Soft Evidential Update for Communication in Multiagent Systems and the Big Clique Algorithm

Young-Gyun Kim and Marco Valtorta[1]
Department of Computer Science and Engineering
University of South Carolina

October 31, 2000

[1]This report is based on a Chapter in Young-Gyun Kim's upcoming dissertation and on [VKV]. This work has been sponsored in part by the Defense Advanced Research Agency (DARPA) and the Air Force Research Laboratory under agreement number F30602-99-2-0513.

Abstract

We address the problem of updating a probability distribution represented by a Bayesian network upon presentation of soft evidence. Our motivation is the desire to let agent communicate with each other by exchanging beliefs represented by probability distributions, as in the Agent Encapsulated Bayesian Network (AEBN) model. We explain how to model soft evidence with *observation variables* in a way that insures that the independence structure of the model is maintained even upon receipt of soft evidence. We describe the *big clique algorithm*, which extends the celebrated junction tree algorithm to soft evidential update, and BC-Hugin, its implementation in Java and the Hugin API.

Chapter 1

Motivation and Preliminaries

We address the problem of updating a probability distribution represented by a Bayesian net upon the presentation of soft evidence. We call this the problem of soft evidential update.

The motivation for this work is our desire to let agents that use probabilistic models (and especially Bayesian nets) communicate with each other by exchanging beliefs.

While this is not the focus of this report, we need to describe briefly our agent model, which is called the Agent-Encapsulated Bayesian Network (AEBN) model, originally due to Bloemeke [Blo98]. Each agent in an AEBN model uses as its model of the world a single Bayesian network (which we also call an AEBN). The agents communicate via message passing. Each message is a distribution on variables shared between the individual networks.

The variables of each AEBN are divided into three groups: those about which other agents have better knowledge (*input set*), those that are only used within the agent (*local set*), and those of which the agent has the best knowledge, and which other agents may want (*output set*). The variables in the input set and the output set are shared, while those in the local set are not. An agent consumes (or subscribes to) zero or more variables in the input set and produces (or publishes) zero or more variables in the output set.

The mechanism for integrating the view of the other agents on a shared variable is to replace the agent's current belief in that variable with that of the communicating agent. When an agent receives a message from a publisher, it modifies the probabilities in its internal model, so that its local distribution either becomes consistent with the other agent's view or is inconsistent with it. In the rest of this report, we assume the former case, but the implementation described at the end of this report identifies inconsistency and warns the user appropriately.

Therefore, after updating using all evidence, we still require that all appropriate marginals of the updated distribution be equal to the evidence entered. The deservedly celebrated junction tree algorithm for probability update [LS88, SS90, Jen95, LJ97] was not designed to satisfy this requirement, and in fact it does not, as we will show in Section 1.2.

When a publisher makes a new observation, it sends a message to its subscribers. In turn, the subscribers adjust their internal view of the world and send their published values to their subscribers. Assuming that the graph of agent communication (which we simply call *agent graph* as in [Blo98]) is a DAG, equilibrium is reached, and a kind of global consistency is assured, because the belief in each shared variable is the same in every agent.

The restriction that one of the agents has oracular knowledge of a variable may seem excessive. However, it is permissible to have multiple views of a common variable. For example, in a multiagent system for interpretation, two agents may issue a report that corresponds to the same (unknown) physical quantity. Nothing prevents another agent from integrating the reports of these agents and effectively obtain a new (and possibly more accurate) view of the same underlying quantity. As another example, it is possible for a subscriber agent to model known reporting errors or biases of the publisher.

When the agent graph is not a tree, great care must be taken to deal with undirected cycles (loops). Such cycles lead to possible double counting of information, which is often known as the rumor problem. We do not address this important problem in this report, but cf. [Blo98, BS99, BV]. It is also possible to consider directed cycles (and in particular, the tight cycles resulting from bi-directional communication in agent graphs), by appropriately sequencing messages between agents. We do not address this extension further in this report, but the reader must be made aware of the very interesting and important work by Xiang on Multiply Sectioned Bayesian networks for related results [Xia96, XL00].

Evidence is a collection of findings on variables. A finding may be hard or soft. A *hard finding* specifies which value a variable is in. A *soft finding* specifies the probability distribution of a variable. *Hard evidence* is a collection of hard findings. *Soft evidence* is a collection of soft findings.

There are more general definitions of finding in the Bayesian network literature, but the one we just presented covers the most useful particular case [CDLS99, pp.93-94] and is the one supported in implementations of Bayesian network and decision network (influence diagram) shells. In [VKV], we generalize the definition of soft evidence in three ways. Firstly, we may extend the definition of finding to allow conditional distributions. Secondly, we may allow joint (and possibly, conditional) distributions on a collection of variables. Thirdly, we may allow distributions on arbitrary events (equivalently, arbitrary logic formulae).

1.1 Two simple examples for soft evidence

Here are two simple examples that illustrate the notion of soft evidence.

Suppose that the initial position and direction of an object are known precisely and that its acceleration is known to be zero. The position (P) of the object is then determined by its actual speed (S), according to some probabilistic law. This simple situation may be modeled by the Bayesian network in Figure 1.1, together with appropriate conditional probability tables for $P(S)$ and $P(P|S)$.

