Multi-Entity Bayesian Networks for Situation Assessment

E. Wright, S. Mahoney, K. Laskey, M. Takikawa, T. Levitt

Information Extraction & Transport, Inc.
1911 N. Ft Myer Drive, Suite 600, Arlington, VA 22209
Corresponding Author: ewright@iet.com

Abstract - *Reasoning about military situations requires a scientifically sound and computationally robust uncertainty calculus, a supporting inference engine that procedurally encodes the axioms of the calculus, the capability to fuse information at multiple levels of abstraction, and the ability to respond to dynamic situations. The inference engine also needs to be able to encapsulate expert knowledge, including deep human doctrinal and domain knowledge. At Information Extraction & Transport, Inc. (IET), we have developed techniques to encode domain and doctrinal expertise in reusable knowledge chunks, based on the technology of Bayesian Network Fragments, and the capability to automatically construct situation specific Bayesian Networks based on a combination of top down control and bottom up evidence-driven processes. These techniques have been used to prototype fusion systems capable of reasoning about uncertain numbers of uncertain hierarchically organized entities based on incomplete observations. These systems have demonstrated success in generating force level situation hypotheses from vehicle tracks and other evidence generated by level 1 fusion systems. This paper presents an overview of our technical approach with applications from recent projects.*

Keywords: Bayesian Networks, multi-entity Bayesian Networks, situation assessment, hypothesis management

1 Introduction

Military situation assessment requires reasoning about an unknown number of hierarchically organized entities interacting with each other in varied ways. The entities are observed by various sensors, which generate sensor reports or feed level 1 fusion systems that generate estimates of various attributes of the entities. In general, these reports cannot be unambiguously associated with the domain entities generating them. For example, in ground combat scenario, level 1 fusion systems generate a stream of tracks, or tracklets, with likelihoods for vehicle types and activities. This evidence stream is subject to errors – including missed detections, false alarms, misassociation, misclassification. This paper describes IET's approach for processing this kind of errorful evidence stream to generate a situation assessment.

1.1 Organization

Section 2 provides an overview of the application of Bayesian Networks (BNs) and Bayesian Network Fragments (BNFrags) to information fusion, and introduces our BN inference engine. Section 3 introduces Multi-Entity Bayesian Networks (MEBNs), a specialization of BNFrags. Section 4 introduces hierarchical models for classification, and section 5 presents the technology of situation specific network construction, hypothesis management, and evaluation. Section 6 provides a summary of an example from a recent research program.

2 Bayesian Networks for Information Fusion

A *BN* represents the probabilistic dependencies among a set of random variables by a directed acyclic graph [1]. Each node in the network represents a random variable with a set of defined states that together typically represent a set of mutually exclusive and exhaustive possible values for some hypothesis. A node in a BN is conditionally independent of all non-descendant nodes given its direct parents. Each node in the network stores its probability distribution given its direct parents and any evidence that has been observed concerning the node. This information is sufficient to implicitly represent the full joint probability distribution over all random variables in the network, as well as the conditional joint distribution given observed evidence about some nodes in the network.

The knowledge required to construct the BN can be learned from existing data, can be defined by known functional relationships, by elicitation from human experts, or by any combination of the above.

2.1 Bayesian Network Fragments

The vast majority of published applications of BNs consist of *template models*. A template model is appropriate for problem domains in which the relevant variables, their state spaces, and their probabilistic relationships do not vary from problem instance to problem instance. Thus, generic knowledge about the domain can be represented by a fixed BN over a fixed set of variables, obtained by some combination of expert judgment and learning from observation. Problem solving for a particular case is performed by conditioning the network on case-specific evidence and computing the posterior distributions of variables of interest.

For example, a medical diagnosis template network would contain variables representing background information about a patient, possible medical conditions the patient might be experiencing, and clinical findings that might be observed. The network encodes probabilistic relationships among these variables. To perform diagnosis on a particular patient, background information and findings for the patient are entered as evidence and the posterior probabilities of the possible medical conditions are reported. Although values of the evidence variables vary from patient to patient, the relevant variables and their probabilistic relationships are assumed to be the same for all patients. It is this assumption that justifies the use of template models.

