Guided Incremental Construction of Belief Networks

Charles Sutton, Brendan Burns, Clayton Morrison, and Paul R. Cohen

Department of Computer Science,
University of Massachusetts, Amherst, MA 01002
{casutton,bburns,cohen,morrison}@cs.umass.edu

Abstract. Because uncertain reasoning is often intractable, it is hard to reason with a large amount of knowledge. One solution to this problem is to specify a set of possible models, some simple and some complex, and choose which to use based on the problem. We present an architecture for interpreting temporal data, called AIID, that incrementally constructs belief networks based on data that arrives asynchronously. It synthesizes the opportunistic control of the blackboard architecture with recent work on constructing belief networks from fragments. We have implemented this architecture in the domain of military analysis.

1 Introduction

Reasoning problems in many domains, such as intelligence analysis and common-sense reasoning, seem to require both large amounts of knowledge and reasoning under uncertainty. For example, a military commander may try to intuit an enemy's intended attack based on knowledge of the likely terrain, weather, enemy strength, and other factors, as well as actual sightings of enemy forces. Or, finding a forgotten book can make use of (highly uncertain!) memories of where it was last seen, places where it might have been used, and people who might have wanted to borrow it, as well as data acquired by looking around the office. In both cases, the reasoner is uncertain about features of the world, and can marshal both observations and large amounts of general and domain-specific knowledge to resolve the uncertainty.

Belief networks are a popular method for reasoning with uncertainty, but traditionally they require specifying a complete model of the domain. Such expert systems assume that in advance a designer can winnow the set of possible influences, which can be large, down to the few that are relevant.

This approach has two serious limitations. First, in intelligence domains, part of the uncertainty is knowing how many influences there *are*. We may not know exactly how many units are on a battlefield, for example. It is difficult to design a complete probabilistic model for a domain without knowing how large the domain is.

Second, in common-sense reasoning, any observation (such as a car not starting) has many possible explanations, ranging from plausible (out of gas) to implausible (wrong car) to downright ridiculous (starter motor stolen by aliens to

repair spaceship). To reason with a lot of general knowledge—imagine a probabilistic knowledge base as large as Cyc! [11]—it helps to be able to work with a plausible subset. But if this subset is selected in advance, we cannot handle situations where implausible rules suddenly become plausible, for example, if the car contains unfamiliar belongings or small grey men with large eyes. Knowledge-based systems could be both more robust and generally applicable if they contained implausible knowledge, but were able to ignore it until new observations called for its use.

The *blackboard architecture* [5, 13] provides a natural framework for incrementally interpreting large amounts of data. The architecture was designed for problems that require many heterogeneous types of knowledge. Blackboard systems have a set of reasoning modules, called *knowledge sources* (KSs), that develop hypotheses using a global data structure called a *blackboard*.

In this paper, we present AIID, an architecture for incrementally interpreting temporal data. AIID is a blackboard architecture in which hypotheses on the blackboard are nodes in a belief network. It extends traditional blackboard techniques by using a principled method for representing uncertainty, and it extends traditional belief network techniques by incrementally building a model as incoming data warrant.

The principal type of knowledge source in AIID is a *network fragment*, a reusable mini-belief network from which larger networks are constructed. Using fragments, AIID can construct a network incrementally in response to data that arrive asynchronously. Network fragments can be reused across domains. Previous work [6, 9, 10] has discussed methods for constructing belief networks using network fragments. AIID extends this work by incorporating procedural knowledge sources that programmatically alter the network, and by focusing on how the growth of the network will be controlled.

In sections 2 and 3, we summarize the blackboard architecture and the existing literature on incrementally constructing belief networks. In section 4, we describe AIID, including its blackboard, knowledge sources, and control. In section 5, we illustrate AIID's operation on an example. We conclude by discussing our prototype implementation, in the domain of military analysis.

2 Blackboard Systems

Blackboard systems are knowledge-based problem solvers that work through the collaboration of independent reasoning modules. They were developed in the 1970s and originally applied to signal-processing tasks. The first, HEARSAY-II [5], was used for speech recognition, employing acoustic, lexical, syntactic, and semantic knowledge. Other systems were applied to problems as diverse as interpretation of sonar data, protein folding, and robot control.

