Distributed Belief Propagation in Multi-agent Environment

Subrata Das(✉) and Ria Ascano

Machine Analytics, Cambridge, MA, USA
sdas@machineanalytics.com

Abstract. A distributed net-centric environment consist of a large variety of data fusion nodes, where each node represents a sensor, software program, machine, human operator, warfighter, or a combat unit. Fusion nodes can be conceptualized as intelligent autonomous agents that communicate, coordinate, and cooperate with each other in order to improve their local situational awareness (SA), and to assess the situation of the operational environment as a whole. In this paper, we describe how we model this net-centric SA problem using a distributed belief propagation paradigm. A local fusion node maintains the joint state of the set of variables modeling a local SA task at hand using Bayesian network (BN) fragments. Local fusion nodes communicate their beliefs and coordinate with each other to update their local estimates of the situation and contribute to the global SA of the environment. We have implemented the propagation paradigm to determine threat out of terrorist dirty bombs with agents searching unstructured intelligence reports for evidence and assessing local situations via BN fragments. The paradigm is a part of our company's cutting-edge predictive analytics products offering to solve enterprise distributed big data search problem.

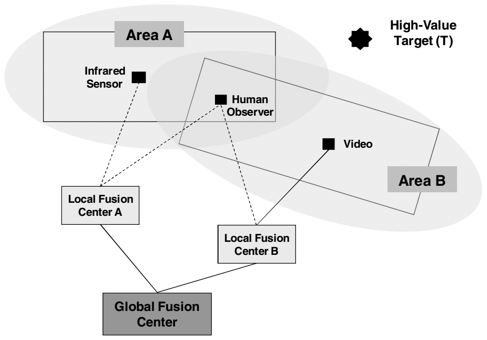
1 Introduction

The concept of distributed fusion (Hall et al., 2012) refers to decentralized processing environments consisting of autonomous sensor nodes, and additional processing nodes without sensors, if necessary, to facilitate message communication, data storage, relaying, information aggregation, and assets scheduling. Some of the advantages of distributed fusion are reduced communication bandwidth, distribution of processing load, aggregation of distributed and proprietary knowledge sources, and improved system survivability from a single point failure. The distributed fusion concept naturally fits within the net-centric multi-agent paradigm.

As a concrete example of distributed fusion, consider the decentralized processing environment as shown in Figure 1 (left). In this example, we assume there is a high-value target within a region of interest, and that the designated areas A and B surrounding the target are considered to be the most vulnerable. These two areas must be under surveillance in order to detect any probing activities, which indicate a possible attack threat. The sensor coverage in areas A and B, shown in grey, is by an infrared sensor (MASINT) and a video camera (IMINT), respectively. In addition, a human

DOI: 10.1007/978-3-319-18944-4_5

observer (HUMINT) is watching the area in common between A and B. There are two local fusion centers for the two areas to detect any probing activity. The infrared sensor has wireless connectivity with the local fusion center for area A, whereas the video camera has wired connectivity with the local fusion center for area B for streaming video. Moreover, the human observer communicates wirelessly with both local fusion centers. Each of the two local centers fuses the sensor data it receives in order to identify any possible probing activity. The centers then pass their assessments (i.e., higher-level abstraction, rather than raw sensor information, thus saving bandwidth) to another fusion center that assesses the overall threat level, based on the reports of probing activities and other relevant prior contextual information.



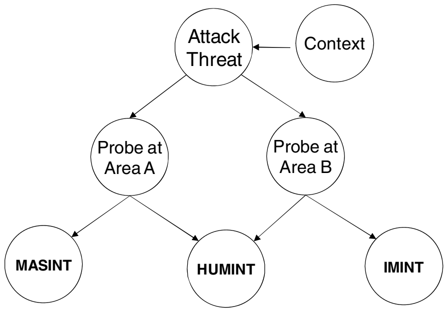


Fig. 1. (left) An example distributed fusion environment; (right) A centralized BN model for situation assessment

In a centralized fusion environment, where observations from IMINT, HUMINT, and MASINT are gathered in one place and fused, a BN model, such as the one in Figure 1 (right), can be used for an overall SA. This model handles dependence among sensors and fusion centers via their representation in nodes and interrelationships. A probing activity at an area will be observed by those sensors covering the area, and the lower half of the BN models this. For example, MASINT and HUMINT reports will be generated due to a probing activity at area A. Similarly, IMINT and HUMINT reports will be generated due to a probing activity at area B. The upper half of the BN models the threat of an attack based on the probing activities at areas A and B, together with other contextual information.

