 4.5 (b) shows the profile for Figure 4.4 with the time varying influences, but no self loop. Note the impact of the time varying influence. Because the influence of event A is much weaker when the change

in probability B occurs at time 11, the impact of B on C is much more dramatic. C does not account for positive influence of A that happened sometime earlier as much as it would if the influence of A had occurred closer in time to that of B. Figure 4.5 (c) shows the impact of the assumption that the belief of event C at a particular time depends upon its own belief at a previous time instance. Note that at time 8 when new information about A arrives at C, the probability of C does not decrease as much as it did in Figure 4.3 (b) because of previous value of C was relatively high so that tempered the negative impact of B. The same result also occurs at time 11.

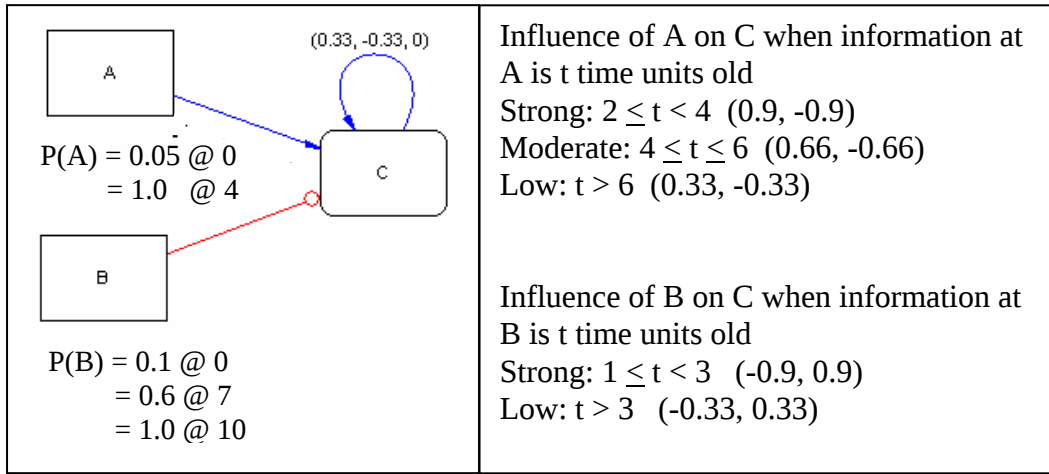
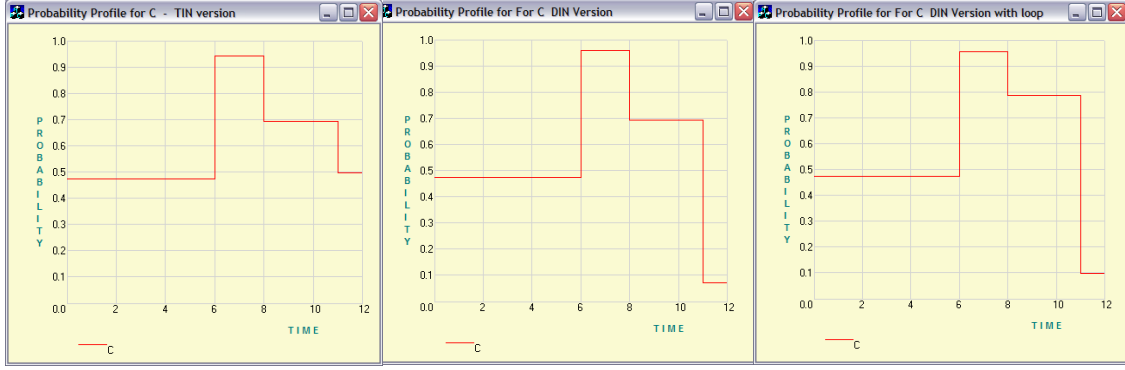


Figure 4.4 A TIN with Self-Loop and Time-Varying Influences

The incorporation of the new constructs (self-loop and time-varying parameters) in the framework of TIN based modeling and reasoning enhances the capabilities of this modeling paradigm in terms of capturing dynamic uncertain scenarios. A TIN with these additional constructs has been defined as Dynamic Influence Net (DIN). The following items characterize a DIN while a formal definition is given in Definition 3.



(a) Profile in a TIN

(b) Profile in a DIN

(c) Profile in a DIN with Loop

Figure 4.5 Comparison of Profiles Generated by a TIN and a DIN

1. The nodes of a DIN are set of random variables. All the variables in the DIN have binary states.
2. A set of directed links that connect pairs of nodes. A node can also have an optional self-loop.
3. A pair (c, t) for each link, where c is a list of tuples representing the CAST logic parameters. For each element in c , a corresponding time interval is defined in t . This interval represents the time during which the corresponding element in c is in effect. In general, (c, t) is defined as

$$[(h_1, g_1) (h_2, g_2), \dots, (h_n, g_n)], [(t_{11}, t_{12}), (t_{21}, t_{22}), \dots, (t_{n1}, t_{n2})]$$

where $t_{i1} < t_{i2}$ and $t_{ij} > 0 \forall i = 1, 2, \dots, n$ and $j = 1, 2$

4. Each non-root node has an associated baseline probability, while a prior probability is associated with each root node.
5. Each link has a corresponding delay d (where $d \geq 0$) that represents the communication delay.
6. Each node has a corresponding delay e (where $e \geq 0$) that represents the information processing delay.
7. A pair (p, t) for each root node, where p is a list of real numbers representing probability values. For each probability value, a corresponding time interval is defined in t . In general, (p, t) is defined as

$$([p_1, p_2, \dots, p_n], [[t_{11}, t_{12}], [t_{21}, t_{22}], \dots, [t_{n1}, t_{n2}]])$$

where $t_{i1} < t_{i2}$ and $t_{ij} > 0 \forall i = 1, 2, \dots, n$ and $j = 1, 2$

Definition 3

A *Dynamic Influence Net* is a tuple $(V, E, C, B, D_E, D_V, A)$ where

V : set of Nodes,

E : set of Edges,

C represents causal strengths:

$$E \rightarrow \{([(h_1, g_1) (h_2, g_2), \dots, (h_n, g_n)], [(t_{11}, t_{12}), (t_{21}, t_{22}), \dots, (t_{n1}, t_{n2})])\}$$

such that $-1 < h_i, g_i < 1$, $t_{ij} \rightarrow \mathbb{Z}^+$ and $t_{i1} \leq t_{i2}, \forall i = 1, 2, \dots, n$ and $j = 1, 2$

B represents Baseline / Prior probability: $V \rightarrow [0,1]$,

D_E represents Delays on Edges: $E \rightarrow Z^+$,

D_V represents Delays on Nodes: $V \rightarrow Z^+$, and

A (input scenario) represents the probabilities associated with the set of actions and the time associated with them.

A: R $\rightarrow \{([p_1, p_2, \dots, p_n], [[t_{11}, t_{12}], [t_{21}, t_{22}], \dots, [t_{n1}, t_{n2}]])$

such that $p_i = [0, 1]$, $t_{ij} \rightarrow Z^*$ and $t_{i1} \leq t_{i2}, \forall i = 1, 2, \dots, n$ and $j = 1, 2$ where $R \subset V$ }

4.2 Technical Features of Pythia

Pythia is the integrated tool suite that was developed by this effort to support the EBO concept and activities described in Section 2. It is the result of integrating and enhancing the five technologies described in section 3 (CAT, CAESAR II/EB, TEMPER, R-TEM, and the Visualization-Module). This section describes the various technical features that have been incorporated into the tool. The description is purposely general; the details of each feature may be found in the various references. It should be noted that the CAT tool has continued to evolve in a parallel effort led by Dr. John Lemmer at AFRL. The Pythia tool that was developed under the effort that is the subject of this report has some of the features of the current CAT, but also has different ones as well.

The underlying concept of EBO is based on creating a model that relates desired (and undesired) effects to potential actions. The modeling paradigm chosen for this effort is influence nets and their extensions including Timed Influence Nets. The EBO concept activities revolve around creating a TIN of the situation, and performing analysis on that TIN to evaluate and refine alternative Courses of action, support the decision making process for selecting a COA by comparing COAs, and then using the TIN to support the assessment of progress toward achieving the desired effects based on new information about actions and elements of the TIN.

Pythia provides the graphical interface and mathematical algorithms to support the modeling and analysis of an uncertain situation using Timed Influence Nets and thus can be used to support the EBO concept. As is the case with any modeling paradigm, the creation of Influence Nets is both art and science. The modeling process starts with the identification of desired effects. With the help of the subject matter experts (SMEs), a system modeler (or a knowledge engineer) then identifies the events that are related to the military, economic, diplomatic, and information

aspects of the situation. These events influence the occurrence of the desired effects. In the IN terminology, the desired effects are modeled as leaf nodes while the military, diplomatic, economic, and other events are modeled as ancestors of these desired effects. The events which influence these diplomatic and economic events are further added in the model. The process continues until the SMEs identify those events (actions) that are under the control of a decision maker. These actions become the root nodes of the IN.

Pythia not only provides a graphical interface to support this modeling process but it also provides several algorithms that help in analyzing a particular situation for the purpose of refining the course of action. *Sensitivity of actions/influences analysis* identifies the individual impact of each actionable event/influence on the desired effect. These analyses help the system analyst in identifying the pressure points. Thus, a system analyst can identify the events/influences which cause a major change in the probability of achieving a desired effect. This feature can be helpful to an analyst because it indicates the actions-effects chains that have the greatest impact on desired or undesired effects.

The *Sets of Actions Finder (SAF)* algorithm [Haider, et al. 2004] is another useful tool implemented in Pythia which identifies the combinations of actions that cause the probability of a desired effect above a certain threshold. Together with sensitivity analyses, SAF can provide useful insights during the model building phase of the static Influence Nets. In both the Sensitivity of Actions and the SAF algorithms the analysis is done without the use of temporal information so that it is analysis of untimed influence nets.

Once a system analyst becomes comfortable with the untimed IN, the analyst can start adding the temporal information associated with the arcs and nodes, respectively. These delays along with the time stamp(s) associated with the actionable events are used by Pythia to generate *probability profiles* of variable of interests. A probability profile displays the probability of a particular event over a specified interval of time. It provides a visual mechanism to assess the effectiveness of a particular COA. Pythia also allows the comparison of multiple profiles generated by different COAs. This feature is useful while comparing the performances of alternative COAs.

To provide a deeper insight into the trajectory taken by a probability profile, Pythia uses a temporal logic based library (named PIL Engine⁸) that answers several types of temporal queries related to the profile. PIL Engine also is used to perform ‘What-If’ analysis that provides the potential temporal relationship which should exist between two actionable events in order to achieve a desired effect at a particular point in time. This capability is the result of incorporating the TEMPER module.

COA comparison and temporal logic based utilities aid a system analyst in selecting a desirable course of action. It should be noted, however, that the manual trial and error based approach for COA selection is a very tedious process and can only work for small size networks. It is very likely that the process will not work for large or even moderate size networks. For instance, suppose there are 39 actionable events in an IN. Further suppose that all of these actions need to be taken within 20 time units (this duration is also referred to as the window of opportunity). Assuming binary state for each actionable events, there are $2^{39} \times 20^{39}$ (order of a trillion trillion trillion trillion, 1 with 62 zeros after it) possible courses of action. It is quite evident that manual trial and error based methods could not exhaust all the possibilities and thus are likely to select a suboptimal course of action which is only the best of few possibilities explored by a system analyst. Pythia uses an Evolutionary Algorithm (EA) based approach to automate the process of effective COAs determination. The approach generates several (typically four) alternative courses of action that have a *similar* likelihood of producing the desired effect.

Once a COA is identified (either manually or using the EA based approach) and executed, the next task to be performed by a system analyst is the real time execution monitoring of the COA. During the COA execution, it is quite possible that new information about some uncertain events becomes available. As a result, the events may no longer be uncertain. This would result in revising the beliefs about other relevant events in the network. In the Influence Nets (or the Bayesian Networks) terminology, the process is referred to as *belief updating*. Pythia and its predecessor CAT provide a couple of algorithms to perform belief updating in Timed Influence Nets. It should be mentioned, however, that the belief updating is a very complex problem and the work is currently in progress. The influence net community in particular, and BN community

⁸ PIL Engine is an API developed at George Mason University that models formal logic for temporal and spatial aspects of large scale discrete event systems.

in general, is working on developing new techniques to solve this problem in real time. With respect to Pythia, the capability is the result of incorporating the R-TEM technology.

The above description not only describes the process a system analyst has to follow while developing and analyzing a situation but it also describes the major algorithms available in Pythia. The following list organizes the algorithms in different categories and a further explanation of these categories is provided below. While not specifically discussed in this subsection, the Graphical User Interface comprises the Visualization Module.

- a) Probability propagation algorithms in static Influence Nets
- b) Sensitivity analyses
- c) Incorporation of temporal information
- d) Algorithms/Analyses using temporal information
- e) Integration of the temporal logic
- f) Effective courses of action determination

4.2.1 Probability Propagation in Static Influence Nets

The probability propagation algorithm used by INs is based on the independence of parents assumption which is similar to loopy belief propagation. Pythia uses this algorithm as the default probability propagation algorithm. Pythia, however, also allows a user to generate a Bayesian Network from a user-specified Influence Net. The task is accomplished with the help of the SMILE⁹ library developed by the University of Pittsburgh. Once an IN is converted into a BN, a user can take advantage of several simulation-based and exact propagation algorithms available in SMILE.

Pythia supports multiple schemes for knowledge elicitation. A user can specify the strength of an influence using either Noisy-Or [Agosta 1990; Drudzel and Henrion 1993] or CAST logic. Besides giving flexibility, the schemes relinquish a user from the daunting task of specifying the complete conditional probability table.

⁹ SMILE (Structural Modeling, Inference, and Learning Engine) is a fully portable library of C++ classes implementing graphical decision-theoretic methods, such as Bayesian net-works and influence diagrams (<http://genie.sis.pitt.edu/>)

4.2.2 Sensitivity Analyses

Once an influence net is specified by a system analyst, it is typically of interest to study the sensitivity of a desired effect to actionable events and influences in the network. More than often, an analyst is also interested in identifying the combination of actions that cause the probability of a desired effect above/below a certain threshold. Pythia provides the algorithms that support the above mentioned tasks.

4.2.2.1 Sensitivity to Action Analysis

The sensitivity to action analysis looks at how sensitive an effect is with respect to the actionable events when actions are considered one at a time. The analysis runs in linear time, i.e., if there are 'n' actionable events then 'n' iterations are required to perform the analysis. A rather detailed explanation of the sensitivity of action analysis is given in Haider et al. [2004].

4.2.2.2 Sensitivity to Influence Analysis

The sensitivity to influence analysis looks at how sensitive an effect is with respect to the influences (or the “strength” of the influences) when influences are considered one at a time.

Pythia provides a grid based output for both sensitivity to actions and sensitivity to influences analyses that allows a user to sort actions/influences based on different criteria. Some of these criteria for Sensitivity of Actions include:

- maximum/minimum probability achieved by an effect when a particular action is true
- maximum/minimum probability achieved by an effect when a particular action is false
- difference between the maximum and minimum probabilities of an effect

The criteria for Sensitivity to Influence Analysis include:

- maximum/minimum probability achieved by an effect when a particular influence has a very strong/low positive impact (That is the strength of the influence when the parent is true is set of positively very strong (99) or negatively very strong (-99).
- maximum/minimum probability achieved by an effect when a particular influence has a very strong/low negative impact, etc.

4.2.2.3 Sets of Actions Finder (SAF) Algorithm

The SAF algorithm is a heuristic algorithm that identifies sets of actions which cause the probability of desired effect above/below a certain threshold. It employs a greedy approach, during its search for the best set of action, and works in polynomial time. The algorithm starts with a single action which, when considered individually, causes the highest increase in the probability of the objective being true. This is followed by the selection of a second action from the remaining set of actions that together with the first action cause the highest increase in the probability of the objective node. Other actions are added iteratively in a similar manner. The process stops at a point where the inclusion of an action causes the probability of the objective node to decrease and to fall below the given probability threshold. Once alternative sets of actions are obtained, they can be grouped together to form more general sets of rules. Haider et al. [2004] describes the SAF algorithm in detail.

4.2.3 Incorporation of Temporal Information

Influence Nets were originally designed to capture static uncertain situations. Wagenhals et al. [1998] added several temporal constructs that allow an analyst to capture particular types of temporal information using Influence Nets. The resultant class of Influence Nets was later named as Timed Influence Nets [Lindstrøm and Haider 2001]. Pythia allows the modeling of an uncertain situation using TINs and provides algorithms to support several types of analyses. Using Pythia, an analyst can associate delays to influences and events, and can assign time stamps to actionable events. Instead of having a single time stamp associated with an actionable event, Pythia also allows an analyst to assign multiple time stamps to an event. This feature provides the capability to model *persistence of action*.

Recently Haider and Levis [2005] have added new structural and parametric constructs to Timed Influence Nets based framework. Previously, nodes in a TIN were considered memoryless. This feature resulted in an inability to modeling the impact of different sequences of actions on a desired effect. TINs also failed to capture time-varying influences, a feature of the concept of persistence. The new enhancements allow a system analyst to specify such influences. Thus, the analyst can specify both stationary and non-stationary influences. Furthermore, the dependency of an event on its previous state can also be modeled by adding a self-loop to the corresponding node. The incorporation of self-loop adds memory to the existing memory-less TIN and can be

used to represent the change in the impact of an influencing event over time. The addition of both self-loop and time-varying influences enables an analyst to model impacts of repeated actions on an effect. The new class of TIN is named Dynamic Influence Nets (DINs). Previously, in the event of repeated actions, a TIN only considers the latest impact on the effect while ignoring the previous attempts. A DIN, on the contrary, would convolve the impact of repeated actions on the desired effect. A detailed description of DINs is given in Haider and Levis [2005]. Pythia supports both TIN and DIN based modeling paradigm.

4.2.4 Algorithms/Analyses Using Temporal Information

Together with Section 2.3, this section describes the real time execution monitoring (R-TEM) and visualization (V-Mod) modules of Pythia.

4.2.4.1 Visualization Module (V-MOD)

Once a COA is specified, Pythia generates *probability profiles* of selected events. A probability profile shows the likelihood of occurrences of events over a period of time. The period of time is determined from the COA and temporal information available in the form of arc and node delays. The probability profile is also considered while performing temporal analysis using the TEMPER logic module.

Pythia allows a user to compare several courses of actions. The comparison is performed using probability profiles of variable(s) of interest generated from the selected courses of actions. This feature is very helpful during the task of COA selection.

4.2.4.2 Real Time Execution Monitoring Module (R-TEM)

As events unfold during the execution of a selected COA, it is often the case that new information (evidence) about uncertain events is received. The incorporation of this evidence in the corresponding TIN requires *belief updating* of the remaining events in the net. Because of certain mathematical properties, this is a very complex problem. In fact, the problem is NP-Hard and it cannot be solved in real time unless the model is very small. Several approximation and simulations based schemes have been proposed in the literature to overcome the complexity of this problem. One such heuristic-based algorithm was developed at GMU [Haider and Levis 2004]. The technique updates the nodes in a TIN in the sequential manner using the constraint that all the children of a node affected by the new evidence will be updated first before updating

the belief in that particular parent node. The algorithm also takes advantage of the fact that in TINs, the focus is on observing the behavior of few nodes in the network. Hence there is no need to update all the nodes of the network. The nodes that receive impact of evidence and have a path to the target nodes only need to be updated. This relaxation helps in applying graph reduction techniques on TINs.