Blackboard systems have three main components: the blackboard itself, knowledge sources (KSs), and control. The blackboard is a global data structure that contains hypotheses or partial solutions to the problem. The blackboard is typi-

cally organized into sections by levels of abstraction. For example, HEARSAY-II had different levels for phrases, words, syllables, and so forth.

Knowledge sources are small programs that write new hypotheses to the blackboard. Ideally, knowledge sources interact only by posting to the blackboard. Different KSs use different types of knowledge: for example, one might use a syntactic grammar to generate words that are likely to occur next, while another might detect phonemes directly from the acoustic signal. While no single knowledge source could solve the problem, working together they can, and the blackboard model provides an organized means of collaboration.

One focus of the architecture is control [4]. The operation of a blackboard system can be seen as search for hypotheses that explain the data at each level of abstraction, using the KSs as operators. Rather than search bottom-up (i.e., from the data level to the most abstract level) or top-down, blackboard systems can search opportunistically, dynamically rating KSs based on the current data and on the partial solutions that exist so far.

Because hypotheses posted to the blackboard are often uncertain, systems have needed to represent that uncertainty. Heuristic methods have generally been used [4]: for example, HEARSAY-II used a numerical confidence score that ranged from 1 to 100. We are not aware of a blackboard system that represents the links between hypotheses using conditional probabilities.

3 Belief Networks

A belief network is a directed graphical model that represents a probability distribution by a graph that encodes conditional independence relationships, and a set of conditional distributions that give a local distribution for variables based on their parents in the graph. Belief networks are described formally in Russell and Norvig [16].

Belief networks that describe several similar objects often have repetitive structure. For example, in the military domain, every unit has attributes like Unit-Type (e.g., tanks, infantry, artillery) and Direct-Fire-Radius. These attributes have relationships that do not depend on the particular unit: for example, tanks can shoot farther than infantry. If we simply have nodes called Unit-Type-for-Unit1, Direct-Fire-Radius-for-Unit1, etc., then the humans constructing the network need to specify separate, identical CPTs for each unit, which is impractical because there could be many units, and we do not know in advance how many.

Several authors [6, 9, 10] have addressed this problem by breaking up large belief networks into smaller subnetworks. Subnetworks have designated input nodes—which have no conditional distribution, requiring that their distribution be specified in a different subnetwork—and resident nodes, which do have CPTs. A standard belief network can be created from subnetworks by unifying the input nodes of one subnetwork with the resident nodes of another. Repetitive structure can be specified once in a subnetwork and instantiated multiple times to exploit redundancies in the domain.

Object-oriented Bayesian networks (OOBNs) [9, 15] employ strong encapsulation between subnetworks. Each subnetwork defines a set of output variables, and combinations between subnetworks are made only by connecting the output variables of one subnetwork to the input variables of another. Each subnetwork can be seen as a single cluster node in a higher-level belief network, so that an OOBN defines a single probability distribution over its variables. An attractive feature of OOBNs is that their hierarchical structure can be exploited to guide clustering for the junction-tree algorithm, making inference more efficient. Since the subnetworks are connected by a knowledge engineer, rather than automatically, OOBNs are not a technique for incrementally building models based on incoming evidence.

Network fragments [10] are another approach to constructing modular subnetworks. Unlike OOBNs, nodes can be resident in more than one fragment, so they designate influence combination methods for combining distributions from multiple fragments. So network fragments can combine in more unexpected ways than in OOBNs, which precludes specialized inference algorithms, but can be more flexible for specifying complicated belief networks.

4 Architecture

4.1 The Blackboard

The blackboard of AIID represents the system's current beliefs about the domain. The blackboard contains a possibly disconnected belief network that includes previous observations, background knowledge, and hypotheses about the data. In the military domain, the blackboard contains nodes that include sightings and hypothesized locations of enemy units, locations of key terrain, and hypotheses about the enemy's tactics and strategy. A sample blackboard is shown in figure 1.

As in the subnetwork literature, we use a first-order extension to belief networks to represent multiple similar entities more conveniently, analogous to the extension of propositional logic to predicate logic. Instead of naming the random variables by a single atom, e.g. UNIT-MASS, each node has a *node-type*, for example, UNIT-MASS, and a set of *arguments*, for example, "Tank Regiment 1." Logic variables can be used as arguments in KSs to describe a relationship that does not depend on the particular argument values. The combination of a node-type and arguments uniquely specifies a node on the blackboard.

