Inconsistent Knowledge Integration with Bayesian Network

Yi Sun and Yun Peng
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County

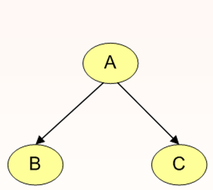
Background

Given a Bayesian network (BN) representing a probabilistic knowledge base of a domain, and a set of low-dimensional probability distributions (also called constraints) representing pieces of new knowledge coming from more up-to-date or more specific observations for a certain perspective of the domain, we present a theoretical framework and related methods for integrating the constraints into the BN, even when these constraints are inconsistent with the structure of the BN due to dependencies among relevant variables in the constraint being absent in the BN.

Inconsistent Constraints

- Type I inconsistency: there does not exist a joint probability distribution that can satisfy all the constraints in the constraint set.
- Type II inconsistency: some dependency implied in the constraint does not hold in the DAG of the BN.

Example:
R1 and R2 have Type I inconsistency because R1(A)≠ R2(A). R3 has Type II inconsistency with the DAG because R3(B,C|A)≠ R3(B|A) R3(C|A).



A	
T	F
0.5	0.5

A	B	
	T	F
T	0.5	0.5
F	0.5	0.5

A	C	
	T	F
T	0.5	0.5
F	0.5	0.5

A	B	R1(A,B)
T	T	0.2
T	F	0.4
F	T	0.2
F	F	0.2

A	C	R2(A,C)
T	T	0.2
T	F	0.2
F	T	0.4
F	F	0.2

A	B	C	R3(A,B,C)
T	T	T	0.1
T	T	F	0.15
T	F	T	0.15
T	F	F	0.1
F	T	T	0.1
F	T	F	0.15
F	F	T	0.15
F	F	F	0.1

The Problem

Given BN $G=(G_S,G_P)$ with JPD $P(X)$, and a set of constraints $\mathbf{R}=\{R_1(Y^1),...,R_m(Y^m)\}$ with at least one constraint having Type II inconsistency, construct a new BN $G'=(G'_S,G'_P)$ with probabilility distribution $P'(X)$ that meet the following conditions:
C1: Constraint satisfaction: $\forall R_i(Y^j)\in\mathbf{R}, P'(Y^j)=R_i(Y^j)$;
C2: Minimality: $I(P'(X)\,\|\,P(X))$ is as small as possible.

Methods

Our framework allows for identifying the structural inconsistencies between the constraints and the existing BN. We use the d-separation method to find out dependency relations in the BN, and use the dependency test to find out dependency relations in each constraint. The dependency relations that exist in the constraint but are missing in the BN are the cause of the structural inconsistencies.

Algorithm INCIDENT
Input: BN $G=(G_s,G_P)$ with ordering of nodes in G_s, and constraints $\mathbf{R}=\{R_1,R_2,...,R_m\}$.
Output: Structural Consistent constraint set $\mathbf{R}^+$, structural inconsistent constraint set $\mathbf{R}^-$ and Dependency List DL.
Steps:

1. Create an empty consistent constraint set $\mathbf{R}^+$, an empty inconsistent constraint set $\mathbf{R}^-$, and an empty Dependency List DL;
2. Perform the following steps for each constraint R_i in $\mathbf{R}$:
   2.1 Create an empty Dependency List DL_i;
   2.2 For each pair of variables A and B in R_i, do from zero-order to ($|R_i|-2$) -order independence test on them, where $|R_i|$ is the number of variables in R_i. If the test fails, add $<R_i,A,B,Z>$ to DL_i;
   2.3 For each entry $<R_i,A,B,Z>$ in DL_i, use D-Separation method to test whether A and B are independent given Z, i.e., $A\perp B\,|\,Z$. If the test fails, remove this entry from DL_i;
   2.4 If DL_i is empty, add R_i to $\mathbf{R}^+$. Otherwise, add R_i to $\mathbf{R}^-$, and merge DL_i into DL;
3. Return $\mathbf{R}^+$, $\mathbf{R}^-$, and DL.

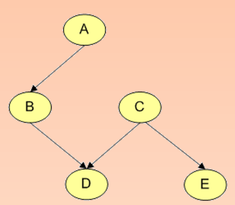
Methods

Our framework also allows for overcoming the identified structural inconsistencies during the knowledge integration by changing the structure of the BN. This is done by adding one node for each structural inconsistent constraint in a way similar to the use of the virtual evidence node in Pearl's virtual evidence method. Other more computationally efficient variations of adding node methods are also considered.

