Analyzing Intelligence on WMD Attacks Using Threaded Event-Based Simulation

Qi Fang, Peng Liu, John Yen, Jonathan Morgan, Donald Shemanski, Frank Ritter

Chapter 14

ANALYZING INTELLIGENCE ON WMD ATTACKS USING THREADED EVENT-BASED SIMULATION

Qi Fang, Peng Liu, John Yen, Jonathan Morgan, Donald Shemanski and Frank Ritter

Abstract Data available for intelligence analysis is often incomplete, ambiguous and voluminous. Also, the data may be unorganized, the details overwhelming, and considerable analysis may be required to uncover adversarial activities. This paper describes a simulation-based approach that helps analysts understand data and use it to predict future events and possible scenarios. In particular, the approach enables intelligence analysts to find, display and understand data relationships by connecting the dots of data to create network of information. The approach also generates alternative storylines, allowing analysts to view other possible outcomes. It facilitates the automation of reasoning and the detection of inconsistent data, which provides more reliable information for analysis. A case study using data from the TV series, *24*, demonstrates the feasibility of approach and its application to intelligence analysis of WMD attacks against the critical infrastructure.

Keywords: Intelligence analysis, threaded event-driven simulation

1. Introduction

Terrorists often target critical infrastructures. The U.S. State Department defines terrorism as "premeditated, politically motivated violence perpetrated against noncombatant targets by sub-national groups or clandestine agents, usually intended to influence the audience" [17]. This paper focuses on the analysis of intelligence related to weapons of mass destruction (WMD) attacks against critical infrastructures.

WMD attacks could be chemical, biological, radiological, nuclear or combinations thereof [5]. The general characteristics of terrorists and other clandestine groups who seek to acquire WMD include cause, commitment, cama-

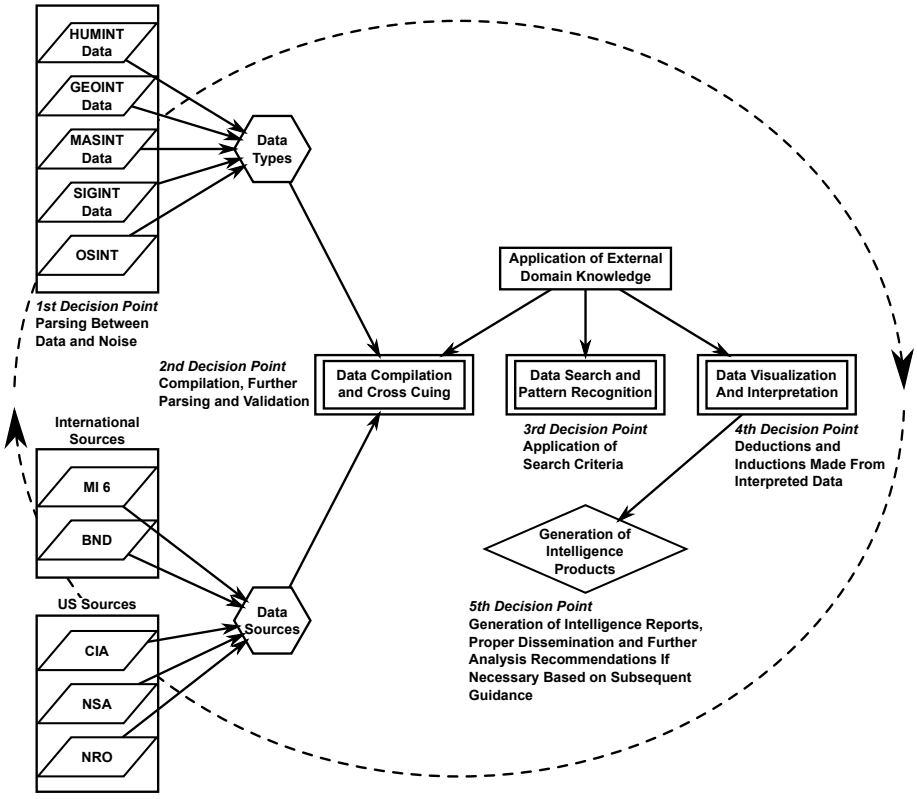

Figure 1. Intelligence analysis workflow.

raderie, charismatic leadership, cash and resources, and cells [19]: Organizational characteristics include command, control and communications, recruitment, weapons procurement, logistics, surveillance, operations and finance. Organizational complexity, characterized by the division of responsibility within the group with respect to the various tasks outlined above, generally contributes to the likelihood of a successful, high-yield WMD event, while also generating a greater amount of traceable data.

In order to combat terrorism in a timely and effective manner, intelligence analysts need to continuously analyze incoming information related to key actors, organizations and events, identify patterns, anomalies, relationships and causal influences, and provide alternative explanations and possible outcomes for decision making. When given an assignment, intelligence analysts search for information, assemble and organize the information in a manner designed to facilitate retrieval and analysis, analyze the information to make an estimative judgment, and write a report [8]. Figure 1 shows the workflow and key decision points in the intelligence gathering and analysis cycle [8].

