

A hybrid metareasoning architecture combining case-based reasoning and Bayesian networks

Tor Gunnar HOUELAND^a, Tore BRULAND^b, Agnar AAMODT^a and Helge LANGSETH^a

^a *Department of Computer and Information Science*

^b *Department of Cancer Research and Molecular Medicine
Norwegian University of Science and Technology, Norway*

Abstract. In complex domains, a single type of knowledge and reasoning method is often not sufficient for a decision support system to address the variety of tasks a user performs. It is often necessary to determine which reasoning method would be the most appropriate for each task, and a combination of different methods has often shown the best results. We examine the strengths and weaknesses of two complementary reasoning methods, case-based reasoning and Bayesian networks, and discuss how they can be combined to form a more robust and better-performing hybrid. We present a metareasoning system for automatically selecting the most viable reasoning method for a particular input query at runtime, for a sustained learning scenario where an expert provides initial domain knowledge through modeling illustrative cases.

Keywords. metareasoning, hybrid reasoning systems, case-based reasoning, bayesian networks

1. Introduction

Automated computer reasoning is a challenging field. There are a multitude of different reasoning methods available, with different strengths, assumptions and learning biases. The accuracy of a method depends on the specific domain and can be difficult to predict in general, while a human expert can often determine whether a reasoning method will be either particularly useful or inappropriate for a specific problem. Recently, there has been a renewed interest in reasoning about reasoning, i.e. *metareasoning*, within automated computer reasoning systems [1]. In metareasoning there are generally two important aspects: introspection and meta-level control. Introspection refers to a system's ability to gather information about its own (object-level) reasoning processes, e.g. computational performance data, and records of the steps taken during reasoning. Meta-level control aims to increase the quality of the combined reasoning system by deciding what type of object-level reasoning to perform, based on the information collected from introspection. Meta-level control can for example be used to balance a system's limited resources between computations and external (ground-level) actions that affect the world, and dynamically adapting the specific computations and actions that are performed.

When using reasoning methods for real life applications, there are two different categories of uncertainty that are usually both present to some degree: aleatoric uncertainty and epistemic uncertainty. Aleatoric uncertainty refers to the general stochastic nature of the domain, and refers to events having a certain probability of happening given the right conditions. This is in contrast to e.g. axiomatic logic where a valid implication means that a conclusion *always* occurs given the premise. On the other hand, epistemic uncertainty is a general lack of knowledge. This refers to our incomplete understanding of the domain, e.g. inaccurate beliefs about the causal relationships and models that do not include all significant effects. To address these different types of uncertainty, we are examining hybrid reasoning systems that combine reasoning methods that handle each of the uncertainty categories well. In the research presented in this paper the two methods we study are Bayesian networks (BN) for aleatoric uncertainty and case-based reasoning (CBR) for epistemic uncertainty.

We present a reasoning architecture that uses metareasoning to automatically detect at runtime whether CBR or BN should be used to solve a new problem query, given the system's current state of uncertainty. The approach is based on the identified characteristics of CBR and BN regarding uncertainty, and the decision for each problem query is based on the empirical results of using CBR and BN to solve other recently seen input queries. This approach applies to domains with both aleatoric and epistemic uncertainty, where there is a human expert who can provide a set of illustrative cases to initially model the domain, and the system receives feedback about the actual outcomes of using its predicted solutions to the problems it solves.

In the next two sections we summarize the essential properties of CBR and BN, with an emphasis on their complementary strengths and weaknesses. Section 4 discusses how hybrid combinations can be made from CBR and BN methods, and section 5 examines how a meta-reasoner can be used to create such hybrid reasoning systems automatically. A discussion and concluding remarks end the paper.