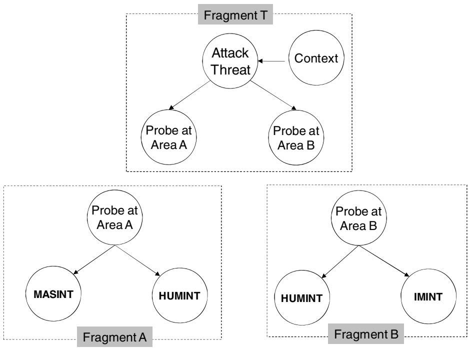


Fig. 2. Distributed parts of the BN models

In a decentralized environment, as illustrated in Figure 2, each of the three fusion centers contains only a fragment of the above BN model. Local fusion centers A and B assess probing activities based on their local model fragments, and send their assessments to the global fusion center via messages. The global fusion center then uses its own models to determine the overall attack threat. If the same HUMINT report is received by both local fusion centers, the process has to ensure that this common information is used only once; otherwise, there will be a higher-than-actual level of support for a threat to be determined by the global fusion model. This is called the *data incest problem* in a distributed fusion environment, which is the result of repeated use of identical information. *Pedigree* needs to be traced, not only to identify common information, but also to assign appropriate trust and confidence to data sources. An information graph (Liggins et al., 1997), for example, allows common prior information to be found.

For situation and threat assessment in a distributed net-centric environment, each node is an agent representing (Das, 2010) a sensor, software program, machine, human operator, warfighter, or a unit. A fusion node maintains the joint state of the set of variables modeling a local SA task at hand. Informally, the set of variables maintained by a fusion node is a *clique* (maximal sets of variables that are all pairwise linked), and the set of cliques in the environment together form a clique network to be transformed into a *junction tree*, where the nodes are the cliques. Thus the cliques of a junction tree are maintained by local fusion nodes within the environment. Local fusion nodes communicate and coordinate with each other to improve their local estimates of the situation, avoiding the repeated use of identical information.

A junction tree can also be obtained by transforming (Jensen, 2001) a Bayesian Belief Network (BN) (Pearl, 1988; Jensen, 2001; Das, 2008b) model representing a global SA model in the context of a mission, thereby contributing to the development of a Common Tactical Picture (CTP) of the mission via shared awareness. Each clique is maintained by a local fusion node. Inference on such a BN model for SA relies on evidence from individual local fusion nodes. We make use of the message-passing inference algorithm for junction trees that naturally fits within distributed NCW environments. A BN structure with nodes and links is a natural fit for distributing tasks in a NCW environment at various levels of abstraction and hierarchy. BNs have been applied extensively for centralized fusion (e.g., Jones et al., 1998; Wright et al., 2002; Das et al., 2002a; Mirmoeini and Krishnamurthy, 2005; Su et al., 2011) where domain variables are represented by nodes.

2 Related Work

There are approaches along these lines, namely Distributed Perception Networks (DPN) (Pavlin et al., 2006) and Multiply Section Bayesian Networks (MSBN) (Xiang et al., 1993), but the proposed approach leverages existing algorithms and reduces the overall message flow to save bandwidth. Please refer to Paskin and Guestrin (2004) for a more detailed account of a junction tree-based distributed fusion algorithm along the lines of the one presented here. The algorithm presented later in the paper, in addition, optimizes the choice of junction tree to minimize the communication and computation required by inference.

There is an abundance of work in the area of distributed agent-based target tracking and, more generally, in the area of distributed fusion. In general, a distributed processing architecture for estimation and fusion consists of multiple processing agents. Here we mention only some of them.

Horling et al. (2001) developed an approach to real-time distributed tracking, where the environment is partitioned into sectors to reduce the level of potential interaction between agents. Hughes and Lewis (2009) investigated the Track-Before-Detect (identify tracks before applying thresholds) problem using multiple intelligent software agents. Martin and Chang (2005) developed a tree based distributed data fusion method for ad hoc networks, where a collection of agents share and fuse data in an ad hoc manner for estimation and decision making.

Graphical Bayesian belief networks have been applied extensively by the fusion community to perform situation assessment (Das et al., 2002). A network structure, modeling a situation assessment problem, with nodes and links is a natural fit for distributing tasks at various levels of abstraction and hierarchy, where nodes represent agents with message flows between agents along the links. An approach along these lines has been adopted by Pavlin et al. (2006). Mastrogiovanni et al. (2007) developed a framework for collaborating agents for distributed knowledge representation and data fusion based on the idea of an ecosystem of interacting artificial entities. Mobile agents have also been employed for distributed fusion.

Mobile agents are able to travel between nodes of a network in order to make use of resources that are not locally available. Mobile agents enable the execution code to be moved to the data sites, thus save the network bandwidth and provide an effective way to overcome network latency. Qi et al. (2001) developed an infrastructure for Mobile-agent-based Distributed Sensor Networks (MADSNs) for multisensor data fusion. Bai et al. (2005) developed a Mobile Agent-Based Distributed Fusion (MADFUSION) system for decision making in Level 2 Fusion. The system environment consists of a peer-to-peer ad-hoc network in which information may be dynamically distributed and collected via publish/subscribe functionality. Jameson's Grapevine architecture (2001) for data fusion integrates intelligent agent technology, where an agent generates the information needs of the peer platform it represents. Gerken et al. (2003) embedded intelligent agents into the Mobile Commander's Associate (MCA) decision aiding system to improve the situational awareness of the commander by monitoring and alerting based on the information gathered.