There are four broad challenges in intelligence analysis: data collection, synthesis, validation and interpretation. Also, simulation tools that facilitate intelligence analysis must operate within the applicable time constraints even when information is abundant or incomplete. To address these challenges, we describe a threaded event-based simulation approach for intelligence analysis. The simulation approach offers intelligence analysts a means for identifying causal relationships and patterns in large data sets, detecting missing data, performing counter-validation and mapping multiple alternative storylines to support emerging analysis priorities.

2. Related Work

Several techniques have been developed to address the challenges of organizing information in order to identify recurring patterns and causal relationships, distinguish relevant information from noise and infer activities of interest from incomplete data. However, these techniques generally place limited, if any, emphasis on counter-validation.

Computer-aided analysis enhances the ability of intelligence analysts to reason about complex problems when the available information is incomplete or ambiguous, as is typically the case in intelligence analysis [8]. Modeling and simulation can provide valuable knowledge, understanding and preparation to combat future attacks [20].

Simulation approaches can be broadly grouped into two categories: agent-based and event-based simulations. Several researchers have used agent-based approaches to model and infer the effects of decisions and actions in social systems [6, 12, 15, 17]. In this paradigm, agent interactions are characterized in different ways: as forms of information diffusion [4]; as mechanisms that leverage social influence [11]; as trades, contracts or negotiations [2]; and as the consequences of some activity or strategy [3]. Agent-based approaches also offer a powerful means for representing agent-level capabilities using constraints such as geospatial effects [13], psychological limitations [18] and socio-cognitive effects [14].

Agent-based approaches vary in their portability (i.e., ability to integrate with simulation environments) and modularity (i.e., ability of the user/analyst to change aspects of the architecture). Agent architectures also differ in their capabilities and in their theoretical entanglements. Powerful complex agents are generally computationally expensive, difficult to integrate into simulation environments and entail significant theoretical commitments, which makes them difficult to modify. Lighter agents, on the other hand, are often modular and easier to integrate into existing simulations, but are relatively brittle. They generally model a small set of behaviors very well, but require significant modification to model other behaviors of interest. Consequently, agent-based approaches, when used exclusively, are unlikely to support – without external assistance – a wide range of evolving analysis priorities or fundamental changes in understanding.

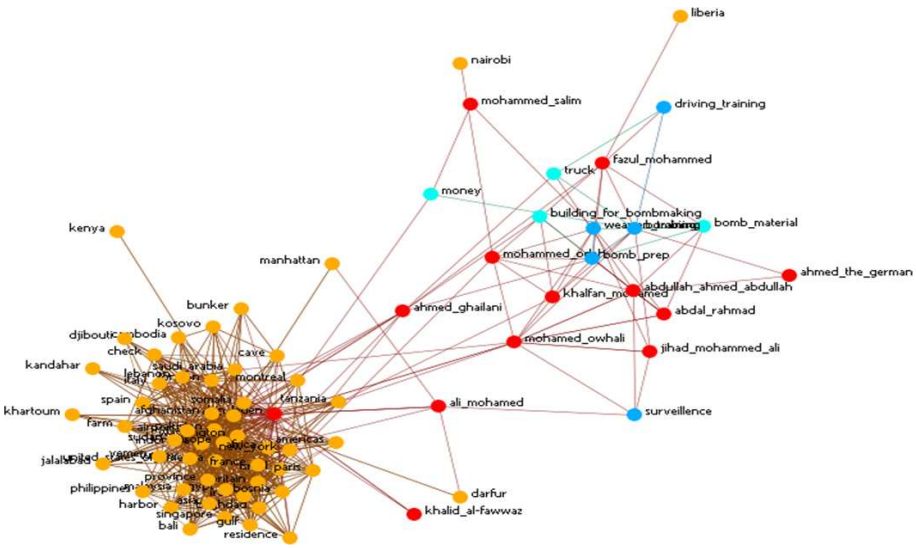

Figure 2. Sparse event network.

Event-based simulations represent behavior by modeling the causal and temporal preconditions, participants, effects, times and locations that characterize an event. Unlike the internal state of an agent, these characteristics are immediately observable and verifiable. Analysts can use hypergraphs or meta-network representations to express causal, temporal, spatial and social relationships as ties between events. Event-based simulations also support deductive and inferential reasoning. By defining the causal and temporal preconditions associated with an event, a set of potential consequences (or storylines) can be deduced. These storylines can be compared with reports gathered from human intelligence sources, providing a form of counter-validation. Alternatively, analysts can identify potential group associations or behaviors by highlighting points of co-occurrence or performing other analyses of the network topology. Note that the notion of events has also been used to model the growth of social networks [16].