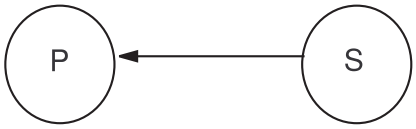

Figure 1.1: Position and speed.

Suppose that we cannot measure the speed precisely, but that we have uncertain information about it. We may think of E as the speed returned by a remote sensor. The sensor is embodied into a separate agent, whose task is to provide an estimate of speed. Since we live in the world of Bayesian networks, the estimate is a belief, i.e. a probability distribution over the possible values of speed.

We call this kind of evidence *soft evidence*. The new situation may be modeled by Figure 1.2, where the dashed line indicates that the evidence (E) about S is uncertain.

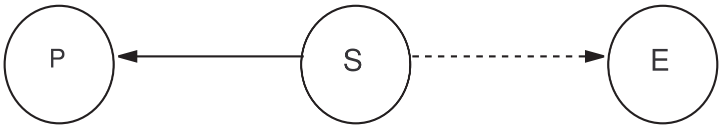

Figure 1.2: Position and speed with soft evidence about speed.

In our second example, position (P) depends on speed (S), again. We again have a separate sensor agent that provides us with soft evidence about the speed of the target. In contrast to the situation in the previous example, however, we now know that the state of the sensor (SS) that gives us the speed is affected by the weather (W). The situation is described in Figure 1.3.

Note that a finding (even when uncertain) about speed sets up a dependence between the weather and the target's position. In particular, if we knew the position and had evidence for the speed, we could tell the weather. Qualitatively, the argument goes like this: the position determines (although uncertainly) the speed of the target; whatever variability remains in the measured speed is explained by the weather (through the sensor model).

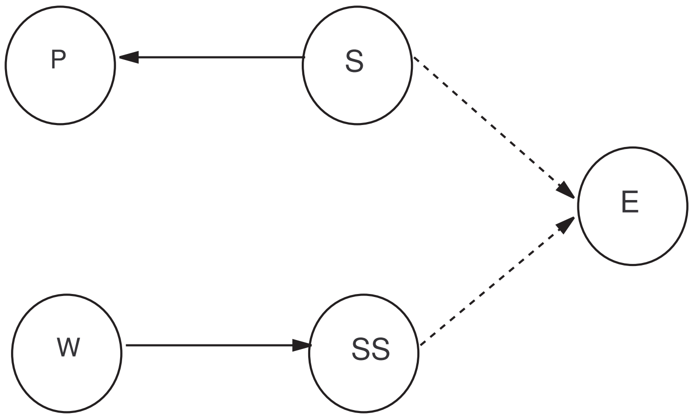

Figure 1.3: Speed with a sensor whose accuracy depends on the weather.

1.2 Why do the classical propagation methods fail?

We show, by a simple example, that the junction tree algorithm does not treat soft evidence properly.

The skeleton of the argument is as follows. In the junction tree algorithm, messages are passed across separators from clique to clique. Exactly two messages are passed between two cliques (say, C_i and C_j), one in each direction, as shown in Figure 1.4. The first message is passed during the *DistributeEvidence* phase of the method (say, from C_i to C_j), the second during the *CollectEvidence* phase (say, from C_j to C_i). suppose that clique C_i contains a node (say, V_i) for which we have a soft finding (say, $P(V_i)$), When C_i sends its message to C_j, $P(C_j)$ in modified (by calibration). After the probability of C_j has been fully updated (say, to $Q(C_j)$), C_j sends its message to C_i, and the probability of C_i is modified by calibration. In general, letting $Q(C_i)$ be the modified probability, we have that $\sum_{C_i \setminus \{V_i\}} Q(C_i) \neq P(V_i)$, which implies that the soft finding $P(V_i)$ is not treated as evidence.

We now present the promised example that establishes our claim that the junction tree algorithm does not handle soft evidence properly. In the classic wet grass example [Jen95], suppose that our hard evidence is that Holmes's lawn is certainly dry (i.e., $H = n$), while we have soft evidence that Watson's lawn, in the form $P(W) = (0.7, 0.3)$. If this soft evidence is absorbed in clique WR and the hard evidence is absorbed in clique HRS, propagation will consist of two messages through the separator R. After the messages are passed, P is updated to Q, and $Q(W) = (.4439, .5561) \neq (.7, .3) = P(W)$.

We have shown that even if soft findings are absorbed correctly into cliques, the propagation method itself would not respect the evidence characteristics of

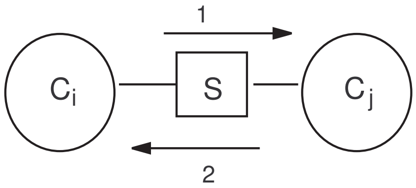

Figure 1.4: The soft finding $P(V_i)$ is not treated as evidence.

the findings.

The reason is that it is not possible to enter new evidence that contradicts the evidence already entered, and therefore zero entries are treated in a special way by the junction tree algorithm: zeroes in probability tables remain zeroes after each message (Cf. [Jen95, Lemma 4.1].)

1.3 Jeffrey's rule

Jeffrey's rule, also known as the rule of probability kinematics, provides a way to update a probability distribution from soft (uncertain, non-categorical, non-propositional) evidence. In this section, we introduce the rule and some problems with its application, following the presentation by Pearl [Pea88, Pea90].

