Incremental Junction Tree Inference

Hamza Agli, Philippe Bonnard, Christophe Gonzales, Pierre-Henri Wuillemin

Incremental Junction Tree Inference

Hamza AGLI[‡], Philippe BONNARD[‡],
Christophe GONZALES[⋆] and Pierre-Henri WUILLEMIN[⋆]

[‡] IBM France Lab, Gentilly, France
[⋆] Sorbonne Universités, UPMC Univ Paris 6, CNRS, UMR 7606 LIP6, Paris, France,
{hamza.agli,philippe.bonnard}@fr.ibm.com
{christophe.gonzales,pierre-henri.wuillemin}@lip6.fr

Abstract. Performing probabilistic inference in multi-target dynamic systems is a challenging task. When the system, its evidence and/or its targets evolve, most of the inference algorithms either recompute everything from scratch, even though incremental changes do not invalidate all the previous computations, or do not fully exploit incrementality to minimize computations. This incurs strong unnecessary overheads when the system under study is large. To alleviate this problem, we propose in this paper a new junction tree-based message-passing inference algorithm that, given a new query, minimizes computations by identifying precisely the set of messages that differ from the preceding computations. Experimental results highlight the efficiency of our approach.

Keywords: Bayesian networks, incremental inference, junction tree.

1 Introduction

Bayesian networks (BN) [17, 10] are one of the most popular framework for reasoning with uncertainty in expert systems. They are used in a wide range of real-world applications, including medical diagnosis, risk management and clinical decision support. A *BN* is a compact graphical representation of a joint probability distribution. It can be considered as a probabilistic knowledge base, in which the process of querying/requesting is called inference. Different queries exist, including the computation of most probable explanations or that of the posterior marginal distributions of some random variables (hereafter called *targets*). In this paper, we focus on the latter. It is known to be NP-hard in general [2, 3] but many exact and approximate inference algorithms have been proposed in the literature [12, 15, 20]. Extensions to handle very large systems [11, 18, 21] and temporal features have also been proposed [6, 16, 19]. Their increased complexity requires even more efficient inference algorithms.

Rule-based systems, which originated our research, are nowadays a very popular tool for automating decision making. To quantify uncertainties in the domain, they most often use heuristic models, e.g., *certainty factors* [1], which have theoretical and practical limitations [8] that could be overcome by exploiting probabilities. In this context, BNs could prove to be useful. In addition, their efficient inference engines, notably cluster-based and junction tree-based algorithms [9, 20, 15], seem to be good candidates to speed-up the rules inference process. But, by essence, rule-based systems are incremental multi-target environments, so BN inference shall also be performed incrementally. Some algorithms exploit partially this feature (see [15]) but they are far from optimal when the set of targets is smaller than the set of all the random variables or when it changes. The problem is even worse when the structure of the junction tree (JT) evolves over time. This can become an issue in rule-based systems in which changes in the *BN* structure, the evidence and the *targets*, occur frequently.

In [4], 4 incrementality criteria relevant to probabilistic inference were introduced: incrementality w.r.t resources, queries, evidence and representation. In this paper, we are interested in all these criteria, especially in the last three. Surprisingly, very few inference algorithms address all these aspects. In [5], for instance, the query point of view is taken into account by reconfiguring dynamically some join trees when queries change but the *BN* structure is assumed to remain static, which may not necessarily be the case in rule-based systems. In [14], the authors exploit relevance-based reasoning to identify the parts of the network that are relevant for computations and, then, update several subnetworks whose union covers the original one. Unfortunately, this algorithm does not take into account computations performed previously. In [15], an incremental JT-based inference algorithm has been proposed that exploits independences induced by incremental evidence updates. But the JT structure never evolves and it is assumed that all the nodes are targets, which is not optimal in our context. On the opposite, the incremental JT structure is addressed in [7] but not the queries incrementality nor the exploitation of previous probabilistic computations. Along similar lines, Li *et al.* argue that compiling the original *BN* into a conjunctive normal form coupled with caching techniques improves inference when the network structure is updated [13]. But this does not take optimally into account evidence and queries. In this paper, we investigate a new approach to overcome the above shortcomings. This approach aims at improving the efficiency of inference for very large and dynamic systems. The key idea of our algorithm, called Incremental Junction Tree Inference (*IJTI*), consists of restricting the computations only to parts of the JT that are relevant to *targets* and that have been invalidated by incremental changes. As a consequence, *IJTI* minimizes the probabilistic computations.

The paper is organized as follows. In the next section, we introduce the necessary background. In Section 3, we present our approach and justify its correctness. Then we highlight the efficiency of our contribution with a set of experiments. Finally, some conclusion and future works are provided in Section 5. All the proofs are given in an appendix.