Information on the blackboard can occur on different temporal scales. For example, we can represent a short meeting between two people as a punctual event, while an activity like "Planning-Attack" takes an extended amount of time. We handle these scales using two temporal representations: a tick-based representation, and an interval representation. At lower levels of the blackboard, where we are considering things like meetings and current locations, each network node is indexed by the time it occurs, and the entire network is a Dynamic Bayesian Network (DBN) [12]. At higher levels of the blackboard, which correspond to

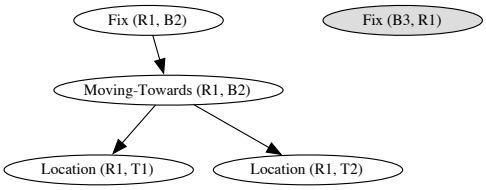

Fig. 1. A sample blackboard in the military analysis domain

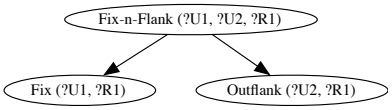

Fig. 2. A sample knowledge fragment for the military analysis domain

long-term actions and intentions, we represent events by the interval in which they occur. Each event has a start-time and an end-time that are explicit nodes in the network. These two representations are integrated as described elsewhere [3].

4.2 Knowledge Sources

A knowledge source is a procedure that modifies the blackboard. Knowledge sources can post new nodes to the blackboard, add edges, alter CPTs, and remove nodes. Every KS has three components, which can be arbitrary procedures: a confidence, a precondition, and an action. The confidence returns a number that indicates how intrinsically useful the KS is. The precondition is run when the blackboard changes and returns true if the KS is applicable. The action is the procedure that actually modifies the blackboard.

As in blackboard systems, KS actions can be full-fledged programs. For example, a KS might use arbitrary heuristics to post simplifying assumptions to make reasoning more tractable. In the military domain, for example, our implementation uses a grouping KS that treats several enemy units as a group if they seem sufficiently close.

Another type of KS cleans up nodes that accumulate from old time steps. Old nodes can slow down inference without greatly affecting current beliefs. Cleanup KSs can remove nodes that are either older than some cutoff or that do not cause a large drop in information about certain nodes of interest. We define the information value of a node in section 4.3.

The most common type of KS is a *network fragment*, which is a belief network that represents a small fragment of knowledge. Example fragments are shown in figures 2 and 6. A node in the fragment *matches* a node on the blackboard when the two nodes have the same type, and their argument lists unify. (Recall that nodes in a KS can have logic variables in their argument lists, and the arguments of a node are distinct from its set of possible outcomes.) By default,

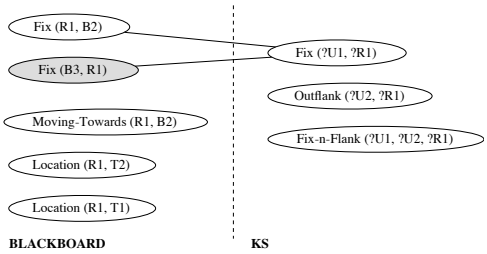

Fig. 3. The bipartite matching problem from matching KS 2 to the blackboard in figure 1

the precondition for a fragment KS is that at least one of the fragment nodes has a match on the blackboard; however, the KS designer can designate certain nodes that must be matched, or write an arbitrary precondition.

Posting Network Fragments Fragments are posted to the blackboard by a process that resembles unification. A fragment can be posted to the blackboard if three conditions hold. First, each of the fragment nodes must match a node on the blackboard; a new node can be created on the blackboard if necessary. Second, a single unifying assignment must unify the argument lists of all the fragment nodes with their corresponding blackboard nodes—this merely ensures that a logic variable like ?U refers to the same thing throughout the fragment. Third, no two fragment nodes can match the same blackboard node.

We can think of fragment matching as a bipartite matching problem, as shown in figure 3. On the left side of the bipartite graph are all the blackboard nodes; on the right are all the fragment nodes. A blackboard node and a fragment node are linked if they have the same node type. Now, any bipartite matching in this graph describes a way the fragment could be posted to a blackboard. A fragment node unifies with its neighbor in the matching. If it has no neighbor, a new node is posted to the blackboard.