Figure 2 shows a sparse event network based on data from [4]. The network has different types of nodes and ties. Red nodes represent agents, orange nodes represent locations, light blue nodes represent resources and dark blue nodes represent tasks/events. The ties between nodes represent several types of relations, such as social relations between agents, spatial relationships between agents and locations, ownership relationships between agents and resources, actor relations between agents and tasks, and distance relations between locations. A sparse network can help visualize nodes with high centrality and cliques, providing early indicators of the nodes of interest.

Event-based simulations are a pragmatic approach to dealing with analysis problems. Many of the basic causal and temporal preconditions associated

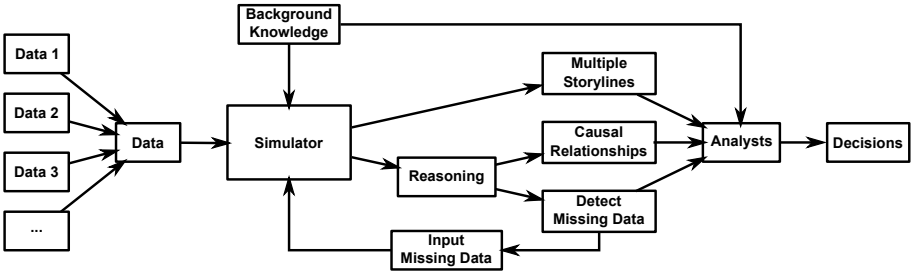

Figure 3. Intelligence analysis workflow with simulator.

with events can be handled using established techniques [1, 10]. The relative theoretical flexibility of event-based simulations also makes them responsive to user demands. Basic deductions require only running the simulation again using a new set of inputs; implementing new causal or temporal rules is facilitated through a GUI. On the other hand, event-based simulations generally possess no mechanisms for bringing to bear external knowledge. Also, the scope of inferences and deductions depends on the completeness of the available data and the expert knowledge encoded into the simulations.

3. Simulation Approach

Figure 3 presents an analysis-driven workflow using our simulator, which supports the second, third and fourth decision points in Figure 1. The simulator provides assisted analysis via information organization, simulation of event possibilities represented as storylines, support of evolving analysis priorities by enabling intelligence analysts to examine the effects of different decision points, discovery of causal relationships, detection of missing data, and, thus, an important aspect of counter-validation.

The simulator decomposes the original data into a set of networks or storylines by defining three types of nodes, events, actors and objects, along with their associated attributes. The simulator can identify and generate multiple divergent storylines as well as alternative storylines from a large dataset. Multiple storylines occur when a decision point generates multiple coexisting possibilities; alternative storylines occur when the decision point generates two mutually exclusive possibilities. Figure 4 shows an example of multiple storylines generated by the simulator along with the decision nodes used in a simulation. Each storyline is indicated by a different color in the figure.

In previous approaches, temporal and causal dependencies between events are specified in advance by the knowledge engineer; if the preconditions are not satisfied, the associated event cannot proceed. In our approach, temporal and causal dependencies between events are discovered; when the preconditions are not satisfied, the gathering of missing information is triggered. The associated event proceeds after the missing information becomes available.

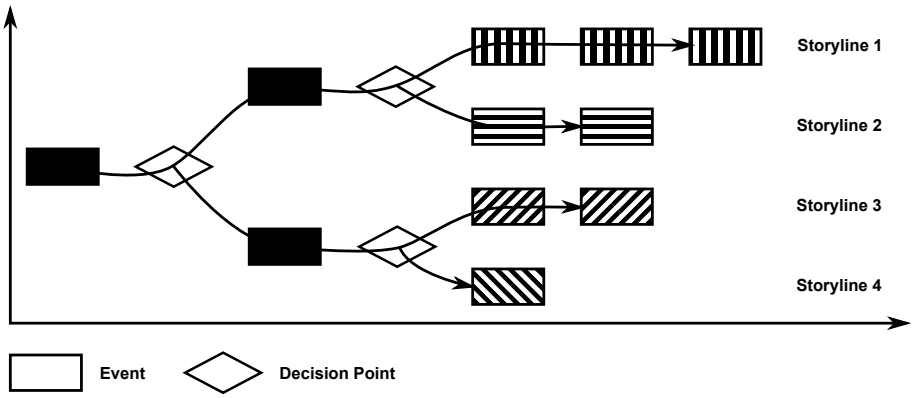

Figure 4. Threaded event-based simulation.

To demonstrate the performance of the simulator, we present a case study using data from the second season of the popular TV show *24*. The entire season was parsed into a set of discrete events along with their attributes, from which the simulator generated multiple storylines with decision nodes and different options. The simulator also discovered various causal relationships and detected the presence of missing information. The test involving missing information was conducted by deliberately deleting some events from the original data and running the simulation.

4. Simulator Design

As mentioned above, the simulator is designed to reduce the workload of intelligence analysts. Specifically, the simulator incorporates mechanisms for connecting unorganized data into an information network, generating multiple storylines to allow analysts to view different outcomes, and automatically detecting causal relationships and missing information.