The development of efficient belief propagation algorithms for template models enabled an explosion of research and applications of probability models in intelligent systems[1, 2]. As BN technology is applied to more complex problems, the limitations of template models become apparent. Even when a domain can be represented by a template model, its size and complexity may make it necessary to represent it implicitly as a collection of modular subunits [3].

It is clear that a template model is inadequate for any type of military situation assessment. The number of actors of any given type is not static, but varies from situation to situation. A reasoning system must be capable of unifying reports with already-hypothesized units and/or hypothesizing new units, as the context for the current problem demands. The relevant variables for reasoning about an actor depend on the actor's type. For example, the mode in which a radar emits is a key variable for inferring the activity of a surface-to-air missile battery. However, this variable is not applicable to units that have no radar. Clearly, a network with a fixed set of variables and a fixed topology is inadequate for this problem.

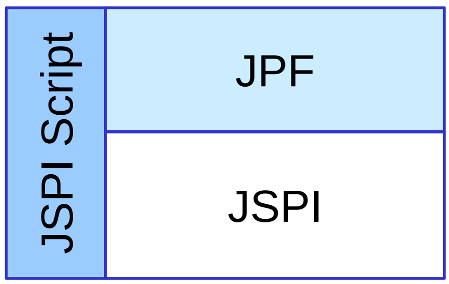
To address this issue, IET has pioneered research that supports the decomposition of complex models into conceptually meaningful and manageable pieces called BNFrags [4]. Opportunities for decomposition and reuse occur when families of variables share the same sets of possible values, when sets of related variables have a common structure and when possible variables, values and conditioning relationships can be parameterized. Structural and parametric regularities also occur within the conditioning distributions of BN. Network fragments represent shared elements of a probabilistic knowledge base. Each network fragment contains probabilistic knowledge about a small set of random variables. Moreover, a network fragment may represent only a portion of a conditional probability distribution for a variable given its parents. Model specification, model maintenance, and communication are facilitated if models are specified as network fragments.

Patterns of entity structure, behavior and relationships can be encoded as fragments of BNs [4]. A knowledge base consisting of a set of BNFrags that encodes human expertise and domain knowledge can provide the building blocks for assembling a situation specific BN.

The ability to select and combine the right BNfrags to build the appropriate BN for the specific inference requires a powerful BN inferencing engine.

2.2 The IET Inference Engine

Over the last decade IET has continuously built and improved the inferencing technology we use to model complex, real world problems. Figure 1 shows the current IET inferencing components.



- JSPI Script: Scripting language for JPF / JSPI
- JAVA Probabilistic Frames (JPF): Capability to manipulate / reuse network fragments
- JAVA Symbolic Probabalistic Inferrencing (JSPI): Propagate Uncertainty In Bayesian Networks

Figure 1. IET Inference Engine Components

Java Symbolic Probabilistic Inferencing (JSPI) Engine is based on the IET's SPI algorithm[5]. It is one of only two known general solution algorithms for BNs. In contrast to the alternate "join tree" approach to inference in BNs, SPI has the following two important characteristics. First, SPI is query based. SPI extracts the minimum subset of a BN that is necessary for each query, minimizing the amount of computation required for answering the query. This is important because the same query can be repeated many times for different points within the area of interest. Second, SPI has local expressions, an extension of BNs, used to express local structure within a node. Local expressions can be used to instantiate many independence relationships including independence of causal influences and context-specific independence. SPI exploits these independence relationships in addition to the conditional independences inherent in BNs for efficient inference in large BNs. SPI has successfully computed queries for large "bench mark" BNs, which the join-tree inference algorithm is unable to compute in reasonable time. In addition, SPI's query-oriented approach allows for compilation of any probabilistic query into an efficient and small procedural code. Because both the memory and CPU requirement of this generated code is fixed; it is readily usable in an embedded and/or real-time environment.