2. Case-based reasoning characteristics

In case-based reasoning (CBR) [2,3] a computer model is built up of a set of concrete past situations, called cases, stored in a knowledge base referred to as a case base. CBR is a method for problem solving that also incorporates learning from problems just solved. The core knowledge component is the case, which in its basic form has two parts, a problem description part and a problem solution part. The notion of a "problem" should here be viewed in a general sense, i.e. as a state for which some inference or action is called for that leads to a "solution", i.e. some result. Hence, a problem may be interpreting a situation, posing a question, assessing a feature value, etc., as well as solving a larger diagnostic, planning, or scheduling problem. The problem description part constitutes the set of input features to the reasoning process, while the problem solution part is the system's suggested solution to the problem. A third part is often added: outcome, i.e. the result after having applied the solution to the problem. In its simplest form a single case is represented as a list of attribute-value pairs, but more complex representations are also frequently adopted, such as hierarchical description structures.

Reasoning methods of similarity assessment, pattern recognition, and analogical mapping, rather than theory-driven methods, operate over this case base. A four-step

cycle describes the CBR problem solving and learning process (Figure 1): In the first step, Retrieve, an input case is matched against the cases in the case base, and the best matching case is retrieved. In the next step, Reuse, the solution part of the best matching case is used to construct a solution to the input case. The two final steps enable the system to learn from this problem solving session and update its case base accordingly. The proposed solution is first, in the Revise step, evaluated in order to assess its quality. This may be done by a human expert, by applying the solution to the problem, or by some additional computer model outside the CBR system - such as a simulation system. After evaluation the solution may be adjusted, and the case then enters the Retain step, in which a new case may be constructed, the new case may be merged with an existing case, or only the index structure may be updated.

The above description has focused solely on the case knowledge, i.e. the cases in the case base. As Figure 1 shows, a CBR system's knowledge base may also contain general domain knowledge, for example a set of rules, a taxonomy, an ontology, or another multi-relational domain model. Although all subareas of CBR have been subjects of active research, up to now the Retrieve step, and in particular similarity assessment, has been the most focused [4]. It is important to note that case retrieval is a partial matching process, in which the best partial match is targeted. The definition of a best match is contained in the similarity assessment procedure. In the simplest CBR method, also referred to as an instance-based method, a case is represented by a flat attribute-value list, and no general domain knowledge is used. The degree of similarity between two cases is calculated by summing up the number of identical attribute-value pairs, if we are dealing with symbolic feature values, and by computing the numerical differences if the attributes have numerical values. When general domain knowledge is used in the similarity assessment method, a similarity metric can take different forms depending on how the knowledge is used. An example expression is given in Equation 1:

$$sim(C_{IN}, C_{RE}) = \frac{\sum_{i=1}^n \sum_{j=1}^m sim(f_i, f_j) \times w(f_j)}{\sum_{j=1}^m w(f_j)} \quad (1)$$

Here each feature in a stored case is given a weight (w) that reflects its relevance for that particular case. General domain knowledge is used to determine the degree of similarity between any two features, independent of their position in the case or whether they are superficially similar or not. In the above expression, C_{IN} and C_{RE} are the input and retrieved cases, and n and m are the number of findings in each of them, respectively. f_i is the i th finding in C_{IN} , f_j the j th finding in C_{RE} , and $w(f_j)$ the weight of that finding.

It should be noted that case-based reasoning has been strongly motivated by cognitive science research [3] and the similar abilities of the human mind [5]. For example, in their daily practice clinicians make use of personal specific experiences gained through daily work [6]. Past patient cases provide a level of specificity that focuses on single patients rather than generalized principles, and are easier to describe than generalized principles and dependencies. When modeling decision knowledge in a computer system, an initial case base is therefore often easier to come up with than a generalized model, and can often be transferred from written documentation such as patient journals.

An important property of the CBR method is its "lazy" approach to modeling and learning. Lazy in this sense means that CBR does not eagerly build generalizations of its experiences in the learning phase, but store them as specific instances, and wait un-

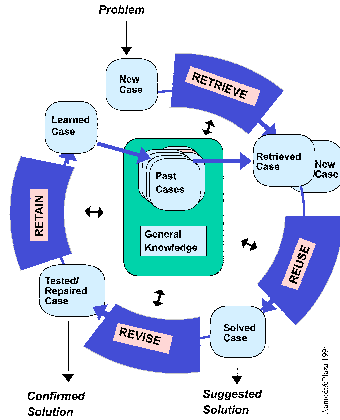


Figure 1. The CBR cycle.