This section clarifies the principal concepts used in the simulation model. Also, it describes the software architecture and the algorithms used by the simulator.

4.1 Key Concepts

An event is something of interest that has occurred (e.g., CTU agents have found a bomb). Actors are participants in events. Objects are target infrastructures or tools. Events and their relationships generate state changes. The "world" is defined by events, actors, objects and their relationships.

Events, actors and objects have various attributes that capture their properties (Tables 1, 2 and 3, respectively). Preconditions and effects are two important attributes of events. Preconditions describe the state of the world before

Table 1. Event attributes.

Name	**Implication**	**Example**
Event ID	Identifier of event	1
Content	Description of event	"Make death allusion"
Start Time	Time of event	–20:00
Location	Place where event occurs	WestBank@USA
Actors	Participants involved in event	Mamud Faheen
Relationships	Relationships shown in event	
Effects	Effects caused by event	Mamud Faheen.status = 2
Preconditions	Requirements for event to occur	Mamud Faheen.type = 1

Table 2. Actor attributes.

Name	**Implication**	**Example**
Actor ID	Identifier of actor	1
Name	Name of actor	Mamud
Sex	1: Male; 2: Female; 0: Unknown	1
Age	Positive integer	49
Type	1: Terrorist; 2: Anti-terrorist agent; 3: Neutral	1
Status	0: Dead; 1: Alive; 2: Arrested; 3: Under surveillance	2
Level	Status level of actor in a task	10
Affiliation	Organization to which actor belongs	Second Wave
Location	Location of actor	Los Angeles

Table 3. Object attributes.

Name	**Implication**	**Example**
Object ID	Identifier of object	1
Object Name	Name of object	Nuclear bomb
Object Status	Status of object	Ready
Object Location	Location of object	Los Angeles

a change that is caused by an event. Effects describe the state of the world (actors and objects) after a change. Events can trigger new events.

The simulator treats preconditions as qualified state(s) of the actors, objects and relationships for which the owner event occurs. A state that does not satisfy the preconditions of an event precludes the event from occurring in the state. Information gathering – currently in the form of a query to the user – is triggered when preconditions are not satisfied.

Causal relationships are present when the effects of an event cause the preconditions of another event to be satisfied. Hidden relationships are discovered by searching the dataset for causal relationships.

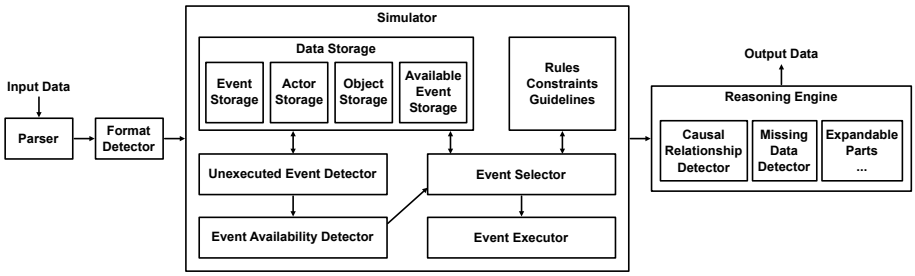

Figure 5. Software architecture of the simulator.

4.2 Software Architecture

Figure 5 shows the software architecture of the simulator, including the components that support the detection of causal relationships and the identification of missing data. The figure also illustrates the selection of events by the simulator. The selection is guided by rules, constraints and guidelines that can be obtained from and/or altered by experienced analysts.

The parser extracts information about events and their attributes. The format detector verifies that the parsed data can be executed by the simulator.

The simulator has six main components: (i) data storage, which saves all the data during a simulation process and includes the available event storage, which holds the events whose preconditions are satisfied; (ii) rules/constraints/guidelines component, which saves the logic rules used by the simulator; (iii) unexecuted event detector, which selects unexecuted events; (iv) event availability detector, which selects events whose preconditions are satisfied from the output of the unexecuted event detector; (v) event selector, which selects the event with the earliest start time from the output of the event availability detector; and (vi) event executor, which executes the selected event, changes the attributes accordingly and marks the event as "executed."

After completing a simulation, the simulator gives a chronological sequence of discrete events for each storyline according to the execution order of events.

The reasoning engine is designed to discover causal relationships between events and to detect missing information based on the output of the simulator. For each event, the reasoning engine matches its preconditions with the effects of all preceding events to check if any relationship exists between the events. The reasoning engine detects missing information when one or more preconditions do not match.

4.3 Algorithms

Figure 6 presents the simulator workflow. The workflow involves the following steps:

- **Step 1:** Parse input data into the appropriate XML format.