One of the possible limitations of the approach is that the algorithm works only if the time stamp of the evidence is later than the time stamp of the last update of the evidence node caused by the *forward propagation* of the effects from all of the action nodes. This constraint might be very strong in some cases. It is quite possible that the evidence could be observed earlier than expected by the model. To overcome this problem, Haider and Zaidi [2004] have developed a transformation algorithm that converts a Timed Influence Nets into a Time Sliced Bayesian Network (TSBN). They have suggested the use of TINs as a front end tool for model building and course of action selection process, and TSBNs for execution monitoring of the selected course of actions by making use of the vast variety of belief updating algorithm developed for TSBN.

The transformation algorithm is implemented in Pythia. The algorithm uses the SMILE library introduced earlier in this report. An analyst can make use of this capability to observe the impact of new evidence on events of interest with the help of the V-Mod. Pythia also provides an initial estimate of the number of time slices that should be drawn in a TSBN. The estimate is based on the maximum path length in the corresponding Timed Influence Nets and the maximum time stamp assigned to actionable events. The user has the provision to alter this initial estimate of the number of time slices and set a different value. The COA comparison utility in Pythia also helps in comparing the results obtained from different approximate and simulation-based belief updating algorithms that are already available in Pythia and those that would be added in the future.

4.2.5 Integration of the Temporal Logic

This section describes the integration of the TEMPER logic module, named the PIL Engine, into Pythia. Algorithms have been implemented in Pythia that take a TIN and generate a corresponding point-graph (PG). Once a PG is constructed, the Point-Interval Temporal Logic (PITL) [Zaidi 1999; Zaidi and Levis 2001] approach helps in analyzing the temporal

relationships among system variables in a complex uncertain situation. The results of this analysis can aid a system analyst in gaining a better insight of the impact of a selected course of action on desired effect(s). The PG representation of a corresponding TIN answers queries regarding certain temporal characteristics of an effect's probability profile. The PG also aids a system modeler by explaining what needs to be done to achieve a certain effect at a specific time instant. If the requirements for achieving effects at certain time instants are not temporally consistent, then the PG helps in understanding the reasons for inconsistencies. A detailed technical description of this integration of Pythia and PIL can be found in Haider et al. [2005] and Zaidi et al. [2005]. The latter reference provides a detailed example of the end-to-end process of creating an influence net of a situation to support and effects based operation and using the integrated TEMPER application capabilities to query and perform "what if" analysis on the model and ultimately refine the course of action.

4.2.6 Effective Courses of Action Determination

Once an uncertain situation is captured using TIN, the next task confronted by a system analyst is the identification of sets of actions and their time of execution that would maximize the likelihood of achieving a desired effect. GMU has developed a methodology, named ECAD-EA (Effective Courses of Action Determination using Evolutionary Algorithms), to accomplish this task [Haider and Levis 2005]. The methodology has been implemented in Pythia. It produces several alternative courses of action that have a *similar* likelihood of producing the desired effect. One advantage of producing alternative COAs is that a decision maker would have more than one option in selecting a particular COA. The second, and equally important, objective is to generalize the relationships that exist among the actions and their execution timings. The generalization would aid the decision maker in understanding the temporal relationships among actionable events at a qualitative level. The generalized qualitative relationships can also be used to provide the sequence in which action should take place, thus giving more flexibility to a decision maker during planning stage of a mission.

Pythia also allows a system analyst to specify several types of causal and temporal constraints that could exist among actionable events and must be satisfied by a feasible COA. Furthermore, Pythia allows an analyst to select a metric (or set of metrics) from several metrics that are used as Measures of Performance (MOPs). Pythia also provides very user friendly interfaces for

desired effect selection, metric selection, constraints input, and windows of opportunity and observation input. A 'window of opportunity' represents the time period during which a plan must be executed while a 'window of observation' represents the time period during which a decision maker is interested in getting the desired effect.

5. CONCLUSION

GMU has successfully integrated a suite of technologies and tools into a single tool environment called Pythia that can support operators and intelligence analysts in conducting Effects Based Operations. As part of the development effort extensions to Influence Nets were developed and formalized. These include Timed Influence Nets and Dynamic Influence Nets. Furthermore, several algorithms were developed to support the analysis of these nets needed for course of action evaluation, comparison, and selection. An installation version of the tool has been created along with a User's Guide.

The development effort was aided by several opportunities to use the tools and algorithms in war games during the development period. This experience enabled the developers to better understand the needs of the operator and to expand the developer's repertoire of modeling types and techniques to provide support to different classes of problems. One of the key lessons learned from these experiences was the difficulty in developing EBO models and conducting EBO analysis and the intensity of the human effort required.

As we have discovered, EBO is a complex undertaking with potentially high payoff. Both the operational and R&D communities need to refine their thinking and create new tools and techniques to manage this complexity. Furthermore, the tools and techniques must be incorporated in an overall process that is used for planning, execution, and assessment across domains and levels. Most of the effort up to this point has been on Effects Based Planning. Additional emphasis needs to be placed on conducting Effects Based Assessment, a process during effects based execution of monitoring and determining if the selected COA and plan is on the right track and determining if there are opportunities or needs to make adjustments to the COA. If conceptualizing the action-effect relationships is at the heart of EBO, we need to devise better ways to help operators and analysts develop good models rapidly. There is considerable room for improvement in this arena such as the use of templates and approaches for pruning models. In addition, we need ways of rapidly finding information and data that can be used in developing the models and methods for transforming the information into the constructs of the models. Since more than one modeling technique is appropriate, we need to determine how to

import the information derived from one model into another. For example, Pythia models may support and be supported by more traditional wargaming models and simulations. We need to determine what the interactions between these models should be in order to enhance the EBO process.

We think that the Pythia tool or its successor can provide a unifying framework for effects based operations across the process of COA selection, planning, execution and assessment. There is still much to be learned about using tools like Pythia in the real operational environment. This will require experimentation efforts involving both the operators and the developers to determine what works and what is the best allocation of tasks between tools and operators. Furthermore, the tools will have to be integrated into the existing command and control Information Technology environments such as the one that exist in a Combined Air Operations Center. Perhaps the largest challenge ahead is to develop a cadre of analysts within the operational community who can quickly create these types of models. Members of that cadre should be made available at multiple echelons of the command and control structure. This cadre could continuously coordinate the development and maintenance of there models as they support the planning, execution and assessment process. As we continue to increase our understanding of the use of this basic model and the techniques and procedures that support those uses, we should begin to be able to better support the operators as they conduct effects-based operations.

REFERENCES

- Agosta, J. M., “Conditional Inter-Causally Independent Node Distributions, a Property of Noisy-OR Models”, In the Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence, 1991.
- Chang, K. C., Lehner, P. E., Levis, A. H., Zaidi, S. A. K., and Zhao, X., “On Causal Influence Logic”, Technical Report, George Mason University, Center of Excellence for C3I, Dec. 1994.
- Drudzel, M. J., and Henrion, M., “Intercausal Reasoning with Uninstantiated Ancestor Nodes”, In the Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence, 1993.
- Haider, S., “On Finding Effective Courses of Actions in Dynamic Uncertain Situations”, PhD Dissertation, School of Information Technology & Engineering, George Mason University, Fairfax, VA, 2005. Available at <http://mutex.gmu.edu:2284/dissertations/search>.
- Haider, S. and Levis, A. H., “An Approximate Technique for Belief Revision in Timed Influence Nets”, In the Proceedings of 2004 Command and Control Research and Technology Symposium, San Diego, June 2004.
- Haider, S. and Levis, A. H., “Dynamic Influence Nets: An Extension of Timed Influence Nets for Modeling Dynamic Uncertain Situations”, In the Proceedings of 10th International Command and Control Research and Technology Symposium, Washington DC, June 2005.
- Haider, S. and Levis, A. H., “On Finding Effective Courses of Action in a Complex Situation using Evolutionary Algorithms”, In the Proceedings of 10th International Command and Control Research and Technology Symposium, Washington DC, June 2005.
- Haider, S. and Zaidi, A. K., “Transforming Timed Influence Nets into Time Sliced Bayesian Networks”, In the Proceedings of Command and Control Research and Technology Symposium, San Diego, June 2004.
- Haider, S., Zaidi, A. K., and Levis, A. H., “A Heuristic Approach for Best Set of Actions Determination in Influence Nets”, In the Proceedings of IEEE International Conference on Information Reuse and Integration, Las Vegas, Nov. 2004.
- Haider, S., Zaidi, A. K., and Levis, A. H., “On Temporal Analysis of Timed Influence Nets using Point Graphs”, In the Proceedings of 18th International FLAIRS Conference, Clearwater Beach, FL, May 2005.
- Kschischang, F. R. and Frey, B. J., “Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models”, IEEE Journal on Selected Areas in Communication, 16(2), 1998.
- Lemmer, J. F. and Gossink, D. E., “Recursive Noisy Or – A Rule for Estimating Complex Probabilistic Interactions”, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 34, No. 6, pp. 2252-2261, Dec. 2004.

- Lindstrom, B. and Haider, S., "Equivalent Colored Petri Nets Models of a Class of Timed Influence Nets with Logic", In the Proceedings of the Third Workshop and Tutorial on Practical Use of Colored Petri Nets and the CPN Tools, Denmark, August 2001.
- McEliece, R. J., MacKay, D. J. C., and Cheng, J. F., "Turbo Decoding as An instance of Pearl's Belief Propagation Algorithm", IEEE Journal on Selected Areas in Communication, 16(2), 1998.
- Murphy, K., Weiss, Y., and Jordan, M., "Loopy-belief Propagation for Approximate Inference: An Empirical Study", In the Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, 1999.
- Neapolitan, R. E., "Learning Bayesian Networks", Prentice Hall, 2003.
- Pearl, J., "Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference", Morgan Kaufmann, 1987.
- Rosen, J. A., and Smith, W. L., "Influence Net Modeling with Causal Strengths: An Evolutionary Approach", In the Proceedings of the Command and Control Research and Technology Symposium, Naval Post Graduate School, Monterey CA, Jun. 1996.
- Wagenhals, L. W., "Course of Action Development and Evaluation Using Discrete Event System Models of Influence Nets", PhD Dissertation, School of Information Technology & Engineering, George Mason University, 2000. Available at <http://mutex.gmu.edu:2284/dissertations/search>.
- Wagenhals, L. W., Levis, A. H., "Course of Action Development and Evaluation", In the Proceedings of the 2000 Command and Control Research and Technology Symposium, 2000.
- Wagenhals, L. W. and Levis, A. H., "Modeling Effects Based Operations in Support of War Games", In the Proceedings of the 15th International Symposium on Aerospace / Defense Sensing, Internal Society for Optical Engineering, Proceedings of SPIE, Vol. # 4367, 2001.
- Wagenhals, L. W., Levis, A. H., "Modeling Support of Effect-Based Operations in War Games", In the Proceedings of the 2002 Command and Control Research and Technology Symposium, 2002.
- Wagenhals, L. W., Shin, I., and Levis, A. H., "Creating Executable Models of Influence Nets with Coloured Petri Nets," Int. J. STTT, Spring-Verlag, Vol. 1998, No. 2, 1998.
- Wagenhals, L. W., T. J. Reid, R. J. Smillie, & A. H. Levis (2001), "Course of Action Analysis for Coalition Operations", In the Proceedings of 6th International Command and Control Research and Technology Symposium, Annapolis, Maryland, Jun. 2001.
- Zaidi, A. K., "On Temporal Logic Programming Using Petri Nets", IEEE Transactions on Systems, Man, Cybernetics A, 29, May 1999.
- Zaidi, A. K. and Levis, A. H., "TEMPER: A Temporal Programmer for Time-Sensitive Control of Discrete Event System", IEEE Transactions on Systems, Man, Cybernetics A, 31, Nov. 2001.
- Zaidi, A. K., Wagenhals, L. W., and Haider, S., "Assessment of Effects Based Operations using Temporal Logic", In the Proceedings of 10th International Command and Control Research and Technology Symposium, Washington DC, June 2005.

LIST OF ACRONYMS

AFOSR	Air Force Office of Scientific Research
AFRL	Air Force Research Laboratory
ATD	Advanced Technology Demonstration
ATO	Air Tasking Order
BN	Bayesian Net
C2	Command and Control
CAESAR II/EB	Computer Aided Evaluation of System Architectures (Effects Based)
CAT	Campaign Assessment Tool
CAST	Causal Strength
COA	<i>Course of Action</i>
COTS	Commercial off the Shelf
CPN	Colored Petri Net
CPT	Conditional Probability Table
DAG	Directed Acyclic Graph
DIN	Dynamic Influence Net
EA	Evolutionary Algorithm
EBO	<i>Effects Based Operations</i>
ECAD-EA	Effective Course of Action-Evolutionary Algorithm
GMU	George Mason University
GUI	Graphical User Interface
IN	Influence Net
IO	<i>Information Operations</i>
ISR	Intelligence Surveillance and Reconnaissance
JEFX	Joint Expeditionary Force Experiment
MOE	Measure of Effectiveness
MOFE	Measure of Force Effectiveness
MOP	Measure of Performance
ONR	Office of Naval Research
PIL	Point-Interval Logic
PITL	Point-Interval Temporal Logic
PG	Point Graph
R&D	Research and Development
R-TEM	Real Time Execution Monitor
SAF	Sets of Actions
SIAM	<i>Situation Influence Assessment Module</i>
SME	Subject Matter Expert
TIN	<i>Timed Influence Net</i>
TSBN	Time-Slice Bayesian Net
V-Mod	Visualization Module

APPENDIX A: PYTHIA Version 1.0 USER MANUAL

TABLE OF CONTENTS

Introduction to <i>Pythia</i>	53
A.1. Salient Features of Pythia	54
A.1.1 Graphical Interface Features	54
A.1.2 Technical Features	55
A.2 Pythia System Requirements and Tutorial.....	57
A.2.1 Pythia Graphical Interface	57
A.2.2 Drawing Of Nodes	59
A.2.3 Saving A Pythia Model	62
A.2.4 Drawing Of An Arc.....	62
A.2.5 Probability Propagation in Static Influence Nets.....	64
A.2.6 Sensitivity Analysis	65
A.2.7 Running Of SAF (Sets Of Actions Finder) Algorithm	68
A.2.8 Course Of Action Assignment	70
A.2.9 Saving A Course Of Action	71
A.2.10 COA Execution / Probability Profile Generation	72
A.2.11 Running ECAD-EA (Effective Courses of Action Determination using Evolutionary Algorithms) Methodology	75
A.2.11.1 Selection of Metric.....	76
A.2.11.2 Evolutionary Algorithm Setting.....	77
A.2.11.3 Constraint Specification.....	79
A.2.12 COA Comparison.....	82
A.2.13 TEMPORAL ANALYSIS	85
A.2.14 Conversion to Dynamic Bayes Nets for Adding Evidence	91
A.2.15 Dynamic Influence Nets (DINs)	97
A.3. Summary	102

Introduction to Pythia

Pythia provides an environment to build graph-based decision theoretic models and to perform several analyses on them. It is the latest software added to the CAESAR suite of tools developed by the System Architectures Lab at GMU to aid decision making in complex situations. Pythia and its earlier version based on CAT¹⁰ have been used in several war games to model Effects Based Operations and courses of action (COAs) evaluation.

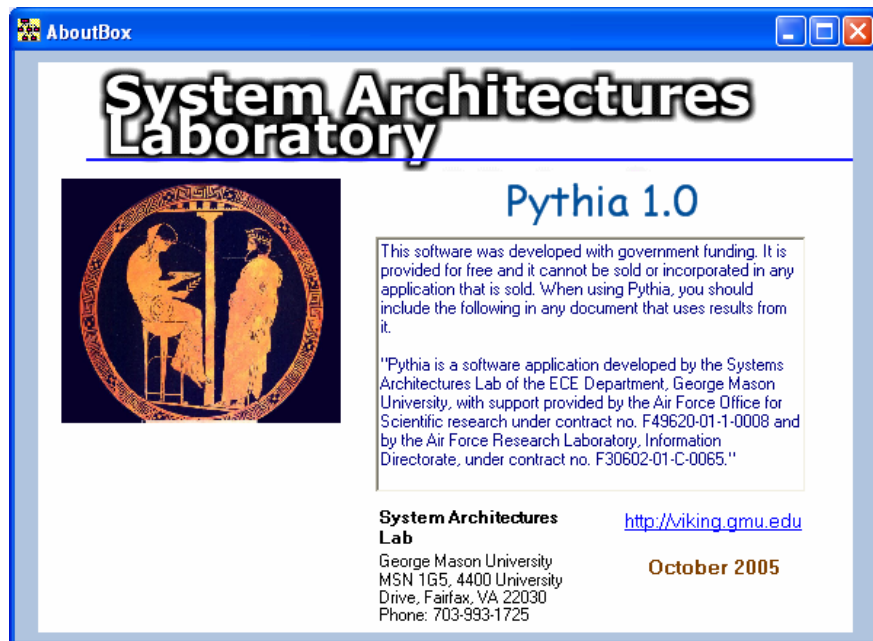


Figure A.1: About Box of Pythia

Pythia is developed using the Visual Studio .NET platform, more specifically, using the C# language. The front end of the tool is designed with the help of AddFlow¹¹, a commercial-off-the-shelf API. Pythia allows building Influence Net models of any size and complexity; the only limitation is the memory of the machine running Pythia. The beta version of the software is currently used by several government and non-government organizations. Version 1.0 of the

¹⁰ CAT (Campaign Assessment Tool) was originally a Bayesian Network based software developed by Air Force Research Lab (AFRL). SAL/GMU modified the source code of CAT to add the TIN related capabilities.

¹¹ Addflow is developed by Lassalle Technologies. A trial version of AddFlow can be downloaded from <http://www.lassalle.com/download.htm#.NET>.

software was released in October 2005. Figure A.1 shows the Pythia's "About" window. The rest of this appendix highlights certain characteristics of Pythia's development environment: Section A1 explains its salient features, while Section A2 contains the tutorial. Whenever appropriate, references (from Haider 2005^{*}) are provided to the keywords used in the description.

A.1. SALIENT FEATURES OF PYTHIA

A.1.1 GRAPHICAL INTERFACE FEATURES

Pythia allows a user to draw and specify the structure and parameters of an Influence Net and its extensions Timed Influence Nets (Section 4.2^{*}) and Dynamic Influence Nets (Chapter 9^{*}). The main graphical interface features are described below:

- Actionable events are drawn as rectangles while non-actionable events are drawn as rounded rectangles thus making them visually distinct.
- Different arc styles are used in Pythia to distinguish between positive and negative influences.
- A color-coding scheme is used that assists a user in estimating the likelihood of occurrence of a particular event in an Influence Net.
- A dual interface is used for the Conditional Probability Tables (CPTs) assignment. A user can select either Noisy-OR or the CAST for CPTs specification.
- Pythia allows a user to specify certain parameters related to a TIN/DIN model:
 - Time delays associated with events and influences.
 - Time stamps and corresponding probabilities associated with actionable events. The set of time stamps and corresponding probabilities are collectively referred to as a course of action (COA).
 - Prior and baseline probabilities associated with events.
 - Strength of influence associated with each arc.
- Pythia supports standard Windows based routines for input and output:
 - A 'Save' routine allows a user to save a new or pre-existing Influence Net.

^{*} Haider, S., On Finding Effective Courses of Actions in Dynamic Uncertain Situations, PhD Dissertation, School of Information Technology & Engineering, George Mason University, Fairfax, VA, 2005. Available at <http://mutex.gmu.edu:2284/dissertations/search>.

- o A 'Save As' routine allows a user to save a pre-existing Influence Net with a different name.
- o An 'Open' routine allows opening of a saved Influence Net.
- o A 'Print' routine allows printing of an Influence Net. This routine contains other standard Windows printing features such as print preview, page setup, etc.
- o A 'Save COA' routine saves the current COA (time stamps and their corresponding probabilities associated with actionable events).
- o A 'Load COA' routine opens a saved COA.
- Pythia exports the image of an Influence Net into a (JPEG format) picture. This picture can subsequently be used in Microsoft Word and/or PowerPoint for documentation and/or presentation purposes. The export feature can also be used for printing large Influence Net model on a single sheet of paper.
- Pythia supports a zoom-in and zoom-out feature that enhances the readability of an Influence Net.
- Besides displaying the structure of an Influence Net, Pythia also displays the event names in a sorted order in a tree view, thus making the selection of an event really easy for a user.