Jeffrey's rule can be written as:

$$Q(A) = \sum_i P(A|B_i) \cdot Q(B_i)$$

where $Q(B)$ is soft evidence, and $P(A|B)$ is the conditional probability of A given B *before evidence.* Jeffrey's rule applies in situations in which $P(A|B)$ is invariant w.r.t. $P(B)$. This is not always the case, as shown in the following example, which is an extension by Pearl of an example by Jeffrey himself [Pea88].

The worth (W) of a piece of cloth depends on its color (C), as modeled by the Bayesian network structure of Figure 1.6, with $P(C)$ and $P(W|C)$. Before observing the cloth (whose color we know to be one of Green, Blue, and Violet), our belief is summarized by $P(Color) = (.3, .3, .4)$. We then observe the cloth by candlelight. The light is not good enough to allow us to distinguish the color precisely, but we revise our belief about the color of the cloth to $(.7, .25, .05)$. How should we revise the worth of the cloth?

Jeffrey's rule for this case is:

$$Q(W) = \sum P(W|c_i) \cdot Q(c_i).$$

The rule applies only if $P(W|c_i)$ is not modified by the observation, which is clearly true in this example. Diaconis and Zabell comment that "this condition

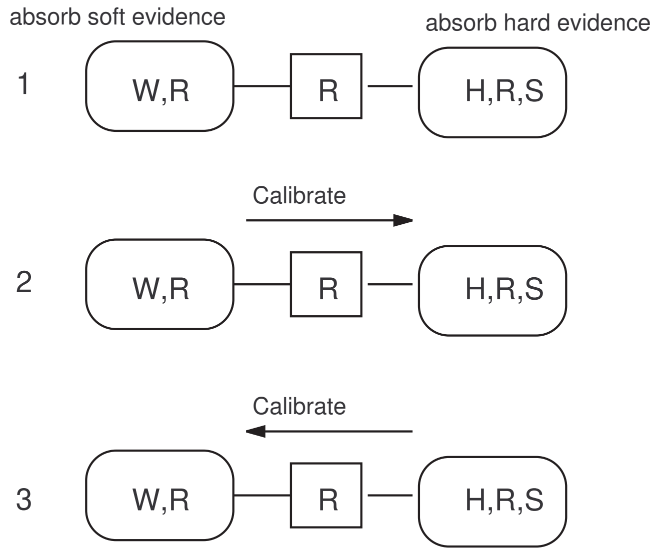

Figure 1.5: The junction tree algorithm and the wet grass network.

is an internal or psychological condition that must be checked or accepted at each stage. Mathematics has nothing to offer here" [DZ82, p.825]. There is, however, a principled way to check whether the condition holds, which is based on the theory of Bayesian networks.

Augment the Bayesian net with a node that represents the evidence (E) and the appropriate edges. In the example, the evidence depends on the (true) color of the cloth, as shown in Figure 1.7. Now, check whether W is independent of E given C (using a d-separation algorithm). If this is the case, then $P(W|C) = P(W|C, E)$, and Jeffrey's rule may be applied. In our example, this is the case.

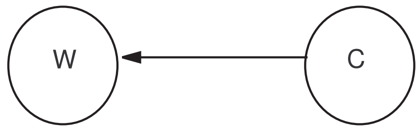

Figure 1.6: The worth of a cloth.

Let us now consider a situation in which Jeffrey's rule is not applicable. Suppose that we know the color of the cloth, and we want to update our belief in the type of candle (CA) used. In the absence of observations, the type of candle used does not affect the color of the cloth, and vice versa, so that:

$$P(CA|C) = P(CA),$$

and the Bayesian network structure is given in Figure 1.8.

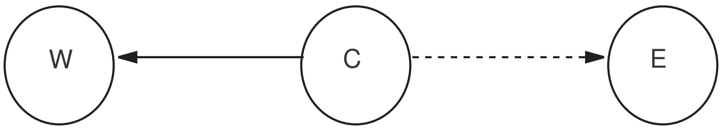

Figure 1.7: Cloth worth and cloth color in the presence of soft evidence.

Figure 1.8: Candle type and cloth color.

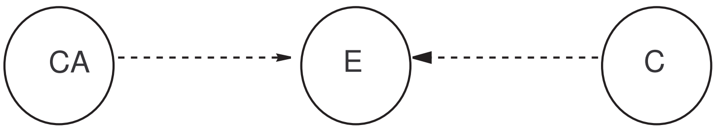

Figure 1.9: Candle type and cloth color in the presence of soft evidence.

The Bayesian network structure in the presence of soft evidence for Color is given in Figure 1.9.

Since CA and C are not d-separated by E, $P(CA|C) \neq P(CA|C, E)$, and therefore Jeffrey's rule is not applicable.

Pearl [Pea88, Pea90] observes that the virtual evidence method (also cf. [Kyb87, Nea90]) can be viewed as formally equivalent to the likelihood ratio version of Jeffrey's rule. Suppose that we obtain evidence on a variable B, with values b_i. Letting $Q(b_i)$ be the evidence and $P(b_i)$ be the prior probability of B, the ratio $Q(B)/P(B)$ may be used in the virtual evidence method, resulting in an updated distribution whose marginal over B is the soft evidence $Q(B)$.[1]

[1]Please refer to [VKV] for details.