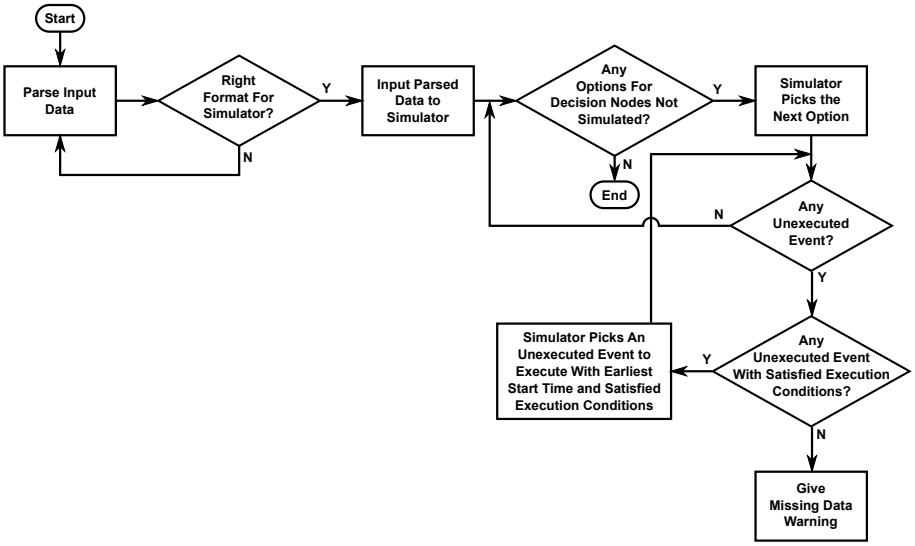

Figure 6. Simulator workflow.

- **Step 2:** Search for unexecuted options; terminate the process if none exist.
- **Step 3:** Select the option with the earliest start time and conditions that are satisfied.
- **Step 4:** Search for subsequent events; query for additional information if necessary.
- **Step 5:** Generate a network of events based on the causal and temporal dependencies.

Algorithms are implemented for information organization, simulation, causal relationship discovery, missing data detection and multiple storyline generation.

- **Information Organization:** The input data for the simulator is a set of events with their associated attributes in XML format. Each storyline has three input files corresponding to events, actors, objects and their associated attributes. Figure 7 presents the input data format.

  The parser processes all the input files, extracts the attributes in each record and saves the objects in event, actor and object storage.

- **Simulation:** After parsing the data, the simulator goes through the event storage and checks if the preconditions of each event are satisfied. If the event is unexecuted and its preconditions are satisfied, the simulator marks the event as "ready." From the ready events, the simulator picks the event with the earliest start time to execute. After the event

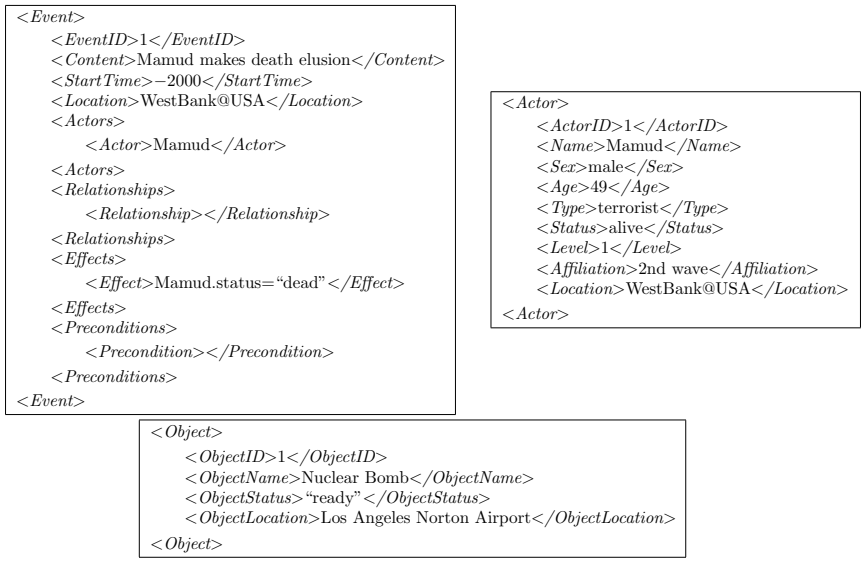

Figure 7. XML format of input data.

is executed, the attributes of the preconditions of other events may be changed; these attributes are listed as effects. The simulator updates attributes in data storage. After an event is executed, the simulator marks the event as "executed."

The simulator repeats the steps until all the events are executed (i.e., there is no missing data about the causal relationships). If missing data exists, the simulator performs a series of steps described below.

After the simulation is complete, the simulator generates a chronological sequence of events according to their execution sequence, each sequence representing a storyline. Algorithm 1 in Figure 8 lists the steps involved in a simulation.

- **Causal Relationship Discovery:** If the effects of one event affect the preconditions of a second event, then a causal relationship exists between the first and second event. The procedure for discovering causal relationships is specified in Algorithm 2 in Figure 8. The preconditions of an event are matched with the effects of all preceding events; causal relationships exist when the preconditions match. Because an event could have several preconditions, several events could have causal relationships with a given event.