2 Preliminaries and Notations

A BN is a pair $(\mathcal{G}, \Theta)$, where $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ is a directed acyclic graph (DAG). $\mathcal{V}$ is a set of nodes representing random variables. $\mathcal{A}$ is a set of arcs and $\Theta =$

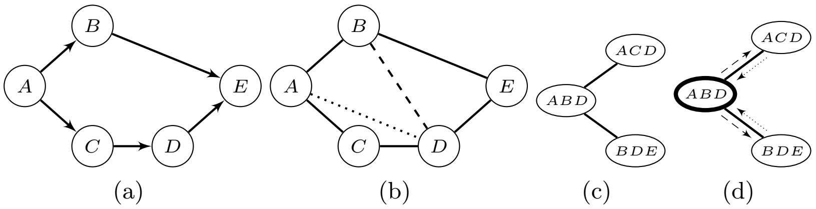


Fig. 1: A JT construction

$\{P(X|\text{Pa}(X)) : X \in \mathcal{V}\}$ is the set of the conditional probability tables (CPT) of the variables in $\mathcal{V}$ given their parents in $\mathcal{G}$. The BN encodes the joint probability over $\mathcal{V}$ as the product of these CPTs. In this paper, probabilistic inference is based on a message-passing algorithm within a JT. Constructing the latter consists of, first, converting DAG $\mathcal{G}$ into an undirected graph by adding, for each node in $\mathcal{V}$, edges between all of its parents (*moralization*) and removing the orientations of the remaining arcs, and, then, by adding an edge between a pair of non-adjacent nodes in every cycle of at least four nodes (*triangulation*). The nodes of the JT correspond to complete maximal subgraphs (cliques) of the resulting graph. These nodes are linked by edges in such a way that *i*) the JT contains no loop; and *ii*) any pair of cliques with a nonempty intersection are linked by a path on which all cliques contain this intersection. Fig. 1a shows an example of a DAG, its moralized and triangulated graph are given in Fig. 1b, where dashed and dotted edges represent those added during *moralization* and *triangulation* respectively. Finally Fig. 1c depicts a corresponding JT. Note that a JT can be a forest, e.g., when DAG $\mathcal{G}$ is not connected. In our approach, dealing with a forest is equivalent to iterate the same process on its connected components. Hence, without loss of generality, we will consider in the sequel that the JT on which we will perform inference is a tree $\mathcal{T}$. Hereafter, for any JT $\mathcal{T}$, we will denote by $\mathcal{V}(\mathcal{T})$ and $\mathcal{E}(\mathcal{T})$ its set of cliques and edges respectively.

The message-passing algorithm consists of performing a *collect* and a *distribution* from a predetermined root $r \in \mathcal{V}(\mathcal{T})$. During the *collect*, messages are sent along edges from leaves toward r and, during the *distribution*, they are sent in the opposite direction. To guarantee the correctness of computations, for any edge $(i, j) \in \mathcal{E}(\mathcal{T})$, the message sent from i to j, denoted by $\psi_{i \to j}$, is computed only when clique i has received messages from all its neighbors except j. Fig. 1d shows an example of message-passing with $r = ABD$ (thick clique) and dotted and dashed arcs representing the *collect* and *distribution* messages respectively. The computation of these messages is beyond the scope of this paper but can be found in [15]. In an incremental environment, not all the messages need be recomputed each time a modification occurs because, in practice, many will remain the same. As we shall see, using the following definitions, we can characterize precisely those that need some update given new set of evidence, structural changes or/and targets.

Definition 1 (Path). *let $\mathcal{T} = (\mathcal{V}(\mathcal{T}), \mathcal{E}(\mathcal{T}))$ be a JT. $i_1, \ldots, i_{n+1}$ is said to be a path in $\mathcal{T}$ if $(i_\alpha, i_{\alpha+1}) \in \mathcal{E}(\mathcal{T})$ for all $\alpha \in \{1, \ldots, n\}$. For simplicity, this path is denoted by $i_1 - i_{n+1}$ and its length by $len(i_1 - i_{n+1})$ (which is equal to n).*

Definition 2 (Adjacency). *Let $i, j \in \mathcal{V}(\mathcal{T})$, $i \neq j$. i and j are adjacent in $\mathcal{T}$ iff $(i, j) \in \mathcal{E}(\mathcal{T})$. The set of cliques adjacent to i is denoted by $\mathrm{Adj}(i)$, i.e., $\mathrm{Adj}(i) := \big\{k \in \mathcal{V}(\mathcal{T}) : (i, k) \in \mathcal{E}(\mathcal{T})\big\}$. Let $r \in \mathcal{V}(\mathcal{T})$, $r \neq i$, then $Adj_r(i)$ denotes the singleton set containing the clique adjacent to i that is on the path between i and r, i.e., $Adj_r(i) := \big\{k \in \mathrm{Adj}(i) : k \in i - r\big\}$. We also define $Adj_r(r) := \emptyset$. Finally, let $\mathrm{Adj}_{\text{-}j}(i) := \mathrm{Adj}(i) \setminus \{j\}$.*

For instance, in Fig. 2a, $Adj_r(i) = \{k_3\}$ and $Adj_r(k_3) = \{r\}$. Finally, let $\mathcal{V}_{\text{-}j}(i)$ stands for the set of nodes of the maximal subtree in $\mathcal{T}$ that contains i and not $Adj_j(i)$, and let $\mathcal{V}_j(i) = \mathcal{V}_{\text{-}j}(i) \cup \{j\}$ (see the shadowed area in Fig.2a).

