Managing Dynamic Contexts Using Failure-Driven Stochastic Models

Nikita A. Sakhanenko and **George F. Luger**
Dept. of Computer Science, Univ. of New Mexico
Albuquerque, New Mexico

Carl R. Stern
Management Sciences, Inc.
Albuquerque, New Mexico

Abstract

We describe an architecture for representing and managing context shifts that supports dynamic data interpretation. This architecture utilizes two layers of learning and three layers of control for adapting and evolving new stochastic models to accurately represent changing and evolving situations. At the core of this architecture is a form of probabilistic logic used to encode sets of recursive relationships defining Dynamic Bayesian Models. This logic is extended with a syntax for defining contextual restrictions on stochastic Horn clauses. EM parameter learning is used to calibrate models as well as to assess the quality of fit between the model and the data. *Model failure*, detected as a poor fit between model and data, triggers a model repair mechanism based on causally informed context splitting and context merging. An implementation of this architecture for distributed weather monitoring is currently under development.

Introduction

Understanding and characterizing context is critical for real-time diagnostic and prognostic reasoning. Context can mediate the interpretation of highly complex data by activating a specific set of inferential strategies and focal points, such as the temporary focus on the causal role of a particular component or subsystem state. The use of inferential strategies and selective focus can serve to reduce the computational size of an estimation task in a probabilistic model by mitigating the need to continuously link local estimation tasks to probability updating over the entire world model.

Modeling context also demands the ability to forget less relevant and/or older information and to shift attention to significant aspects of the current data. Moreover, while modeling a dramatically changing world, transforming context across time can reflect deep changes in the complex internal structure of the world model. Our probabilistic modeling environment supports these and other aspects of context revision.

Many current probabilistic modeling systems, especially those that rest on a knowledge-based model construction approach (Wellman, Breese, & Goldman 1992), map an entire knowledge base into an often complex graphical model. As the size of the resulting network grows large, it becomes time consuming and cumbersome to manipulate, and the inference algorithm can take an exponentially longer time to finish probability updating. Thus, a primary reason for representing contexts dynamically is to reduce the complexity of the constructed model and, consequently, to streamline the inference process. Representing an evolving context requires omitting information that is no longer relevant to the current task from a knowledge base, and at the same time constructing and maintaining the graphical model of the context.

A final reason for creating dynamic contexts in probabilistic modeling systems is the ability to combine multiple *snapshot* models – those models that represent stationary or smoothly evolving stochastic processes. We can think of a single context as a snapshot model; when the context evolves, the modeling system integrates aspects of other snapshot views of the domain. This can be very useful for representing non-stationary processes with abrupt changes. Moreover, our reasoning engine includes failure detection and recovery mechanisms based on causal representations (Pearl 2000). By employing these meta-structures, we can support the explicit characterization and manipulation of context to perform diagnostic and prognostic analysis of complex distributed environments.

In the next section we describe with further detail our dynamic system in which the notion of context plays a major role. Then, after briefly summarizing research related to our approach, we focus on the role of context for reorganizing a current model. We show how context can be used to achieve non-stationary probabilistic modeling with the introduction of contextual mechanisms, such as *context splitting* and *context merging*. We then present the details of our failure-driven dynamic context system through pseudocode, describe the current developmental state of the system, and conclude.

Dynamical Systems Driven by Failure and Recovery Mechanisms

Probabilistic modeling systems that dynamically represent frequently changing data are very important for carrying out complex tasks. Complex modeling systems that can employ available domain knowledge to increase computational efficiency are of the greatest interest. With the increasing use of remote sensing technology continuously and in parallel collecting large sets of data, it becomes more and more necessary to develop the methodology for processing noisy data in a timely manner and for incorporating recovered information into global knowledge about a domain. Since modern sensing systems often employ very large networks, the standard approach of collecting and processing all data at a central location is often not efficient. It sometimes becomes necessary to shift aspects of the computation to the sensors where the data are collected. This introduces additional constraints on the running time and memory of the modeling system.