IET's **JAVA Probabilistic Frames (JPF)** is a knowledge representation language based on frames (a widely used knowledge representation in Artificial Intelligence) augmented in various ways to express uncertainties. In addition to frame (class) abstractions organized by "is-a" hierarchies inherited from the frame system, JPF supports mechanisms to express uncertainties about the value of variables, the reference to instances, the existence of instances, and the type of instances. JPF allows for expressing domain knowledge as pieces of BNs (or network fragments) in a modular and compact way, facilitating reuse. Instances of probabilistic frames are created dynamically for each instance, allowing situation specific probabilistic inference. The probabilistic inference is done by JSPI using a BN created dynamically from the current set of probabilistic frame instances. This generation of BNs from JPF utilizes JSPI's local expressions to exploit all types of independence relationships to speed up the inference.

JSPI Script is an object-oriented scripting language designed specifically for BN applications. It provides full access to all the functions of JSPI and JPF. It can be used to dynamically construct BNs, make situation-specific queries, and define and replace software components on the fly. In addition, the JSPI Script language can be run interactively from a command line, or can be used via an API from within a larger software system - allowing automated control over construction and manipulation of BNs.

3 Multi-Entity Bayesian Networks

An MEBN is a collection of BNFrags that satisfy consistency criteria such that the collection specifies a probability distribution over attributes of and relationships among a collection of interrelated entities. An MEBN implicitly encodes a probability distribution over an unbounded number of hypotheses. For any given problem, only a finite subset of these hypotheses will be relevant. A formal theory for MEBNs is under development [6].

MEBN logic extends standard BNs to allow the kind of replication and combination needed to reason about complex problems in which variable numbers of entities interact in varying ways. Implicitly encodes a joint probability distribution over object-level domain entities. Augmented with an ontology for higher-order probability, MEBN also implicitly specifies higher-order distributions and supports learning. MEBN can serve as both object language and meta-language. When extended to a Multi-Entity Decision Graphs (MEDG), the approach also can support trading off computation against accuracy and/or utility for decision making and resource allocation.

MEBNs are to regular BNs what algebra is to arithmetic. If all we have is arithmetic, to figure out how much carpet to buy for an arbitrary room requires a huge table that lists lengths, widths and the area corresponding to every possible length and width (or an instruction to fill in the blank by multiplication). With algebra we can write a single equation a = l x w, and represent the table in one line.

Similarly, if all we have is BNs, and there are M months of data with N variables per month, we must build a BN with MxN nodes, and fill in identical arcs and local probability distributions at each time step. With MEBNs, we can write a single BNFrag relating the variables at time t with the variables at time t+1 and say "repeat for all t's." Similarly, we can relate a vehicle's type to the type of the unit it is a member of and say "repeat for all vehicles in a unit, and then repeat for all units." Standard implementations of BNs do not provide this capability.

4 Hierarchical Models

BNFrags and MEBNs provide the structure for building the knowledge base. But the knowledge base must capture knowledge about the problem domain. In a military ground combat domain the knowledge is typically organized hierarchically. To build the knowledge base we define BNFrags that correspond to knowledge chunks at different levels of the hierarchy. Then evidence, when it is available, can be applied to the appropriate level of abstraction. Links between BNfrags at different levels of the hierarchy allow evidence at one level to support inference at another level of abstraction.

Knowledge can be refined along two dimensions:

Type – We will usually start with the most specific target type for which the initial detection provides adequate evidence (e.g., tracked vehicle) and refine further as additional information is gathered. This refinement is performed based on both the information gathered so far and the commander's interests. For example, a commander may not care about wheeled vehicles, but be particularly concerned about cross-country attacks from tracked vehicles, and want very refined classification on any tracked vehicle.