- **Missing Data Detection:** Missing data is detected based on the causal relationships. The effects of some events affect the preconditions of other events and, thus, trigger these events. If certain events are missing, their effects will not trigger other events.

Algorithm 1: Simulation

Require: $D = \{E, A, O, EE\}$
Require: E, event list; EE, executed event list
Require: A, actor list; O, object list
1: **while** E.events $\neq 0$ **do**
2: **for** event i in E **do**
3: **if** all preconditions of i == true **then**
4: set i.status = "ready"
5: **end if**
6: **end for**
7: set min_start_time = start_time of event 1 in "ready"
8: **for** event j in "ready" events **do**
9: **if** j.start_time < min_start_time **then**
10: min_start_time = j.start_time
11: picked_event = j
12: **end if**
13: **end for**
14: execute event j, update A, update O
15: EE.add(event j)
16: E.remove(event j)
17: **end while**
18: output events from EE

Algorithm 2: Discovery of Causal Relationships and Missing Data

Require: $D = \{E, EE\}$
Require: E, event list
Require: EE, executed event list
1: **while** E.events $\neq$ NULL **do**
2: **for** event i in E **do**
3: **for** precondition j in i.preconditions **do**
4: **if** j == effect of event m in EE **then**
5: output causal relationship $m \rightarrow i$
6: **end if**
7: **end for**
8: **end for**
9: **if** no "ready" event in E **then**
10: output "missing information"
11: break;
12: **end if**
13: **end while**

Figure 8. Key algorithms.

As shown in Algorithm 2, the simulator checks if events are not executed after a simulation. Missing data is detected when events are not executed. The result is that the storyline generated by the simulator is incomplete. The simulator then alerts the analysts that it requires the data to keep working. Thus, the simulator helps analysts discover useful missing information.

- **Multiple Storyline Generation:** The simulator permits analysts to set decision nodes and choose different options at each step to examine the possible outcomes. When the simulator encounters a decision node, it simulates one of the options associated with the decision node. After the simulator generates a complete storyline, it goes back to the decision node and simulates other options until all possible options are simulated.

  The simulator requires analysts to maintain different storylines in different files. Each file contains the complete sequence of events of a storyline.

5. Case Study

This section describes the results of a case study using data from the popular TV series *24* (Season 2). The plot involves a team of Counter Terrorist Unit (CTU) agents working together to foil a terrorist plot to detonate a nuclear bomb in Los Angeles.

5.1 WMD Attack Data

The dataset was extracted from Episodes 1 through 14 of *24* (Season 2) that spanned a 14-hour time period. The actors include terrorists, agents and civilians. Three story threads are involved. The first thread is about how agents

Table 4. Portion of the storyline.

ID	Content	Time	Location	Actors
1	"Mamud makes death allusion"	–20:00	WestBank	Mamud
2	"Get a nuclear bomb"	–10:00	Norton Airport	Mamud
3	"Mamud introduces Nina to Joe"	09:01	Los Angeles	Mamud, Nina, Joe
4	"Joe gets plans of CTU"	09:00	Los Angeles	Joe, Nina
5	"Marie deals with Ali"	08:50	Warner	Marie, Ali
...	...	...	...	...
57	"Agent found the bomb"	13:09	Norton Airport	Jack

led by Jack Bauer locate the nuclear bomb and prevent the attack. The second thread deals with politics and differences in opinion regarding government policies. The third thread is about the experiences of several civilians during the same time period.

Our case study only included scenarios from the first thread. A total of 57 discrete non-overlapping events, 31 actors and five objects were extracted. Attributes were assigned to events, actors and objects. Also, we tracked possible events that could occur and lead to different outcomes based on conversations between actors and from other possible actions of actors. We set decision nodes between different options and generated different sets of data, each set containing all the information for a storyline.

The set of events was input to the simulator to produce a complete storyline. We checked if the simulator could detect causal relationships between events. We also input an incomplete set of events to test if the simulator could detect inconsistent data. Finally, we set a decision node and provided a different event set to obtain a different storyline.

5.2 Simulation Results

The simulator generated a chronological sequence of events, providing a complete storyline about how agents collaborate to detect and prevent the WMD attack. Because there was no missing data in this experiment, the simulator output all 57 events in sequential order. Table 4 lists a portion of the simulation output. Figure 9 presents the sequence of events and decision points, with boxes representing events and diamonds representing decision points.

5.3 Causal Relationships

The simulator detected several causal relationships between events, some of which are listed in Table 5. For example, Event 3: "Mamud introduces Nina to Jo," triggers the occurrence of Event 4: "Joe gets CTU plans in Los Angeles from Nina." Because the effect of Event 3 is that Nina and Joe know each other and this is the precondition of Event 4, a causal relationship exists between Events 3 and 4.