2.1 Implementation

Our approach to complex analytics[1] in our product is to make use of a computational model and its mobile agent-based distributed belief propagation presented in this paper. Figure 3 presents a Bayesian network model to help in assessing the level of a dirty-bomb threat from a rogue nation. In our model, the site maintaining the root node (for example) continually updates the state (i.e., the probability distribution) of an overall threat based on evidence it receives from its child node, representing terrorism indication and warning, which in turn receives evidence of indications and warnings from its four children, namely, planning, acquisition, making, and deployment. The state of these nodes can be maintained by various sites based on the evidence received from their children nodes. This hierarchical breakdown process continues, and model fragments are determined. A fragment is defined here as a connected

[1] Analytics and data fusion are two sides of the same coin (Das, 2014)

sub-network of the entire belief network model. For the purpose of our demonstration we assume all the fragments are two levels deep as shown in Figure 3.

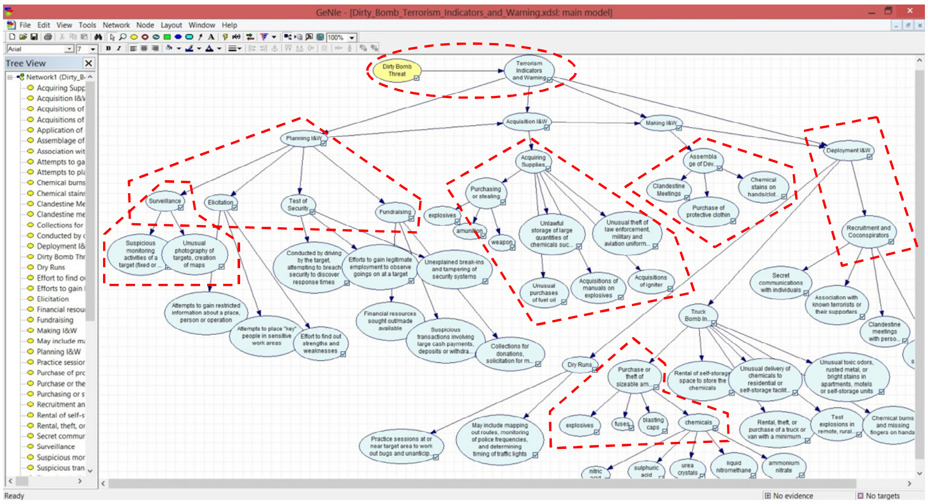
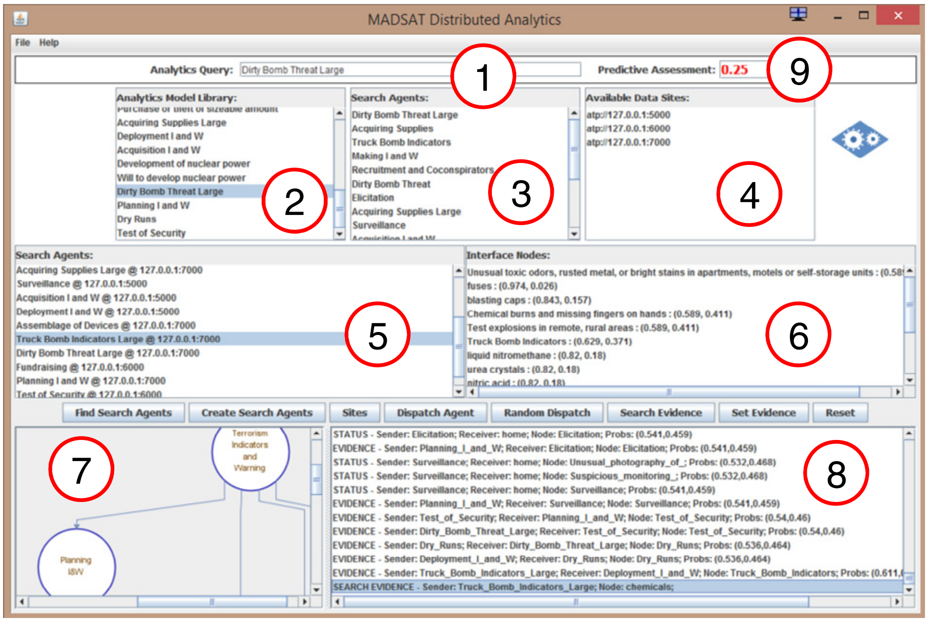

GeNIe - [Dirty_Bomb_Terrorism_Indicators_and_Warning.xdsl: main model]
File Edit View Tools Network Node Layout Window Help
Tree View
Dirty Bomb Threat
Terrorism Indicators and Warning
Acquisition I&W
Making I&W
Planning I&W
Deployment I&W
Surveillance
Elicitation
Test of Security
Fundraising
Acquiring Supplies
Purchasing or stealing
explosives
ammunition
weapon
Assembla ge of Dev...
Clandestine Meetings
Chemical stains on hands/clot...
Purchase of protective clothin
Recruitment and Coconspirators
Secret communications with individuals
Association with known terrorists or their supporters
Clandestine meetings with perso...
Suspicious monitoring activities of a target (fixed or...
Unusual photography of targets, creation of maps
Attempts to gain restricted information about a place, person or operation
Attempts to place "key" people in sensitive work areas
Effort to find out strengths and weaknesses
Conducted by driving by the target, attempting to breach security to discover response times
Efforts to gain legitimate employment to observe goings on at a target
Unexplained break-ins and tampering of security systems
Financial resources sought out/made available
Suspicious transactions involving large cash payments, deposits or withdra...
Collections for donations, solicitation for m...
Unlawful storage of large quantities of chemicals suc...
Unusual theft of law enforcement, military and aviation uniform...
Unusual purchases of fuel oil
Acquisitions of manuals on explosives
Acquisitions of igniter
Truck Bomb In...
Dry Runs
Purchase or theft of sizeable am...
Rental of self-storage space to store the chemicals
Unusual delivery of chemicals to residential or self-storage facilit...
Unusual toxic odors, rusted metal, or bright stains in apartments, motels or self-storage units
Practice sessions at or near target area to work out bugs and unanticip...
May include mapping out routes, monitoring of police frequencies, and determining timing of traffic lights
explosives
fuses
blasting caps
chemicals
Rental, theft, or purchase of a truck or van with a minimum...
Test explosions in remote, rural...
Chemical burns and missing fingers on hands
nitric acid
sulphuric acid
urea crystals
liquid nitromethane
ammonium nitrate
Ready
No evidence
No targets