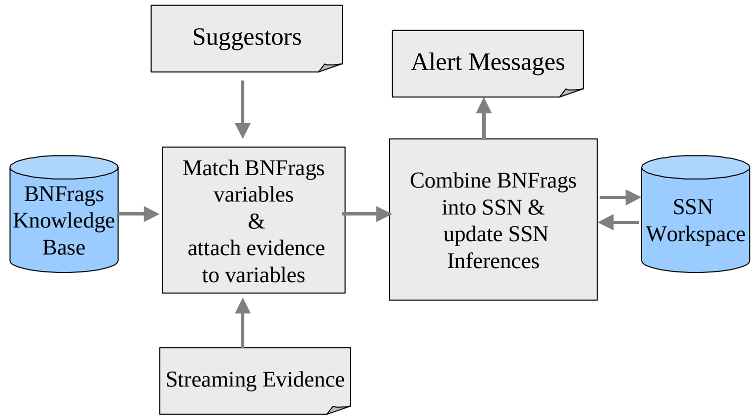


Figure 2 High Level Architecture for SSN Construction from MEBN Knowledge Base

Activity Aspect – Activities can provide powerful information about type, even in the absence of direct observations of type. For example logistics vehicles often operate alone on a closed loop between a supply depot and a supported unit. Combat vehicles – and most types of potential high value targets - do not exhibit this behavior. The activity aspect also applies to the related activities of individual entities. As an example, a HUMINT report may provide information that the fuel truck for a mobile missile launcher has just departed a supply depot. If assessed as credible, this report could cue an activity based query to an MTI database for possible combinations of tracklets leaving the depot at the correct time. The result of this query may lead to the fuel truck entity. Tasking sensors to follow the fuel truck may lead to direct observation of the mobile missile.

Activities also occur in hierarchies. The activities of the individual vehicles in a platoon are related to the platoon's activities, which are in turn related to the company's activities. With a hierarchical activity model, it is possible to infer the activities of higher level units from observations on their members.

The type and activity hierarchies provide a powerful and flexible way to represent doctrinal knowledge – about organization of forces, and about organized activities that military forces engage in.

This kind of a hierarchical knowledge representation, with the capability to add additional BNFrags at the appropriate level of the hierarchy allow us to apply a wide range of evidence from diverse sources at multiple levels of abstraction to our information fusion problem.

5 Situation Specific Bayesian Networks

MEBNs, a powerful inference engine, and a hierarchical model of the problem domain are prerequisites for building a system that will automatically construct situation specific BNs.

To reason about specified target hypotheses given evidence about a particular situation, an ordinary finite BN, called a *situation-specific network* (SSN), is constructed from a MEBN knowledge base. The SSN construction process is initiated when clusters of observations or reports trigger firing of a *suggestor*. Suggestors are modules that use features of the situation to determine which hypotheses need to be represented. The suggestor triggers retrieval of relevant BNFrags. Actual entities from the situation replace the variables in the BNFrags.

After retrieval, the BNFrags are combined, possibly with an already existing SSN, to create a current SSN. Next, evidence is applied to the SSN and inferences are drawn about the target hypotheses. Finally, decision nodes are evaluated to determine what action needs to be taken. An architecture for SSN construction is shown in Figure 2.

5.1 Automated Construction

Hypothesis management is the name of the capability we embody in a software module that manages the composition of the constructed system model. It includes suggestors, described above, as well as rules for pruning hypotheses. Hypotheses management is an important part of the domain knowledge base. Mission parameters establish the decisions to be made and the type of available evidence (e.g. sensor data). The specified decisions determine the relevance of elements of available evidence. Both the decisions and available evidence set the scope for the entities of interest and relationships to be included in the system model.

Individual suggestors examine relevant evidence and the current state of the system model to suggest that new hypotheses be instantiated. For reasons of efficiency, individual suggestors are tailored to types of evidence and the entities of interest about which inferences are to be made. Consequently, the task of hypothesis management meta-level reasoning is to select the suggestors that meet the requirements established by the mission parameters.

Action level hypothesis management makes two types of decisions. **Construction suggestors** make decisions about what hypotheses to add to the constructed model. **Revision suggestors** make decisions about what to change within the model.

The JPF knowledge structures a suggestor may reason about include the following [7]:

Entities: whether an entity of interest exists. This is called existence uncertainty. False alarms are non-existent entities of interest.