Chapter 2

Observation Variables

Suppose that an agent-encapsulated Bayesian network needs to be updated in the light of information from another agent. There are two types of observed variables in a multiagent system:

1. One observed state of a random variable X is reported, but a publishing agent is unsure of its observation. E.g. a particular flip of a coin came up heads, but the agent is not sure since it has observed the coin from a long distance.

2. The probabilities that a random variable X takes certain states $P(X = x_i)$, $i = 1, \ldots, n$ are reported. E.g. an agent will report $P(X = head) = 0.5$ and $P(X = tail) = 0.5$ in the case it believes that it has observed a fair coin.

There is an approach to update from non-crisp information that is often used for in Bayesian networks, e.g. in the Hugin program [AOJJ89], and which corresponds to the first type of variable. Pearl calls it *virtual evidence update* in [Pea88]. (Also cf. [Hec86, HH86, Kyb87, Pea90, Nea90].) Under certain conditions, which we discuss below, virtual evidence update may be an appropriate way to perform inter-agent communication. The second approach, which is the focus of this report, corresponds to the second type of variable. We call it *soft evidential update*. Next, we will describe the basic differences between virtual evidence update and soft evidential update.

Virtual evidence update is a model of updating suitable for situations when agents obey the following property:

(V1) A publishing agent provides its belief in the true state of the observed variable. Its observation can be improved by something that the publishing agent can not observe. Variables in the subscribing agent model may have an influence on the true state of the variable observed by the publishing agent.

A model of virtual evidence update is displayed on Figure 2.1. It consists of the following three steps:

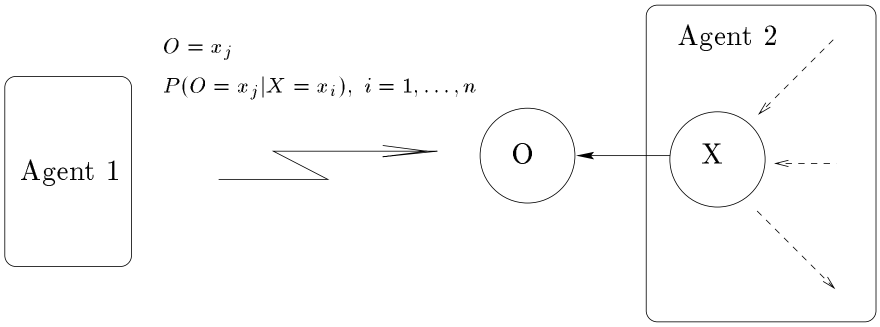

Figure 2.1: Virtual evidence update

1. A dummy node O is created. It has the same states as node X, and X is its only parent.

2. A publishing agent reports one observed state $x_j \in \{x_1, \ldots, x_n\}$ from n possible states of an observed variable X together with his reliability in the form of conditional probability distribution $P(O = x_j \mid X = x_i)$, $i = 1, \ldots, n$ (Note that only values for the observed value j are required). This conditional probability distributions read as: "The probability of observing variable X being in state x_j if its true state is x_i."

3. Updating of the Bayesian network is performed. This corresponds to entering evidence $O = x_j$ with consequent propagation using the values of $P(x_j \mid X = x_i)$, $i = 1, \ldots, n$. E.g. the junction tree algorithm [LS88, SS90, Jen95, LJ97] can be employed.

The approach for modeling soft evidence we propose is fundamentally different. The soft evidential update approach is suitable when the following assumption holds:

(S1) The publishing agent's belief on states of a variable cannot be improved by anything that is observed later, i.e. no variable in the model of a subscribing agent may have any influence on that publishing agent's observation.

The model of this approach is displayed on Figure 2.2. Note that, in contrast with virtual evidence update, the report of the publishing agent does not include the conditional probability table $P(X \mid O)$ but consists only of a distribution on the observation variable.

This kind of situation exists in AEBNs, in which there is precisely one agent with specific authority for each variable that occurs in multiple agents. In this case, it is correct for the subscriber agents to treat their initial belief in that variable not as a prior probability to be updated upon receipt of additional information (as in the virtual evidence method), but as a guess to be *replaced* by the evidence provided by the authority. The soft evidential update approach more

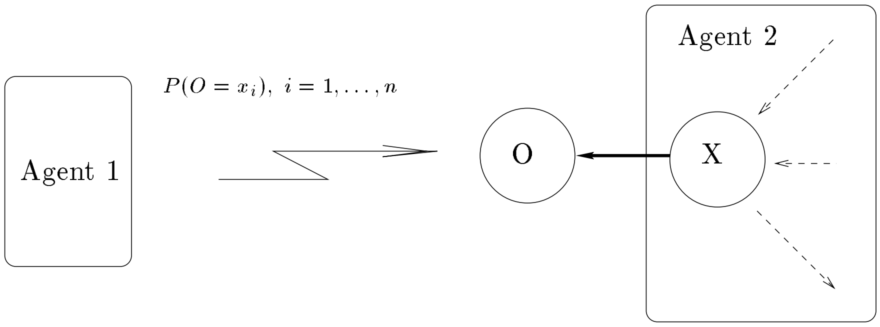

Figure 2.2: Soft evidential update

closely models the notion of observation as a property of the publishing agent and insures consistency across all agents that subscribe to the published variable, in the sense that the belief (marginal posterior probability) in a published variable is the same throughout the agent system.