A.1.2 Technical Features

- Once an Influence Net is completely specified by a user, Pythia computes the marginal probabilities of all the events in the Influence Net. These probabilities show the likelihood of occurrence of corresponding events in a static (time-independent) situation.
- A user can perform two types of sensitivity analyses in Pythia: (a) the sensitivity of action analysis provides the sensitivity of an effect to actionable events (b) the sensitivity of influence analysis provides the sensitivity of an effect to the CAST logic parameters associated with arcs in an Influence Net.
- Pythia allows a user to run the Sets of Actions Finder (SAF) algorithm. The algorithm provides sets of actions that cause the probability of a desired effect to be above a certain threshold.

- Pythia generates an equivalent Bayesian Network from a user-specified Influence Net. The BN is produced in a format readable by SMILE/Genie¹².
- Once a COA is specified, Pythia generates probability profiles of selected events. A profile shows the likelihood of occurrences of events over a period of time. The period of time is determined from the COA and temporal information available in the form of arc and node delays.
- Pythia allows a user to compare several courses of actions. The comparison is performed using probability profiles of variable(s) of interest generated from the selected courses of actions.
- A transformation algorithm has been implemented in Pythia that converts a Timed Influence Net into an equivalent Time Sliced Bayesian Network. The task is accomplished with the help of the SMILE library. The transformation aids in modeling the ISR (Intelligence, Surveillance, and Reconnaissance) problem using the Influence Nets based methodology.
- Pythia allows a user to assign several likelihoods of occurrence of an actionable event at distinct discrete time instances. This feature attempts to capture the persistence of actionable events.
- Pythia allows a user to run the ECAD-EA (Effective Course of Action Determination using Evolutionary Algorithms) methodology. A user can specify:
 - Temporal and causal constraints among actionable events
 - A set of metrics
 - Windows of observation and opportunity
- The PIL Engine¹³, developed separately at SAL/GMU, has been integrated in Pythia. Algorithms have been added that take an Influence Net and generate a corresponding point-graph. The resultant point-graph is subsequently used for answering temporal queries and performing “what-if” analysis.

¹² SMILE/Genie is developed by the University of Pittsburg and can be downloaded from <http://www.sis.pitt.edu/~genie/>

A.2 PYTHIA SYSTEM REQUIREMENTS AND TUTORIAL

Pythia is a stand alone application that runs on the Windows operating systems. It is developed in C# language using the Visual Studio .NET framework. A very small percentage of the code is written using Visual Basic 6.0. The following are the system requirements to run Pythia.

Operating Systems: Microsoft ® Windows 2000/XP (Pythia also runs on Windows 98 but some of the graphical features do not show up as nicely as they do in Windows 2000 or XP).

CPU: Intel ® Pentium II or higher (Pythia has not been testing on AMD® series of processor but it should work without any problem)

Memory: 128 MB or higher

Hard Disk Space: 10 MB

Software: Microsoft .NET Framework (Version 1.1) ¹⁴

Pythia is distributed in a CD and it can be installed by running the “setup.exe” file inside the “DISK1” directory. Once Pythia is installed on a computer, it can be run through Start → Programs → sal gmu → Pythia → Launch Pythia.

A.2.1 Pythia Graphical Interface

There are 4 main components in the GUI (Graphical User Interface) of Pythia (Figure A.2):

- | | |
|-----------------|--------------|
| 1. Drawing Area | 2. Tree View |
| 3. Toolbar | 4. Menu Bar |

¹³ PIL Engine is an API developed at George Mason University that models formal logic for temporal and spatial aspects of large scale discrete event systems.

¹⁴ Microsoft .NET framework can be downloaded free of charge from the Microsoft’s website:
<http://www.microsoft.com>

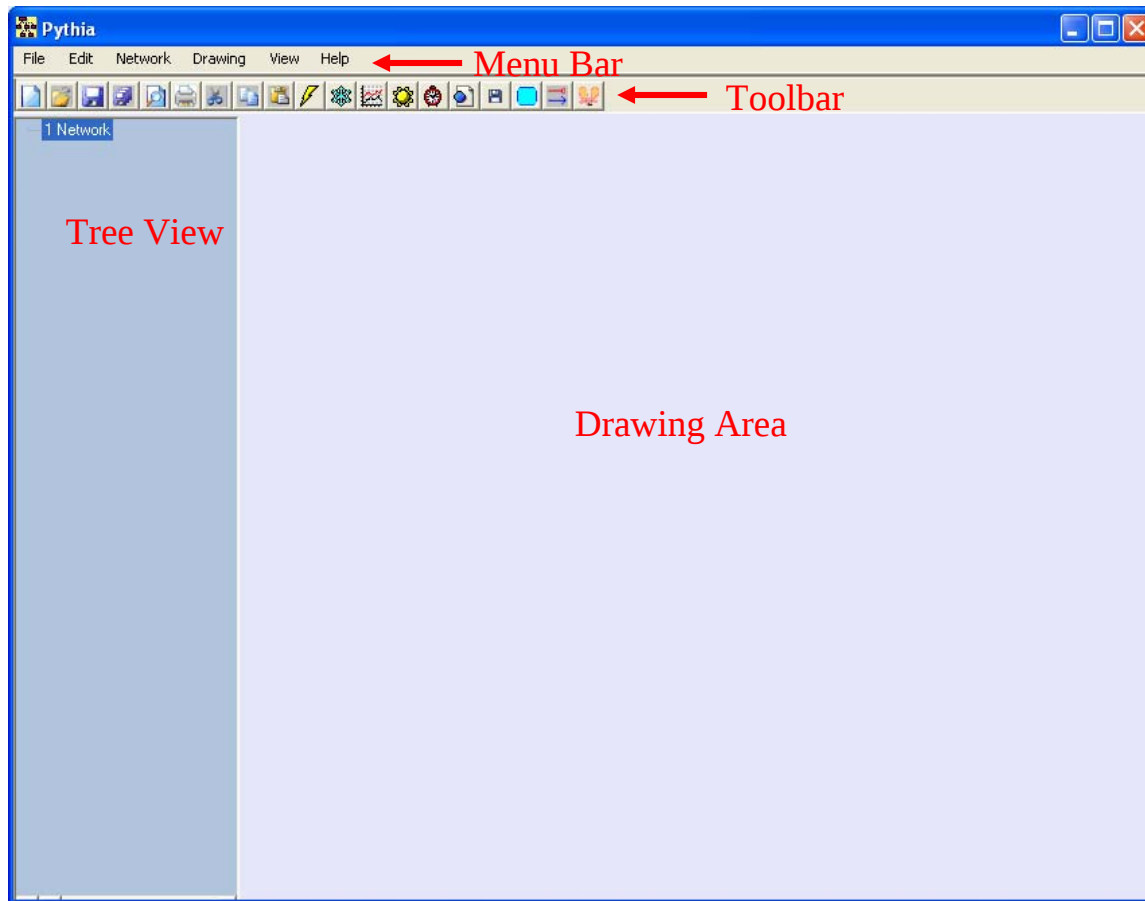


Figure A.2: Graphical Interface of Pythia

A brief description of these components is given below:

Drawing Area: provides the space to draw events and influences in an Influence Net.

Tree View: provides a sorted list of events in an Influence Net.

Toolbar: provides button for running various commands, such as, drawing of events/influences, save/open models/COAs, etc.

Menu Bar: displays the top level menu heading. Clicking on a heading opens the corresponding menu and displays the commands under that menu. These commands include the ones mentioned in the description of the Toolbar and many others such as running of SAF algorithm, ECAD-EA methodology, etc.

The following subsections describe how to use these components for modeling and analyzing an Influence Net. The process is explained in a step-by-step fashion with the help of a small example.

The following definition of terms will aid in understanding the steps. The user is encouraged to review the Final Technical Report for the PYTHIA project for complete descriptions of the algorithms used in PYTHIA.

Root Node: Any node in an Influence Net model that has no parents. Usually represents actions or actionable events.

Non-Root Nodes: Nodes with parents. Usually represent effects.

Marginal Probability: Probability associated with or assigned to nodes in an influence net. Usually assigned to root nodes to indicate whether or not an action will be taken. Probability values of non-root nodes that are calculated by the probability propagation algorithm of influence nets.

Baseline Probability: Probability value used in either CAST logic or the Noisy-Or model¹⁵ to calculate the conditional probability values associated with each non-root node. It represents the probability that the random variable the nodes represents (i.e. the statement in the node name) will be true on its own, without the influence of any parent nodes in the model. Also called the “leak” probability in the Noisy-Or model.

Causal Strengths: values assigned to directed links or arcs between nodes by the modeler that indicate the degree of influence that a parent node has on its child. CAST Logic is the default method of assigning the causal strength values.

The following subsections describe how to use these components for modeling and analyzing an Influence Net. The process is explained in a step-by-step fashion with the help of a small example.

A.2.2 DRAWING OF NODES

1. To draw a node, Pythia needs to be in the “Drawing Node” mode.

¹⁵ A detailed description of the CASE logic and Noisy-Or model is contained in Haider, S., On Finding Effective Courses of Actions in Dynamic Uncertain Situations, PhD Dissertation, School of Information Technology & Engineering, George Mason University, Fairfax, VA, 2005.

2. To bring Pythia to the “Drawing Node” mode; click the button in the toolbar that has a rounded rectangle picture, as shown in Figure A.3.
3. The mouse style is changed from the default to the hand cursor.

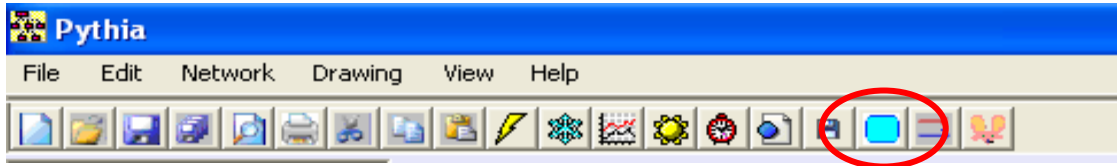


Figure A.3: Button for Drawing Node Mode

4. Click the mouse in an empty space in the Drawing Area, and drag the mouse to draw a node.
5. A Window will pop up as shown in Figure A.4. The user is required to enter the name¹⁶ and (optionally) the marginal probability. The default marginal probability is set to 0.5 (In the figure, the user has changed it to 0.1). To create a small five node Influence Net model, let the name of the node be “International Community Threatens Sanctions” and the marginal probability be set to 0.1. Leave the remaining entries in their default values¹⁷.



Figure A.4: Node Properties Window

¹⁶ Be sure to only use alphabetic characters or numbers. Use of special characters like @, #, \$, %, (,), etc. can result in an error or corruption of the file.

¹⁷ Note that the user can specify the Baseline probability. This is a parameter used in the CAST logic that is equivalent to the “leak” probability in the Noisy-Or model that is used to simplify the creation of Bayesian nets. It represents the probability that the random variable the nodes represents (i.e. the statement in the node name) will be true on its own, without the influence of any parent nodes in the model. The default value is 0.5 to indicate that this probability is considered to be unknown and cannot be estimated with any more accuracy than a flip of a coin.

Repeat Step 5 four more times to add additional nodes. Set the parameters of the node as follows:

- a) Name: Country G Employs Successful Covert Operations
Marginal Probability: 0.1
- b) Name: Country G Begins Diplomatic Efforts
Marginal Probability: 0.1
- c) Name: Leader of Country B Believes He can Win
Marginal Probability: Default (0.5)
- d) Name: Country B Agrees to Withdraw
Marginal Probability: Default (0.5)

The resultant Influence Net is shown in Figure A.5.

- 6. Restore the mouse to the default shape by clicking on the “Restore Mouse” button, as shown in Figure A.6.

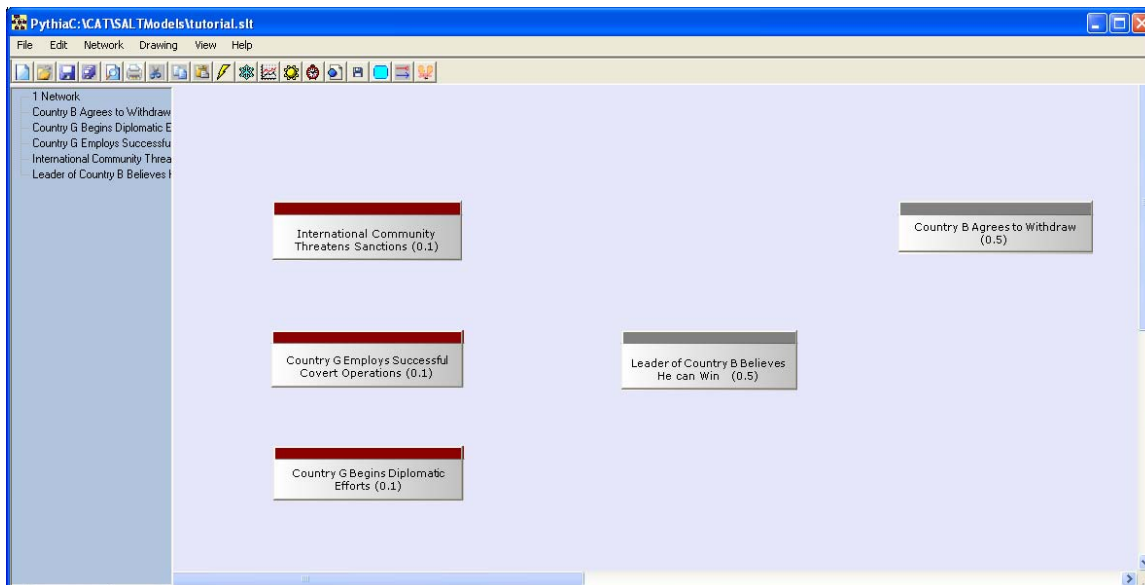


Figure A.5: An Influence Net with 5 Nodes



Figure A.6: Button for Restoring Default Mouse

A.2.3 Saving A Pythia Model

Before proceeding further, let us save the sample Influence Net model.

1. Click on the 'File' menu and click on 'Save'.
2. A dialog box will ask for a name of the model. Let us call the sample TIN "tutorial".
3. Pythia saves the model as 'tutorial.slt'.

A.2.4 Drawing Of An Arc

1. To draw an arc, Pythia needs to be in the "Drawing Arc" mode.
2. To bring Pythia to the "Drawing Arc" mode; click the button in the toolbar that has a picture of 'two arcs', as shown in Figure A.7.
3. The mouse style changes from the default to the arrow cursor.



Figure A.7: Button for Drawing Arc Mode

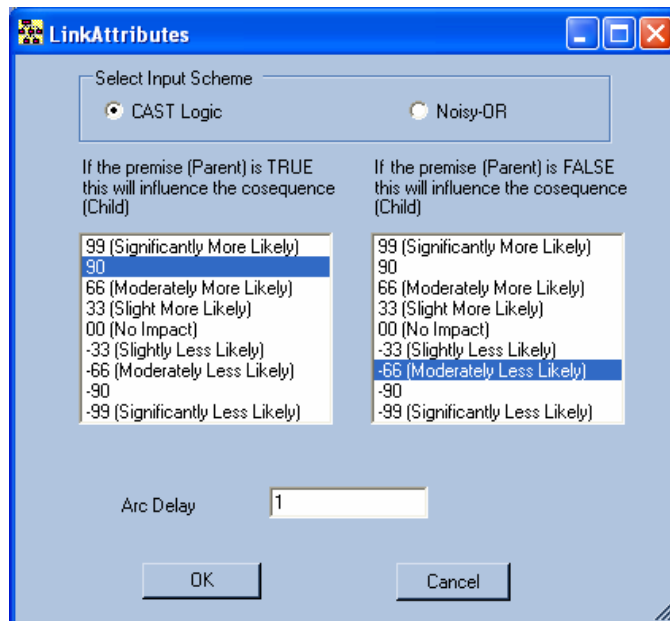


Figure A.8: Arc Properties Window

4. Click on a parent node and drag the Mouse from the parent node to a child node. In the current example, first click on the node ‘International Community Threatens Sanctions’ and then drag the mouse to the node ‘Leader of Country B Believes He can Win’.
5. A Window will pop up, as shown in Figure A.8. The user is required to select the appropriate causal strengths. Suppose we select “90” in the list box on the left of the window (h value), and “-66” in the list box on the right of the window (g value), as shown in the figure. The Arc Delay is set to 1.

Step 5 is repeated four more times to create the following arcs¹⁸.

- a) Parent: Country G Employs Successful Covert Operation
Child: Leader of Country B Believes He can Win
h Value: -66 g Value: -66 Arc Delay: 1
- b) Parent: Country G Begins Diplomatic Efforts
Child: Leader of Country B Believes He can Win
h Value: -66 g Value: 66 Arc Delay: 3
- c) Parent: International Community Threatens Sanctions
Child: Country B Agrees to Withdraw
h Value: 90 g Value: -66 Arc Delay: 6
- d) Parent: Leader of Country B Believes He can Win
Child: Country B Agrees to Withdraw
h Value: -90 g Value: 90 Arc Delay: 1

The resulting TIN is shown in Figure A.9. It contains the five nodes and five arcs. The nodes on the left side of the network have no parents and are sometimes called “root” nodes. They represent the actionable events in the model. The other “non-root” nodes are effects. The g and h parameters and the time delay for each arc are indicated next to the arc. The nodes indicate the marginal probability and have a colored bar at the top that visually show whether this probability is low (considerably greater than 0.5 (Red)), medium (near 0.5 (Grey)), or high (considerably greater than 0.5 (Blue)). At this point the TIN has not been “solved”. The probabilities shown

¹⁸ Note that the user has the option of specifying either CAST logic or Noisy-Or as the method for inputting the “strength” values for the influence. The example shows the CAST logic that requires two inputs. If the Noisy-Or option is selected only the promoting positive values of first column of the window are used.

are the ones that were set when the nodes were drawn. They will change when the TIN is “solved” by conducting the Static Propagation that is discussed below.

6. Restore the mouse to the default shape by clicking on “Restore Mouse” button as shown in Figure A.6.

Before proceeding, save the enhancements by clicking first on the ‘File’ menu and then on ‘Save’.