Relationships among entities**:** which two entities out of several possible pairing share a specified relationship. This is called reference uncertainty. Associating evidence with an object establishes a relationship. Reference uncertainty applies when there is uncertainty about which object actually caused the evidence.

Entity types: When an entity of interest is identified, it may be of one or more possible subtypes. This is called subtype uncertainty.

Variable resolution**:** The best partition of a variable's possible values depends upon the requirements of the situation and the granularity of the available data.

Dependency relation**:** We may want to modify/adapt/replace one conditional relationship with another as the context varies or we learn.

When reasoning about an entity of interest, hypothesis management suggestors make decisions about instantiating instances of the entity, removing instances of the entity and revising entity instances. A suggestor may decide to instantiate the entity as a hypothetical (uncertain) instance. While the suggestor has some information associated with an entity of interest, there is a 'reasonable' chance that this entity may be a false alarm. If there is enough information, the suggestor may recommend that the entity be instantiated as a certain one. Alternatively, the suggestor may choose to not instantiate the entity at this time because of insufficient information. Other suggestors may decide to remove the entity from the situation-specific network, decide to make a hypothetical entity into a certain one, or decide to take no action. Similar types of suggestors make reference uncertainty decision. Subtyping suggestors follow the isa relationship among entities. Variable resolution reasoning requires suggestors guided by the mission parameters to determine the granularity of the states of the variables. Granularity roughly corresponds to discrimination power. Whenever certain variables have known values, they effectively minimize the size of the conditional probability distributions. These are called context variables. However, in the course of a mission, a context variable may change or be found to be in error. Hypothesis management suggestors need to respond to that change and modify the conditional probability distributions of the model accordingly.

An example software component that constructs SSNs is the Tactical Site, Group, Unit, and activity Detection and Assessment (TSGUDA) as it was used in the DARPA Dynamic Data Base (DDB) program [8]. This data consisted of fused tracks produced by the All-source Track and ID Fusion (ATIF) component, as well as MTI flow data and SIGINT emissions density data. The first functional component of TSGUDA is the “suggestors”. They identify possible hypotheses which are passed to a software module called the assessment engine, which builds and maintains the SSN. The suggestors use the available knowledge models, and the current hypotheses maintained in the assessment engine. The role of the suggestors is to detect, with a high probability of detection (and corresponding high false alarm rate) many possible candidate hypotheses from the data. Note that the suggestors are part of the knowledge representation for the problem domain. The suggestors determine how the data is potentially relevant to the current SSN, and suggest the addition of new BNFrags to the existing network. They may also suggest the addition of new levels of the type or activity hierarchy.

The candidate hypotheses, generated by the suggestors, are sent to the assessment engine, that is responsible for building and maintaining the situation specific BN. Hypotheses in the situation estimate are represented by nodes, or collections of nodes, in the BN. The current BN can be queried at any time to provide an assessment of any hypotheses. The assessment engine is also capable of performing hypotheses management, by periodically evaluating all, or a specific subset of the probabilistic hypotheses and either eliminating them or declaring them to be true.

The situation specific BN is maintained by the assessment engine. It changes dynamically over time, as suggestors present new candidate hypotheses, and as the hypothesis management functions prune the network. There is also a capability to store the current state of the network in the “frame cache”. This provides the capability to store the history of the situation estimate as it evolves over time, and a future capability to “backtrack” to an earlier state if the system discovers that it is diverged too far from reality.

5.2 Evaluation

To be of value, the results of Information fusion must be credible to decision makers. This requires that there is a way to evaluate the situation hypotheses that have been generated.

- Location of entity;
- Composition of entity (e.g., number of elements of each type);
- Activity of entity.