The soft evidential update approach allows to combine the variable observed by the publishing agent and the subscribing agent's own opinion on the same variable, by the introduction of additional variables with the same domain. Dependence is modeled within the structure owned by the subscribing agent. By using this simple technique, it is reasonable to require that the probability distribution provided by the subscribing agent should not be changed (assumption S1) in a wider range of situations than it may superficially appear. In contrast with requiring that agents correspond by communicating either hard evidence or likelihood ratios, we can accommodate absolute beliefs, providing that the agent that uses them model them as such.

Soft evidential update approach requires the addition of observation variables to the original Bayesian network, in the following three steps.

- First, we create a new node, which we call *observation node*, for each observation. Let O be a generic such node. Every state of node O corresponds to a possible outcome of an observation.[1]

- Second, we add directed edges to node O from all nodes in the Bayesian network structure that have a direct influence on the observation, so that the set of parents of O in the extended network d-separates O from the rest of the network.

- Third, we model the dependence of the parents of O ($\mathrm{pa}(O)$) on O, by specifying the conditional probability table $P(O \mid \mathrm{pa}(O))$.

[1]Several additional types of evidence, such as higher-dimensional marginal distributions defined on the Cartesian product of two or more variables, can be handled, as described in [VKV], but we limit ourselves to the most common and useful case of findings on individual variables in this report.

Figures 1.2 and 1.7 show situations in which the observation (E in both cases) is a child of a single node (S and C respectively). Figures 1.3 and 1.9 show situations in which the observation (E in both cases) is a child of two nodes. The two parent nodes are S and SS in the case of Figure 1.3 and C and CC in the case of Figure 1.9.

Consider now the case of the specification of the conditional probability for the speed sensor. To update correctly using soft evidence, we need to specify $P(E \mid S, SS)$. Note that such specification only involves the possible values of variables O, S, and SS, and does not require consideration of belief states. In fact, the possible values of O are the same as the possible values of S. In the case of the color of the cloth, we need to specify $P(E \mid C, CC)$. Again, only the possible values of the three variables E, C, and CC need to be considered. Here, the possible values of E are the same as the possible values of C.

Chapter 3

Big Clique Algorithm

Here, we suggest a soft-evidence absorption algorithm. This algorithm combines two methods: the junction tree propagation and Iterative Proportional Fitting Procedure (IPFP).

3.1 Algorithm

The *big clique algorithm* modifies the junction tree algorithm as follows:

1. Build a junction tree that includes all variables for which soft evidence is given in one clique, the *big clique* C_1. (These variables may appear in other cliques as well.)

2. Update $P(V)$ to a distribution $P^*(V)$ by executing the junction tree algorithm using only hard evidence. $P^*(V)$ is a distributed representation of $P(V|$ hard evidence), in the sense of the remark following Theorem 4.2 in [Jen95]: the product of all clique tables divided by the product of all separator tables is equal to $P(V|$ hard evidence).

3. Absorb all soft evidence in C_1 (with the algorithm described above).

4. Call the routine DistributeEvidence from C_1. This routine and the correctness of this step are presented in Section 3.2.

3.2 Propagation of Soft Evidence

First, recall that Step 2 in the modified junction tree algorithm leads to a distributed representation of the posterior probability of all variables given all hard findings. Second, observe that the product of the table for the special clique that has absorbed all soft evidence (as done in Step 3) multiplied by the tables for the other cliques and divided by the tables of the separators is a representation of the posterior probability of all variables given the soft evidence

and the hard evidence. (Hard evidence had already been absorbed in Step 2).
(Hard evidence had already been absorbed in Step 2).

We now need to restore consistency between the special clique and the other clique. To do so, we propagate from the clique that has absorbed soft evidence using the Hugin *DistributeEvidence* algorithm, which is described in [Jen95, Sect. 4.4.1]. This algorithm has three important properties: (1) it updates the probability tables of the other cliques while . it maintains the invariant that the product of the clique tables divided by the separator tables is equal to the joint probability table for all variables in the Bayesian network; (2) it insures local consistency and (Theorem 4.5 in [Jen95]) global consistency; (3) it does not disturb hard findings, because it does not introduce new zeroes.

Finally, observe that the table for the clique that contains all variables for which we have soft findings is unchanged by a *DistributeEvidence* call that starts at that clique. Therefore, the result of propagation is to obtain a globally consistent distributed representation of the posterior in which all findings, hard and soft, hold.

We remark that the big clique algorithm could be simplified by removing the *DistributeEvidence* part from the second step. In other words, it is sufficient to carry out one *CollectEvidence* operation to the special clique (using only hard evidence) and one *DistributeEvidence* from the special clique (after absorbing soft evidence in the special clique). From this point on, we redefine the big clique algorithm to be this simplified version. Figures 3.1 and 3.2 illustrate the *CollectEvidence* and *DistributeEvidence* operations. Figure 3.3 illustrates the operation of the whole big clique algorithm on the lawn example. In this special case, the junction tree is the same as that constructed by the junction tree algorithm (cf. 1.5), but the order of operations is different. In particular, note how the absorption of soft evidence is delayed.