The most suitable systems in these cases, we believe, are those that are able to *evolve* to handle rapidly changing pieces of information. The idea of a system's evolution is also supported by recent psychological research (Granott, Fischer, & Parziale 2002; Gopnik *et al.* 2004) that offers an analysis of the way humans learn. This research suggests that human learning behavior can be represented by our current generation of probabilistic graphical models.

There is a limitation, however, that makes current probabilistic modeling approaches unable to support this evolution: most of these approaches are *static*, namely, they assume that modeling is done only once and that the entire dataset is available ahead of time. In this paper we introduce a failure-driven probabilistic modeling approach that incorporates ideas from developmental learning, including *assimilation* and *accommodation* (Piaget 1983), to model data from dynamic environments.

During learning by assimilation, the system updates the parameters of the *existing* network, which in psychology is similar to an individual incorporating novel events and objects into an *existing* way of thinking. During learning by accommodation, the system uses a contextualized repair mechanism to *reorganize* the model to accommodate new data. In psychology this is similar to an individual discovering new aspects of an environment that do not fit into his/her existing mental structures and, consequently, *reorganizing* these structures to incorporate this new information. Our computational environment extends our earlier first-order, stochastic, and Turing complete "Loopy Logic" language (Pless 2003).

Review of Related Research

There are currently many different probabilistic modeling systems. Ngo and Haddawy (Haddawy 1994; Ngo & Haddawy 1997; Ngo *et al.* 1997) are the first to produce dynamic systems by joining graphical probabilistic models (Bayesian networks) using the first-order predicate calculus. Friedman et al. (Friedman *et al.* 1999) present probabilistic relational models enabling the specification of a probability model on classes of objects rather than on simple attributes.

Kersting and DeRaedt (Kersting & DeRaedt 2000) propose another approach based on knowledge-based model construction that generates Bayesian networks specific for given queries, using a set of first-order rules with uncertainty parameters. Richardson and Domingos (Richardson & Domingos 2006) propose a probabilistic system based on general first-order logic, sentences of which are converted into a conjunctive normal form, as opposed to restricted subsets of the general logic. They are the first researchers who developed a complete mapping from the first-order predicate calculus with function symbols to probability distributions.

Among all of the logic-based probabilistic modeling approaches, (Ngo & Haddawy 1997) is the only research in the field of stochastic logic modeling that explicitly uses context information about the domain of interest to cut down the size of the knowledge base needed to answer a query.

Pearl (Pearl 2000) and Halpern (Halpern & Pearl 2001) emphasize the importance of a causal model when searching for a good explanation of events under consideration. They argue that the explanation must acknowledge the actual cause of events. For example, if lightning strikes a tree and starts a forest fire, then it is reasonable to say that the lightning is the cause of the fire. But what is the role of the amount of oxygen in the air and dryness of the wood? It seems that there would not be a fire, if the wood were wet and the air was missing oxygen. In order to define the actual cause in such situations, Halpern and Pearl (Halpern & Pearl 2001) propose the language of structural equations. In this paper we use causal structures for context specification, since intuitively, a current probabilistic model can be thought of as an explanation of events.

The research work presented in this paper is partially motivated by advances in educational and developmental psychology. Gopnik et al. (Gopnik *et al.* 2004) investigate the importance of causal representation and several related types of learning mechanisms in the cognitive development of children. The authors argue that knowing about causal structure permits humans to make wide-ranging predictions about future events. Importantly, knowing about causal structure also allows humans to interfere in a dynamic world by triggering new events that produce future results.

Gopnik et al. reason (Gopnik *et al.* 2004) that it is unlikely that children store large amounts of data in memory and then use some learning procedure on that data. Most likely, they say, children use small samples of data to form hypotheses. They then forget the data and revise their hypotheses as suggested by new data. Gopnik et al. note that during such revision, children change not only the hypothesized causal relations, but also variables and properties they consider to be useful. Moreover, the authors suggest that causal regularities learned from one context somehow constrain the causal regularities to be learned in other contexts, supporting *learning by analogy*.