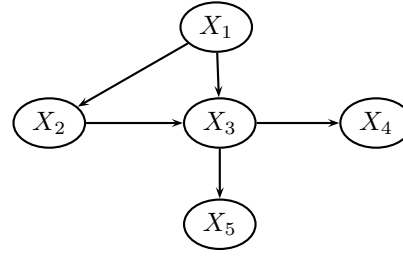


Figure 2. An example BN over the nodes $\{X_1, \dots, X_5\}$. Only the qualitative part of the BN is shown.

til the problem solving phase to determine in what way they should be used. This also makes CBR suitable as a complementary method to generalization-based models: CBR uses localized models to model a domain, and in that sense is able to fill in holes in a knowledge base in a straightforward manner, i.e. just by adding another case. Hence, epistemic uncertainty can easily be captured by a CBR model, whereas aleatoric uncertainty, for which a known probability distribution over a set of states is assumed, favors a generalized knowledge model, such as a BN model.

3. Bayesian networks characteristics

A Bayesian Network (BN), [7,8], is a compact representation of a multivariate statistical distribution function. A BN encodes the probability density function governing a set of random variables $\{X_1, \dots, X_n\}$ by specifying a set of conditional independence statements together with a set of conditional probability functions. More specifically, a BN consists of a qualitative part, a *directed acyclic graph* where the nodes mirror the random variables X_i , and a quantitative part, the set of conditional probability functions. An example of a BN over the variables $\{X_1, \dots, X_5\}$ is shown in Figure 2, where only the qualitative part is given. We call the nodes with outgoing edges pointing into a specific node the *parents* of that node, and say that X_j is a *descendant* of X_i if and only if there exists a directed path from X_i to X_j in the graph. In Figure 2 X_1 and X_2 are the parents of X_3 , written $\text{pa}(X_3) = \{X_1, X_2\}$ for short. Furthermore, $\text{pa}(X_4) = \{X_3\}$ and since there are no directed path from X_4 to any of the other nodes, the descendants of X_4 are given by the empty set and, accordingly, its non-descendants are $\{X_1, X_2, X_3, X_5\}$.

The edges of the graph represents the assertion that a variable is conditionally independent of its non-descendants in the graph given its parents in the same graph. Other conditional independence statements can be read off the graph by using the rules of *d-separation* [7]. The graph in Figure 2 does for instance assert that for all distributions compatible with it, we have that X_4 is conditionally independent of $\{X_1, X_2, X_5\}$ when conditioned on $\{X_3\}$.

When it comes to the quantitative part, each variable is described by the conditional probability function of that variable *given the parents* in the graph, i.e., the collection of conditional probability functions $\{f(x_i|\text{pa}(x_i))\}_{i=1}^n$ is required. The underlying assumptions of conditional independence encoded in the graph allow us to calculate the joint probability function as

$$f(x_1, \dots, x_n) = \prod_{i=1}^n f(x_i|\text{pa}(x_i)). \quad (2)$$

A BN is a model of general domain knowledge. One of the main arguments for using BNs over other model-based reasoners is the framework's sound inference engine. One can calculate the belief in any query given any evidence (e.g. the probability of an object being from a specific class given the attributes describing that object). Mathematically, we say that we can calculate arbitrary marginal distributions, $f(x_i, x_j, x_k)$ as well as arbitrary conditional distributions, $f(x_i, x_j|x_k, x_\ell)$. The inference engine utilises the conditional independence statements asserted by the model rather efficiently, which makes BNs well suited for modelling complex systems. Models over thousands of variables are not uncommon.

The qualitative part of the BN (the graph) has an intuitive interpretation as a model of causal influence. Although this interpretation is not necessarily entirely correct, it is helpful when the BN structure is to be elicited from experts. Furthermore, it can also be defended if some additional assumptions are made [9]. These assumptions are often acceptable in real-life situations, in particular if cause-effect relations are the most important pieces of knowledge to encode. In such cases, the building of a BN structure is a viable (although sometimes rather time-consuming) task to perform by domain experts.