Evaluating the quality of a situation estimate in comparison with a ground truth scenario requires first

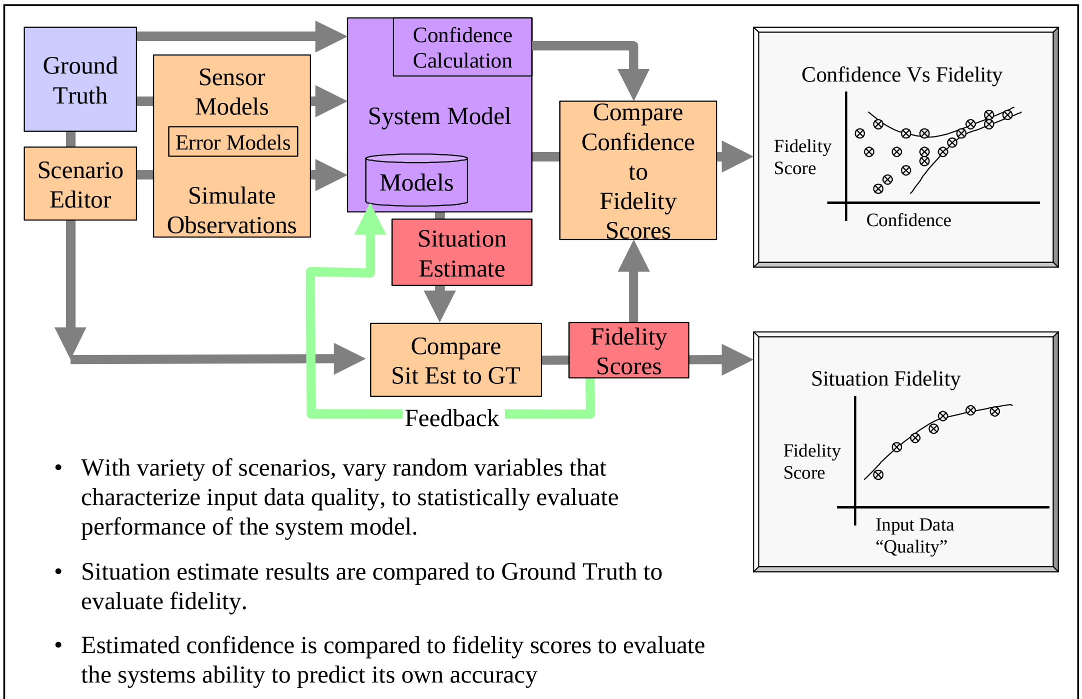


Figure 3. Experimental architecture.

In previous work [8, 9] we demonstrated the capability to measure the fidelity of a situation estimate to ground truth, at either a single or at multiple levels of a force hierarchy.

The elements evaluated include:

- Hypotheses about entities of interest and their attributes;
- An indication of which reports and/or lower level situation elements are associated with each hypothesized entity of interest;
- Inferences about features of the entities of interest.

Relevant features may include:

- Type of entity (e.g., maneuver company, engineer platoon);

associating situation hypotheses with the ground truth elements that gave rise to them and then evaluating how well the situation hypotheses represent the ground truth elements that they represent. In particular, a situation estimate is a faithful representation of a ground truth scenario to the extent that:

- Most ground truth elements are associated with situation hypotheses (there are few *missed detections*);
- Most situation hypotheses have exactly one ground truth element associated with them (there are few *false alarms*);
- The features of interest (e.g., type, location, composition, activity) of the ground truth element are faithfully represented in the situation hypothesis.

The ability to generate a situation estimate has little value unless there is some confidence that the estimate is accurate enough to be useful to a decision maker. An evaluation can be done subjectively by visually comparing a situation estimate with known ground truth. But a subjective evaluation is time consuming to perform and is of little value in quantifying the effects of small changes in domain models, suggestor logic, or hypothesis management. In addition, there are usually only a limited number of ground truth data sets available. Based on similar concerns in other problem domains we have developed an experimental architecture that allows us to systematically evaluate a system model and its components. The experimental architecture is shown in figure 3.