Fig. 3. Some fragmented models distributed across remote sites

The specific strategy for evaluating a fragment at a remote site is determined based on the site's capability of accumulating evidence from multiple sources. Figure 4 shows the current state of the interface for controlling and monitoring the distributed execution, with nine list components for the following nine purposes respectively:

1. Text area where the analyst poses a full or partial analytics query in key words.
2. Lists all the model fragments stored in a directory such as the ones from the BN in Figure 3, filters the models based on the analyst query, and lets user select one.
3. The selected dependent model fragments based on the user selection that are to be distributed and maintained across remote sites.
4. Publishes available http addresses of the remote sites running Aglets servers to host computation.
5. Lists search nodes of the fragments (same as the list of model fragments above).
6. Provides probability distributions corresponding to the search nodes as model fragments.
7. Graphically display a selected fragment from the library.
8. Displays messages that are received and also the evidence that are found.
9. Overall assessment of threat which is 0.25 based on evidence searched so far.

Users can dispatch fragments individually by selecting a fragment from the area marked 3 to a remote site selected from the area marked 4 by pressing the Dispatch Agent button. Users can also dispatch all fragments at once just by pressing the Random Dispatch button. Evidence on a child node at a remote site can be set by selecting the node in the area marked 6 and then by pressing the button Set Evidence. Various messages will be passed among fragments as described earlier in the section on complex analytics. These messages will be displayed in the area marked 8. The probability distributions of each child node will be updated in areas marked by 6. To start execution of the analytics model, a user dispatches all fragments at once by pressing the Random Dispatch button in the analytics interface.



Analytics Query: Dirty Bomb Threat Large

Predictive Assessment: 0.25

Analytics Model Library:
Purchase or theft of sizeable amount
Acquiring Supplies Large
Deployment I and W
Acquisition I and W
Development of nuclear power
Will to develop nuclear power
Dirty Bomb Threat Large
Planning I and W
Dry Runs
Test of Security

Search Agents:
Dirty Bomb Threat Large
Acquiring Supplies
Truck Bomb Indicators
Making I and W
Recruitment and Coconspirators
Dirty Bomb Threat
Elicitation
Acquiring Supplies Large
Surveillance
Acquisition I and W

Available Data Sites:
atp://127.0.0.1:5000
atp://127.0.0.1:6000
atp://127.0.0.1:7000

Search Agents:
Acquiring Supplies Large @ 127.0.0.1:7000
Surveillance @ 127.0.0.1:5000
Acquisition I and W @ 127.0.0.1:5000
Deployment I and W @ 127.0.0.1:5000
Assemblage of Devices @ 127.0.0.1:7000
Truck Bomb Indicators Large @ 127.0.0.1:7000
Dirty Bomb Threat Large @ 127.0.0.1:7000
Fundraising @ 127.0.0.1:6000
Planning I and W @ 127.0.0.1:7000
Test of Security @ 127.0.0.1:6000

Interface Nodes:
Unusual toxic odors, rusted metal, or bright stains in apartments, motels or self-storage units : (0.58
fuses : (0.974, 0.026)
blasting caps : (0.843, 0.157)
Chemical burns and missing fingers on hands : (0.589, 0.411)
Test explosions in remote, rural areas : (0.589, 0.411)
Truck Bomb Indicators : (0.629, 0.371)
liquid nitromethane : (0.82, 0.18)
urea crystals : (0.82, 0.18)
nitric acid : (0.82, 0.18)

```
STATUS - Sender: Elicitation; Receiver: home; Node: Elicitation; Probs: (0.541,0.459)
EVIDENCE - Sender: Planning_I_and_W; Receiver: Elicitation; Node: Elicitation; Probs: (0.541,0.459)
STATUS - Sender: Surveillance; Receiver: home; Node: Unusual_photography_of_; Probs: (0.532,0.468)
STATUS - Sender: Surveillance; Receiver: home; Node: Suspicious_monitoring_; Probs: (0.532,0.468)
STATUS - Sender: Surveillance; Receiver: home; Node: Surveillance; Probs: (0.541,0.459)
EVIDENCE - Sender: Planning_I_and_W; Receiver: Surveillance; Node: Surveillance; Probs: (0.541,0.459)
EVIDENCE - Sender: Test_of_Security; Receiver: Planning_I_and_W; Node: Test_of_Security; Probs: (0.54,0.46)
EVIDENCE - Sender: Dirty_Bomb_Threat_Large; Receiver: Test_of_Security; Node: Test_of_Security; Probs: (0.54,0.46)
EVIDENCE - Sender: Dry_Runs; Receiver: Dirty_Bomb_Threat_Large; Node: Dry_Runs; Probs: (0.536,0.464)
EVIDENCE - Sender: Deployment_I_and_W; Receiver: Dry_Runs; Node: Dry_Runs; Probs: (0.536,0.464)
EVIDENCE - Sender: Truck_Bomb_Indicators_Large; Receiver: Deployment_I_and_W; Node: Truck_Bomb_Indicators; Probs: (0.611,
SEARCH EVIDENCE - Sender: Truck_Bomb_Indicators_Large; Node: chemicals;
```