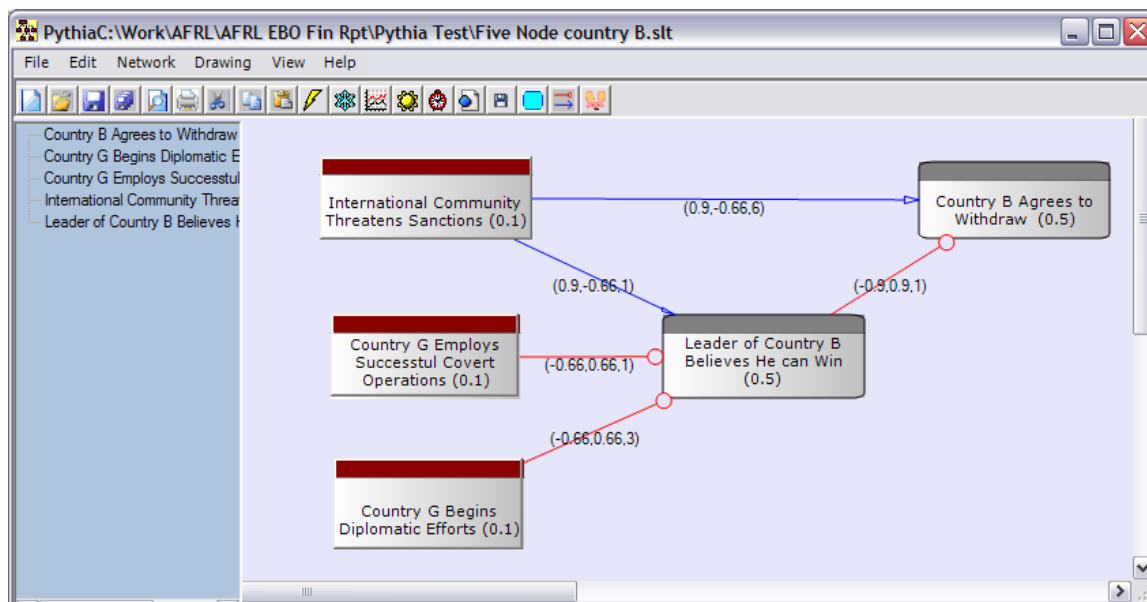


Figure A.9: A Completely Specified Influence Net

A.2.5 Probability Propagation in Static Influence Nets

Before the static propagation is performed, it is assumed that an Influence Net has been completely specified, i.e., the following parameters have been assigned:

- a) Marginal probabilities – for root nodes
 - b) Baseline probabilities – for non-root nodes
 - c) Causal strengths – for each arc
1. To perform the static propagation, click on the ‘Network’ menu and then on ‘Static Propagation’ (Figure A.10). Alternatively, one can click on the lightning bolt on the tool bar.
 2. Pythia updates the marginal probabilities of all the non-root nodes in the Influence Net.

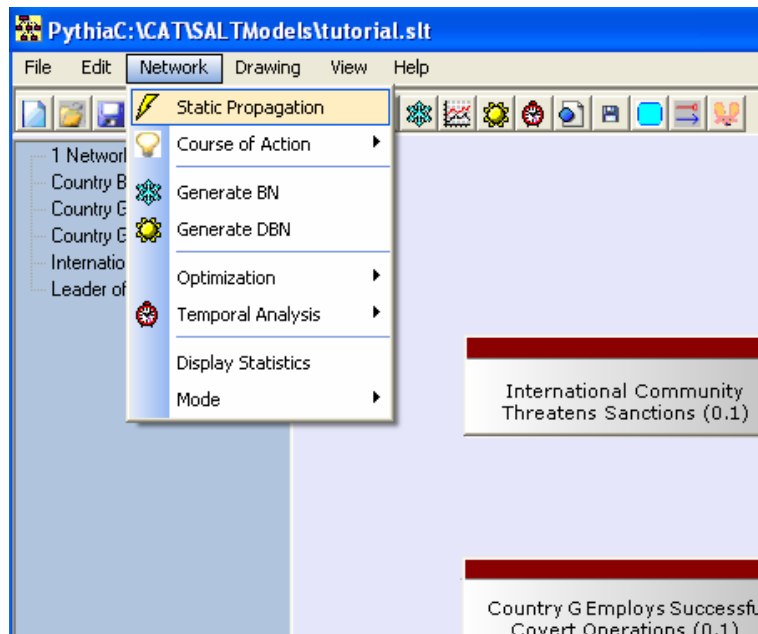


Figure A.10: Static Propagation Menu

A.2.6 Sensitivity Analysis

Once an influence net is specified by a system analyst, it is typically of interest to study the sensitivity of a desired effect to actionable events and influences in the network. More than often, an analyst is also interested in identifying the combination of actions that cause the probability of a desired effect to be above or below a certain threshold. Pythia provides the algorithms that support the above mentioned tasks. Two types of sensitivity analysis are available, ‘Sensitivity to Action’ and ‘Sensitivity to Influence’. The first determines the maximum and minimum probabilities that can be generated by each root node and the second examines the sensitivity of the influence net to changes in the causal strength values in the network.

1. Click on the ‘Network’ and then on ‘Optimization’.
2. Under the ‘Optimization’ sub-menu, click on ‘Sensitivity Analysis’. Figure (A.11)
3. A window will pop up as shown in Figure A.12. The window has two tabs for ‘Sensitivity to Action’ and ‘Sensitivity to Influence’ analyses. The ‘Sensitivity to Action’ analysis tab is the default tab. Select a node from the combo box¹⁹ in the upper middle

¹⁹ A combo box is a user interface control in a graphical user interface that is a combination of a single-line textbox and a menu. It is also commonly referred to as a drop down box, drop down list, or listbox.

portion of the window. For the current example, select the event ‘Country B Agrees to Withdraw’.

4. Press the ‘Sensitivity Analysis’ button. The grid in the middle portion of the window displays the sensitivity of the selected node to the actionable events. Figure (A.13). The table shows the lower and upper probabilities values of the selected node (“Country B agrees to Withdraw”) as the probability of each action node is set to either 0 or 1. The difference column reflects the amount of change that can occur as each action is set to 0 and then to 1.

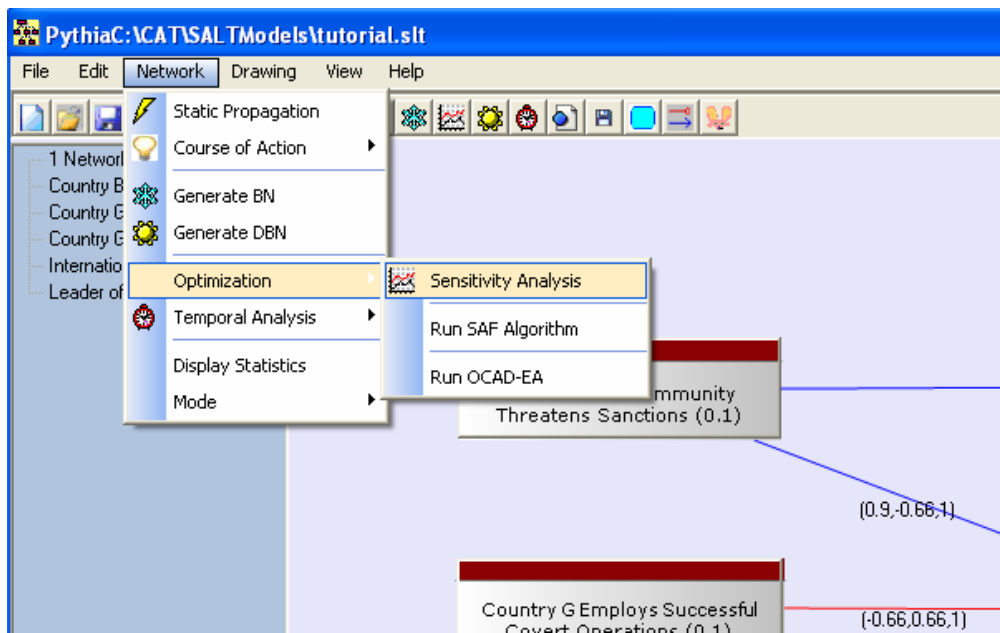


Figure A.11: Sensitivity Analysis Menu

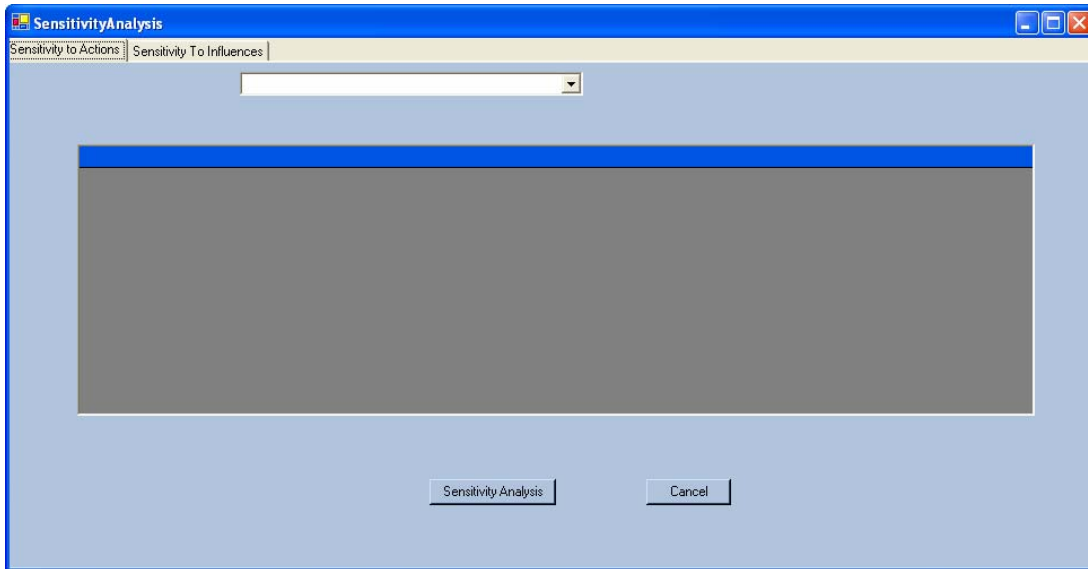


Figure A.12: Sensitivity Analysis Window

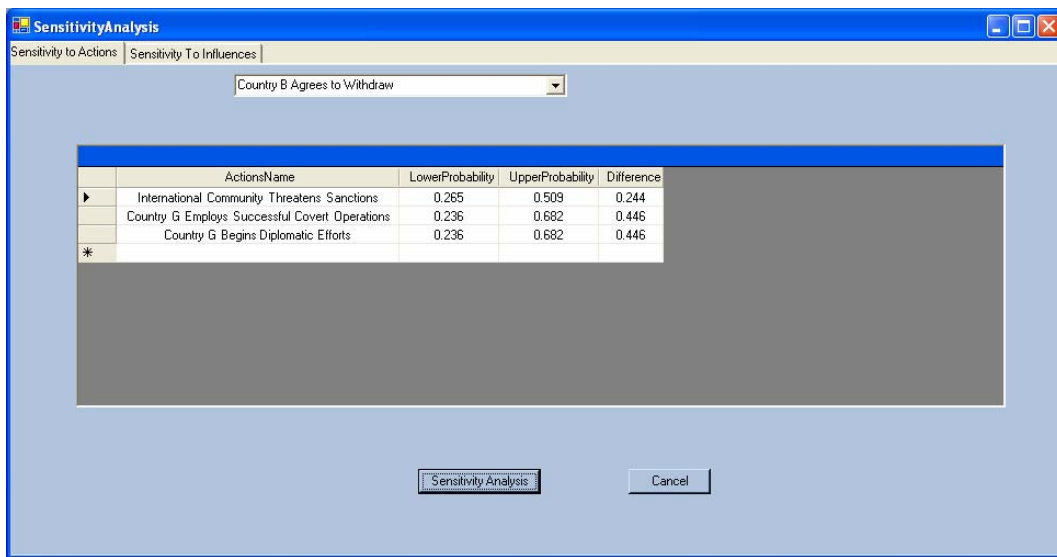


Figure A.13: Result of Sensitivity to Action Analysis

To run the ‘Sensitivity to Influence’ Analysis, click on the right tab of the window of Figure A.12. Select an effect from the combo box and then press the ‘Sensitivity Analysis’ button. The results are shown in Figure A.14. For each link the probability of the selected node is shown as the g or h value is changed from minimum (-.99) to maximum (0.99).

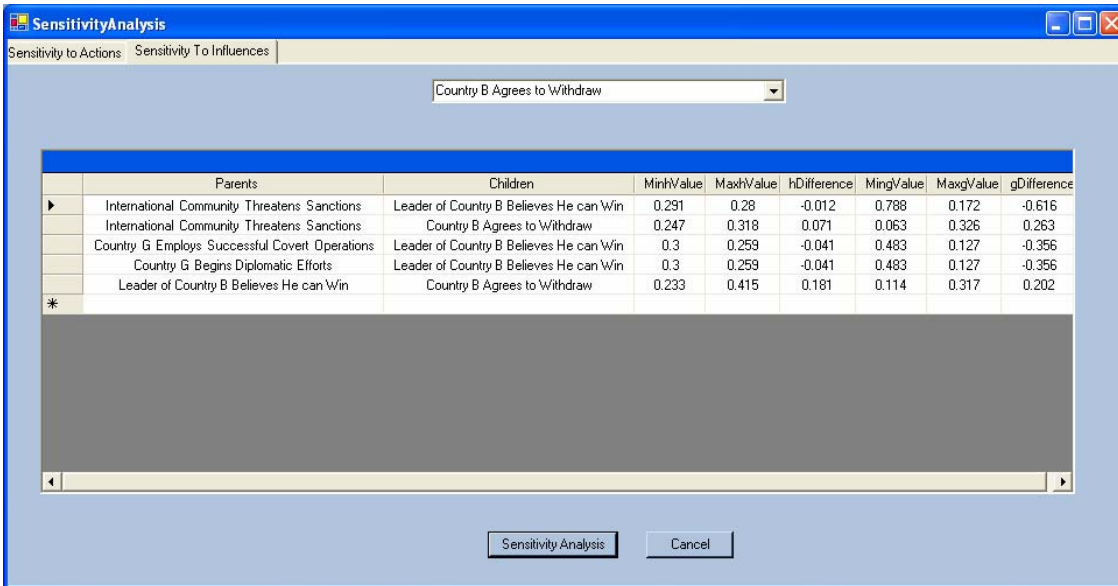


Figure A.14: Result of Sensitivity to Influence Analysis

A.2.7 Running Of SAF (Sets Of Actions Finder) Algorithm

Having done sensitivity analysis on individual actions, the SAF Algorithm will do analysis on sets of actions to determine the combination of actions that provide the best probability for a selected effect.

1. Click on the 'Network' and then on 'Optimization'.
2. Under the 'Optimization' sub-menu, click on 'Run SAF Algorithm'. Figure (A.15)

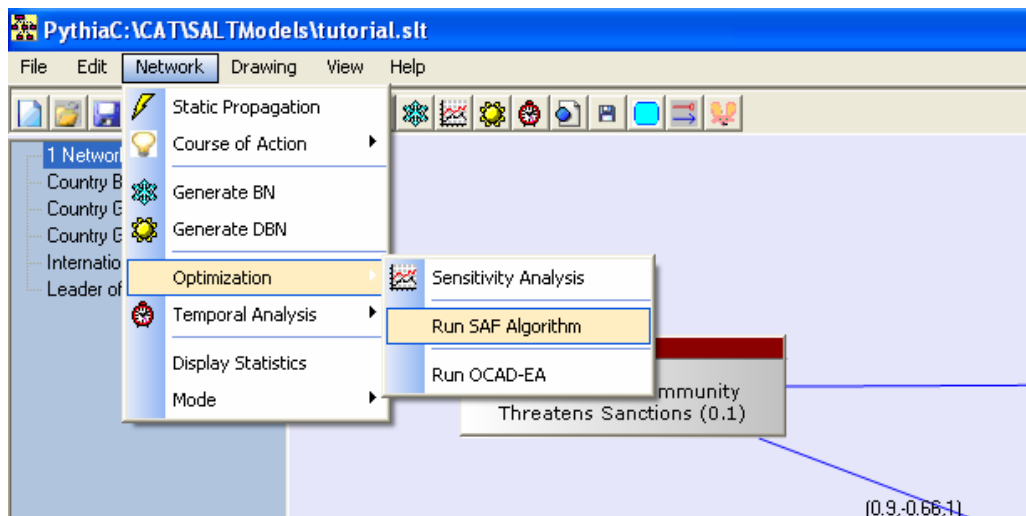


Figure A.15: Run SAF Algorithm Menu

3. A window will pop up as shown in Figure A.16. The window has two tabs for 'Single Objective Optimization' and 'Multi-Objective Optimization'. The 'Single Objective Optimization' tab is the default tab.
4. Select a node from the combo box in the upper middle portion of the window. For the current example, select the node 'Country B Agrees to Withdraw'.
5. Set the 'threshold probability' to 0.7. (Figure A.16)
6. Press the 'Run SAF Algorithm' button. The text box in the middle portion of the window, under the heading 'Set of Actions', displays the sets of actions that cause the selected event's probability to be above 0.70. The order of actions should be read from the left most text box containing event node's label. For instance, the first set of action says 'F T F' and the corresponding probability is 0.711. The order of actions from the left most text box is as follows:
 - a. International Community Threatens Sanctions (F=False; probability = 0)
 - b. Country G Employs Successful Covert Operations (T=True; probability = 1)
 - c. Country G Begins Diplomatic Efforts (F=False; probability = 0)
7. The SAF algorithm has found four sets that cause the probability of the selected event to be above 0.70. Clearly, the best set of actions is to do b and c and not do a.

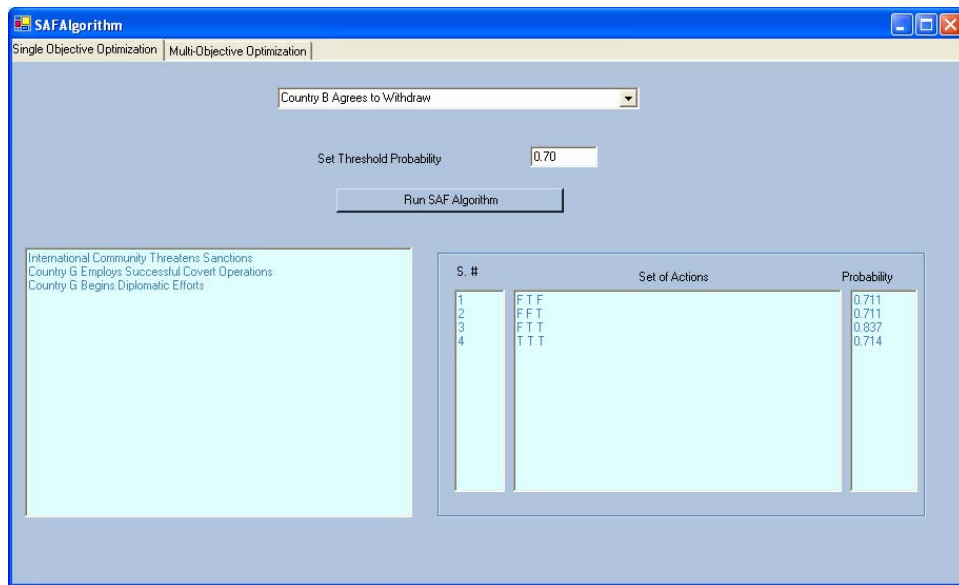


Figure A.16: SAF Algorithm Output Window

A.2.8 Course Of Action Assignment

1. Click on the 'Network' menu and then on 'Course of Action'.
2. Under the 'Course of Action' sub-menu, click on 'Assign COA'. (Figure A.17)
3. A window will pop up (similar to the one shown in Figure A.18). Select an actionable event from the combo box and assign a time stamp and corresponding probability to it. For the current example, select 'International Community Threatens Sanctions' and set time to 2 and the probability to 1 as shown in Figure A.18.
4. Press the 'Update' button.