Algorithm AddNode+E-IPFP
Input: BN $G=(G_S,G_P)$ with ordering of nodes in G_S, and constraint set $\mathbf{R}=\{R_1,R_2,...,R_m\}$.
Output: BN $G'=(G'_S,G'_P)$ that satisfies $\mathbf{R}$ with JPD as close to that of G as possible.
Steps:

1. Run INCIDENT to partition $\mathbf{R}$ into $\mathbf{R}^+$ and $\mathbf{R}^-$;
2. For each constraint R_i in $\mathbf{R}^-$, add a new node V_i to G_S with variables in R_i as its parents;
3. /* this step is the same as AddNode method */
   For each constraint R_i in $\mathbf{R}^-$,
   3.1 Calculate likelihood ratio for R_i using
   $$L(Y^i)=\frac{R_i}{P(Y^i)}=\frac{R_i(y_{(1)})}{P(y_{(1)})}:\frac{R_i(y_{(2)})}{P(y_{(2)})}:...:\frac{R_i(y_{(l)})}{P(y_{(l)})}$$
   where Y^i is the variable set of R_i, and l is the number of distinct instantiations for Y^i;
   3.2 Construct CPT of V_i with likelihood ratio $L(Y^i)$;
   3.3 Set the state of V_i to be true;
4. Apply one iteration of E-IPFP with $\mathbf{R}^+$ and the updated BN as input, i.e.,
   4.1 Do I-projection for the current probability distribution $P_{k-1}(X)$ on each constraint R_j in $\mathbf{R}^+$:
   $$P_k(X)=P_{k-1}(X)\cdot\frac{R_j(Y^j)}{P_{k-1}(Y^j)}$$
   where Y^j is the variable set of R_j, and X is the set of all variables in G;
   4.2 Form and apply the structure constraint:
   $$P_k(X)=\prod_{i=1}^{n}P_{k-1}(X_i\,|\,\pi_i)$$
   where $(X_i,\pi_i)\in G_s$;
5. Repeat step 3 and step 4 until the JPD of the updated BN does not change;
6. Return $G'=(G'_S,G'_P)$, where G'_S is the updated structure after adding nodes to G_S, and G'_P is the CPTs for the nodes in G'_S.

Experiment



A	
T	F
0.5	0.5

C	E	
	T	F
T	0.5	0.5
F	0.5	0.5

A	B	
	T	F
T	0.5	0.5
F	0.5	0.5

B	C	D	
		T	F
T	T	0.5	0.5
T	F	0.5	0.5
F	T	0.5	0.5
F	F	0.5	0.5

C	
T	F
0.5	0.5

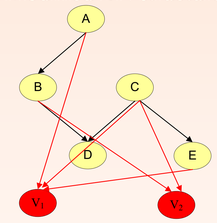
A	B	E	R1(A,C,E)
T	T	T	0.1368
T	T	F	0.2232
T	F	T	0.03
T	F	F	0.03
F	T	T	0.1672
F	T	F	0.2728
F	F	T	0.07
F	F	F	0.07

B	C	R2(B,C)
T	T	0.6
T	F	0.1
F	T	0.2
F	F	0.1

C	D	E	R3(C,D,E)
T	T	T	0.228
T	T	F	0.372
T	F	T	0.076
T	F	F	0.124
F	T	T	0.04
F	T	F	0.04
F	F	T	0.06
F	F	F	0.06

Result of INCIDENT:
$\mathbf{R}^-=\{R1,R2\}$
$\mathbf{R}^+=\{R3\}$
DL= {<R1, A, C, Ø>,
<R1, A, E, Ø>,
<R1, A, C, {E}>,
<R2, B, C, Ø>}

Modified BN structure:



Comparison of different methods for the above BN

Method	Added Nodes	Iterations	Run Time	I-aggregate	I-divergence
E-IPFP-SMOOTH	0	149	195s	0.062	0.5
AddNode+E-IPFP	2	3	3.22s	0	0.65
AddNode+Merge	1	3	2.32s	0	0.65
AddNode+Factorization	1	3	2.27s	0	0.65

Comparison of different methods for 10 node BN

Method	Added Nodes	Iterations	Run Time	I-aggregate	I-divergence
AddNode+E-IPFP	3	1	102s	0	2.28
AddNode+Merge	1	1	15.8s	0	2.28
AddNode+Factorization	1	1	14.2s	0	2.28

Conclusion

Compared to the existing methods for probabilistic knowledge integration, our methods have the advantage of being able to integrate new dependencies into the existing knowledge base as constraints become available from more reliable sources.