Granott et al. (Granott, Fischer, & Parziale 2002) give further psychological perspective on the basis of human learning. The key notion of their research is *bridging*. Using a term from dynamical systems, bridging is an *attractor* that draws development of a system toward more advanced and more stable levels. The bridging mechanism is carried out by partially defined *shells* that are *scaffolds* directing the development of new knowledge by providing a perspective for processing new experiences.

Granott et al. argue that bridging is a transition mechanism that people use while learning. They claim that different types of bridging are created by using the shells of more advanced knowledge, while leaving out the shell's particular content. They also state that bridging is not the same as hypothesizing, since bridging operates with unknown, not yet realized objects. Hence, hypotheses are the results of bridging. In other words, bridging is a process of drafting a viable path towards an unknown goal.

Non-stationary Stochastic Modeling Using Context

Most of the probabilistic modeling research cited previously tends to make minimal assumptions about the data, hence producing general representations. One of the major shortcomings of these approaches is that they describe stationary probability distributions, that is, these models assume constant statistical regularities across the data. Over a large set of data the stationarity assumption produces a good approximation of correct probability density. However, models with this assumption do not always reflect variations across contexts.

Researchers (Lee, Lewicki, & Sejnowski 2000) have also attempted to use mixtures of models to employ local data representations; however later studies showed that such models do not perform well when the structure of statistical regularities of the data is significantly more complex. Other methods (Sutton & Barto 1998; Everson & Roberts 1999; Pham & Cardoso 2001), that are more successful in representing non-stationary stochastic processes, assume that data statistics vary smoothly from sample to sample. This assumption frequently does not hold: many complex modeling tasks involve data with steep changes that cannot be represented by slowly evolving processes. In all of this research there is no attempt to characterize explicitly the notion of context.

In this paper we use an explicit context to *switch* between local representations, that is, between probabilistic models that represent locally stationary processes. We illustrate this statement with an example.

Context splitting

Suppose there is a burglar alarm installed with remote communication guarding someone's residence. Suddenly the owner receives a message that the alarm has been triggered. We would like to compute the probability that the owner's home has been burglarized. In this example let us assume that the system was produced and sold exclusively in Albuquerque. Given that the alarm system was used by customers in Albuquerque, the local police department recorded the data when the alarm went off and when the residence with the installed alarm was actually burglarized.

Using a declarative language representation (Pless 2003), we add the following sentence to the knowledge base (KB):

$$\mathrm{alarm}(x)|\mathrm{burglary}(x) = L.$$

This sentence shows that the event that the alarm goes off is conditionally dependent on the event that the residence was burglarized, where the conditional probability distribution is unknown (L). In our system the rules of the KB are mapped into a Markov network and then, after inferencing, the KB is updated with the following facts:

$$\mathrm{alarm}(x)|\mathrm{burglary}(x) = [.9, .1, .001, .999].$$

This indicates that the alarm goes off in 90% of the cases involving a burglary, and if the alarm does not go off, 99% of the time there was no burglary. It is important to note that this information is learned from the data from Albuquerque: at this time we do not realize that the particular location of the alarm might be relevant. As far as we are concerned, the KB represents the whole world.

Now let us extend the example. Assume the company starts selling its alarm system in Los Angeles, and the police database expands with new data tuples obtained from LA. Suppose also that the distribution of data from LA is very different from that of Albuquerque, because there are many more false positives in the Los Angeles area. Thus, the graphical model created with Albuquerque data does not fit the data from LA.

Consequently, we need to reorganize the model in order to account for the new data that do not fit the existing structure of the model. In this example we see that by splitting the rule on cases depending on location gives a better predictor.[1]

We can distinguish contexts corresponding to Albuquerque and Los Angeles, and as a result the original rule is split into these two contexts. The rule for Albuquerque stays the same as the original one, but the rule for LA is an unknown distribution, which can be learned using parameter estimation as was shown above. As the result, the structure of the KB is changed and its distributions are updated:

$$\mathrm{alarm}|\mathrm{burglary} = [.9, .1, .001, .999] \leftarrow \mathrm{Albuquerque}$$
$$\mathrm{alarm}|\mathrm{burglary} = [.9, .1, .1, .9] \leftarrow \mathrm{LosAngeles}.$$

For these rules we have used the notation of Haddawy (Haddawy 1994) that represents a context as a predicate after the symbol $\leftarrow$. The context distinguishes two cases, each of which corresponds to a separate Markov network.