Once a fragment has been matched to the blackboard, it can be posted. An example of a fragment posting is given in figure 4. A fragment is posted to the blackboard in three steps. First, new nodes are posted if they are required by the match. Second, for every pair of fragment nodes that are linked, a corresponding edge is added to the blackboard. Now the nodes on the blackboard have both their original parents $\mathbf{V}_{BB}$ and the new parents that were specified by the fragment, $\mathbf{V}_F$. Third, since every node V in the fragment has both a conditional distribution in the fragment, $\mathrm{P}(V \mid \mathbf{V}_F)$, and a one on the blackboard, $\mathrm{P}(V \mid \mathbf{V}_{BB})$, these two distributions are combined to get $\mathrm{P}(V \mid \mathbf{V}_F, \mathbf{V}_{BB})$. Ways this can be done are given in the next section.

Influence Combination Since network fragments are themselves belief networks, they specify a complete probabilistic model over their variables. But nodes

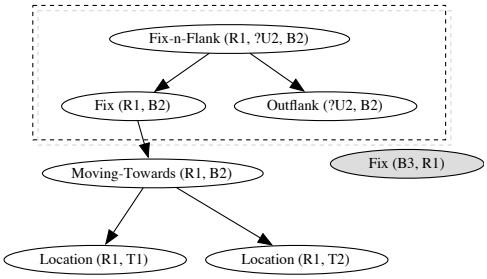

Fig. 4. The blackboard in figure 1 after KS 2 posts

on the blackboard already have a probability distribution. Suppose that some node V has parents $\mathbf{V}_F$ in the fragment and parents $\mathbf{V}_{BB}$ on the blackboard, so that we have probability distributions $\mathrm{P}(V \mid \mathbf{V}_F)$ and $\mathrm{P}(V \mid \mathbf{V}_{BB})$. When the KS posts, the parents of V will be $\{\mathbf{V}_F \cup \mathbf{V}_{BB}\}$, so we must merge these two models to get a CPT for $\mathrm{P}(V \mid \mathbf{V}_F, \mathbf{V}_{BB})$. For example, between figures 7 and 7, the distribution P(ALARM | BURGLAR) must be combined with the distribution P(ALARM | BASEBALL-BREAKS-WINDOW).

There are several ways to do this. Laskey and Mahoney [10] define several *influence combination methods* to combine conditional probability distributions, one of the principal types being parametric causal models like *noisy-or*. The noisy-or model [14, 17] allows one to compactly specify a conditional distribution when the parents are independent, stochastic causes of the child. The knowledge engineer can specify which combination method should be used for a given node type.

4.3 Control

Many knowledge sources are applicable at any given time, but only a few can be selected to run. This is both because our implementation of AIID runs on a single processor, so only one KS can be run at a time, and because if too many KSs fire, the blackboard could become too large for probabilistic inference to be tractable. We describe three types of control regimes: a simple one based on KSs' confidence methods, an information-theoretic one based on nodes of interest to the user, and a Bayesian one based on the probability of the blackboard structure given the observed data.

First, KSs can be ordered by confidence. This provides a gross ordering among KSs, but it can be difficult to know in advance, or write a procedure that computes, how useful a KS will be.

Second, certain nodes of the blackboard have more interest to the user. For example, an intelligence analyst may want to know whether two people have communicated, or a military commander may want to know whether a certain attack is a feint. Let $\mathbf{V}$ be a set of these nodes of interest. We can choose to post the knowledge sources that provide the most *information* about $\mathbf{V}$, in the

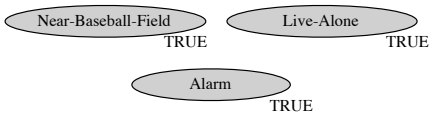

Fig. 5. Initial knowledge in the burglar-alarm example. Darkened nodes are observed, with the observed value printed underneath the node