To elicit the quantitative part from experts, one must acquire all conditional distributions ($\{f(x_i|\text{pa}(x_i))\}_{i=1}^n$ in Equation 2). Once again the causal interpretation can come in as a handy tool, but the quantification of the probability distributions is commonly regarded as the most challenging phase of the BN modelling process. Empiric studies (e.g., [10]) have shown that the results of a query are rather insensitive to small variations in the definition of the conditional distributions, but rather sensitive to variations in the model structure.

As a modelling framework thoroughly grounded in probability theory, there are efficient methods for learning BNs from data. Methods for batch learning of the quantitative part of the BN from data date back to the work of [11], see also [12]. Methods for batch learning the qualitative part from data was pioneered by [13], see also [14,15]. Incremental techniques have also been explored, see e.g. [16].

The BN theory mentioned so far assumes that all the variables in the domain are *categorical*. Defining extensions of the BN framework to also support models that contain both discrete and continuous variables is currently a lively research area, but not yet fully matured. The common approach is to translate continuous variables to discrete interval-variables, then consider all discrete variables (including categorical, ordinal, and interval variables) as if they were categorical. This process inevitably leads to a loss of precision.

Another property of the BN modelling framework worth mentioning, is that as it uses probability distributions to model a domain, it is viable to model aleatoric uncertainty. Epistemic uncertainty cannot easily be captured naturally by a BN model, and if a decision maker is faced with this type of uncertainty, one may consider alternative modelling frameworks, such as CBR.

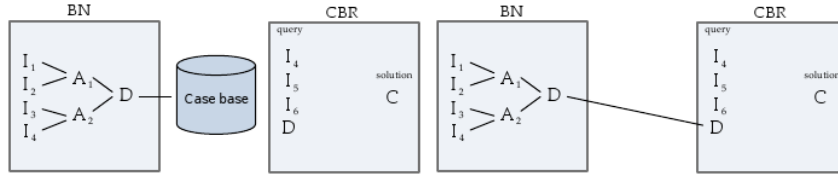


Figure 3. BN-CBR-1 Architecture

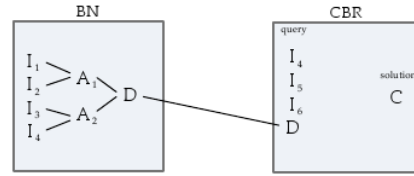


Figure 4. BN-CBR-2 Architecture

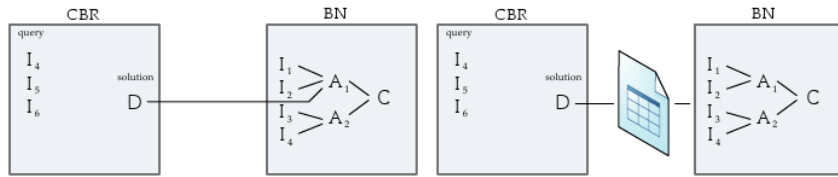


Figure 5. CBR-BN-1 Architecture

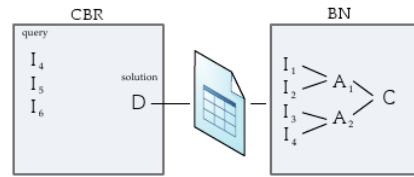


Figure 6. CBR-BN-2 Architecture

4. Hybrid CBR-BN combinations

Before we build a hybrid with BN and CBR, we must determine how the systems should be combined. We split the reasoning about the problem data according to their characteristics. For example, BN is suitable for global models with uncertainty. CBR on the other hand is suitable for local models with a lot of details and a good similarity function, and for easily representing situations to avoid as exceptions.

In a recent study we identified four basic sequential combinations for integrating BN and CBR reasoning processes [17]. These combinations are depicted in Figures 3 to 6, and use the following variable types: input variables I_i that are taken from the problem description, mediating variables A_j that represent nodes in the BN model, an intermediate variable D that is calculated from the first reasoner and transferred to the second, and the final classification variable C that is computed by the second reasoner. There are two fundamental BN-CBR sequences and two CBR-BN sequences. Figure 3 shows the BN-CBR-1 sequence, where the BN component is used to identify and activates the most relevant cases in the case base, and then the CBR component solves the problem by using only the activated cases as its case base. This architecture was used by Gomes [18] and early work in our research group [19].