Since splitting on the location parameter was successful when the company expanded to LA, we can do the same again when the alarm company goes to Moscow. The new data from Moscow is used separately for this new location distribution. This approach is inspired in developmental learning: an individual learns a strategy that works and then uses it, until it stops working, in which case the individual will have to then learn a new strategy. The technique described here we call *context splitting*.

Context merging

We expand our burglar alarm example even further and assume that eventually the company grows very large and has a lot of different retail locations. Consequently, the corresponding modeling system contains the KB with contexts for each location. So let us assume that San Diego, Los Angeles, and San Francisco are among other locations of the company. We would like to simplify the KB by generalizing it. Generalization may be accomplished by merging rules with contexts.

[1] There are many ways to determine the most appropriate attributes of the database to use for context splitting. A simple approach is to iterate across the attributes of the table and find which attribute partitions the distribution such that we get the maximum information gain (Luger 2005).

Suppose that it can be determined that the distributions corresponding to San Diego, LA, and San Francisco have similar properties. Consequently, these three rules can be generalized into one with a context $\mathrm{California}$. If there is another location, Tokyo for instance, we might detect that the distributions for California and Tokyo are also very similar. The corresponding rules can be generalized into one with a context called *foo* or with some higher level understanding: *Earthquake-Zone*. The technique described here we call *context merging*.

Intuitively, context merging can be supported by a form of causal reasoning. It might be observed in the KB that in San Diego, LA, Tokyo, and San Francisco the frequency of an actual burglary being associated with a triggered alarm is lower, hence, it is possible to reason that there must be another cause for the alarm to go off. Consequently, the new cause simply gets a new name, say *foo*. Later, if needed, a user can provide a better name and the explanation for this phenomenon. Note that this is the way to learn latent variables: the systems suggest correlations, that is, that there is a latent relationship (a possible meaning for the hidden parameter), and then a human comes and identifies the relationship (being in an earthquake zone, in our example). We discuss the connection between the notion of context and causality in a later section.

Flexible Modeling Using Context

In this paper a context mechanism is seen as a tool that introduces more flexibility to the modeling process. Context can be used to granulate the overall representation of a problem into a set of finer modeling tasks, such that depending on the current task, the size of a resulting model is much smaller than that of a general model that tries to incorporate all the knowledge. Hence, the inference on contextualized models is faster, because they are more *lightweight*, i.e., they contain less information that proves to be irrelevant to the current task.

Besides being able to efficiently manipulate with context (as shown in the previous section), we have to address another important issue of how to construct a relatively simple model containing only relevant information. Since the system proposed in this paper uses a knowledge-based model construction technique, one solution is to use context to *filter* which rules of the KB are relevant in the current situation.

A similar approach to ours was proposed by (Ngo & Haddawy 1997). They used simple predicates to identify the relevant subset of a KB, thus producing only a rough approximation of the relevant knowledge. Note that we are willing to spend more computational resources on constructing a tight relevant graphical model corresponding to a current task, as long as the consequent reasoning across the constructed network is efficient. For instance, in the probabilistic modeling system of an airplane, it is reasonable to invest several seconds to reorganize a current model when a plane flies into a turbulence zone, if the inferencing on the resulting model will only take milliseconds.

Importance of Context Definition

As described in the previous sections, in order to successfully operate with context (perform context splitting and merging) and, as a result, obtain smaller and more relevant models, a clear definition of context must be provided. In the burglary alarm example above, the system splits on context when new data does not fit in a current model. We can see this as if a so-far-stationary underlying stochastic process presents non-stationary behavior. We identify this situation via a *failure* of the system to incorporate new data into its current model.