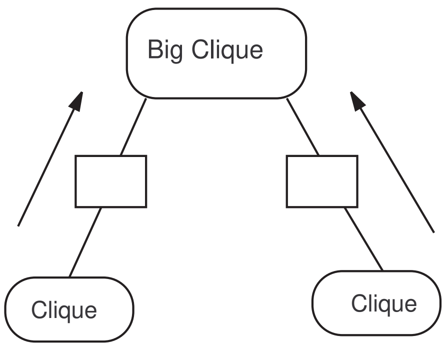

Figure 3.1: The big clique calls *CollectEvidence*.

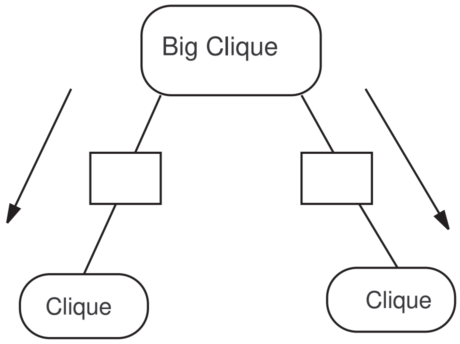

Figure 3.2: The big clique calls *DistributeEvidence*.

3.3 Absorption of Soft Evidence

We define absorption in the special big clique C_1 as the process by which the joint probability. $P(C_1)$, is updated to conform to soft evidence on variables $A \subseteq C_1$, where $A = \{A_1, A_2, \ldots, A_k\}$.

Let $Q(C_1)$ be the joint probability after absorption. Then $\forall i \sum_{C_1 \setminus A_i} Q(C_1) = P(A_i)$, where $P(A_i)$ is the soft evidence on $A_i, i = 1, \ldots, k$. Absorption of soft evidence in clique C_1 is carried out by using the Iterative Proportional Fitting Procedure (IPFP) and consists of cycles of k steps, one per finding. Each step corresponds to one soft finding. The appropriate formulae are:

$$
\begin{aligned}
Q_{(0)}(C_1) &= P(C_1) \\
Q_{(i)}(C_1) &= \frac{Q_{i-1}(C_1) \cdot P(A_j)}{Q_{i-1}(A_j)}
\end{aligned}
$$

where $j = (i-1) \bmod k + 1$.

For a simple example, suppose we have the clique $\{A, B\}$ with joint probability as given in Table 3.1 (all variables are binary). Suppose that soft evidence on variable B is available in the form of $P(B) = (.7, .3)$. We compute the updated joint probability $Q(B)$ in two steps.

The result of the multiplication by $P(B)$ is in Table 3.2. The result of the division by $Q_{(0)} = (.59, .41)$ is in Table 3.3. Note that $\sum_{\{A,B\} \setminus \{B\}} Q(A, B) = P(B) = (0.7, 0.3)$, as claimed.

One step of IPFP is sufficient when there is only one soft finding. In general, however, several cycles may be necessary for IPFP to converge. See [Csi75, Hab74, Vom99] for the proof of convergence in the general discrete case and for bounds on the number of cycles in special cases.

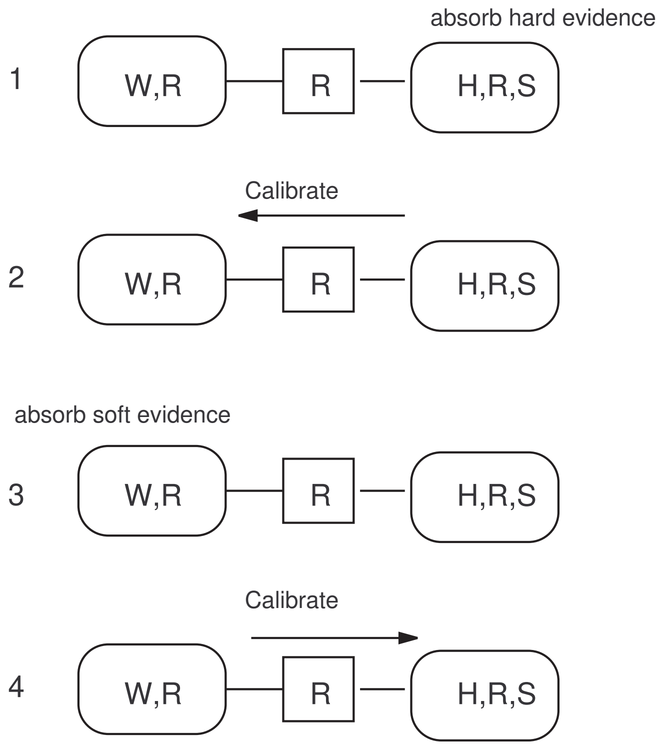

Figure 3.3: Operation of the big clique algorithm on the lawn example.

	A	
B	y	n
y	.56	.03
n	.14	.27
y	.56	.03
n	.14	.27

Table 3.1: Table for $P(A, B)$.

	A	
B	y	n
y	.392	.021
n	.042	.081

Table 3.2: Table for $P(A, B, e)$.

B	A y	n
y	.664	.036
n	.102	.198

Table 3.3: Table for $P(A, B, e)$ after normalization.