Figure 4 shows the BN-CBR-2 sequence. The BN component calculates an intermediate variable D from the input variables, which contains additional useful information for retrieving cases and is transferred to the CBR system. The CBR component uses the problem's input variables together with this additional variable D to find a similar case to solve the problem. A similar architecture was used by early work in our research group [19].

The CBR-BN-1 sequence is shown in Figure 5, which follows the same principle as BN-CBR-2 but adapted to transfer from BN to CBR. The CBR component finds a similar case from the input variables and transfers an intermediate variable D to the BN system. The BN system calculates the class C from the given input variables and variable D .

CBR-BN-2 is the last sequence and it is shown in Figure 6. In this combination the case base contains different BN models, and CBR is first used to find a similar case which has a specific BN model as its solution. This BN model is then loaded and used to

calculate the class C based on the input variables. This architecture was used by Pavón et al. [20] and Louvieris et al. [21].

One or more of these basic sequences can also be put together into a larger hybrid system. As an illustrative example, let us assume that the task is to recommend a suitable place of study for a person, using CBR-BN-1 and BN-CBR-2 to create a CBR-BN-CBR hybrid. CBR is first used to detect exceptions, such as the student already being enrolled elsewhere, or that the student has cheated and is not qualified. If one of these exceptions is found the reasoning stops and no new place of study is found. When there are no exceptions, the next step is using a BN to calculate a general place of study. If the student is good with numbers then a school of management is suggested, or if the student has good grades in natural science then an engineering program from a college is suggested. CBR is then used again to take the suggestion and find a concrete program at a particular higher education facility, perhaps with further constraints based on the student's grades. In this way the first CBR system filters out persons, the BN system calculates a type of school, and the second CBR system finds the most suitable school or university.

5. Automatic combination using metareasoning

A practical problem for creating hybrid systems is that there are many different architectures to choose from, and it is time-consuming to build a reasoning system to examine and evaluate even just one of the possible combinations. This is partially alleviated by following sound software engineering concepts, and providing modular implementations with abstractions that encapsulate the executable code in components that can be reused. The same BN implementation can be used both for a system that only uses a BN to solve problems directly, and for a combination that first uses CBR to retrieve a problem-specific BN to be used. In this way a new system can be specified easily, by just changing which component sends its output to the other, or by changing a parameter such as which similarity measure to use.

In our metareasoning architecture algorithms that are implemented as reusable components in this way are used as building blocks for reasoning during run-time. They can be viewed as a form of language for specifying reasoning systems at the meta-level. This connection to (formal) languages is not accidental [22]. A *metalanguage* is a language used to discuss a language, and such metalanguages have been formally explored in logic and linguistics. The act of introspection or reflection generally corresponds to an *upward shift* from one level to a meta-level, and for languages this means *referring* to a word instead of *using* a word in written text. For a reasoning system it means referring to a reasoning process through a form of possibly implicit naming or labeling, instead of directly referring to the problems that are solved. In an earlier paper we introduced a vocabulary to describe problems and reasoning methods for such a meta-level reasoning system [23].

A popular direction for using metareasoning in CBR systems is to use traces of object-level reasoner behavior as the meta-data [24]. These traces list the steps of the process the object-level reasoner performs, and then a form of meta-level control is used to detect and correct problems. In contrast to this error-correcting approach, we focus on using computational performance data as the meta-data. Based on their complementarity strength related to the two categories of uncertainty, our metareasoning architecture

combines CBR and BN, by detecting at runtime which algorithm is producing the best solutions. We focus on a class of domains containing both aleatoric and epistemic uncertainty. The domain has stochastic elements but human experts use experience and remember previous incidents and consider them when solving new problems, and do not think about the domain as conditional probability distributions. For such domains, a human expert provides an initial set of illustrative cases for the CBR component, which attempts to cover the range of problem categories the expert usually considers in their work.