Because of the limited real world ground truth, and the need to evaluate system model performance against a variety of scenarios, evaluations were performed with simulated scenarios. The simulations defined the types, membership, and activities of a hierarchy of military units, and then generate each specific vehicle track. The vehicle tracks were then processed to simulate the TSGUDA inputs, normally received from the ATIF system. The processing may be “error free” to provide ground truth data to TSGUDA, or it may simulate the types of errors characteristic of real TSGUDA input data. We implemented error models for probability of vehicle type ID, probability of detection, probability of correct association (of a vehicle track at one time step with the correct track at the next time step), and a false alarm rate.

The simulated ground truth, with simulated error models applied, was then input to the system model, which generated situation estimates over time as the situation developed. These situation estimates were compared to the original ground truth situation to generate a fidelity score. The fidelity scoring process is described in [8]. The results of the fidelity score can be used in a feedback process to tune the models, suggestor logic, and hypothesis management logic.

It is understood that a fidelity measure can only be used when ground truth is available, so will be of no use in a real military situation where a TSGUDA like situation estimation capability is needed. Figure 3 also shows a confidence calculation. The confidence measure is a metric developed by the system based on the quantity and believed quality of the input data, consistency between the available evidence, and the fit to existing models. The confidence measure will provide an estimate of the quality of the situation estimate independent of ground truth. The theory for the confidence calculation has been developed [9], and implementation is in progress.

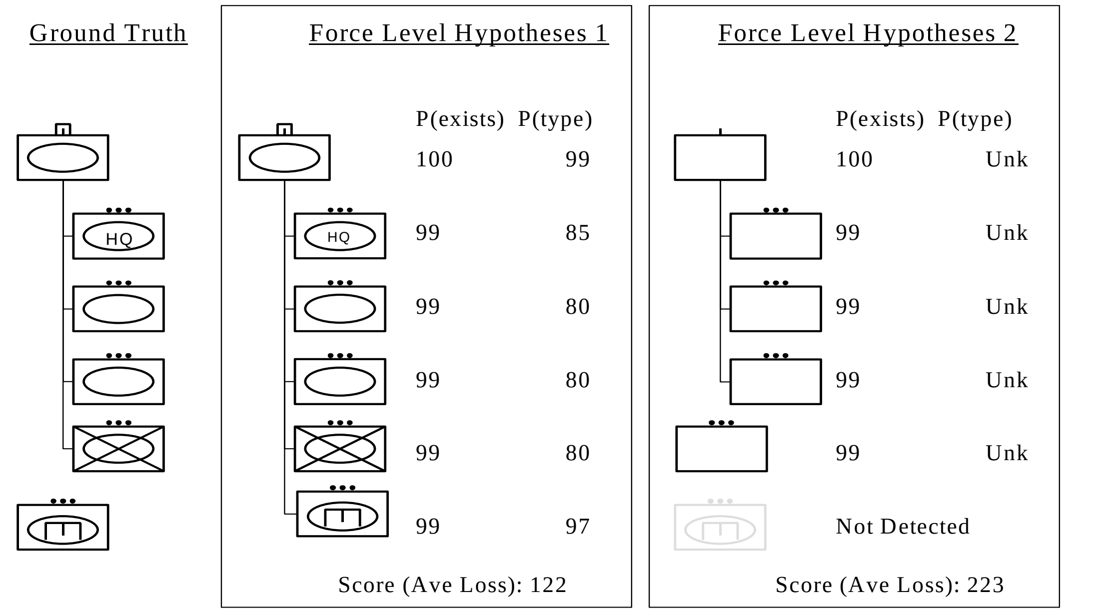


Figure 4. Ground truth force level hypotheses with two example situation hypotheses.

6 Example

An example that illustrates the ability to recognize a military force hierarchy as part of a situation assessment in Figure 4. The assessment was performed using TSGUDA with simulated data. The ground truth force hierarchy is shown consisting of a Armor Heavy Company Team, with a HQs Plt, two Armor Platoons, and a Mechanized Infantry Platoon. A simulated scenario that included coordinated activities of these units was used to generate ground truth tracks for all the individual vehicles. Then a level 1 fusion simulator was used to generate *observed* vehicle tracks. These observed tracks included missed detections, false alarms, track misassociations and type misclassifications. This evidence was input to the TSGUDA system. There is also a separate Engineer Platoon.