Failure is syntactically straight forward to calculate, for example, by monitoring the changes of a model's distributions across time. It is also possible to set triggers that either inform the model's observers, or better, ask the model to recalibrate its current context as it processes newly arriving data. Semantically defined calls for model recalibration are another matter altogether, perhaps left to the a priori concerns of the human user or results from previous modeling activity.

Motivated by research in educational psychology (Granott, Fischer, & Parziale 2002; Gopnik *et al.* 2004), our context-based model tries to adapt to abrupt changes it encounters by reorganizing its model. In a failure situation, indicated by a significant shift in the internal structure of the data, the model's new structure is captured through the definition of a new context. Intuitively, this is achieved using causal meta-structures across the components of the model.

Pearl and Halpern (Pearl 2000; Halpern & Pearl 2001) propose so-called *structural equations* to describe causal relations. Following their lead, we employ structural equations in the definition of context. An additional benefit of defining context using causality enables context merging to be done using causal reasoning. Recall, in the burglary alarm example, the situation when we want to merge the models corresponding to San Diego, Los Angeles, and San Francisco. It is possible to reason that there must be another cause for the alarm to go off in these three cities, because we observe that the frequency of a burglary being associated with a triggered alarm is lower. As a result, we identify the new unknown cause as *foo*, and later as *Earthquake-Zone*.

Our Approach: The Context-Oriented Stochastic System

The contextual mechanism discussed earlier is incorporated in our failure driven probabilistic modeling system as a top layer that controls knowledge-based model construction (KBMC). As can be seen in figure 1, the architecture of the system looks like two concentric loops. The inner loop represents the execution sequence in the usual case of stationary data: a knowledge base (KB) is used to construct a network using a KBMC algorithm. As new data are obtained, we adjust the parameters of the network to incorporate these new data with parameter fitting algorithms such as expectation maximization (EM). Note that we can successfully update the network (and, consequently, the KB) if the data do not significantly deviate from some stable distribution.

When the new data change significantly and we are not successful in updating the network, the system's execution shifts to the outer loop. The contextual mechanism then determines the causal structure of the new data, which in turn constrains the choice of rules from the KB used by the KBMC algorithm. Note that the world and the user do not directly control the KB, but rather influence it via the current network.

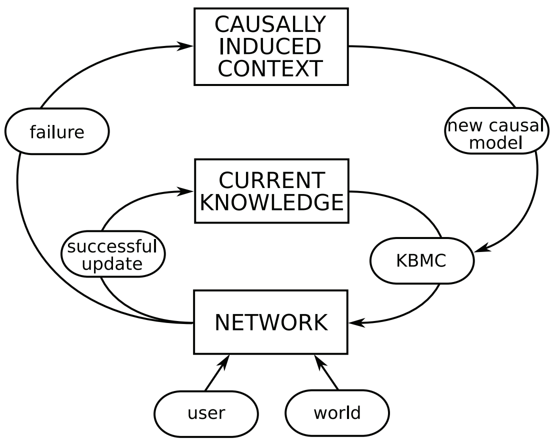

Figure 1: A schematic illustration of the architecture of the failure-driven dynamic probabilistic modeling system that uses two levels of learning (two loops) and three layers of control.

A more formal description of our prototype system is reflected in the following pseudo-code:

```
1 KB=InitKB, Context=InitContext
2 (Net,Map)=_kbmc(KB,Context,Queries)
3 repeat_until(no data received)
3.1 Data=_receive()
3.2 if(_no_failure(Net,Data)) then
3.2.1 Net=_get_updated_net(Net,Data)
3.2.2 KB=_get_updated_kb(KB,Map,Net)
3.3 else
3.3.1 Context=_get_new_context(Net,Data)
3.3.2 SimilarModel=
      _get_similar_model(Context,KB)
3.3.3 if(_is_null(SimilarModel)) then
3.3.3.1 (Net,Map)=
        _kbmc(KB,Context,Queries)
3.3.4 else
3.3.4.1 (Net,Map)=SimilarContext
3.3.5 Net=_get_updated_net(Net,Data)
3.3.6 KB=_get_updated_kb(KB,Map,Net)
```

Table 1: The prototype of a context-based, failure-driven, and event-oriented stochastic modeling system.