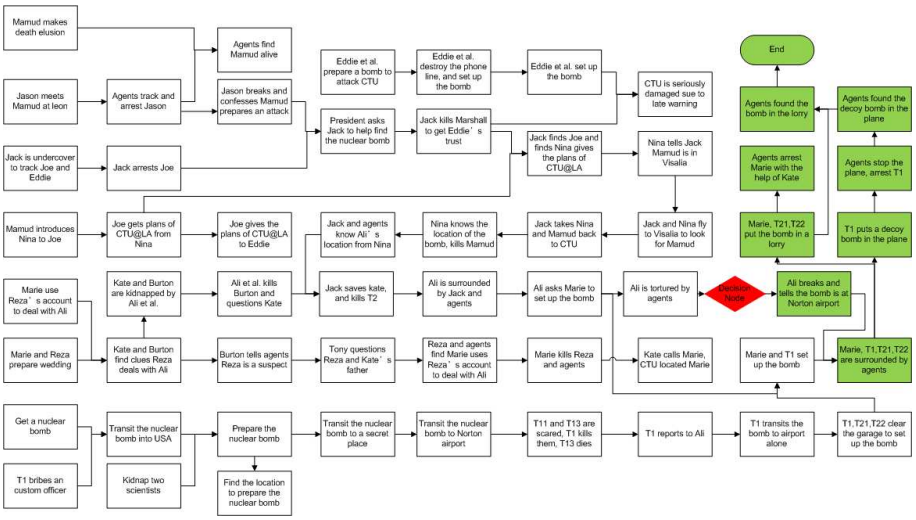

Figure 9. Network of events in *24*.

Table 5. Causal relationships between events.

Event	**Preceding Event(s)**
Event 4: "Joe gets CTU plans"	Event 3: "Mamud introduces Nina to Joe"
Event 7: "Joe gives CTU plans to Eddie"	Event 4: "Joe gets CTU plans"
Event 8: "Transport nuclear bomb to USA"	Event 2: "Get nuclear bomb"
Event 11: "Ready nuclear bomb"	Event 8: "Transport nuclear bomb to USA"

5.4 Missing Data

In the case study, when Event 8: "Transport nuclear bomb to USA" was deleted, as expected, Event 9: "Ready nuclear bomb" and all related events were affected. The simulator provided a warning that some information was missing and requested the user to provide more input, in this case, location information for the nuclear bomb. In addition to identifying instances of incomplete information in the dataset, this mechanism can also identify holes in an account about a set of events, allowing intelligence analysts to follow-up on the key details.

5.5 Multiple Storylines

The *24* TV show only presents one storyline. We generated different storylines based on other possible decisions in the TV show. We set several decision nodes, generated different options and ran the simulator again. Figure 9 shows an alternative storyline where Ali refuses to confess. Once again, events are represented by boxes and decision points by diamonds.

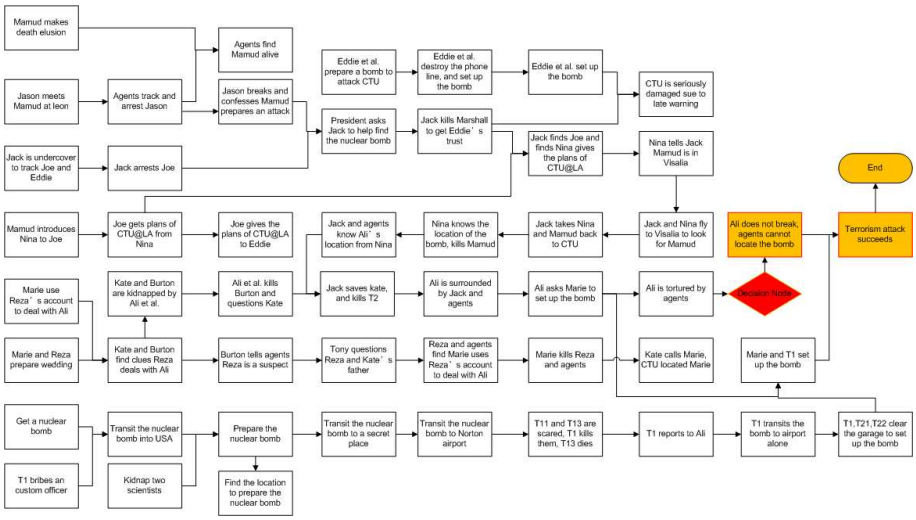

Figure 10. Network events in an alternative storyline.

6. Conclusions

A threaded event-based simulator can significantly enhance intelligence analysis. The simulator presented in this paper offers several functions that enhance the productivity and work quality of analysts. It automatically detects causal relationships between events and missing information, both of which are very important in intelligence analysis. Also, the simulator generates multiple storylines, which enable intelligence analysts to view all possible outcomes given different options for events. A case study involving the TV series *24* demonstrates the utility of threaded event-based simulation in analyzing intelligence related to WMD attacks against critical infrastructures.