A message $\psi_{i \to j}$ sent within $\mathcal{T}$ is directed by nature. It propagates toward j (and, by induction, toward $\mathcal{V}_{\text{-}i}(j)$) all the relevant information coming from the cliques in $\mathcal{V}_{\text{-}j}(i)$, notably all the evidence they received (by abuse, we say that a clique received evidence when at least one of its random variables received evidence). As a consequence, if $\psi_{i \to j}$ has already been computed previously and no new evidence has been received nor structural changes occured in $\mathcal{V}_{\text{-}j}(i)$, there is no need to recompute it. But even if $\mathcal{V}_{\text{-}j}(i)$ received evidence, $\psi_{i \to j}$ needs not be computed/updated if $\mathcal{V}_{\text{-}i}(j)$ contains no target. In this case, $\psi_{i \to j}$'s state becomes "invalid" since the content of $\psi_{i \to j}$ is now incorrect. This is not an issue for the current inference but, for future ones, we have to take this state into account to recompute $\psi_{i \to j}$ if it is to be used. Let $\mathcal{A}(\mathcal{T})$ be the set of all arcs induced from $\mathcal{E}(\mathcal{T})$, taking into account orientations, i.e., $\mathcal{A}(\mathcal{T}) := \bigcup_{(i,j) \in \mathcal{E}(\mathcal{T})} \big\{(i, j)\big\} \cup \big\{(j, i)\big\}$. To formalize the above conditions, we begin with characterizing the information that is "local" to i and j by:

Definition 3 (Local label-message λ). *$\lambda : \mathcal{A}(\mathcal{T}) \mapsto 2^{\{\epsilon, T\}}$ is a function s.t.*

$$(i, j) \longmapsto \lambda_{i \to j} := \begin{cases} \{\epsilon\} & \text{if } \psi_{i \to j} \text{ is in "invalid state" or "new evidence or} \\ & \text{structural changes" have affected } i \text{ (1)} \\ \{T\} & \text{if } i \text{ contains targets (2)} \\ \{T, \epsilon\} & \text{if (1) and (2)} \\ \emptyset & \text{otherwise} \end{cases}$$

To simplify the notation, hereafter, we will remove braces and denote $\{T, \epsilon\}$ by $T\epsilon$. Then, the idea of our algorithm consists of marking every arc (i, j) in $\mathcal{A}(\mathcal{T})$ by labels $\mu_{i \to j}$ expressing all the "local" information that $\mathcal{V}_{\text{-}j}(i)$ contains.

Definition 4 (Label-message μ). *For $(i, j) \in \mathcal{A}(\mathcal{T})$, the label-message sent from i to j is a function $\mu : \mathcal{A}(\mathcal{T}) \mapsto 2^{\{\epsilon, T\}}$ such that $\mu_{i \to j} := \bigcup_{\substack{k' \in \mathcal{V}_{\text{-}j}(i) \\ \{k\} = Adj_j(k')}} \lambda_{k' \to k}$.*

As an example of the previous discussion, imagine that a first incremental update impacts the initial DAG and consequently the initial $\mathcal{T}$ of Fig. 2a. This

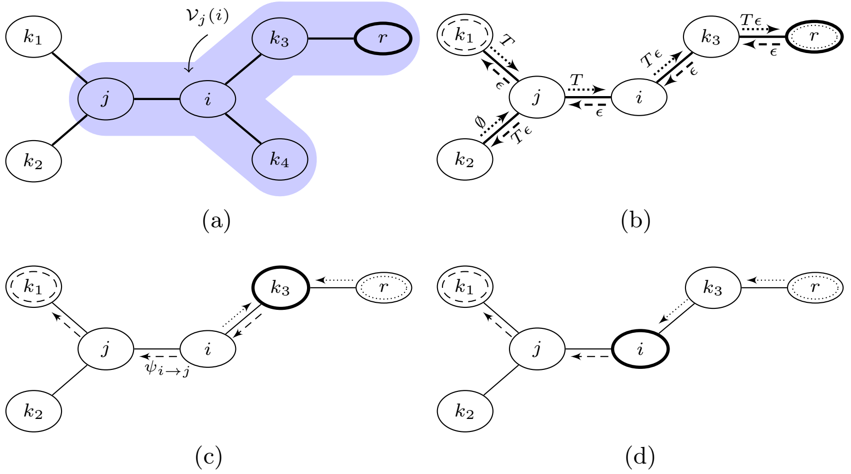


Fig. 2: Message passing within a JT $\mathcal{T}$.

consists of the removal of k_4, the insertion of an evidence on r and a new target on k_1. Fig. 2b depicts the μ-messaging within $\mathcal{T}$ after this update, where dashed and dotted ellipses stand for the cliques containing targets and evidence respectively. One can easily see that, for instance, $\mu_{i\to j} = \epsilon$, $\mu_{j\to i} = T$ and $\mu_{j\to k_2} = T\epsilon$. The following proposition allows to recursively construct the μ-messages:

Proposition 1 (μ construction). *Let $(i,j) \in \mathcal{A}(\mathcal{T})$, then we have : $\mu_{i\to j} = \lambda_{i\to j} \cup \bigcup_{k\in \mathrm{Adj}_{\text{-}j}(i)} \mu_{k\to i}$.*

3 Inference optimization: *IJTI*

Let us recall that $\psi_{i\to j}$ denotes the message exchanged between cliques i and j during an inference computation. It shall not be confused with the label message $\mu_{i\to j}$ of Definition 4.