Additionally, our architecture is designed to support sustained learning, where the system continues to learn from new problem solving experiences while it is running, by evaluating the proposed solution of a problem based on its outcome. In the CBR component this is the purpose of the retain step, which adds new cases to the case base and generalizes and improves existing cases. On the other hand, the underlying aleatoric uncertainty in the domain means that a BN system can perform well if it is trained on statistically representative training data.

For a new problem query, our system chooses either the CBR or BN component based on their empirical performance on the most recent problems the system has solved. The underlying assumption in this approach to hybridization is that a reasoning method which performs well on a certain task will continue to do so in the future, and that changes in classification accuracy when learning occur gradually.

6. Discussion

When our assumptions are correct, the characteristics of such a domain means that a CBR system created using expert knowledge will quickly be able to produce fairly good results, by using a simplified model that's a direct expression of the expert's methodology and understanding of the domain. However, for the way the CBR system is used in this scenario, the sustained learning aspect will be limited. This is because the cases learned from the domain will not have the additional richness of an expert trying to communicate the significant effects in the domain, they will just be the problems the system happens to encounter. The sustained learning aspect will therefore mostly be accomplished by filling in data for which the expert didn't already provide guidance, while the performance on already covered cases will stay but not increase significantly.

On the other hand, the BN reasoner will start out from a very disadvantaged position. BN's statistical learning is founded on learning the probabilistic relations in the domain from training data, and using these learned relations to solve problems. A set of illustrative cases provided by an expert will typically not have the statistical properties as the real problems to be solved. However, the BN will be able to take full advantage of the new problems the system encounters. These problems will naturally be drawn from the same distribution the system has to solve, because they *are* the problems the system encounters. Because of how the BN does not receive useful information from the expert in this scenario, the BN component will initially perform poorly, but learn well from the problem solving experiences that contain the real outcome from using the solutions to address the problems in the real world. In our scenario the BN component uses a fundamentally different approach to solving problems in the domain, which will improve as new problems are solved and can go beyond the human expert's and the CBR component's approach.

A benefit of using an automatic metareasoning approach to create these combinations is that the computations can be dynamically adapted. For our scenario, this means that we empirically detect when there is enough representative training data for the BN to generally produce better results than the modeled CBR knowledge base. Even when our qualitative assumptions hold, we cannot reliably predict in general how much training data would be needed for the BN to be preferable, which depends on the exact characteristics of the specific task being performed.

The metareasoning layer also adds additional learning challenges to the combined reasoning system. If only the outcome of the final chosen solution can be retrieved, then there is an additional exploration/exploitation aspect connected to choosing whether to update the performance data for the assumed "worst" method, or simply using the "best" method to get the best results. The meta-level learning also adds an additional learning bias about the reasoning methods. For an especially peculiar domain where the relative performance of CBR and BN methods fluctuated wildly as they learned, the metareasoning layer would not be able to learn from its performance measurement meta-data, and this approach would not work.

Our metareasoning architecture can be extended in numerous ways, using both CBR and BN methods and possibly including other reasoning methods as well. The presented architecture matches the identified strengths and weaknesses of CBR and BN regarding types of domain uncertainty, but it is possible to add additional improvements on top. One approach is to also include hybrid sequential combinations as base reasoning components, and choosing the hybrid combination sequence that produces the best empirical results. Another approach is to use a form of CBR at the meta-level: instead of evaluating the most recent performance, we can evaluate the most recent performance *on similar problems*, using an additional meta-level similarity measure that is suitable for this purpose.

The CBR and BN methods used in the metareasoning hybrid approach can also provide extended functionality in addition to solving problems. As two notable examples, the CBR approach is particularly good at adding exceptions that can be used to retrieve short-cut solutions instead of performing the normal reasoning, and BN can answer many other questions about the statistical properties of the domain, such as the probability for every possible classification as a solution, or the chance of a problem having particular attributes given the observed outcome.

7. Acknowledgements

The reported research is partially funded by the TLCPC project, Norwegian Research Foundation under contract no. NFR-183362.