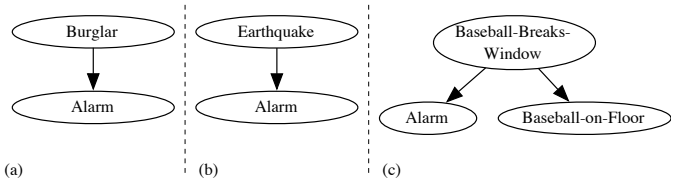

Fig. 6. Three knowledge sources for the burglar-alarm example

sense of reducing its Shannon entropy. The information gained from firing a KS K is given by $I(K) = H(\mathbf{V}) - H_K(\mathbf{V})$, where $H(\mathbf{V})$ is the entropy of $\mathbf{V}$ before K is posted, and $H_K(\mathbf{V})$ is the entropy afterward. We can compute this directly by temporarily posting K, computing the marginal distribution $\mathrm{P}(\mathbf{V})$, and calculating its entropy.

Since this is probably too expensive to use if many KSs are applicable, we can try to approximate this effect. We can get a cheap approximation by simply looking at the distance between where the KS will post and $\mathbf{V}$, that is, the length of the shortest undirected path between a node used by the KS and a member of $\mathbf{V}$. Then we prefer the KSs with the shortest distance, on the assumption that nodes that are closer to $\mathbf{V}$ have more influence on its distribution.

Third, previous work on structure learning in belief networks is applicable to control decisions. Bayesian structure learning is concerned with finding the structure for a belief network B_s which maximizes the posterior probability $\mathrm{P}(B_s \mid D, \xi)$ for some set of observed data D and an environment ξ. For making control decisions, we can select the KS whose application would maximize the posterior probability of the blackboard structure given the observations.

Although there is a closed-form solution for this probability, its calculation is computationally intractable [7]. As a result of this numerous approximation techniques for estimating this probability have been proposed. Heckerman [7] gives a good overview of these approximation techniques, most of which are applicable to the task of control, although some require multiple independent observations, which may not always be available.

5 Example

In this section we illustrate the action of the architecture by a diagnostic example inspired by one from Judea Pearl [16]. Suppose you are upstairs in your house and hear your burglar alarm go off. The alarm might have gone off for many

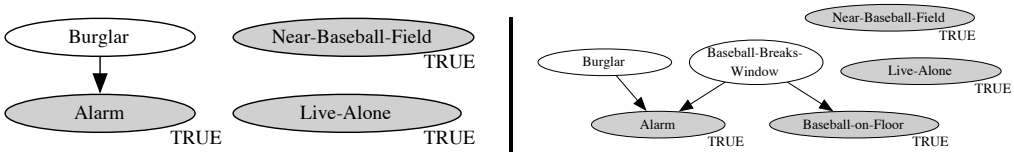

Fig. 7. At left, he blackboard after 6(a) has fired. At right, the blackboard after KS 6(c) has fired

reasons: an earthquake, a forgetful member of the household, a baseball breaking a window, an actual burglar, and many others. Of course you'll investigate, but as you get up to do this, you naturally wonder what you'll find downstairs.

Your initial knowledge might be represented in AIID as figure 5: you have just heard an alarm and you live alone near a baseball field. Since ALARM has just been posted, KSs that use that node are applicable. Some available KSs are listed in figure 6.

Now we have to choose which of those KSs should activate. Suppose we use the Bayesian control regime. In this case, ALARM is observed, so we choose the KS that maximizes its posterior probability. If we assume pessimistically that burglaries are more likely than baseball games, it may be that 6(a) results in the more probable structure, so assume that it is chosen. The result of posting KS 6(a) is shown in figure 7.

Now, suppose that you come downstairs and see a broken window and a baseball on the floor. With this information, KS 6(b) becomes very relevant, so suppose it fires. Now influence combination needs to be performed on the ALARM node. Since BURGLARY and BASEBALL are independent causes of the alarm, it is reasonable to use the noisy-or combination method. At this point P(BASEBALL) is likely to be high and P(BURGLAR) to be low (because of explaining away), so the blackboard now (figure 7) contains a good explanation of the observations.

6 Implementation

We have built a prototype of AIID in the domain of military analysis. We simulate military engagements at the battalion level (roughly a thousand troops), using the Capture the Flag simulator [1, 2]. The simulator includes such effects as terrain, fog of war, artillery, combat aviation, and morale.