3.1 Optimal roots

Usually, the number of computations performed by a JT-based message-passing algorithm does not depend on the root clique selected for collect/distribution because the $\psi_{i\to j}$ messages are sent on both directions on *all* the edges of the JT. For IJTI, this is not the case, since this algorithm computes and sends *only* the $\psi_{i\to j}$ messages necessary for the computation of the posterior distributions of its target nodes. On some edges, IJTI will therefore not compute some $\psi_{i\to j}$ messages because they are irrelevant w.r.t. the targets posterior distributions. As a consequence, in IJTI, the number of computations performed is sensitive to the selection of the root: for instance, in the JT of Fig. 2a, if clique i received evidence and the only target is j, only message $\psi_{i\to j}$ from i to j is necessary, which is precisely what is sent if clique i is selected as root (here, only a distribution is necessary). But if clique k_4 is selected instead, message $\psi_{i\to k_4}$ needs to be sent during the collect and messages $\psi_{k_4\to i}$ and $\psi_{i\to j}$ need to be sent during the distribution, which is clearly not optimal. To determine the optimal roots, let us define $\delta_{i\to j}(r)$ as an indicator of whether message $\psi_{i\to j}$ is recomputed (in this case, $\delta_{i\to j}(r) = 1$) or not ($\delta_{i\to j}(r) = 0$) when r is selected as a root. In IJTI, we therefore seek to minimize $\delta(r) = \sum_{(k',k)\in\mathcal{A}(\mathcal{T})} \delta_{k'\to k}(r)$, which corresponds to the total number of messages recomputed and sent. Based on the discussion of the preceding section, we can write:

$$\delta_{i\to j}(r) = \begin{cases} 1 \text{ if } (\epsilon \in \mu_{i\to j} \text{ and } \{j\} = Adj_r(i)) \text{ or} \\ \qquad (\epsilon \in \mu_{i\to j} \text{ and } \{i\} = Adj_r(j) \text{ and } T \in \mu_{j\to i}) \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

The first line of Eq. (1) concerns *collect* messages ($\{j\} = Adj_r(i)$). It asserts that *collect* message $\psi_{i\to j}$ needs to be recomputed only if it is currently in an invalid state or if new evidence or structural changes have occurred in $\mathcal{V}_{\text{-}j}(i)$ ($\epsilon \in \mu_{i\to j}$). When this is not the case, clearly, this message is up to date and does not need recomputation. The second line of Eq. (1) concerns *distribution* messages ($\{i\} = Adj_r(j)$). It asserts that $\psi_{i\to j}$ needs to be recomputed only if there exists a target farther toward the leaves of the JT ($T \in \mu_{j\to i}$) and if some evidence has been received on $\mathcal{V}_{\text{-}j}(i)$ or some message coming from $\mathcal{V}_{\text{-}j}(i)$ has been updated ($\epsilon \in \mu_{i\to j}$). Eq. (1) can be rewritten more compactly as:

$$\delta_{i\to j}(r) = \begin{cases} 1 \text{ if } \epsilon \in \mu_{i\to j} \text{ and} \\ \qquad (\{j\} = Adj_r(i) \text{ or } (\{i\} = Adj_r(j) \text{ and } T \in \mu_{j\to i})) \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

Fig. 2c and Fig. 2d illustrate that $\delta(k_3) = 5$ and $\delta(i) = 4$ respectively. In this case, it is better to select i as a root rather than k_3 since this avoids the unnecessary computation of one message. The following theorem states the existence of some optimal roots and characterize them:

Theorem 1 (Optimal roots). *Suppose we computed the μ-messages within $\mathcal{T}$. Then there exists $r \in \mathcal{V}(\mathcal{T})$ fulfilling one of the following mutually exclusive and exhaustive properties:*

a) $(\mathcal{V}(\mathcal{T}), \mathcal{E}(\mathcal{T})) = (\{r\}, \emptyset)$
b) $\exists r' \in \mathcal{V}(\mathcal{T}) : \mu_{r'\to r} = \mu_{r\to r'} = T\epsilon$
c) $\forall k \in Adj(r) : \mu_{k\to r} \in \{T, \epsilon, \emptyset\}$

In addition, $r \in Argmin_{k\in\mathcal{V}(\mathcal{T})}\delta(k)$, i.e., r is an optimal root w.r.t. inference computations.

3.2 A new incremental inference

In this section, we propose a new algorithm designed to deal with incremental inference. We assume that a first inference has been performed by message-passing within $\mathcal{T}$, using for instance a collect-distribute algorithm in a Lazy

Algorithm 1: IJTI

```
   input  : modified $\mathcal{T}$, $Q$ targets cliques
   output : posteriors on targets
   // set the number of neighbors
      visited during the collect
1  for $i \in \mathcal{V}(\mathcal{T})$ do
2      $i.nbVN \leftarrow 0$
3  Compute the $\mu$-labels in $\mathcal{T}$
4  Find $r$ using Theorem 1
5  $\mathbf{L} \leftarrow$ the set of leaves of $\mathcal{T}$
   // collect phase
6  foreach clique $i \in \mathbf{L}$ do
7      $p \leftarrow Adj_r(i)$
8      if $\delta_{i \to p}(r) = 1$ then
9          Compute $\psi_{i \to p}$
10     $p.nbVN \leftarrow p.nbVN + 1$
11     if $p \neq r$ and $p.nbVN = |Adj(p)| - 1$
         then $\mathbf{L} \leftarrow \mathbf{L} \cup \{p\}$
12     $\mathbf{L} \leftarrow \mathbf{L} \setminus \{i\}$
   // distribution phase
13 $\mathbf{L} \leftarrow \{r\}$
14 foreach clique $i \in \mathbf{L}$ do
15     foreach $j \in Adj(i) \setminus Adj_r(i)$ do
16         if $\delta_{i \to j}(r) = 1$ then
17             Compute $\psi_{i \to j}$
18             $\mathbf{L} \leftarrow \mathbf{L} \cup \{j\}$
19 foreach clique $t \in Q$ do
20     Compute the posterior distributions
       of the target nodes in clique $t$
21 return posterior distributions
```