Fig. 4. Distributed analytics interface

3 Distributed Fusion Environments

As shown in Figure 5, a typical distributed fusion environment is likely to contain a variety of fusion nodes that do a variety of tasks:

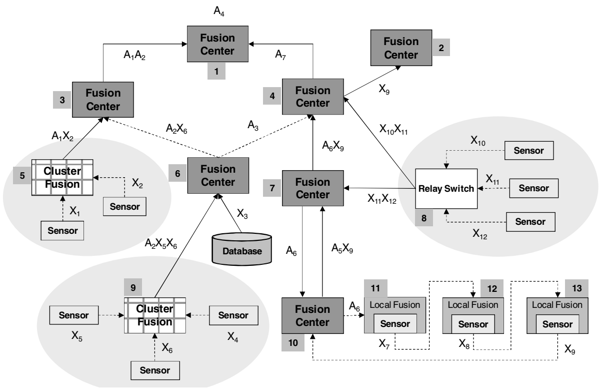


Fig. 5. A typical distributed fusion environment

- Process observations generated from a cluster of heterogeneous sensors (e.g., the local fusion centers A and B in Figure 1, and nodes labeled 5 and 9 in Figure 5).
- Process observations generated from a single sensor (e.g., nodes labeled 11, 12, and 13 in Figure 5).

- Perform a task (e.g., Situation Assessment (SA) and Threat Assessment (TA), Course of Action (COA) generation, planning and scheduling, Common Tactical Picture (CTP) generation, collection management) based on information received from other sensors in the environment and from other information stored in databases (e.g., nodes labeled 1, 2, 3, 4, 6, 7, and 10 in Figure 5).
- Relay observations generated from sensors to other nodes (e.g., the node labeled 8 in Figure 5).

As shown in Figure 5, a fusion node receives values of some variables obtained either from sensor observations (X variables) or via information aggregation by other nodes (A variables). Such values can also be obtained from databases. For example, the fusion center labeled 6 receives values of the variables A2, X5, and X6 from the cluster fusion node labeled 9, and values of the variable X3 from a database. Note that an arrow between two nodes indicates the flow of information in the direction of the arrow as opposed to a communication link. The existence of an arrow indicates the presence of at least a one-way communication link, though not necessarily a direct link, via some communication network route. For example, there is a one-way communication link from node 3 to node 1. A reverse communication link between these two nodes will be necessary in implementing our message-passing distributed fusion algorithm to be presented later.

Each node (fusion center, cluster fusion, relay switch, or local fusion) in a distributed fusion environment has knowledge of the states of some variables, called *local variables*, as shown in Figure 6 (ignore red cross for now). For example, the fusion node labeled 6 has knowledge of the X variables X3, X5, and X6, and A variables A2 and A3. The node receives values of the variables A2, X5, and X6 from the node labeled 9, and the variable X3 from a database. The node generates values of the variable A3 via some information aggregation operation. On the other hand, fusion node 9 receives measurements X4, X5, and X6 from a cluster of sensors and generates A2; fusion node 8 relays values of the variables X10, X1, and X12 to other nodes; and fusion node 12 obtains measurements of X8 from a single sensor.

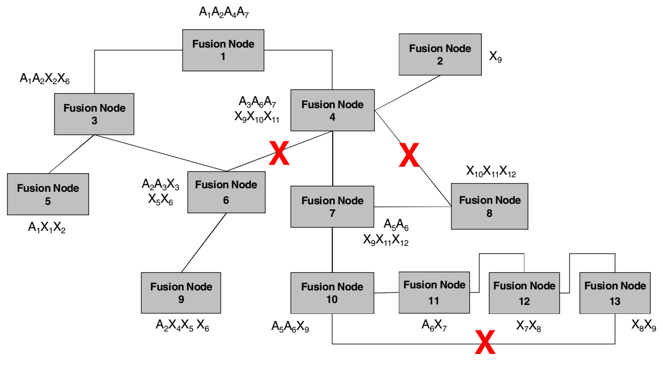


Fig. 6. Network of distributed fusion nodes

There are four possible distributed fusion environments: centralized, hierarchical, peer-to-peer, and grid-based. In a *centralized* environment, only the sensors are distributed, sending their observations to a centralized fusion node. The centralized node combines the sensor information to perform tracking or SA. In a *hierarchical* environment, the fusion nodes are arranged in a hierarchy, with the higher-level nodes processing results from the lower-level nodes and possibly providing some feedback. The hierarchical architecture will be natural for applications where situations are assessed with an increasing level of abstraction along a command hierarchy, starting with the tracking of targets at the bottom level. Considerable savings in communication effort can be achieved in a hierarchical fusion environment. In both *peer-to-peer* and *grid-based* distributed environments, every node is capable of communicating with every other node. This internode communication is direct in the case of a peer-to-peer environment, but some form of "publish and subscribe" communication mechanism is required in a grid-based environment.