Chapter 4

Implementation of the Big Clique Algorithm

In order to support the above algorithms, we implemented BC-Hugin, an extension of the Hugin system, in Java.[1] As mentioned earlier, the big clique algorithm requires new methods such as creating a big clique and soft-evidence absorption, that are not supported by the junction tree propagation that is already implemented by Hugin API. Also the Hugin API does not allow us to control basic programs that are necessary for those new methods.

Therefore, we implemented the BC-Hugin from scratch.[2] The construction of a junction tree and evidence propagation were implemented by following Jensen's variant of the Lauritzen-Spiegelhalter algorithm [LS88, Jen95, CDLS99]. BC-Hugin reads *net* files or *hkb* files[3] and propagates the evidence (hard or soft) entered by the user.

4.1 Example1: Rain

This example is adapted from Jensen [Jen95] The Figure 4.1 shows the given rain model, and we set the *rain* node as the only soft-evidence node. Figure 4.3 shows the junction tree created by BC-Hugin. Clique 1 on the top of the tree is the big clique that includes all soft-evidence node. The value window (in Figure 4.2) is designed to provide visual results for each propagation. The hard finding is represented by radio buttons while text-box represents a soft-finding.

[1] We thank Christian Riekehr, general manager of Hugin Expert A/S, for granting authorization to use the name BC-Hugin.

[2] Exception: In order to save development time, we used Hugin API functions to parse the input networks.

[3] These files are normally created using the Hugin GUI.

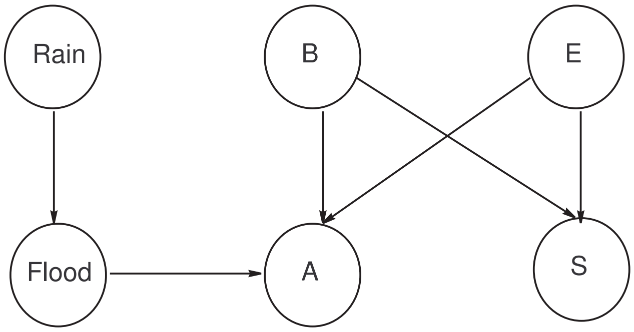

Figure 4.1: Rain Example.

4.2 Evaluation

Soft evidential updating can be formalized as a constrained optimization task. The goal is to find a probability distribution such that

1. it satisfies all the constraints introduced by soft evidence on the observation variables,

2. it optimizes a chosen criterion among all distributions satisfying these constraints.

It can be proved that the big clique algorithm computes a distribution that both (1) satisfies all the constraints introduced by soft evidence on the observation variables and (2) minimizes I-divergence with respect to the original probability distribution. It is very important to stress that the proper use of observation variables insures that the qualitative independence structure is maintained even upon soft evidence propagation. See [VKV] for details and proofs.

A detailed evaluation of BC-Hugin is still underway. The evaluation addresses two aspects:

1. Comparing result with Hugin: When only hard evidence is entered, BC-Hugin produces results identical to those obtained with Hugin.

2. Creating result with hard and soft evidence: In addition to the hard evidence, soft evidence is fed into BC-Hugin. The results are compared to hand-computed results or (for larger networks) are examined for plausibility.

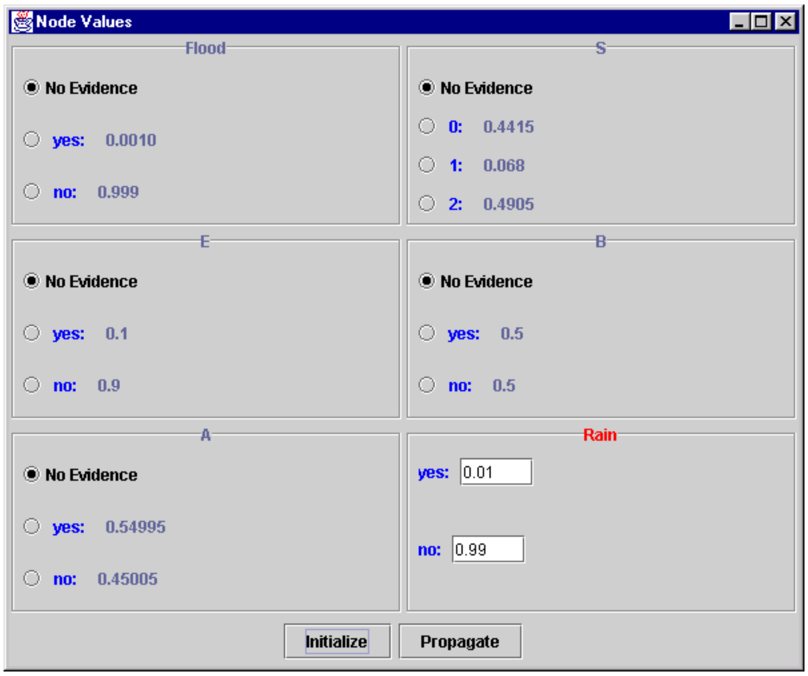

Figure 4.2: BC-Hugin Node window for Rain Example. The node *Rain* is the soft evidence.