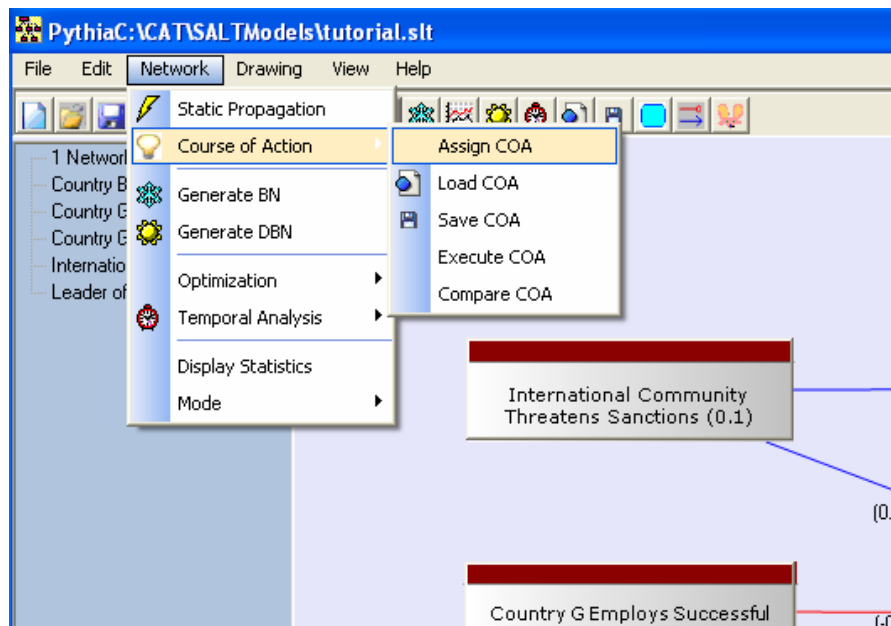


Figure A.17: Assign COA Menu

AssignCOA

Select an Action

International Community Threatens Sanctions

Time	Probability
2	1

Update

Close

Figure A.18: Course of Action Assignment Window

Repeat Steps 3 and 4 for the remaining two actionable events:

- a) Action: 'Country G Employs Successful Covert Operations'
Time Stamp: 5
Probability: 1
- b) Action: 'Country G Begins Diplomatic Efforts'
Time Stamp: 1
Probability: 1

A.2.9 Saving A Course Of Action

1. Click on the 'Network' menu and then on 'Course of Action'.
2. Under the 'Course of Action' sub-menu, click on 'Save COA'. (Figure A.19)

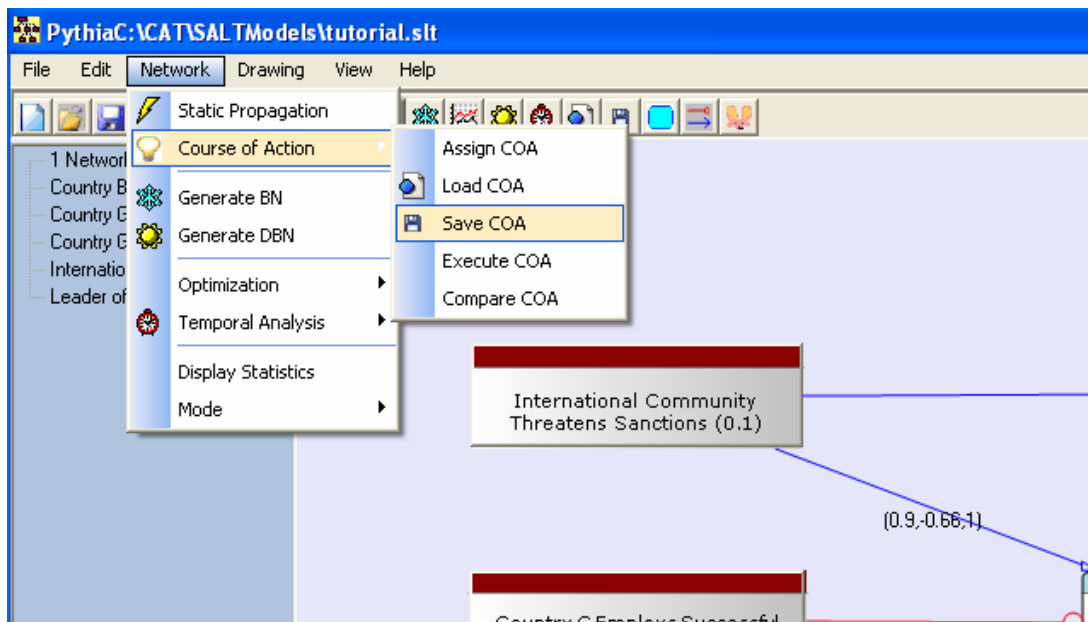


Figure A.19: Save COA Menu

3. Pythia will ask for a name of the course of action. Name the course of action as “tutorial COA1”.
4. Pythia saves the course of action as “tutorial COA1.coa”.

A.2.10 COA Execution / Probability Profile Generation

Before executing a COA and observing its impact on a desired effect, select events whose behavior is of interest. For the current example, let us select ‘Country B Agrees to Withdraw’.

1. Right click on the event ‘Country B Agrees to Withdraw’, and then click on ‘Select Node for Graph’. (Figure A.20)

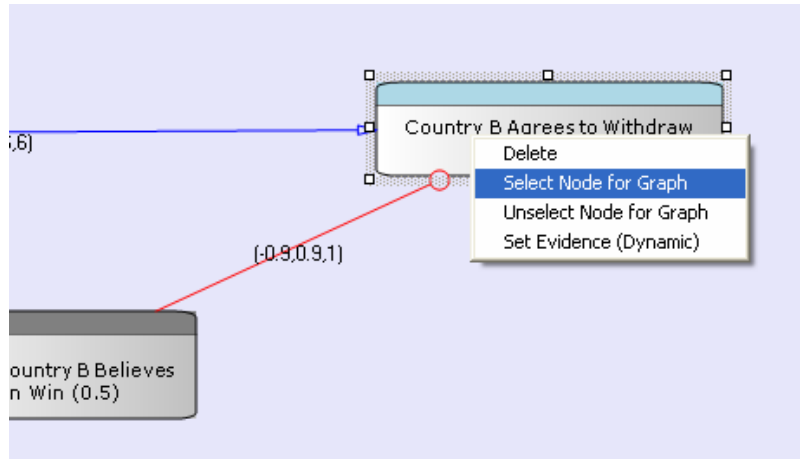


Figure A.20: Event Selection for Probability Profile

3. The border of the node would become thicker; a visual indication that the changes in the belief of corresponding event would be shown in the probability profile.
4. Click on the 'Network' menu and then on 'Course of Action'.
5. Under the 'Course of Action' sub-menu, click on 'Execute COA'. (Figure A.21)
6. Pythia will ask for a name of the probability profile. Name the profile as "tutorial COA1".
7. Pythia saves the profile as "tutorial COA1.ppf".
8. A window will pop up asking for the title of the profile. Enter 'Country B Agrees to Withdraw'. (Figure A.22)
9. A probability profile for the event 'Country B Agrees to Withdraw' is displayed as shown in Figure A.23.

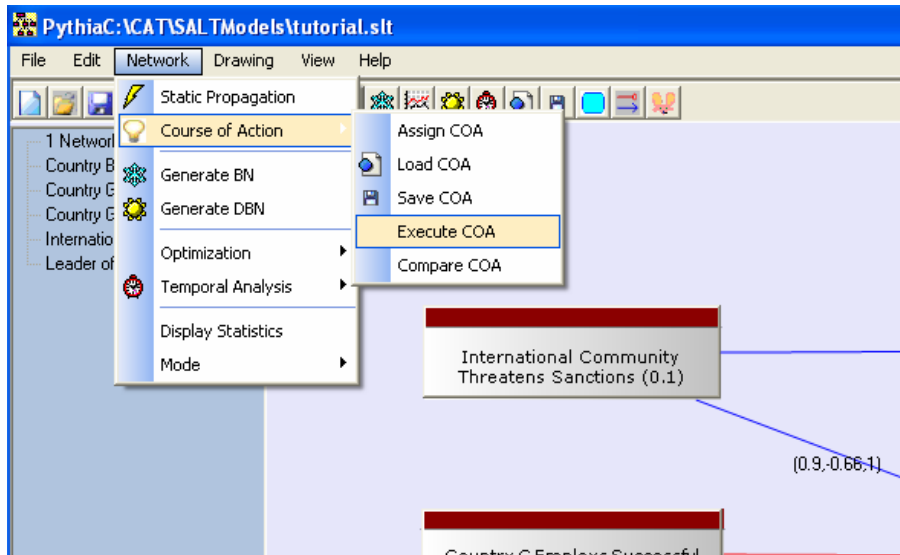


Figure A.21: Execute COA Menu

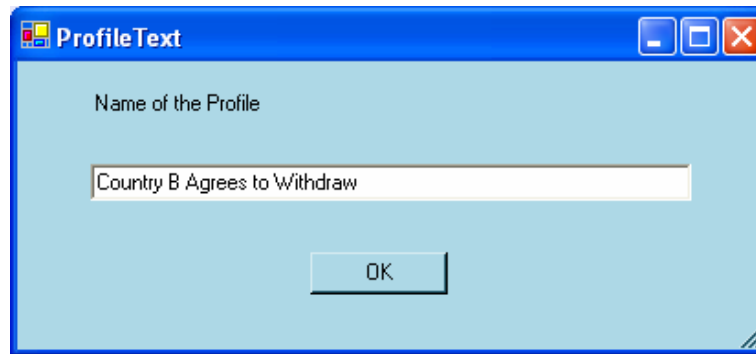


Figure A.22: Title of the Probability Profile Window

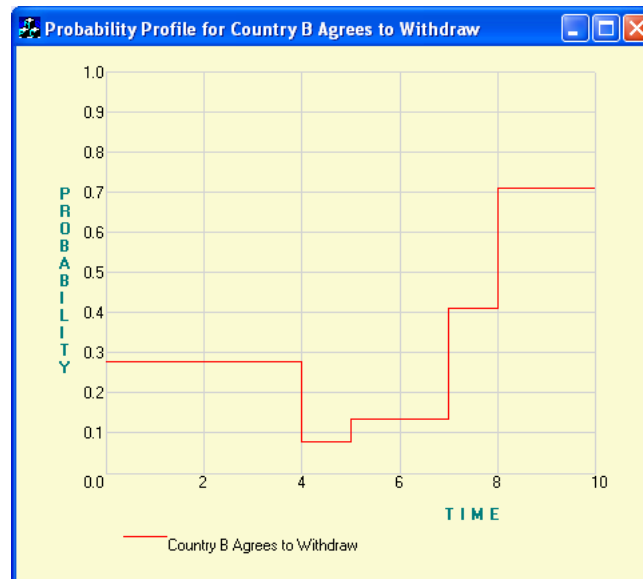


Figure A.23: Probability Profile of the Event 'Country B Agrees to Withdraw'

A.2.11 Running ECAD-EA (Effective Courses of Action Determination using Evolutionary Algorithms) Methodology

The ECAD-EA algorithm is designed to help a system analyst is the identification of sets of actions *and their time of execution* that would maximize the likelihood of achieving a desired effect. It produces several alternative courses of action that have a *similar* likelihood of producing the desired effect.

1. Click on the 'Network' menu and then on 'Optimization'.
2. Under the 'Optimization' sub-menu, click 'Run ECAD-EA'. (Figure A.24)
3. A window will pop up as shown in Figure A.25.

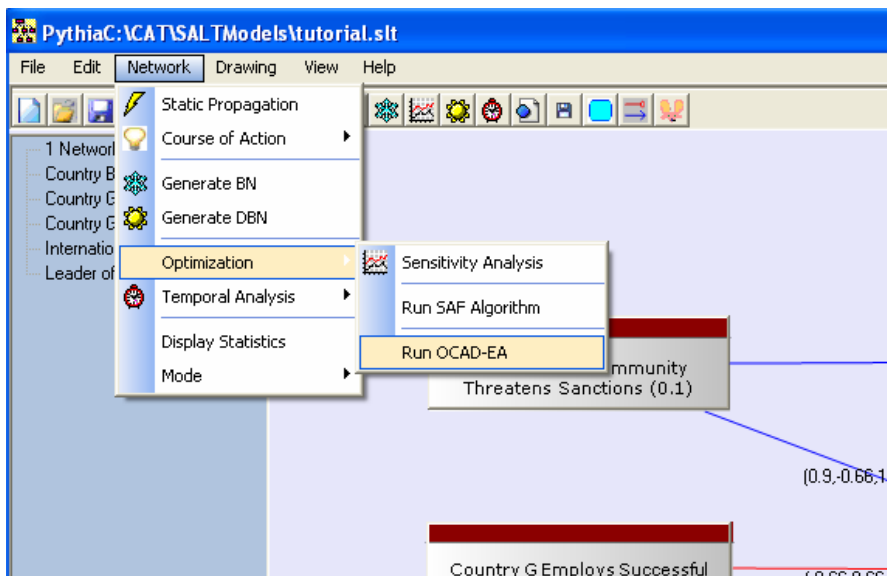


Figure A.24: Run ECAD-EA Menu

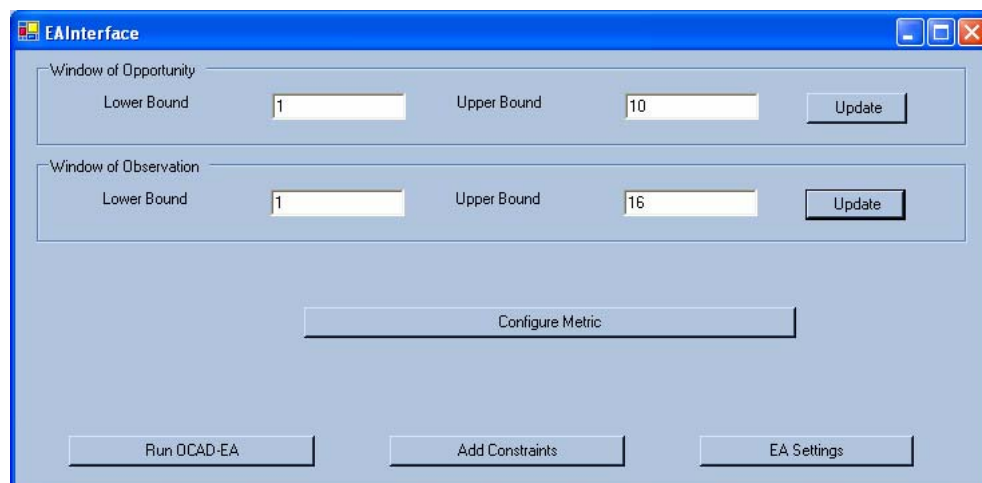


Figure A.25: ECAD-EA Specification Window

A user needs to specify windows of opportunity and observation. A ‘window of opportunity’ represents the time period during which a plan must be executed (all the actionable events must be taken within this time window) while a ‘window of observation’ represents the time period during which a decision maker is interested in getting the desired effect.

4. For the current example, set the upper bounds of windows of opportunity and observation to 10 and 16, respectively. The lower bounds of both windows are set to 1 as shown in Figure A.25.
5. Click on the ‘Configure Metric’ button to select a desired effect that needs to be maximized. A window will pop up as shown in Figure A.26.

A.2.11.1 Selection of Metric

To judge the fitness of a COA, a set of metrics is needed that can be used as Measures of Performance (MOPs). The metrics in Pythia are used to evaluate a COA in terms of the desired effect’s probability profile produced by that particular COA. The fitness of a COA can be measured using either a simple or a composite metrics. In case of a composite metric, weights should be assigned to each of the individual metric. Weights assigned to a metric may vary from situation to situation and are set upon the discretion of the system modeler. A selected metric (or set of metrics) guides the EA through probability profiles’ space while the EA searches for an effective COA.

6. Select a desired effect from the combo box. For the current example, select ‘Country B Agrees to Withdraw’.
7. Click on the ‘Max’ radio button. This tells Pythia that the problem is a maximization problem and we want to maximize the chances of ‘Country B Agrees to Withdraw’ event being true.
8. Enter ‘1’ in the text box next to the ‘Max/Min’ radio buttons.

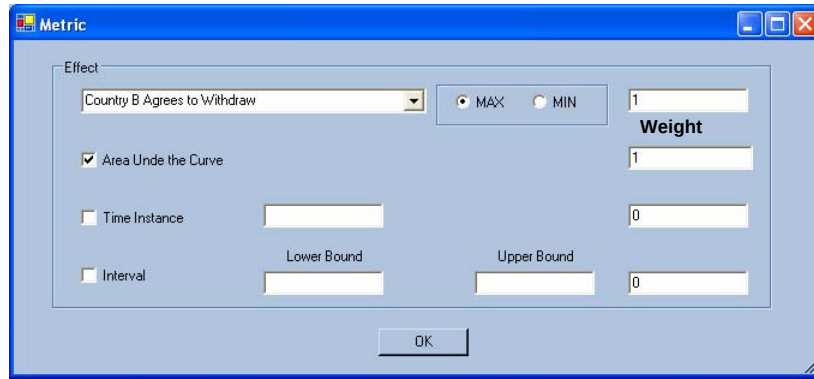


Figure A.26: Metric Selection Window

9. A user needs to specify the metrics that would be used as fitness value of a particular COA. Pythia provides three metrics, Area Under the Curve (AUC), Time Instance, and Time Interval. AUC represents the area under a probability profile. Time Instance and Time Interval metrics cause Pythia to search for COAs that generate probability profiles that have the highest (or lowest if MIN is selected) probabilities at specific time instances or intervals. For the current example, select 'Area under the Curve' metric and assign a weight of 1 to this metric.
10. Press the 'OK' button. The metric assignment window will be closed and the window of Figure A.25 will re-appear.
11. A user can modify the default Evolutionary Algorithm Settings by pressing the 'EA Settings' button. A window will pop up as shown in Figure A.27.

Note: This option is intended to be used by advanced users only²⁰.

A.2.11.2 Evolutionary Algorithm Setting

12. Modify the 'Population' and 'Generation' size of the Evolutionary Algorithm. For the current example, let the population size be 10 and the generation size be 10.
13. The 'Parent Selection' scheme is set to 'Uniform Stochastic' while the 'Generation Selection' scheme is set to 'Truncation'.

²⁰ A detailed discussion of the Evolutionary Algorithm Settings can be found in Haider, S. and Levis, A. H., "On Finding Effective Courses of Action in a Complex Situation using Evolutionary Algorithms", In the Proceedings of 10th International Command and Control Research and Technology Symposium, Washington DC, June 2005.

14. Press the 'OK' button. The EA setting window will be closed and the window of Figure A.25 will re-appear.

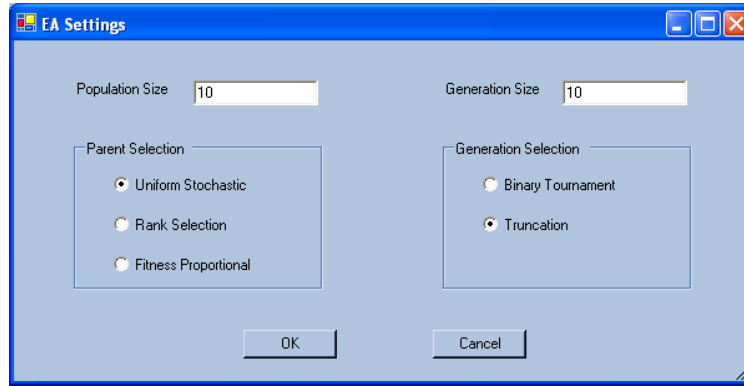


Figure A.27: Evolutionary Algorithm Setting Window

15. For the time being, suppose that there are no temporal and causal constraints among actionable events. Press the 'Run OCAD-EA' button to run the Evolutionary Algorithm.

Note: At the time of writing of this document, Pythia generates the output in a text file. The development of a graphical interface is currently under development and it should be available in the next version of Pythia.

16. Pythia stores the top five COAs in a text file named 'bestCOA'. (Figure A.28) Each row of the file shows a COA with the state of each action (either True or False) and the time that it should be set to that state. Note that the algorithm assumes that all actionable event nodes are initially set to the marginal probability value specified for the node.

17. The file is located in the same directory where the tutorial is saved.

A	time	B	time	C	time	-- Fitness
F	8	T	1	T	2	-- 0.70
F	2	T	2	T	2	-- 0.70
F	6	T	1	T	4	-- 0.69
F	5	T	1	T	6	-- 0.68
F	2	T	1	T	8	-- 0.64

Figure A.28: Five COAs produced by the ECAD-EA

18. To see the performance of the top COA, assign the time stamp and the corresponding probabilities to actionable events as described earlier in Section A.2.8.