Propagation-like architecture. Afterwards, incremental changes occur. Then *IJTI* is called to optimize the inference process. We recall that we use a target-driven approach, hence, we recompute only invalidated *collect* messages and we only *distribute* messages up to the *targets*. Under these assumptions, the proposed algorithm is described in Algo. 1. It runs a revised message-passing algorithm to compute $\psi_{i \to j}$ only when $\delta_{i \to j}(r) = 1$ for all i, j in the modified junction tree $\mathcal{T}$. In line 5, a leaf clique i is such that $|\mathrm{Adj}(i)| = 1$. We emphasize that computing messages is performed similarly to a classic JT-based inference algorithm. The correctness of IJTI is guaranteed by the following proposition:

Proposition 2. *The IJTI algorithm is sound, i.e., computing only messages* $\psi_{i \to j}$ *such that* $\delta_{i \to j}(r) = 1$*, for all* $(i, j) \in \mathcal{A}(\mathcal{T})$*, results in the correct computation of the posterior distributions of the target variables.*

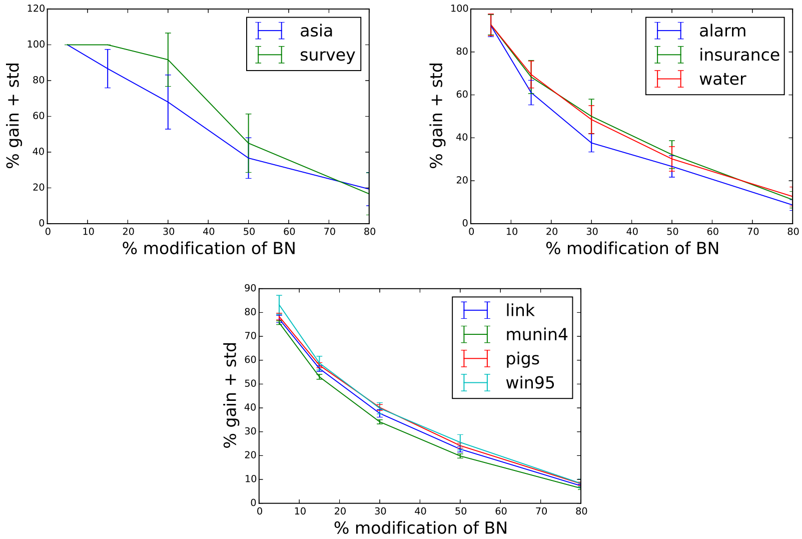
4 Experiments

In this section, we highlight the effectiveness of our algorithm by comparing the gain of using it instead of any non-incremental JT-based inference algorithm. This gain is equal to $1 - \delta(r)/(2|\mathcal{E}(\mathcal{T})|)$, i.e., this is the percentage of unnecessary messages that IJTI avoids to compute compared to the messages sent by classical inference algorithms on both directions on all the edges.

For this purpose, we performed tests using the aGrUM library[1] on 9 real-world BNs of different complexities as well as on randomly generated BNs. The

[1] `http://agrum.lip6.fr`

Fig. 3: *IJTI* gain for real BNs



latter contained $nbNodes$ Boolean random variables, ($6 \leq nbNodes \leq 900$, see Fig. 4) and, for each value of $nbNodes$, 3 BNs were generated with $nbArcs$ arcs, $nbArcs$ being chosen randomly in the interval $[nbNodes - 1, 4/3 * nbNodes - 1]$.

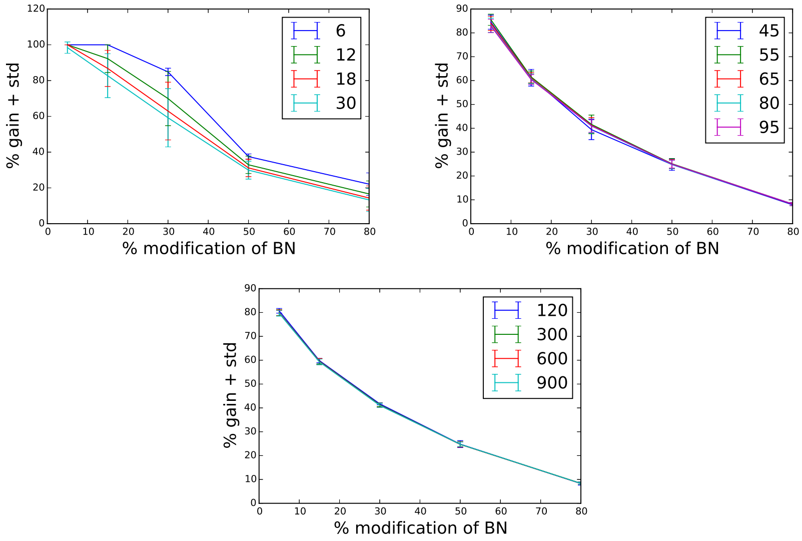
We simulated the incrementality by randomly choosing for each inference a set of targets and modified cliques. This induced invalid messages in $\mathcal{T}$. Fig. 3 and Fig. 4 show the average resulting gains and their standard deviations (error bars) over 20 incremental inference queries. Note that the behavior of the algorithm is the same for real-world BNs and for randomly generated ones. As could be expected, the smaller the modifications, the bigger the gain. Note also that the gain is not too sensitive to the size of the BN.