4 Algorithm for Distributed Belief Propagation

As mentioned in the introduction, there are two ways in which we can accomplish SA in a distributed environment: 1) each local fusion node maintains the state of a set of variables; 2) there is a BN model for global SA.

In the first case, we start with a distributed fusion environment such as the one shown in Figure 5. Our distributed SA framework in this case has four steps: 1) Network formation; 2) Spanning tree formation; 3) Junction tree formation; and 4) Message passing. The nodes of the sensor network first organize themselves into a network of fusion nodes, similar to the one shown in Figure 6. Each fusion node has partial knowledge of the whole environment. This network is then transformed into a *spanning tree* (a spanning tree of a connected, undirected graph, such as the one in Figure 6, is a tree composed of all the vertices and some or all of the edges of the graph), so that neighbor nodes establish high-quality connections. In addition, the spanning tree formation algorithm optimizes the communication required by inference in junction trees. The algorithm can recover from communication and node failures by regenerating the spanning tree. Figure 6 with (red crosses indicating link severed) describes a spanning tree obtained from the network in Figure 5. The decision to sever the link between nodes 4 and 6, as opposed to between nodes 3 and 6, can be mitigated using the communication bandwidth and reliability information in the cycle of nodes 1, 3, 6, and 4.

Using pairwise communication-link information sent between neighbors in a spanning tree, the nodes compute the information necessary to transform the spanning tree into a junction tree for the inference problem. Finally, the inference problem is solved via message-passing on the junction tree.

During the formation of a spanning tree, each node chooses a set of neighbors, so that the nodes form a spanning tree where adjacent nodes have high-quality communication links. Each node's clique is then determined as follows. If *i* is a node and *j* is a neighbor of *i*, then the variables reachable to *j* from *i*, R_{ij}, are defined recursively as

$$R_{ij} = D_i \bigcup_{k \in nbr(i) - \{j\}} R_{ki}$$

where D_i is the set of local variables of node *i*. A base case corresponds to a leaf node, which is simply a collection of a node's local variables. If a node has two sets of reachable variables to two of its neighbors that both include some variable *V*, then the node must also carry *V* to satisfy the running intersection property of a junction tree. Formally, node *i* computes its clique C_i as $C_i = D_i \bigcup_{\substack{j,k \in nbr(i) \\ j \neq k}} R_{ji} \cap R_{ki}$

A node *i* can also compute its *separator set* $S_{ij} = C_i \cap C_j$ with its neighbor *j* using reachable variables as $S_{ij} = C_i \cap R_{ji}$.

Figure 7 shows the junction tree obtained from the spanning tree in Figure 6. The variables reachable to a leaf node, for example, fusion node 9, are its local variables A_2, X_4, X_5, X_6. The variables reachable to an intermediate node, for example, fusion node 1, from its neighboring nodes 3 and 4 are

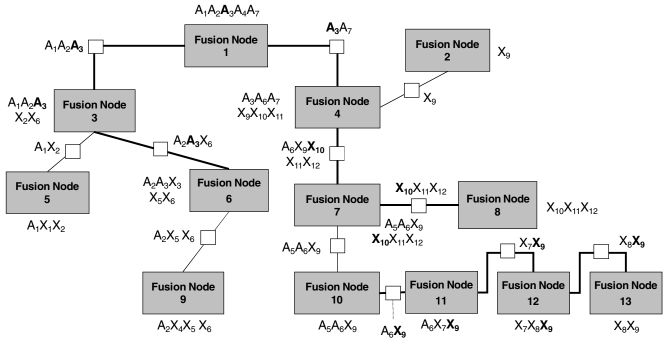


Fig. 7. A junction tree from the distributed fusion environment

$$R_{31} = \{A_1, A_2, A_3, X_1, X_2, X_3, X_4, X_5, X_6\}$$
$$R_{41} = \{A_3, A_5, A_6, A_7, X_7, X_8, X_9, X_{10}, X_{11}, X_{12}\}$$

The local variable of the fusion node 1 is $D_1 = \{A_1, A_2, A_4, A_7\}$. Therefore, its clique is $C_1 = \{A_1, A_2, A_3, A_4, A_7\}$. The formation of a suitable junction tree from a BN model for SA is the only part of our distributed fusion approach that is global in nature.

4.1 Junction Tree Construction and Inference

The *moral graph* of a BN is obtained by adding a link between any pair of variables with a common "child," and dropping the directions of the original links in the BN. An undirected graph is *triangulated* if any cycle of length greater than 3 has a chord, that is, an edge joining two nonconsecutive nodes along the cycle. The nodes of a junction tree for a graph are the *cliques* in the graph (maximal sets of variables that are all pairwise linked).