The data consist of reports about friendly and enemy units, for example, "Unit Red-1 has sound contact with a brigade-sized unit in the east." The prototype does not address the problems of identifying the number and composition of units from individual sightings, which are hard. Rather, our problem is to infer the enemy commander's strategy and, more specifically, the objective of each enemy unit.

The blackboard contains reports about enemy units and hypotheses such as individual unit's objectives, coordinated actions between unit, and theater-wide objectives. We have implemented 30 fragment KSs: for example, one computes

the relative combat strength of two opposing units, and others model of military actions such as defeat (i.e., to attack with overwhelming force), outflank, and fix. One procedural KS clusters enemy units by location, and hypothesizes that the units are groups acting in close concert. This kind of geometric reasoning is difficult to implement in a network fragment.

7 Conclusion

We have presented an architecture for solving knowledge-intensive problems under uncertainty by incrementally constructing probabilistic models. The architecture synthesizes ideas from the older literature on blackboard systems and the newer literature on construction of belief networks from fragments. Blackboard systems are a method of incrementally building symbolic models, while the network fragment systems incrementally build probabilistic models. The contribution of the current work is, in making the connection between the two literatures, to point out that both have good ideas the other hasn't used: probabilistic models have a principled method of reasoning under uncertainty, while blackboard systems have focused on controlling the search through the space of possible models.

It would be natural to extend the architecture to handle influence diagrams, the largest requirement being influence combination methods for decision and utility nodes. The most important open questions in this architecture are better methods of evidence combination and evaluating different control methods. We would also like to apply this architecture to other domains, both within intelligence analysis and common-sense reasoning.

References

1. Marc Atkin, Gary W. King, David Westbrook, Brent Heeringa, Andrew Hannon, and Paul Cohen. SPT: Hierarchical Agent Control: A framework for defining agent behavior. In *Proceedings of Fifth International Conference on Autonomous Agents*, pages 425–432, 2001.
2. Marc Atkin, David L. Westbrook, and Paul Cohen. Domain-general simulation and planning with physical schemas. In *Proceedings of the Winter Simulation Conference*, pages 464–470, 2000.
3. Brendan Burns and Clayton Morrison. Temporal abstraction in Bayesian networks. In *AAAI Spring Symposium*, Palo Alto, CA, 2003.
4. Norman Carver and Victor Lesser. The evolution of blackboard control architectures. In *Expert Systems with Applications 7*, pages 1–30, 1994.
5. L.D. Erman, F. Hayes-Roth, V.R. Lesser, and D.R. Reddy. The HEARSAY-II speech understanding system: Integrating knowledge to resolve uncertainty. *ACM Computing Survey*, 12:213–253, 1980.
6. Robert P. Goldman and Eugene Charniak. A language for construction of belief networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):196–208, 1993.

7. David Heckerman. A tutorial on learning with Bayesian networks. In Michael Jordan, editor, *Learning in Graphical Models*. MIT Press, 1995.
8. Eric Horvitz. *Computation and Action Under Bounded Resources*. PhD thesis, Stanford University, December 1990.
9. Daphne Koller and Avi Pfeffer. Object-oriented Bayesian networks. *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pages 302–313, 1997.
10. Kathryn Blackmond Laskey and Suzanne M. Mahoney. Network fragments: Representing knowledge for constructing probabilistic models. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97)*, San Mateo, CA, 1997. Morgan Kaufmann.
11. DB Lenat and RV Guha. *Building large knowledge-based systems: Representation and inference in the Cyc project*. addison Wesley, 1990.
12. Kevin P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, U.C. Berkeley, July 2002.
13. H. Penny Nii. Blackboard systems. In Avron Barr, Paul R. Cohen, and Edward A. Feigenbaum, editors, *The Handbook of Artificial Intelligence: Volume IV*, pages 1–82. Addison-Wesley, 1989.
14. Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.
15. Avi Pfeffer, Daphne Koller, Brian Milch, and Ken T. Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proceedings of the 14th Annual Conference on Uncertainty in AI (UAI-99)*, pages 541–550. Morgan Kaufmann, 1999.
16. Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Addison-Wesley, 2002.
17. Sampath Srinivas. A generalization of the noisy-or model. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence (UAI-93)*, San Francisco, 1993. Morgan Kaufmann.