5 Discussion and future work

In this paper, we introduced *IJTI*, a new incremental junction-tree-based inference algorithm for multi-target dynamic systems. Assuming that a first complete inference has been performed, it extracts an optimal root and optimizes the inference accordingly. The correctness of these two optimizations is proved and experiments highlight that our approach allows for important savings compared to classical ones. For future works, we plan to improve our algorithm, notably by taking into account caching for the determination of the roots. We also plan to apply IJTI in Probabilistic Relational Models in order to speed-up their inference. Finally, we aim at coupling our approach with rule-based expert systems to improve their probabilistic reasoning.

Acknowledgments: This work was partially supported by IBM France Lab/ANRT CIFRE grant #2014/421.

Fig. 4: *IJTI* gain for artificial BNs



Appendix: Proofs

Proof of Proposition 1: Note that $\mathcal{V}_{\text{-}j}(i) = \{i\} \cup \bigcup_{k\in \mathrm{Adj}_{\text{-}j}(i)} \mathcal{V}_{\text{-}i}(k)$ and, for $k \in \mathrm{Adj}_{\text{-}j}(i)$, $l' \in \mathcal{V}_{\text{-}i}(k)$, we have $Adj_j(l') = Adj_i(l')$. Using Definition 4, one can thus rewrite $\mu_{i\to j}$ into:

$$\mu_{i\to j} = \bigcup_{\substack{l'\in\mathcal{V}_{\text{-}j}(i)\\ \{l\}=Adj_j(l')}} \lambda_{l'\to l} = \lambda_{i\to j} \cup \bigcup_{k\in \mathrm{Adj}_{\text{-}j}(i)} \overbrace{\bigcup_{\substack{l'\in\mathcal{V}_{\text{-}i}(k)\\ \{l\}=Adj_j(l')}} \lambda_{l'\to l}}^{\mu_{k\to i}} = \lambda_{i\to j} \cup \bigcup_{k\in \mathrm{Adj}_{\text{-}j}(i)} \mu_{k\to i}$$

■

Proof of Theorem 1 – mutual exclusivity: if property a) is satisfied, then $\mathcal{T}$ contains no edge, therefore properties b) and c) cannot be satisfied.

Now, assume that there exist r_1, r_1' such that $\mu_{r_1'\to r_1} = \mu_{r_1\to r_1'} = T\epsilon$ (property b). Let r_2 be any clique in $\mathcal{V}(\mathcal{T})$. Without loss of generality, assume that r_1 lies on the path $i_1 = r_2, i_2, \ldots, i_p = r_1'$ between r_2 and r_1'. Then, by Proposition 1, $\mu_{i_2\to r_2} \supseteq \mu_{i_3\to i_2} \supseteq \cdots \supseteq \mu_{r_1'\to r_1} = T\epsilon$. Therefore, properties b) and c) cannot hold simultaneously. ■

Proof of Theorem 1 – r's existence: if $\mathcal{A}(\mathcal{T}) = \emptyset$, then property a) holds and r is the unique node of $\mathcal{T}$. Now, assume that $\mathcal{A}(\mathcal{T}) \neq \emptyset$. If there exists an edge $(i,j) \in \mathcal{E}(\mathcal{T})$ such that $\mu_{i\to j} = \mu_{j\to i} = T\epsilon$, then $r = i$ satifies property b). Otherwise, neither properties a) nor b) hold. Assume that property c) neither holds. Then, for all edges (i,j), exactly one of $\mu_{i\to j}$ or $\mu_{j\to i}$ is equal to $T\epsilon$ and the other one belongs to $\{\emptyset, \epsilon, T\}$. Let (i_0, j_0) be such that $\mu_{i_0\to j_0} = T\epsilon$ and $\mu_{j_0\to i_0} \neq T\epsilon$. Then, if $|\mathrm{Adj}(i_0)| = 1$, clique i_0 satisfies property c), a contradiction. As we assume that property b) neither holds, there exists $i_1 \in \mathrm{Adj}(i_0)$ such that $\mu_{i_1 \to i_0} = T\epsilon$ and $\mu_{i_0 \to i_1} \neq T\epsilon$. The same reasoning holds for i_1, hence either i_1 is a leaf, which contradicts property c) or i_1 has another neighbor i_2 such that $\mu_{i_2 \to i_1} = T\epsilon$ and $\mu_{i_1 \to i_2} \neq T\epsilon$. By induction, we create a path $i_1, \ldots, i_n$ of maximal size. This path is necessarily finite since $\mathcal{T}$ is a finite tree, hence clique i_n is a leaf which, therefore, satisfies property c), a contradiction. Consequently, when properties a) and b) do not hold, property c) holds. ■