Once we have formed a junction tree from either of the above two cases, such as the one in Figure 7, a message-passing algorithm then computes prior beliefs of the variables in the network via an initialization of the junction tree structure, followed by evidence propagation and marginalization. The algorithm can be run asynchronously on each node responding to changes in other nodes' states. Each time a node *i* receives a new separator variables message from a neighbor *j*, it recomputes its own clique and separator variables messages to all neighbors except *j*, and transmits them if they have changed from their previous values. Here we briefly discuss the algorithm, and how to handle evidence by computing the posterior beliefs of the variables in the network.

A junction tree maintains a joint probability distribution at each node, cluster, or separator set in terms of a belief *potential*, which is a function that maps each instantiation of the set of variables in the node into a real number. The belief potential of a set of variables **X** will be denoted as φ_{X}, and $\varphi_{\mathrm{X}}(x)$ is the number onto which the belief potential maps *x*. The probability distribution of a set of variables **X** is just the special case of a potential whose elements add up to 1. In other words,

$$\sum_{x \in \mathbf{X}} \varphi_{\mathrm{X}}(x) = \sum_{x \in \mathbf{X}} p(x) = 1$$

The marginalization and multiplication operations on potentials are defined in a manner similar to the same operations on probability distributions.

Belief potentials encode the joint distribution $p(\mathbf{X})$ of the BN according to the following:

$$p(\mathbf{X}) = \frac{\prod_i \phi_{\mathrm{C}_i}}{\prod_j \phi_{\mathrm{S}_j}}$$

where φ_{C_i} and φ_{S_j} are the cluster and separator set potentials, respectively. We have the following joint distribution for the junction tree in Figure 7:

$$p(A_1,...,A_9,X_1,...,X_{12}) = \frac{\phi_{C_1}\phi_{C_2}...\phi_{C_{13}}}{\phi_{\mathrm{S}_{13}}\phi_{\mathrm{S}_{14}}\phi_{\mathrm{S}_{24}}\phi_{\mathrm{S}_{35}}...\phi_{\mathrm{S}_{12\,13}}}$$

where C_i represents the variable in clique *i* and $S_{ij} = C_i \cap C_j$ represents the separator set between nodes *i* and *j*. It is imperative that a cluster potential agrees with its neighboring separator sets on the variables in common, up to marginalization. This imperative is formalized by the concept of local consistency. A junction tree is *locally consistent* if, for each cluster **C** and neighboring separator set **S**, the following holds: $\sum_{\mathrm{C} \setminus \mathrm{S}} \phi_{\mathrm{C}} = \phi_{\mathrm{S}}$. To start initialization, for each cluster **C** and separator set **S**, set the following: $\phi_{\mathrm{C}} \leftarrow 1,\ \phi_{\mathrm{S}} \leftarrow 1$. Then assign each variable *X* to a cluster **C** that contains *X* and its parents $pa(X)$. Then set the following: $\phi_{\mathrm{C}} \leftarrow \phi_{\mathrm{C}}\, p(X \mid pa(X))$.

When new evidence on a variable is entered into the tree, it becomes inconsistent and requires a global propagation to make it consistent. The posterior probabilities can be computed via marginalization and normalization from the global propagation. If evidence on a variable is updated, the tree requires re-initialization. Next, we present initialization, normalization, and marginalization procedures for handling evidence.

As before, to start initialization, for each cluster **C** and separator set **S**, set the following: $\phi_C \leftarrow 1,\ \phi_S \leftarrow 1$. Then assign each variable *X* to a cluster **C** that contains *X* and its parents $pa(X)$, and then set the following: $\phi_C \leftarrow \phi_C p(X \mid pa(X))$, $\lambda_X \leftarrow 1$, where λ_X is the likelihood vector for the variable *X*. Now, perform the following steps for each piece of evidence on a variable *X*:

- Encode the evidence on the variable as a likelihood λ_X^{new}.
- Identify a cluster **C** that contains *X* (e.g., one containing X and its parents).
- Update as follows: $\phi_C \leftarrow \phi_C \dfrac{\lambda_X^{new}}{\lambda_X},\ \lambda_X \leftarrow \lambda_X^{new}$

Now perform a global propagation using the two recursive procedures *Collect Evidence* and *Distribute Evidence*. Note that if the belief potential of one cluster **C** is modified, then it is sufficient to unmark all clusters and call only *Distribute Evidence*(**C**). The potential φ_C for each cluster **C** is now $p(\mathbf{C},e)$, where *e* denotes evidence incorporated into the tree. Now marginalize C into the variable as $p(X,e) = \sum_{C\setminus\{X\}} \phi_C$. Compute posterior $p(X \mid e)$ as follows:

$$p(X \mid e) = \frac{p(X,e)}{p(e)} = \frac{p(X,e)}{\sum_X p(X,e)}.$$

To update evidence for each variable *X* on which evidence has been obtained, first update its likelihood vector. Then initialize the junction tree by incorporating the observations. Finally, perform global propagation, marginalization, etc.