19. Following the steps described in Section A.2.10, the probability profile of Figure A.29 is generated from the selected COA.

Before proceeding, save the COA and the corresponding profile with the name bestCOA_ECADEA.coa and bestCOA_ECADEA.ppf, respectively.

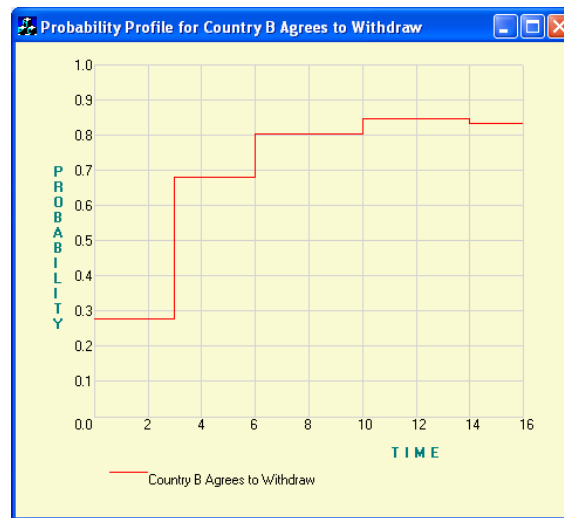


Figure A.29: Probability Profile Produced by the Best COA

A.2.11.3 Constraint Specification

Now suppose there are the following constraints among the actionable events in tutorial example:

- Country G Employs Successful Covert Operations **Before** International Community Threaten Sanctions
- Country G Employs Successful Covert Operations **Before** Country G Begins Diplomatic Efforts
- International Community Threatens Sanctions **Must be True**

20. Assuming that the user has already performed Steps 1-14, click on the button 'Add Constraints' in the window of Figure A.25. A window, similar to the one shown in Figure A.30 will pop up.

21. Under the Heading 'Before/Equal Time Constraint', select 'Country G Employs Successful Covert Operations' in the left side combo box and select 'International Community Threatens Sanctions' in the right side combo box. Select Before from the 'Before/Equal' combo box and keep the time to its default value (0).
22. Press the Add Button. The constraint will be shown in the lower portion of the window.

Figure A.30: Constraint Assignment Window

23. Repeat Steps 20 and 21 for constraint b mentioned above.
24. Under the heading 'Fixed State Constraint', select 'International Community Threatens Sanctions' in the combo box and select 'True' from 'State' combo box.
25. Press the Add Button.
26. The lower portion of the window shows all three constraints as shown in Figure A.31.
27. Repeat Steps 15-18 described above to generate the five COAs in the presence of the given constraints. The COAs are shown in Figure A.32 and the profile generated from the best COA is shown in Figure A.33.

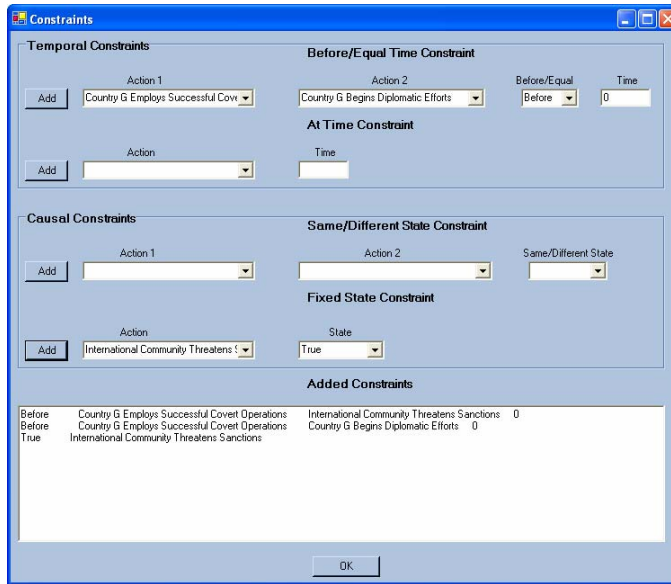


Figure A.31: Constraint Assignment Window with Constraints

A time	B time	C time	Fitness
T 9	T 1	T 6	-- 0.53
T 8	T 4	T 5	-- 0.44
T 4	T 1	T 2	-- 0.44
T 9	T 4	T 8	-- 0.43
T 8	T 1	F 7	-- 0.39

Figure A.32: Five COAs Produced by the ECAD-EA under Certain Constraints

Before proceeding, save the COA and the corresponding profile with the names bestCOA_ECADEA_constraints.coa and bestCOA_ECADEA_constraints.ppf, respectively.

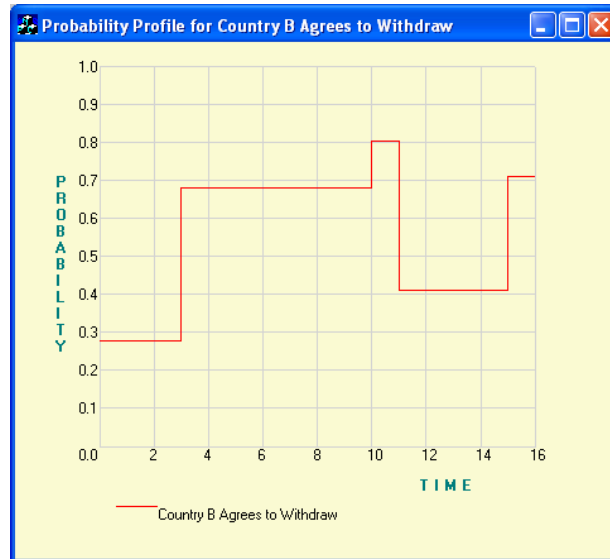


Figure A.33: Probability Profile Produced by the Best COA under certain Constraints

A.2.12 COA Comparison

1. To compare different probability profiles, click on the 'Network' menu and then on 'Course of Action'.
2. Under the 'Course of Action' submenu, click on 'Compare COA'. (Figure A.34)
3. A window, similar to the one shown in Figure A.35, will pop up. Click on the 'Browse1' button to locate the profile saved in Section A.2.10.
4. In the 'Label1' text box, type 'Manual COA'.

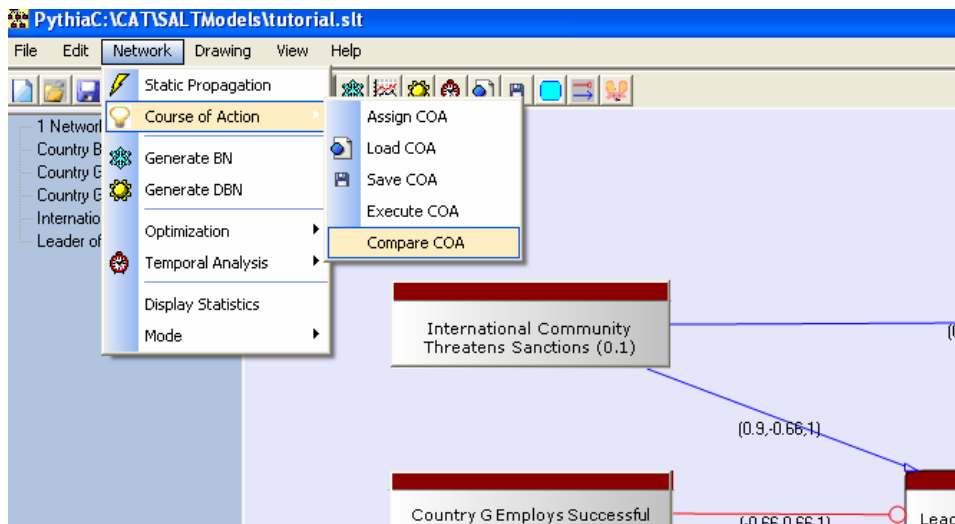


Figure A.34: Compare COA Menu

Repeat Steps 3-4 to locate the profiles saved in Sections A.2.11.2 and A.2.11.3. Type 'ECAD-EA COA' and 'ECAD-EA Constraints COA' in 'Label1' and 'Label2' text boxes, respectively. The window will look like the one shown in Figure A.35.

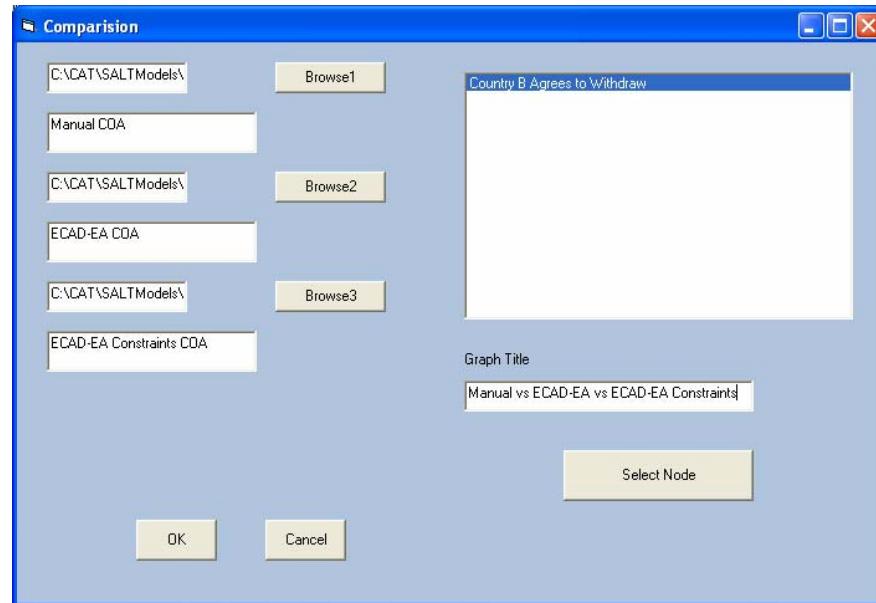


Figure A.35: Compare COA Window

5. Press the 'OK' button. A text titled 'Country B Agrees to Withdraw' will appear in the text area on the right side of the window. (Fig. A.35)
6. In the 'Graph Title' text box, type 'Manual vs ECAD-EA vs ECAD-EA Constraints'.
7. Press the 'Select Node' button. A window with the three profiles will appear (Fig. A.36).

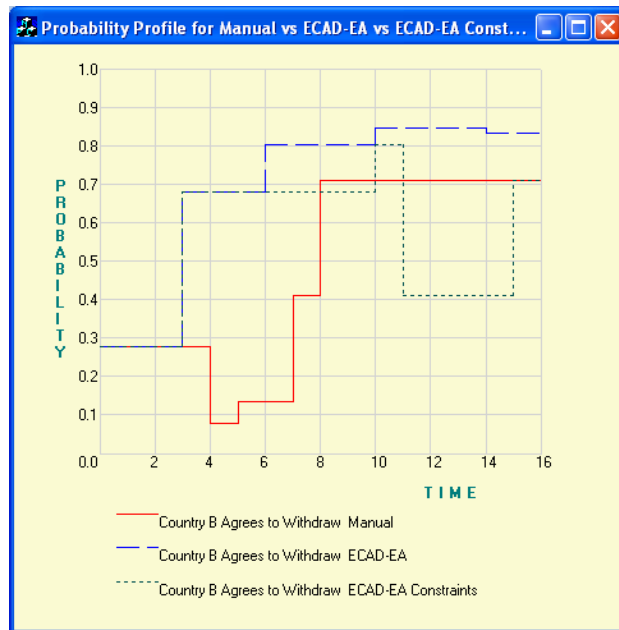


Figure A.36: Comparison of Different Profiles

A2.13 TEMPORAL ANALYSIS

Pythia enables an analyst to conduct temporal analyses of the Timed Influence Net. This analysis can be used to compliment the analysis done using the ECAD-EA algorithms. Two types of analysis are available, Temporal Queries, and If-then Analysis. Assume we are continuing the use the same TIN and that it has been constructed or loaded into the tool.

1. Construct a Point Graph: Click on the Network Menu, then on Temporal Analysis, and then on Construct PG, as shown in Figure A.37.

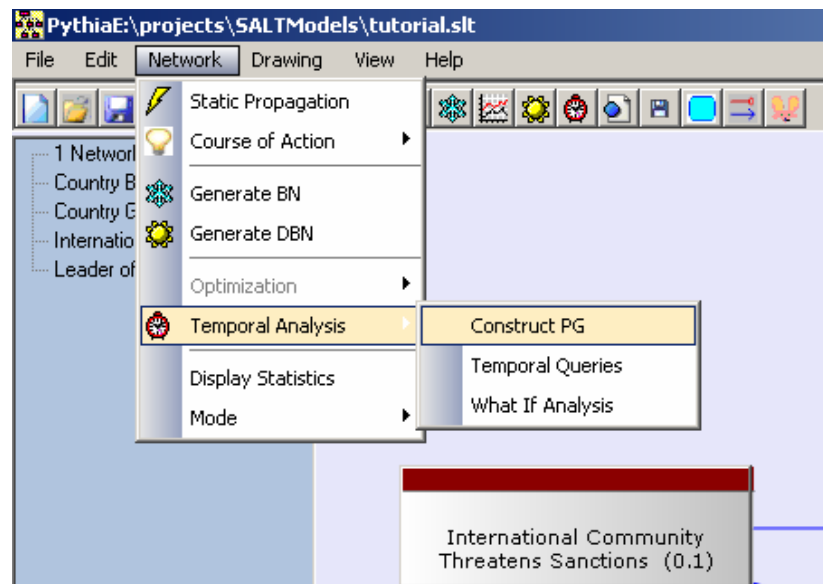


Figure A.37: Construct PG Menu

2. After few seconds, a message will appear saying “Point Graph Constructed”, as shown in Figure A.38.

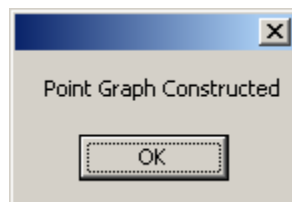


Figure A.38: Message after PG Construction

3. Click “OK”.

4. Open the Temporal Queries Window: Click on the Network menu, then on Temporal Analysis, and then on Temporal Queries, as shown in Figure A.39.

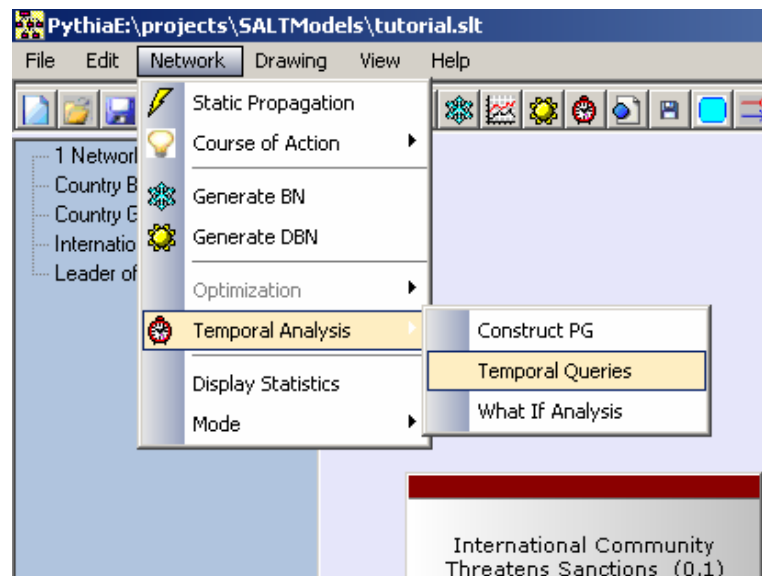


Figure A.39: Temporal Queries Menu

5. A Window, similar to Figure A.40, will pop-up.

Suppose we are interested in knowing what causes a decrease in the probability of “Country G Agrees to Withdraw” at time 4, as was shown in the profile of Figure A.23.

6. In the window of Figure A.40, select “Country G Agrees to Withdraw” from the ‘Select a Node’ combo box and select “4” from the “Select a Time Stamp” combo box.

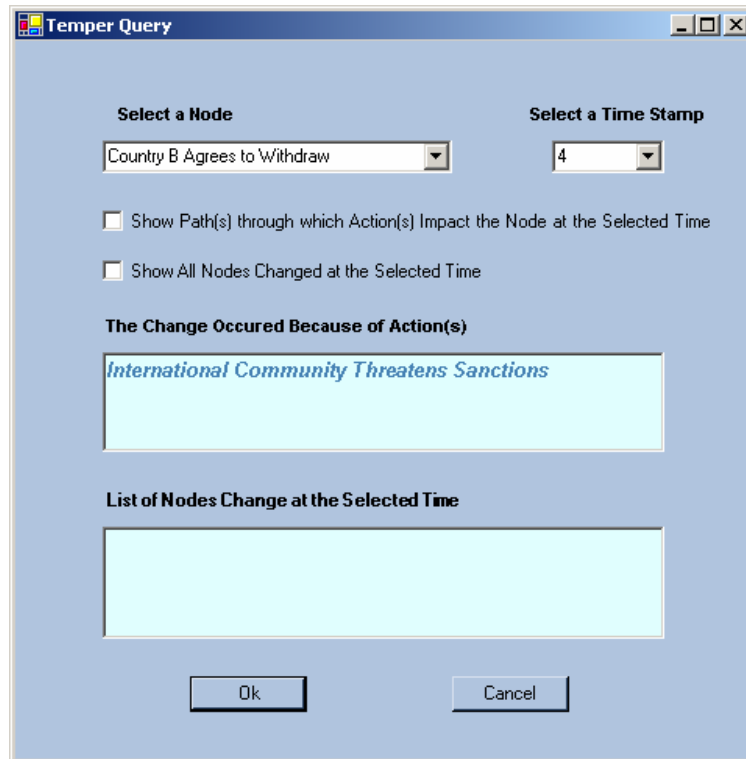


Figure A.40: Temporal Queries Window

7. Click on the OK Button.

The results of the query are shown in the upper text box having label “The Change Occurred Because of Action(s)”, as shown in Figure A.40. The result shows that the change at time 4 occurred because of the action “International Community Threatens Sanctions”.

8. Select the Show Path(s) box and the Show All Nodes box, and click the OK button.

The results of the query are shown in the lower text box labeled “Listed of Nodes Change at the Selected Time”, as shown in Figure A.41. The result shows that the path that caused the change at time 4 included “Leader B Believes He can Win”.