Step 1 of the code is the initialization. `InitContext` and `InitKB` are supplied by a user in the beginning of the execution. In step 2, the initial network is created. We use a knowledge-based model construction approach represented by the function `_kbmc()` to build the network. The function `_kbmc()` takes three arguments – a knowledge base (`KB`), a context as a current causal structure of the data (`Context`), and queries (`Queries`), that are provided by the user – and returns a tuple of a network (`Net`) and a one-to-one mapping (`Map`) between the network and the KB. The mapping `Map` updates the KB after parameter fitting on the network, since the updated parameters of the network can be easily located in the KB. Note that the function `_kbmc()` takes a context as an argument, hence it selectively chooses the relevant rules from the KB for model construction. This is one of the major differences of the proposed system from our earlier stochastic language, Loopy Logic (Pless 2003). The rest of the program is described in step 3, which is a loop that iterates each time new data (`NewData`) are obtained.

In step 3.1, the new data are received through the *ports* of the network. Steps 3.2 and 3.3 represent the switch between the stationary and non-stationary execution scenarios of the system. The predicate `_no_failure()` checks whether the current network fails to fit the new data. If the predicate is true (the network adequately captures the new data), then we perform parameter fitting (step 3.2.1) and update the distributions of the KB (step 3.2.2). Note that in step 3.2.2 we use the correspondence (`Map`) between the KB and the updated network to carry the new distributions over to the KB.

Steps 3.3.1-6 capture the situation when the network does not fit the new data. The system attempts to determine a new context by identifying the new causal structure based on the current network (`Net`) and the new data (`NewData`) (see step 3.3.1 of the pseudo-code). After a new context is determined, we check if there is already a model that has a causal structure similar to the new one. If there are no similar models, the new network is constructed, the parameters of the network are estimated, and the resulting distributions are transferred into the KB (steps 3.3.5 – 3.3.6).

There are several specific issues that need further clarification in our pseudo-code for it to be successfully implemented. First, we must rigorously define the notion of a context using causal models, and then specify what it means for a system to break down. Second, precise methods for learning causal structures from the data are also needed. Further, we can see that the `_no_failure()` predicate also depends on the ability to learn causal models from data. Finally, note that the algorithm implicitly assumes high efficiency of the update functions (`_get_updated_net()`, `_get_updated_kb()`), because of the high amount of consecutive inner loops in the execution of the system.

Conclusions and Future Work

In this paper we identify two major advantages of characterizing context and demonstrate these advantages in our logic-based probabilistic modeling system. First, modeling context allows the system to reduce the size of the resulting graphical model. This reduction improves the efficiency of inferencing across the model, which is especially crucial in modeling dynamically changing systems. The second advantage of modeling context is the ability to handle non-stationarity in the data. Intuitively, we envision a complex non-stationary stochastic process that underlies data as a composition of simpler stationary or slowly evolving processes. Modeling context represents the switching between the snapshot models that correspond to these simpler stochastic processes.

Further, we propose a strong connection between causal relationships and context. We feel that contextualized mechanisms that support causality can capture and simplify the regularities underlying sets of data. Our decision to create "causality organized contexts" is influenced and guided by Pearl and Halpern's investigations (Pearl 2000; Halpern & Pearl 2001) of causal models. The inherent structuring of causal relationships as well as their modularity and stability make them good candidates for specifying context. Additionally, these studies suggest that a satisfactory explanation of many events must acknowledge their causes; hence our probabilistic graphical model identifies and utilizes causal structure, since our model is an attempt to "explain" the events as supported by the data.

In this paper we also describe how our design of context mechanisms is inspired by recent research in developmental learning. On this view, context is used to filter out irrelevant information while reorganizing a current model; similarly, humans ignore redundant and irrelevant information when updating or rebuilding their internal knowledge of situations. We consider the psychological principles of learning through assimilation and accommodation together with the notion of context and propose our architecture for failure-driven and event-oriented stochastic modeling.