Our future work will focus on layered iterative intelligence analysis, counter-validation and hypothesis testing, multi-dimensional relational inference, alternative storylines and hypothetical reasoning.

Acknowledgements

This work was supported by the Defense Threat Reduction Agency under HDTRA 1-09-1-0054. The authors also wish to thank Baojun Qiu for helping gather data for the case study.

References

[1] A. Arnold, Y. Liu and N. Abe, Temporal causal modeling with graphical Granger methods, *Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 66–75, 2007.

[2] R. Axelrod and R. Hammond, The evolution of ethnocentric behavior, presented at the *Midwest Political Science Convention*, 2003.

[3] M. Cohen, F. Ritter and S. Haynes, Using reflective learning to master opponent strategy in competitive environments, *Proceedings of the Eighth International Conference on Cognitive Modeling*, pp. 157–162, 2007.

[4] Computational Analysis of Social and Organizational Systems, Dataset – Tanzania Embassy Bombing – 2006, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2006.

[5] C. Deo, M. Kosal and S. Dhongde, Multidisciplinary modeling of socio-economic influence on adversarial intent to acquire, proliferate and use chemical, biological, radiological and nuclear weapons, presented at the *DTRA Counter-WMD Basic Research Technical Review*, 2009.

[6] X. Fan, S. Sun and J. Yen, On shared situation awareness for supporting human decision-making teams, *Proceedings of the AAAI Spring Symposium on AI Technologies for Homeland Security*, pp. 17–24, 2005.

[7] S. Galam and S. Wonczak, Dictatorship from majority rule voting, *European Physical Journal B – Condensed Matter and Complex Systems*, vol. 18(1), pp. 183–186, 2000.

[8] R. George and J. Bruce, *Analyzing Intelligence: Origins, Obstacles and Innovations*, Georgetown University Press, Washington, DC, 2008.

[9] S. Haynes, M. Cohen and F. Ritter, Designs for explaining intelligent agents, *International Journal of Human-Computer Studies*, vol. 67(1), pp. 90–110, 2009.

[10] K. Karimi and H. Hamilton, Finding temporal relations: Causal Bayesian networks vs. C4.5, *Proceedings of the Twelfth International Symposium on the Foundations of Intelligent Systems*, pp. 266–273, 2000.

[11] U. Krause, A discrete nonlinear and non-autonomous model of consensus formation, in *Communications in Difference Equations*, S. Elaydi, G. Ladas, J. Popenda and J. Rakowski (Eds.), Gordon and Breach, Amsterdam, The Netherlands, pp. 227–236, 2000.

[12] B. Latane and A. Nowak, Self-organizing social systems: Necessary and sufficient conditions for the emergence of clustering, consolidation and continuing diversity, in *Progress in Communication Sciences: Advances in Persuasion*, G. Barnett and F. Boster (Eds.), Ablex Publishers, Greenwich, Connecticut, pp. 43–74, 1997.

[13] I. Moon and K. Carley, Modeling and simulating terrorist networks in social and geospatial dimensions, *IEEE Intelligent Systems*, vol. 22(5), pp. 40–49, 2007.

[14] J. Morgan, G. Morgan and F. Ritter, A preliminary model of participation for small groups, *Computational and Mathematical Organization Theory*, vol. 16(3), pp. 246–270, 2010.

[15] C. Morrison, P. Cohen, G. King, J. Moody and A. Hannon, Simulating terrorist threat in the Hats Simulator, *Proceedings of the First International Conference on Intelligence Analysis*, 2005.

[16] B. Qiu, K. Ivanova, J. Yen and P. Liu, Behavior evolution and event-driven growth dynamics in social networks, *Proceedings of the Second IEEE International Conference on Social Computing*, pp. 217–224, 2010.

[17] S. Raczynski, Simulation of the dynamic interactions between terror and anti-terror organizational structures, *Journal of Artificial Societies and Social Simulation*, vol. 7(2), 2004.

[18] F. Ritter, S. Kase, L. Klein, J. Bennett and M. Schoelles, Fitting a model to behavior tells us what changes cognitively when under stress and with caffeine, *Proceedings of the Biologically Inspired Cognitive Architectures Symposium*, pp. 109–115, 2009.

[19] D. Shemanski, Characteristics and Attributes of Real-World Terrorist Organizations: Implications for Effective Network Analysis, Technical Report, College of Information Sciences and Technology, Pennsylvania State University, University Park, Pennsylvania, 2009.

[20] R. Smith, Counter terrorism simulation: A new breed of federation, *Proceedings of the Spring Simulation Interoperability Workshop*, 2002.

[21] M. Taylor and J. Horgan, A conceptual framework for addressing psychological process in the development of the terrorist, *Terrorism and Political Violence*, vol. 18(4), pp. 585–601, 2006.