9. Observe the path on the main Pythia screen as shown in Figure A.42.

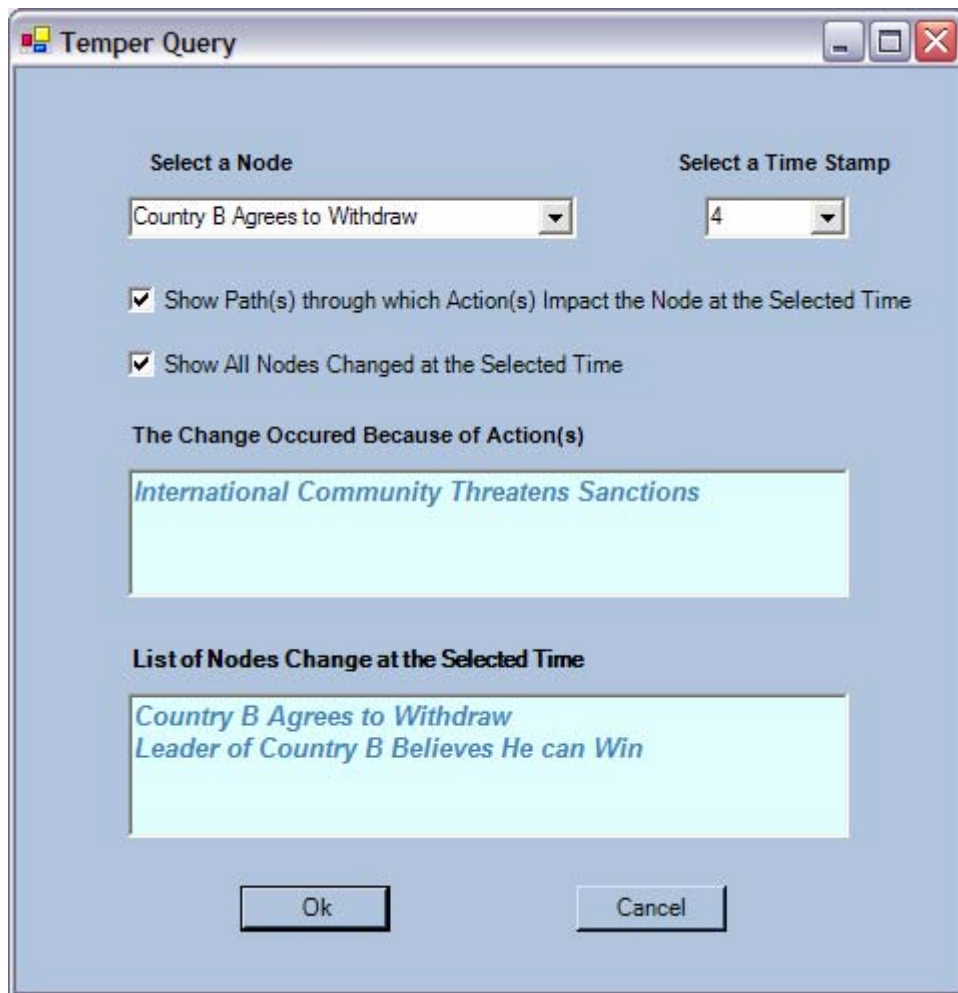


Figure A.41: Temporal Queries Window

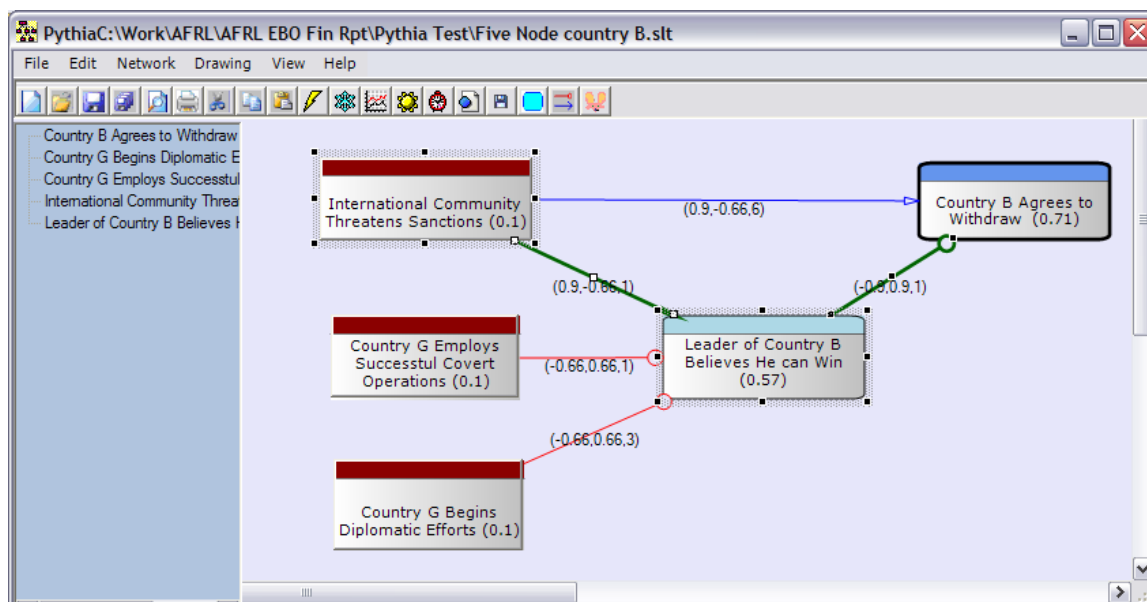


Figure A.42: Path Displayed on TIN

10. What If analysis allows the analyst to determine relationships between actions and effects. Open the Temporal Queries Window: Click on the Network menu, then on Temporal Analysis, and then on What If Analysis, as shown in Figure A.43.

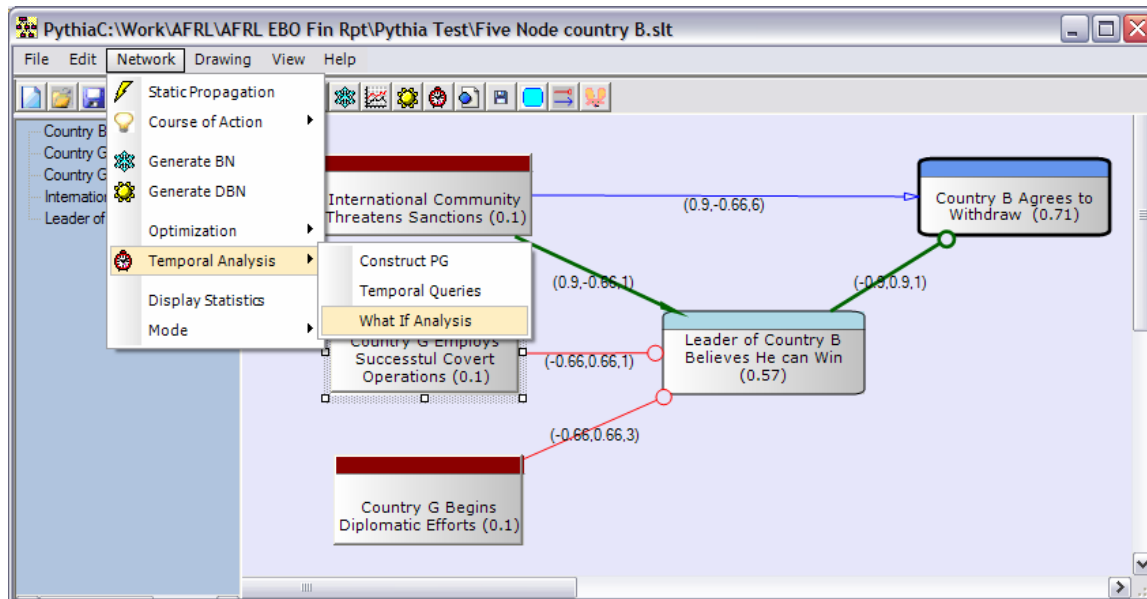


Figure A.43: Temporal Queries Menu

11. A Window, similar to Figure A.44, will pop-up.

The screenshot shows a window titled "What If Analysis" with a light blue background. It contains several dropdown menus and a button. At the top right are standard window control buttons (minimize, maximize, close). The interface is organized into sections:

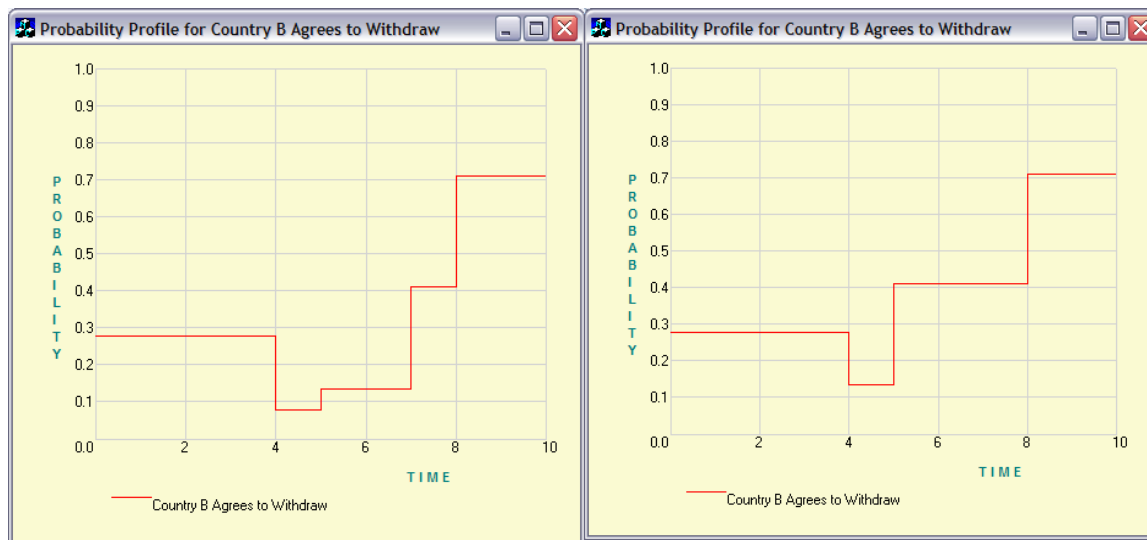
- Select the First Node:** A dropdown menu showing "Country B Agrees to Withdraw".
- Select a Time:** A dropdown menu showing "4".
- Select the Second Node:** A dropdown menu showing "Country B Agrees to Withdraw".
- Select a Time:** A dropdown menu showing "7".
- Relationship Between Following Actions is:** A section with two sub-sections:
 - Action 1:** A dropdown menu showing "International Community Threatens San".
 - Action 2:** A dropdown menu showing "Country G Employs Successtul Covert".
- Below these, a text box displays the relationship: "International Community Threatens Sanctions = Country G Employs Successtul Covert O".
- At the bottom center is a button labeled "Run Query".

Figure A.44: If Then Queries Menu

Suppose we are interested in seeing if we can eliminate the decrease in the probability profile at time 4 by “moving” the action that causes the increase at time 7 as was shown in the profile of Figure A.23.

12. In the window of Figure A.44, select “Country G Agrees to Withdraw” from the ‘Select the First Node’ combo box and select “4” from the “Select a Time” combo box. Again select “Country G Agrees to Withdraw” from the ‘Select the Second Node’ combo box and select “7” from the “Select a Time” combo box. Select “International Community Threatens Sanctions” and Country G Begins Diplomatic Efforts in the boxes labeled Action 1 and Action 2, respectively. (We know from a temporal query that Action 2, Covert Operations, affects the Country B Agrees to Withdraw node at time 7).

13. Click on the Run Query Button. The result is shown in the bottom box of the window as shown in Figure A.44. It says that the Diplomatic Efforts must occur at the same time as the International Community Threatens Sanctions. Since we believe that the International Community will threaten sanctions at time 2, we need to change the time of the covert action to time 2. Change the COA in Pythia and execute the new COA. The probability profile changes to the one shown in Figure A.45 (b), which shows some improvement over the original probability profile shown in Figure A.45 (a). One could continue to refine the probability profile by attempting to eliminate the “dip” at time 4. Note that the ECAD-EA analysis without constraints yielded a better probability profile, but required the International Community NOT to threaten sanctions, even though it may appear that these will help. Save the new COA as COA2.



(a) Initial Probability Profile (see Fig A.23) (b) Profile After First What If Analysis

Figure A.45 Comparison of Profiles Generated from If Then Analysis

A.2.14 CONVERSION TO DYNAMIC BAYES NETS FOR ADDING EVIDENCE

Pythia can convert TINs to Dynamic Bayes nets (DBN) (sometimes called time sliced Bayes nets). This is useful for comparing the probability profiles generated by the TIN algorithm with the ones generated by a Dynamic Bayes net. Generally the results are nearly identical. Even more importantly, the DBN can be used to add “hard” evidence of the true or falsity of a non root node that may become known from evidence gathered as a COA is executed. To demonstrate, let us continue working with the TIN and the COA2. If you are starting from

scratch, open the TIN (or create it), load the COA (or create it), and execute the net with the lightning bolt.

1. Open the Temporal Queries Window: Click on the Network menu, then on Generate BN, as shown in Figure A.46.

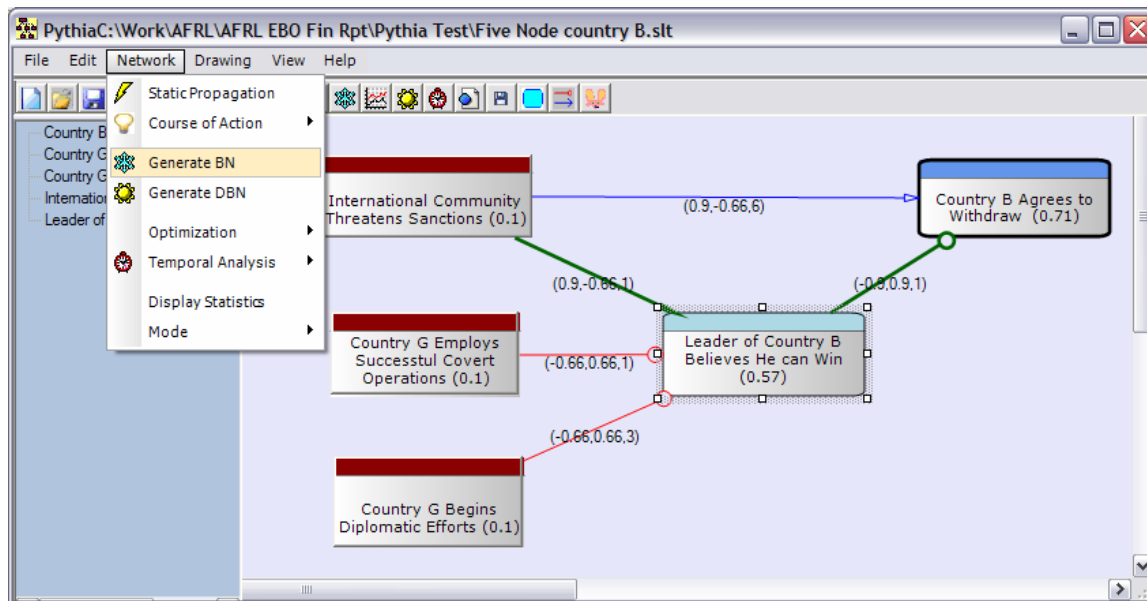


Figure A.46: Generating a Bayes Net

Nothing will happen, but the Bayes net will have been created.

2. Open the Temporal Queries Window: Click on the Network menu, then on Generate DBN, as shown in Figure A.47.

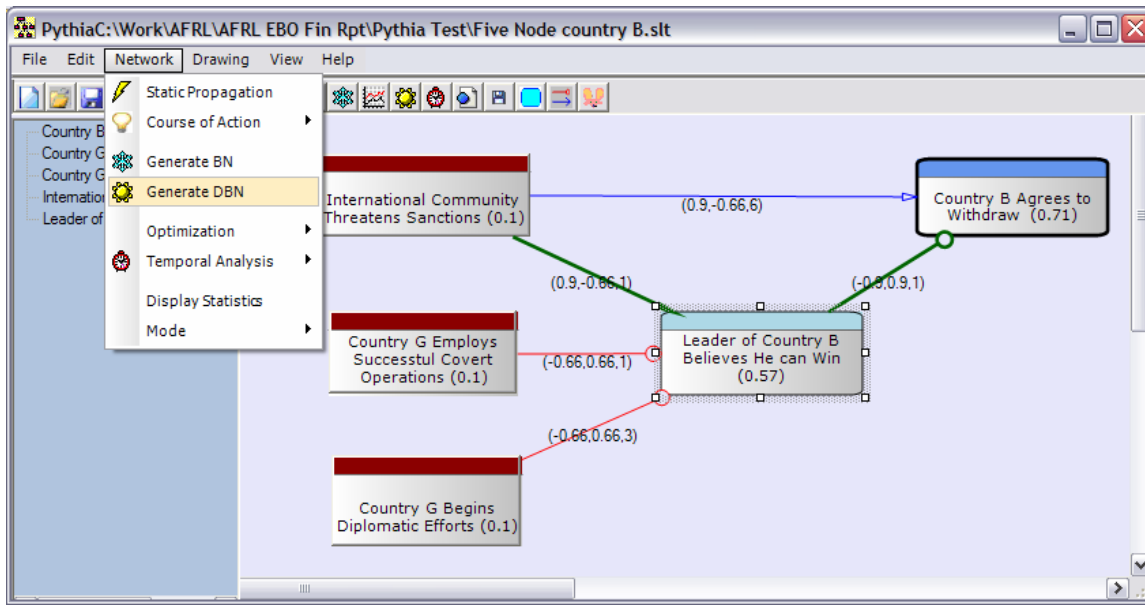


Figure A.47: Generating a Dynamic Bayes Net

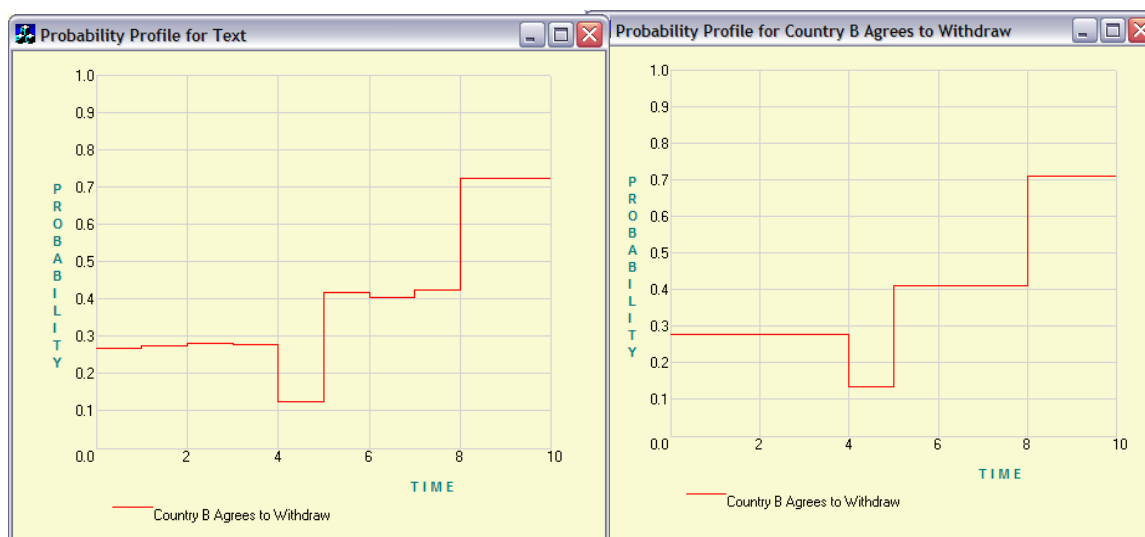
3. A Window, similar to Figure A.48, will pop-up.

The screenshot shows a dialog box titled 'NumberOfTimeSlice'. It has three input fields: 'Maximum Path Length in the TIN' with a value of 6, 'Maximum Time Stamp in the Selected COA' with a value of 2, and 'Total Number of Time Slices' with a value of 8. There is an 'OK' button at the bottom. On the right side, there is a text label: 'You Can Change the Number of Slices'.

Figure A.48: Selecting the number of time slices

This window allows the modeler to set the number of instantiations or time slices of the Bayes net that will be used in the DBN. The default value is based on the maximum length of time units the TIN and the COA needed to complete the probability profile. Click the OK button.

4. The standard window will open asking you to name and save the .ppf file that will be generated by the DBN. Assign a name and click save. The probability profile generated by the DBN will appear. Figure A.49 (a) shows it along side the one generated by the TIN in Figure A.49 (b).



(a) DBN Probability Profile

(b) TIN Profile (See Figure A.45 (b))

Figure A.45 Comparison of DBN and TIN Profiles

Suppose we learn from intelligence, that the Leader of Country B has definitely decided he can win and that he made this decision at time 4. This evidence can be added to the DBN, and its impact on the probability profile can be examined.

5. In the main window select the “Leader of Country B Believes he can Win” node. Right click on the node and a window pops up as shown in Figure A.46.