One can now prove separately the optimality for each property of Theorem 1, since these properties are mutually exclusive:

Proof of Theorem 1 – property a's optimality: r is the only node in $\mathcal{T}$. Choosing it as a root is therefore optimal. ■

Lemma 1. *Let $i, j \in \mathcal{V}(\mathcal{T})$ be such that $\epsilon \in \mu_{j \to i}$ and $\mu_{i \to j} = \emptyset$, then $\forall l \in \mathcal{V}_{\text{-}j}(i) : \delta(l) = \delta(j) + len(l - j)$.*

Proof. Note that when $\epsilon \notin \mu_{j \to i}$, $\mathcal{T}$ is up-to-date in the current inference and there is no need to perform any computation. The proof is achieved by induction on $n = len(l - j)$. For $n = 1$, we have $l = i$, so by Equation (2) and the fact that $\epsilon \in \mu_{j \to i}$ and $i \in Adj_i(j)$, we get $\delta_{j \to i}(i) = 1$. As a consequence, $\delta(i) = \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(j,i)\}} \delta_{k' \to k}(i) + 1$. Yet, as $T \notin \mu_{i \to j}$ we have $\delta_{j \to i}(j) = 0$; so $\delta(j) = \sum_{(k',k) \in \mathcal{A}(\mathcal{T})) \setminus \{(j,i)\}} \delta_{k' \to k}(j)$. Since $\epsilon \notin \mu_{i \to j}$, $\delta_{i \to j}(i) = \delta_{i \to j}(j) = 0$. For $(k', k) \neq (i, j), (j, i)$, we have $Adj_i(k) = Adj_j(k)$ and $Adj_i(k') = Adj_j(k')$. In this case, it follows that $\delta_{k' \to k}(i) = \delta_{k' \to k}(j)$. We conclude that $\delta(i) = \delta(j) + 1$.

Now suppose this property is satisfied for $n - 1 > 1$, let us prove that it remains true for n. Let l be such that $len(l - j) = n - 1$. Let $\{p\} = Adj_i(l)$. Then $\delta(l) = 1 + \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(p,l)\}} \delta_{k' \to k}(l)$ because $\delta_{p \to l}(l) = 1$ (since $\epsilon \in \mu_{p \to l}$ and $\{l\} = Adj_l(p)$). Knowing that $T \notin \mu_{l \to p}$, we get $\delta_{p \to l}(p) = 0$, it follows that $\delta(p) = \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(p,l)\}} \delta_{k' \to k}(p)$. Now using the same reasoning as in the case $n = 1$ and by remarking $\delta_{l \to p}(p) = \delta_{l \to p}(l) = 0$ because $\epsilon \notin \mu_{l \to p}$, we conclude that $\delta(l) = 1 + \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(p,l)\}} \delta_{k' \to k}(l) = 1 + \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(p,l)\}} \delta_{k' \to k}(p) = 1 + \delta(p)$. By applying the induction hypothesis on l, where $len(l - j) = n - 1$, we obtain : $\delta(l) = 1 + \delta(p) = 1 + n - 1 + \delta(j) = \delta(j) + n$. ■

Lemma 2. *Let $\mathcal{V}_1 = \{r \in \mathcal{V}(\mathcal{T}) : \exists k \in \mathrm{Adj}(r), \mu_{r \to k} = \mu_{k \to r} = T\epsilon\}$, then for any r, r' in $\mathcal{V}_1$ we have $\delta(r) = \delta(r')$.*

Proof. Assume that $|\mathcal{V}_1| > 1$. By Proposition 1, the nodes in $\mathcal{V}_1$ form a connected subgraph. Let $r, r' \in \mathcal{V}_1$ be such that $(r, r') \in \mathcal{E}(\mathcal{T})$. Finally, let $(k', k) \in \mathcal{A}(\mathcal{T}) \setminus \{(r, r'), (r', r)\}$. If $k' \notin \{r, r'\}$, then either $k = r$, $k = r'$ or $k \notin \{r, r'\}$ and in all these cases we have: $Adj_r(k') = Adj_{r'}(k')$, hence $\delta_{k' \to k}(r) = \delta_{k' \to k}(r')$. Otherwise, let $k' = r'$ then $k \neq r$ and we have also [2] $Adj_r(k) = Adj_{r'}(k)$ and again $\delta_{k' \to k}(r) = \delta_{k' \to k}(r')$. As a consequence: $\sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(r,r'),(r',r)\}} \delta_{k' \to k}(r) = \sum_{(k',k) \in \mathcal{A}(\mathcal{T}) \setminus \{(r,r'),(r',r)\}} \delta_{k' \to k}(r')$. By Equation (2), we get:

[2] if $k' = r$ then $k \neq r'$ and the equality also verified.