5 Conclusions

We have presented an agent based approach to distributed belief propagation in net-centric environments. The approach provides the foundation of the company's predictive analytics products. We are currently enhancing the product with agent-based approach distributed semantic search to find evidence to propagate in Bayesian network fragments. We are investigating the best way to make use of any types of local model fragments such as rules, neural networks, and decision trees.

References

Bai, L., Landis, J., Salerno, J., Hinman, M., Boulware, D.: Mobile agent-based distributed fusion (MADFUSION) system. In: Proceeding of the 8th International conference on Information Fusion, Philadelphia (2005)

Chong, C-Y., Mori, S.: Graphical models for nonlinear distributed estimation. In: Proceedings of the Conference on Information Fusion, vol. I, pp. 614–621 (2004)

Das, S., Grecu, D.: COGENT: cognitive agent to amplify human perception and cognition. In: Proc. of the 4th Int. Conf. on Autonomous Agents, Barcelona, June 2000

Das, S., Grey, R., Gonsalves, P.: Situation assessment via bayesian belief networks. In: Proc. of the 5th Int. Conference on Information Fusion, Annapolis, Maryland (2002a)

Das, S., Shuster, K., Wu, C.: ACQUIRE: agent-based complex query and information retrieval engine. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy (2002b)

Das, S.: Foundations of Decision-Making Agents: Logic, Probability, and Modality. World Scientific/Imperial College Press, Singapore/London (2008a)

Das, S.: High-Level Data Fusion. Artech House, Norwood (2008b)

Das, S.: Agent-based information fusion. Guest Editorial, Information Fusion, Elsevier Science **11**, 216–219 (2010)

Das, S.: Computational Business Analytics. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series (2014)

Gerken, P., Jameson, S., Sidharta, B., Barton, J.: Improving army aviation situational awareness with agent-based data discovery. In: American Helicopter Society 59th Annual Forum, Phoenix, Arizona (2003)

Hall, D., Liggins, M., Chong, C. (eds) Distributed Data Fusion for Network-Centric Operations. CRC Press (2012)

Horling, B., Vincent, R., Mailler, R., Shen, J., Becker, R, Rawlins, K., Lesser, V.: Distributed sensor network for real time tracking. In: Proceedings of the 5th International Conference on Autonomous Agents, Montreal, pp. 417–424 (2001)

Hughes, E., Lewis, M.: An intelligent agent based track-before-detect system applied to a range and velocity ambiguous radar. In: Electro Magnetic Remote Sensing Defence Technology Center (EMRS DTC) Technical Conference (2009)

Jameson, S.: Architectures for distributed information fusion to support situation awareness on the digital battlefield. In: Proc. of the 4th Int. Conf. on Data Fusion, pp. 7–10 (2001)

Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer, NY (2001)

Lichtblau, D.E.: The critical role of intelligent software agents in enabling net-centric command and control. In: Command and Control Research and Technology Symposium, The Power of Information Age Concepts and Technologies, San Diego, CA (2004)

Liggins, M.E., Chong, C.-Y., Kadar, I., Alford, M.G., Vannicola, V., Thomopoulos, S.: Distributed fusion architectures and algorithms for target tracking. Proceedings of the IEEE **85**(1), 95–107 (1997)

Martin, T., Chang, K.: A distributed data fusion approach for mobile ad hoc networks. In: Proceedings of the 8th Int. Conference on Information Fusion, pp. 25–28 (2005)

Mastrogiovanni, F., Sgorbissa A., Zaccaria, R.: A distributed architecture for symbolic data fusion. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India (2007)

Mirmoeini, F., Krishnamurthy, V.: Reconfigurable bayesian networks for adaptive situation assessment in battlespace. In: Proceedings of the IEEE Conference on Networking, Sensing and Control, pp. 810–815 (2005)

Paskin, M., Guestrin, C.: Robust probabilistic inference in distributed systems. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI), Banff, Canada (2004)

Pavlin, G., de Oude, P., Maris, M., Hood, T.: Distributed perception networks. In: Proc. of the International Conference on Multisensor Fusion and Integration for Intelligent Systems, Heidelberg, Germany (2006)

Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Mateo (1988)

Qi, H., Wang, X., Iyengar, S., Chakrabarty, K.: Multisensor data fusion in distributed sensor networks using mobile agents. In: Proceedings of 5th International Conference on Information Fusion, pp. 11–16 (2001)

Su, X., Bai, P., Du, F., Feng, Y.: Application of Bayesian Networks in Situation Assessment. In: Chen, R. (ed.) ICICIS 2011 Part I. CCIS, vol. 134, pp. 643–648. Springer, Heidelberg (2011)

Waldock, A., Nicholson, D.: Cooperative decentralised data fusion using probability collectives. In: Proc. of the 1st Int. Work. on Agent Technology for Sensor Networks (2007)

Wright, E., et al, T.: Multi-entity bayesian networks for situation assessment. In: Proceedings of the 5th International Conference on Information Fusion, pp. 804–811 (2002)

Xiang, Y., Poole, D., Beddoes, M.: Multiply sectioned Bayesian networks and junction forests for large knowledge based systems. Computational Intelligence **9**(2), 171–220 (1993)