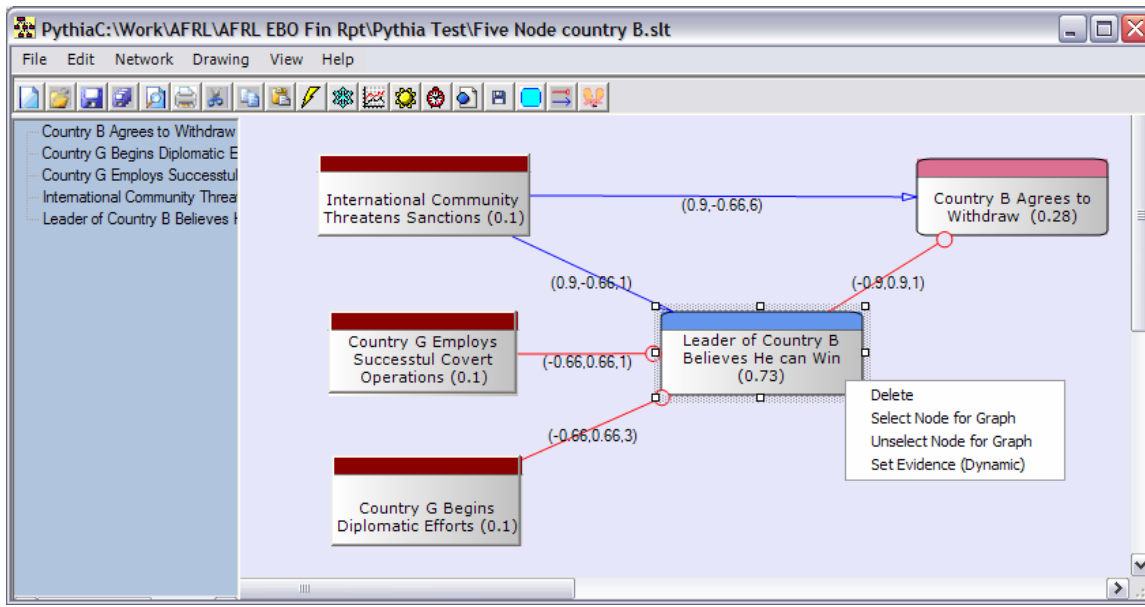


Figure A.45 Select and Setting a node for evidence

6. Select the “Set Evidence (Dynamic)” choice in the window. A window like the one in Figure A.46 will appear. Enter the time of the evidence and its probability (1 in this case). Click the OK button. (Note that Pythia will only accept the value of 1 or 0 for the probability because it is assumed the evidence confirms that the statement in the node is either true or it is false. Putting in other values may result in an error message.)

Figure A.46 Inputing Evidence

Pythia opens a response window as shown in Figure A.47.

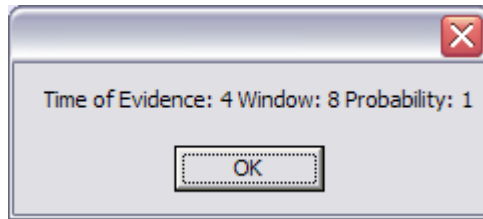
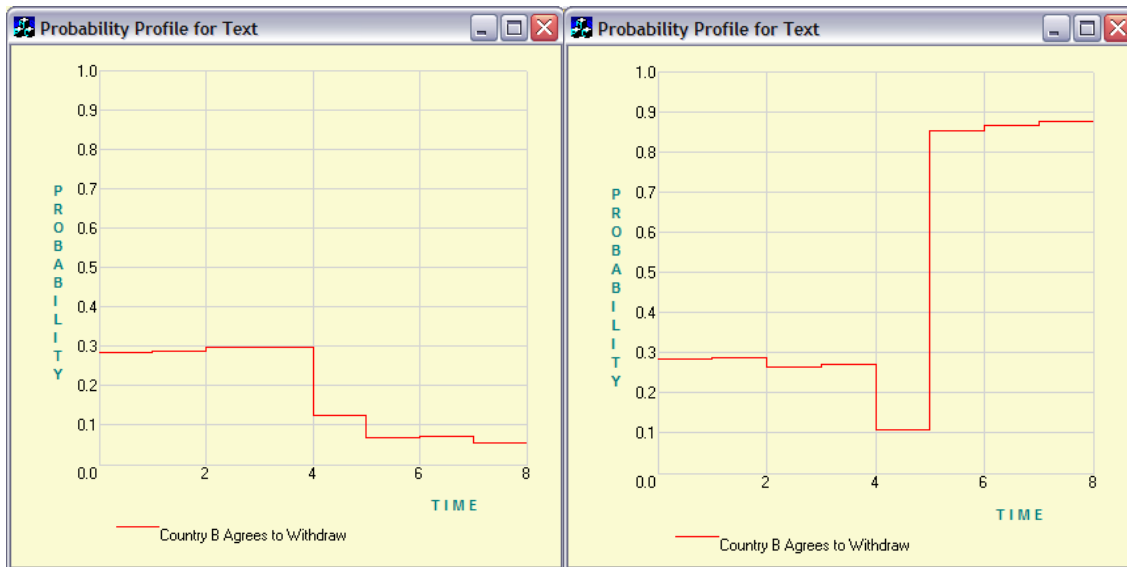


Figure A.48 Evidence Confirmation

7. Click the OK button. Pythia will open the standard window asking for you to name the new .ppf file (probability profile file) that will be created with the new evidence. Provide a name and save the file. The probability profile shown in Figure A.49 (a) is generated. Note how this “hard” evidence strongly affects the overall effect node (“Country B Agrees to Withdraw”). For illustration purposes, the same evidence entering sequence was followed with the evidence being false and the probability profile shown in Figure A.49 (b) was generated.



(a) Evidence Leader Can Win = 1 at time 4 (b) Leader Can Win = 0 at time 4

Figure A.49 Comparison of effects of Evidence

A.2.15 DYNAMIC INFLUENCE NETS (DINS)

Pythia enables an analyst to create Dynamic Influence Nets. These extensions of the TIN support concept of persistence of influence in which the strength an influence can be specified to vary as a function of how old inputs to a node are. This allows the concept that the strength of an influence can change over time representing the notion that as time goes by the strength and memory of an influencing event may change (typically it goes down). With DINs, modelers are allowed to specify various strengths of influences and their corresponding window of effectiveness.

Let us continue to use the TIN and COA2. If you are starting from scratch, open the TIN (or create it), load the COA (or create it), and execute the net with the lightning bolt.

1. Open the Temporal Queries Window: Click on the Network menu, click on the “Mode” item, then on “Dynamic Influence Nets”, as shown in Figure A.50.

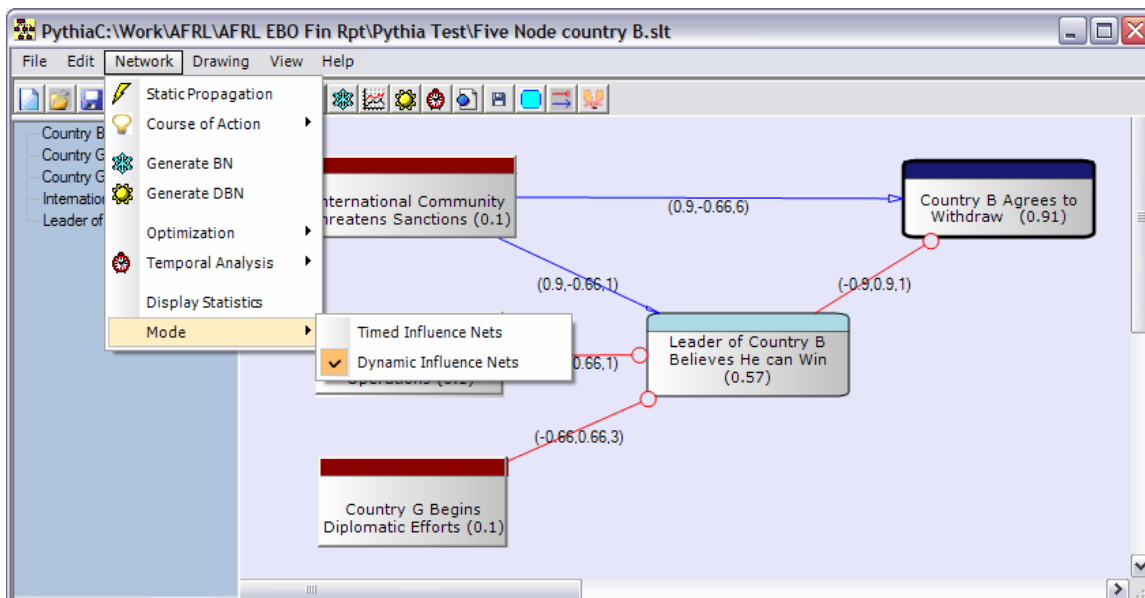


Figure A.50 Select Dynamic Influence Net Mode

2. Click on the arc that goes from “International Community Threatens Sanctions” to “Country B Agrees to Withdraw”. A window like the one in Figure A.51 will open.

The screenshot shows a window titled "TimeVaryingCASTInterface". It contains a table for setting causal strength parameters. The table has four columns: "If the Premise (Parent) is TRUE this will influence the Consequence (Child)", "If the Premise (Parent) is FALSE this will influence the Consequence (Child)", "Lower Bound", and "Upper Bound". There are three rows of data, each representing a different time interval. The first row has values 90, -66 (Moderately Less Likely), 6, and 7. The second row has values 66 (Moderately More Likely), -66 (Moderately Less Likely), 7, and 8. The third row has values 33 (Slight More Likely), 33 (Slight More Likely), 8, and 10. Below the table is an "Arc Delay" field with the value 6. At the bottom are "OK" and "Cancel" buttons.

If the Premise (Parent) is TRUE this will influence the Consequence (Child)	If the Premise (Parent) is FALSE this will influence the Consequence (Child)	Lower Bound	Upper Bound
90	-66 (Moderately Less Likely)	6	7
66 (Moderately More Likely)	-66 (Moderately Less Likely)	7	8
33 (Slight More Likely)	33 (Slight More Likely)	8	10

Arc Delay: 6

OK Cancel

Figure A.51 Setting DIN Causal Strength Parameters

3. Use the pull down windows to select the Causal Strengths and set the time intervals during which those strengths will apply. The Arc Delay can be set also.

In setting the parameter values consider that the first row means that the causal strengths shown will be used if the node needs to update its probability (because information arrived on a different arc) and at the time of update the information from the arc for which the parameters are being set is between 6 and 7 units old. (Note that with the time delay of 6, the information will always be at least 6 time units old). The second and third rows in the window show that the strength of the influence from the “International Community Threatens Sanctions” decreases to 66 and -66 if the information is 7 units old and to 33 and -33 if the information is 8 or 9 units old. (We know from the COA and delays that the information will never be older than this). Click the OK button.

4. Let us use the same procedure to set DIN Causal Strength Parameters on the arc from the node “Leader of Country B Believes He can Win” to the node “Country B Agrees to Withdraw”. Set the parameters as shown in Figure A.52 and click the OK button.

The dialog box is titled "TimeVaryingCASTInterface". It contains two columns of dropdown menus for causal strength, a "Time Interval" section with "Lower Bound" and "Upper Bound" input fields, and an "Arc Delay" input field.

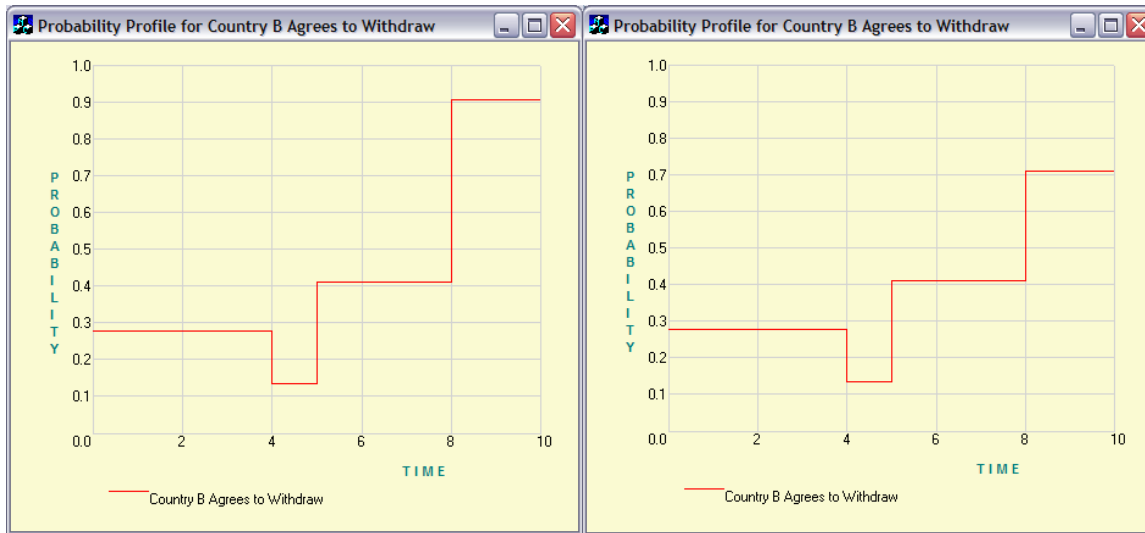
If the Premise (Parent) is TRUE this will influence the Consequence (Child)	If the Premise (Parent) is FALSE this will influence the Consequence (Child)	Lower Bound	Upper Bound
-90	90	1	2
-66 (Moderately Less Likely)	66 (Moderately More Likely)	2	4
-66 (Moderately Less Likely)	66 (Moderately More Likely)	4	10

Arc Delay: 1

Buttons: OK, Cancel

Figure A.52 Setting DIN Causal Strength Parameters

5. Execute the COA as described in A.2.10, and the probability profile shown in Figure A.53 (a) is generated. Figure A.53 (b) shows the TIN profile for comparison purposes.



(a) Profile for DIN

(b) Profile for TIN

Figure A.53 Comparison of DIN and TIN

The impact of the time varying influence is evident at time 8 when the “International Community Threatens Sanctions” influence arrives. At this time, the input from the node “Leader of Country B Believes he can Win” is 5 units old and so a lesser Causal Strength is used in the DIN than in the case of the non varying causal strengths of the TIN.

Within the DIN mode Pythia allows an additional persistence of influence concept to be modeled using self loops. This allows the model to reflect the concept that the change in state (probability) of a node may not only depend on the arrival of new information from its parents, but will also take into account its own current state. This is a form of memory. It represents the concept that one considers current thinking in addition to the input of new information.

Let us incorporate this feature in the DIN we have just created. Remain in the DIN mode.

To create a self loop, a node must not have any arcs connected to it. Let us assume that we want to add the self loop concept to the node “Country B Decides to Withdraw”. First, delete the two arcs that are coming into that node by first selecting the arc and then right clicking to select “delete link”. Now the node has no arcs connected to it.

1. Select the arc drawing mode as described in A.2.4. Draw the self loop arc by starting inside the node moving the mouse outside the node and then back into the node while holding the left button down. Release the button when you have re-entered the node. The Setting Causal Strength window like the one in Figure A.54 will appear. Fill in all of the windows as shown in the figure. (We are assuming a moderate amount of consideration will be given to the current state before changing state, thus the selection of 66 and -66). The time delay window is not available. When done, click the OK button.

The screenshot shows a window titled "TimeVaryingCASTInterface" with a light blue background. It contains three columns of controls. The first column is labeled "If the Premise (Parent) is TRUE this will influence the Consequence (Child)" and contains three dropdown menus with values "66 (Moderately More Likely)", "66 (Moderately More Likely)", and "33 (Slight More Likely)". The second column is labeled "If the Premise (Parent) is FALSE this will influence the Consequence (Child)" and contains three dropdown menus with values "-66 (Moderately Less Likely)", "-66 (Moderately Less Likely)", and "-33 (Slightly Less Likely)". The third column is labeled "Time Interval" and has two sub-columns: "Lower Bound" and "Upper Bound". The values in these fields are 1 and 2 for the first row, 2 and 3 for the second row, and 4 and 10 for the third row. Below these columns is an "Arc Delay" field with the value 0. At the bottom are "OK" and "Cancel" buttons.

If the Premise (Parent) is TRUE this will influence the Consequence (Child)	If the Premise (Parent) is FALSE this will influence the Consequence (Child)	Lower Bound	Upper Bound
66 (Moderately More Likely)	-66 (Moderately Less Likely)	1	2
66 (Moderately More Likely)	-66 (Moderately Less Likely)	2	3
33 (Slight More Likely)	-33 (Slightly Less Likely)	4	10

Arc Delay: 0

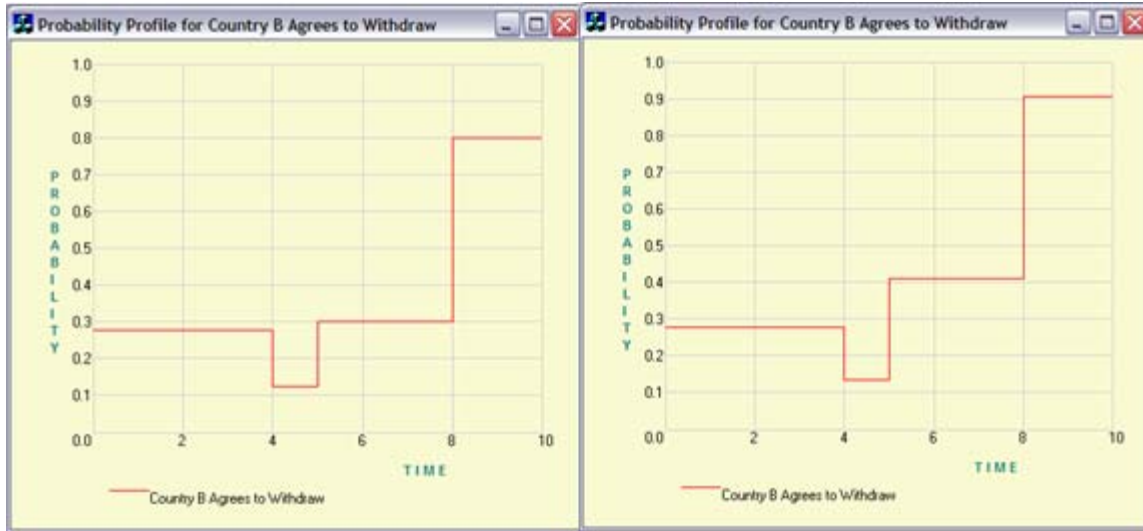
OK Cancel

Figure A.54 Setting DIN Causal Strength Parameters for Self Loop

2. Re-establish the two arcs that were deleted and re-enter their causal strength parameters (See Figures A.51 and A.52).

3. Execute the DIN with the lightning bolt and execute the COA as before, saving the .ppf file. The result is shown in Figure A.55 (a) with the comparison of the DIN without the self loop in Figure A.55 (b). Note that the self loop “tempers” the probability that “Country B Agrees to Withdraw”.

It should be noted that there is no algorithm for converting DINs to DBNs so that the algorithms available for DBNs do not apply to DINs.



(a) Profile for DIN with Self Loop

(b) Profile for DIN without Self Loop

Figure A.55 Effects of Self Loops

A.3. SUMMARY

This manual described the workings of Pythia. Pythia is a new tool that has undergone only limited testing. Any feedback on its use would be appreciated. Be careful when entering information to be sure that you are in the correct mode and that the object you are working on has been selected. In text fields like the name of the node, only use alpha-numeric characters; do not use special keys like #, \$, %, (,), etc. Their use may corrupt the file. If you get an error box, try clicking on continue. Be sure to save your work often, and test it after you save. Inquiries about the future releases and/or Pythia 1.0 can be directed to lwagenha@gmu.edu.