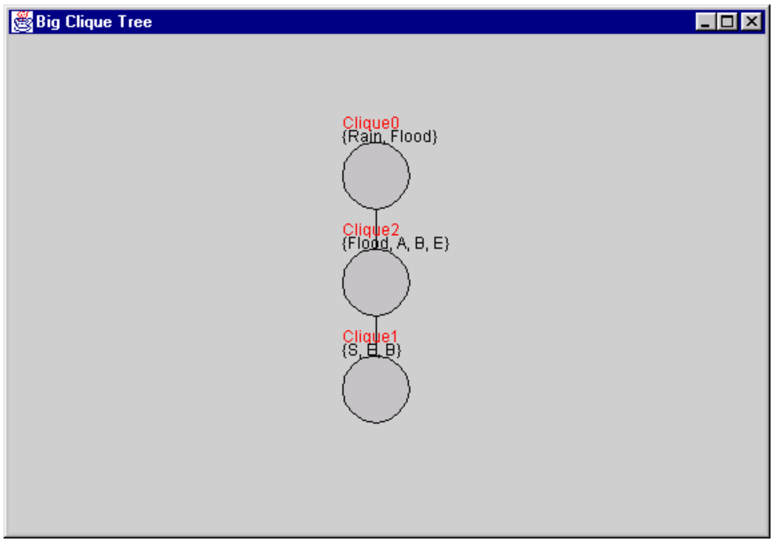

Figure 4.3: BC-Hugin clique tree window for Rain Example. The node *Rain* is the soft evidence.

Chapter 5

Appendix: Java Code Documentation for BC-Hugin

See WWW.PROJECTSITE.SC.EDU for the documentation of the Java code.

Bibliography

[AOJJ89] Stig K. Andersen, Kristian G. Olsen, Finn V. Jensen, and Frank Jensen. Hugin–a shell for building Bayesian belief universes for expert systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1080–1085, Detroit, 1989.

[Blo98] Mark Bloemeke. *Agent Encapsulated Bayesian Networks.* PhD thesis, Department of Computer Science, University of South Carolina, 1998.

[BS99] Mark Bloemeke and Wayne Smith. Agent encapsulated Bayesian networks—a solution to the rumor problem, Technical Report TR9908. Technical report, Department of Computer Science, University of South Carolina, 1999.

[BV] Mark Bloemeke and Marco Valtorta. Agent encapsulated Bayesian networks—a framework for probabilistic agent systems. in preparation.

[CDLS99] Robert G. Cowell, A. Philip Dawid, Steffen L. Lauritzen, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems.* Springer-Verlag, New York, NY, 1999.

[Csi75] I. Csiszár. *I*-divergence geometry of probability distributions and minimization problems. *Ann. Prob.*, 3(1):146–158, 1975.

[DZ82] Persi Diaconis and Sandi Zabell. Updating subjective probability. *Journal of the American Statistical Association*, 77:822–830, 1982.

[Hab74] S. Haberman. *The Analysis of Frequency Data.* The University of Chicago Press, Chicago and London, 1974.

[Hec86] David Heckerman. Probabilistic interpretation for MYCIN's certainty factors. In L. N. Kanal and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, pages 167–196. Kluwer Science Publishers, 1986.

[HH86] Eric Horvitz and David Heckerman. The inconsistent use of measures of certainty in artificial intelligence research. In L. N. Kanal and J.F.

Lemmer, editors, *Uncertainty in Artificial Intelligence*, pages 137–151. Kluwer Science Publishers, 1986.

[Jen95] Finn V. Jensen. *An Introduction to Bayesian Networks.* Springer-Verlag, New York, NY, 1995.

[Kyb87] Henry E Kyburg, Jr. Bayesian and non-Bayesian evidential updating. *Artificial Intelligence*, 31:271–293, 1987.

[LJ97] S.L. Lauritzen and F.V. Jensen. Local computation with valuations from a commutative semigroup. In *Annals of Mathematics and Artificial Intelligence*, volume 21, pages 51–69. J.C. Baltzer A.G., 1997.

[LS88] Steffen L. Lauritzen and David Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society, Series B*, 50:157–224, 1988.

[Nea90] Richard E. Neapolitan. *Probabilistic Reasoning in Expert Systems: Theory and Algorithms.* John Wiley and Sons, New York, NY, 1990.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufman, San Mateo, CA, 1988.

[Pea90] Judea Pearl. Jeffrey's rule, passage of experience, and *neo*-Bayesianism. In H.E. et al. Kyburg, Jr., editor, *Knowledge Representation and Defeasible Reasoning*, pages 245–265. Kluwer Academic Publishers, 1990.

[SS90] Glenn R. Shafer and Prakash P. Shenoy. Probability propagation. *Annals of Mathematics and Artificial Intelligence*, 2:327–352, 1990.

[VKV] Marco Valtorta, Young-Gyun Kim, and Jiri Vomlel. Soft evidential update for multiagent systems. under review.

[Vom99] Jiri Vomlel. *Methods of Probabilistic Knowledge Integration.* PhD thesis, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, December 1999.

[Xia96] Y. Xiang. A probabilistic framework for cooperative multi-agent distributed interpretation and optimization of communication. *Artificial Intelligence*, 86:295–342, 1996.

[XL00] Y. Xiang and V. Lesser. Justifying multiply sectioned Bayesian networks. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS-2000)*, Boston, MA, July 2000. To appear; preprint available at the first author's web site.