This paper only begins to address the deep issues related to context specification and its application in the design and use of stochastic models. There are a number of various research directions for further investigation, for instance, to consider the possibility of combining the original notion of context proposed by (Ngo & Haddawy 1997) for a *set* of systems, with causal models that correspond to a *specific* system. Additionally, we need to further develop computational ways to implement Pearl and Halpern's notion of causality as well as to investigate how causal structures can be learned from data. Although our Scheme-based prototype has been developed for several simple models, our current focus of research is to analyze weather data taken from multiple distributed sensors supplied by the US Air Force.

Acknowledgements

We would like to thank the National Science foundation (115-9800929, INT-9900485) and the US Navy (SBIR N00T001, STTR N0421-03-C-0041) for their earlier support in the design of our Turing-complete and first-order stochastic logic language, Loopy Logic. The research presented in this paper is funded under an Air Force Research Laboratory SBIR contract (FA8750-06-C0016). We would like to acknowledge Martin Walter at AFRL Rome for his supportive management and direction. We also thank Daniel Pless, Chayan Chakrabarti, and Roshan Rammohan for numerous thought-provoking discussions.

References

Everson, R., and Roberts, S. 1999. Non-stationary independent component analysis. 503–508. Proc. of 8th Intl. Conf. on Artificial Neural Networks.

Friedman, N.; Getoor, L.; Koller, D.; and Pfeffer, A. 1999. Learning Probabilistic Relational Models. 1300–1307. Proc. of 16th Intl. Joint Conf. on AI (IJCAI).

Gopnik, A.; Glymour, C.; Sobel, D. M.; Schulz, L. E.; Kushnir, T.; and Danks, D. 2004. A theory of causal learning in children: Causal maps and Bayes nets. *Psychological Review* 111(1):3–32.

Granott, N.; Fischer, K. W.; and Parziale, J. 2002. Bridging to the unknown: a transition mechanism in learning and development. *Microdevelopment: transition processes in development and learning*.

Haddawy, P. 1994. Generating Bayesian Networks from Probability Logic Knowledge Bases. 262–269. Morgan Kaufmann. Proc. of 10th Conf. on Uncertainty in AI.

Halpern, J., and Pearl, J. 2001. Causes and Explanations: A Structural-Model Approach — Part 1: Causes. 194–202. San Francisco, CA: Morgan Kaufmann. Proc. of 17th Conf. on Uncertainty in AI (UAI-01).

Kersting, K., and DeRaedt, L. 2000. Bayesian Logic Programs. Technical Report 00151. Albert-Ludwigs University at Freiburg.

Lee, T.-W.; Lewicki, M. S.; and Sejnowski, T. J. 2000. ICA mixture models for unsupervised classification of non-Gaussian sources and automatic context switching in blind signal separation. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 22(10):1078–1089.

Luger, G. F. 2005. *Artificial intelligence: Structures and Strategies for Complex Problem Solving*. Addison-Wesley.

Ngo, L., and Haddawy, P. 1997. Answering queries from context-sensitive probabilistic knowledge bases. *Theoretical Computer Science* 171(1–2):147–177.

Ngo, L.; Haddawy, P.; Krieger, R. A.; and Helwig, J. 1997. Efficient Temporal Probabilistic Reasoning via Context-Sensitive Model Construction. *Computers in Biology and Medicine* 27(5):453–476.

Pearl, J. 2000. *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Pham, D.-T., and Cardoso, J.-F. 2001. Blind separation of instantaneous mixtures of non stationary sources. *IEEE Trans. Signal Processing* 49(9):1837–1848.

Piaget, J. 1983. Piaget's theory. *Handbook of Child Psychology* 1.

Pless, D. J. 2003. *First-Order Stochastic Modeling*. Ph.D. Dissertation. University of New Mexico.

Richardson, M., and Domingos, P. 2006. Markov Logic Networks. *Machine Learning* 62(1–2):107–136.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Wellman, M. P.; Breese, J. S.; and Goldman, R. P. 1992. From knowledge bases to decision models. *Knowledge Engineering Review* 7(1):35–53.