The TSGUDA system included domain models for platoon an company level force organization and activities. Suggestors looked for clusters of vehicles and hypothesized that they were groups. Other suggestors looked at the evidence for vehicle types in a hypothesized group, and suggested group, or unit, types. Additional suggestors identified clusters of platoons to hypothesize company sized groups, and assessed platoon type hypotheses to suggest the company type. In addition, other suggestors, evaluated the locations of individual vehicles to suggest possible platoon formations, which combined with activities of individual vehicles, suggested unit activities.

Two example force level hypotheses, generated from two different sets of simulated level 1 fusion inputs, are also shown. In the first, the simulated level 1 fusion errors were consistent with the performance of the ATIF system. In this example all platoons have been detected and identified with high probabilities. The differences between the ground truth and the hypotheses is in the hypothesized membership of the Engineer Platoon in the Company, and in the vehicles hypothesized as members of the Engineer Platoon (not shown). The score (average loss from the fidelity scoring algorithm) is 122.

The second force level hypotheses was generated from a simulation with significantly higher simulated level 1 fusion errors. This set of hypotheses contains a company of unknown type, consisting of three platoons of unknown type, and one additional platoon. The separate Engineer platoon was not detected in this example. The score (average loss from the fidelity scoring algorithm) for this example is 223. While these results are poorer then the first example, they do demonstrate that our approach for situation assessment can identify force hierarchies, even in the presence of significant input errors.

7 Conclusions

This paper has outlined the IET approach to situation assessment. Our key tenets are

i) that hiearchical Bayesian inference as embodied in BNs provide a scientifically sound and robust uncertainty calculus;

ii) the SPI algorithm provides the necessary solution modularity to drive dynamic situation assessment;

iii) that human expertise and domain knowledge can be captured in hierarchical models of the task domain;

iv) the knowledge can be represented in modular and parametrizable BNFrags and MEBNs;

v) robust decision logic can be designed to guide the construction of situation specific BNs;

vi) there are practical methodologies to evaluate the fidelity of assessment;

vii) together, these elements provide a demonstrated capability to perform information fusion for dynamic, scalable situation assessment.

References

[1] Pearl, J., *Probabilistic Reasoning in Intelligent Systems*. 1988, San Mateo, CA: Morgan Kaufmann.

[2] Jensen, F.V., *An Introduction to BNs*. 1997, New York: Springer-Verlag.

[3] Pradhan, M., G. Provan, B. Middleton, and M. Henrion. *Knowledge Engineering for Large BNs*. in *Uncertainty in Artificial Intelligence*. 1994. San Francisco, CA: Morgan Kaufmann.

[4] Laskey, K.B. and S.M. Mahoney. *Network Fragments: Representing Knowledge for Constructing Probabilistic Models*. in *Uncertainty in Artificial Intelligence*. 1997. San Francisco, CA: Morgan Kaufmann.

[5] Li, Z. and B. D'Ambrosio, *Efficient inference in Bayes Networks as a combinatorial optimization problem.* International Journal of Approximate Reasoning, 1994. **11**: p. 55-81.

[6] Laskey, K.B., *Multi-Entity BNs: A Logic for Plausible Reasoning. Unpublished Draft*. 2002, George Mason University: Fairfax, VA.

[7] D'Ambrosio, B., M. Takikawa, D. Upper, J. Fitzgerald, and S.M. Mahoney. *Security Situation Assessment and Response Evaluation (SSARE)*. in *DISCEX-II*. 2001. Los Angeles, CA.

[8] Wright, E., S.M. Mahoney, and K.B. Laskey. *Use of Domain Knowledge Models to Recognize Cooperative Force Activities*. in *Proc. 2001 MSS National Symposium on Sensor and Data Fusion*. 2001. San Diego, CA.

[9] Mahoney, S.M., K.B. Laskey, E. Wright, and K.C. Ng. Measuring Performance for Situation Assessment. in 2000 MSS National Symposium on Sensor and Data Fusion. 2000. San Antonio, TX.