References

- [1] M. T. Cox and A. Raja, "Metareasoning: A manifesto," tech. rep., BBN TM-2028, BBN Technologies, 2007.
- [2] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Communications*, vol. 7, pp. 39–59, March 1994.
- [3] J. Kolodner, *Case-based reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

- [4] R. L. de Mántaras, D. McSherry, D. G. Bridge, D. B. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T. Cox, K. D. Forbus, M. T. Keane, A. Aamodt, and I. D. Watson, "Retrieval, reuse, revision and retention in case-based reasoning," *Knowledge Eng. Review*, vol. 20, no. 3, pp. 215–240, 2005.
- [5] R. C. Schank, *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. New York, NY, USA: Cambridge University Press, 1983.
- [6] V. L. Patel, D. R. Kaufman, and J. F. Arocha, "Emerging paradigms of cognition in medical decision-making," *J. of Biomedical Informatics*, vol. 35, pp. 52–75, February 2002.
- [7] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA.: Morgan Kaufmann Publishers, 1988.
- [8] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. Berlin, Germany: Springer-Verlag, 2007.
- [9] J. Pearl, *Causality – Models, Reasoning, and Inference*. Cambridge, UK: Cambridge University Press, 2000.
- [10] M. Henrion, M. Pradhan, B. Del Favero, K. Huang, G. Provan, and P. ORorke, "Why is diagnosis using belief networks insensitive to imprecision in probabilities?," in *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, (San Mateo, CA.), pp. 307–314, Morgan Kaufmann Publishers, 1996.
- [11] D. J. Spiegelhalter and S. L. Lauritzen, "Sequential updating of conditional probabilities on directed graphical structures," *Networks*, vol. 20, pp. 579–605, 1990.
- [12] S. L. Lauritzen, "The EM-algorithm for graphical association models with missing data," *Computational Statistics and Data Analysis*, vol. 19, pp. 191–201, 1995.
- [13] G. F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, pp. 309–347, 1992.
- [14] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," *Machine Learning*, vol. 20, pp. 197–243, 1995.
- [15] N. Friedman, "The Bayesian structural EM algorithm," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, (San Francisco, CA.), pp. 129–138, Morgan Kaufmann Publishers, 1998.
- [16] J. R. Alcobé, *Incremental Methods for Bayesian Network Structure Learning*. PhD thesis, Universitat Politècnica de Catalunya, 2004.
- [17] T. Bruland, A. Aamodt, and H. Langseth, "Architectures Integrating Case-Based Reasoning and Bayesian Networks for Clinical Decision Support," in *Intelligent Information Processing V* (Z. Shi, S. Vadera, A. Aamodt, and D. Leake, eds.), pp. 82–91, Springer, 2010.
- [18] P. Gomes, "Software design retrieval using Bayesian Networks and WordNet," *Lecture Notes in Computer Science*, pp. 184–197, 2004.
- [19] A. Aamodt and H. Langseth, "Integrating Bayesian Networks into Knowledge-Intensive CBR," in *AAAI Workshop on Case-Based Reasoning Integrations*, 1998.
- [20] R. Pavón, F. Díaz, R. Laza, and V. Luzón, "Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study," *Expert Systems With Applications*, vol. 36, no. 2P2, pp. 3407–3420, 2009.
- [21] P. Louvieris, A. Gregoriades, and W. Garn, "Assessing critical success factors for military decision support," *Expert Systems with Applications*, 2010.
- [22] S. Costantini, "Meta-reasoning: A survey," in *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, (London, UK), pp. 253–288, Springer-Verlag, 2002.
- [23] T. G. Houeland and A. Aamodt, "An introspective component-based approach for meta-level reasoning in clinical decision-support systems," in *Proceedings of the First Norwegian Artificial Intelligence Symposium (NAIS'09)*, pp. 121–132, Tapir Forlag, 2009.
- [24] M. T. Cox, K. Eiselt, J. Kolodner, N. Nersessian, M. Recker, and T. Simon, "Introspective multistrategy learning: On the construction of learning strategies," *Artificial Intelligence*, vol. 112, pp. 1–55, 1999.