$\delta_{r\to r'}(r) + \delta_{r'\to r}(r) = \delta_{r\to r'}(r') + \delta_{r'\to r}(r') = 2$. We conclude that $\delta(r) - \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(r,r'),(r',r)\}}\delta_{k'\to k}(r) = \delta(r') - \sum_{(k',k)\in\mathcal{A}(\mathcal{T})\setminus\{(r,r'),(r',r)\}}\delta_{k'\to k}(r')$. Hence $\delta(r) = \delta(r')$. ■

Proof of Theorem 1 – property b's optimality: Under the notations of property b), it is sufficient to prove that for any i not in $\mathcal{V}_1, \delta(r) \leq \delta(i)$ [3]. Without loss of generality, assume that $i \in \mathcal{V}_{\text{-}r'}(r)$. Let $(k,k') \in \mathcal{A}(i-r)$, where $\mathcal{A}(i-r)$ is the set of arcs induced from $i-r$. We either have $\{k'\} = Adj_r(k)$ or $\{k\} = Adj_r(k')$. Assume for instance that $\{k'\} = Adj_r(k)\,, k \neq r$, the second case should be treated similarly. Then $\mu_{k'\to k} = T\epsilon$ and by applying Equation 2, we summarize the results on the following table:

$\mu_{k\to k'}$	$\delta_{k\to k'}(i) + \delta_{k'\to k}(i)$	$\delta_{k\to k'}(r) + \delta_{k'\to k}(r)$
$\emptyset$	1	0
T	1	1
ϵ	2	1

we conclude that $\sum_{(k',k)\in\mathcal{A}(i-r)} \delta_{k'\to k}(r) \leq \sum_{(k',k)\in\mathcal{A}(i-r)} \delta_{k'\to k}(i)$.(1)
Now for $(k,k') \notin \mathcal{A}(i-r)$ it is easy to see that $\delta_{k\to k'}(i) = \delta_{k\to k'}(r)$ and hence : $\sum_{(k,k')\in\mathcal{A}(\mathcal{T})\setminus\mathcal{A}(i-r)} \delta_{k'\to k}(r) = \sum_{(k,k')\in\mathcal{A}(\mathcal{T})\setminus\mathcal{A}(i-r)} \delta_{k'\to k}(i)$. (2).
By comparing (1) and (2) we get that $\delta(r) \leq \delta(i)$ for $i \notin \mathcal{V}_1$. So far, we obtain, by Lemma 2, for any i in $\mathcal{V}_1$, $\delta(r) = \delta(i)$ and for any i not in $\mathcal{V}_1, \delta(r) \leq \delta(i)$, therefore we have $r \in Argmin_{i\in\mathcal{V}(\mathcal{T})}\ \delta(i)$. ■

Proof of Theorem 1 – property c's optimality: Let i in $\mathcal{V}(\mathcal{T})$ s.t. $i \neq r$.
first case: $\mu_{Adj_i(r)\to r} = \emptyset$. Assume that $T, \epsilon \in \mathcal{V}_{\text{-}i}(r)$, because otherwise there is no need to perform any computation, as either there is no query or no modification in $\mathcal{T}$; so by Lemma 1 we have $\delta(i) = \delta(r) + len(i-r)$ because $i \in \mathcal{V}_{\text{-}r}(Adj_i(r))$. Hence $\delta(r) < \delta(i)$.
second case: we omit the case $\mu_{Adj_i(r)\to r} \in \{T, \epsilon\}$, but one should use the same methodology as in property b)'s proof and the fact that for any k, k' in $i-r$ s.t $\{k'\} = Adj_r(k) : \mu_{k\to k'} = \mu_{i\to Adj_r(i)}$ and examine $\delta_{k'\to k}(r)$ and $\delta_{k'\to k}(i)$. ■

Proof of Proposition 2 : Given a root r, $\delta_{i\to j}(r)$ corresponds, by construction, to the fact that $\psi_{i\to j}$ is necessary during the current inference and was invalidated in the previous one. As a consequence, the current inference needs to recompute only such a message for any i, j in $\mathcal{V}(\mathcal{T})$. ■

References

1. Buchanan, B.G., Shortliffe, E.H.: Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project (The Addison-Wesley Series in Artificial Intelligence). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1984)

[3] all the nodes are computationally equivalent if $\forall i \in \mathcal{V}(\mathcal{T}), i \in \mathcal{V}_1$ since $\mathcal{V}(\mathcal{T}) = \mathcal{V}_1$

2. Cooper, G.F.: The computational complexity of probabilistic inference using Bayesian belief networks (research note). Artif. Intell. 42(2-3), 393–405 (Mar 1990)
3. Dagum, P., Luby, M.: Approximating probabilistic inference in Bayesian belief networks is np-hard. Artif. Intell. 60(1), 141–153 (Mar 1993)
4. D'Ambrosio, B.: Incremental probabilistic inference. In: Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence. pp. 301–308 (1993)
5. Darwiche, A.: Dynamic join trees. In: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. pp. 97–104. UAI'98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998)
6. Dean, T., Kanazawa, K.: A model for reasoning about persistence and causation. Computational Intelligence 5(2), 142–150 (1989)
7. Flores, M.J., Gámez, J.A., Olesen, K.G.: Incremental compilation of Bayesian networks. In: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence. pp. 233–240. UAI'03, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2003)
8. Heckerman, D.E., Shortliffe, E.H.: From certainty factors to belief networks. Artif. Intell. Med. 4(1), 35–52 (Feb 1992)
9. Jensen, F., Lauritzen, S., Olesen, K.: Bayesian updating in causal probabilistic networks by local computations. Computational Statistics Quarterly 4, 269–282 (1990)
10. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
11. Koller, D., Pfeffer, A.: Probabilistic frame-based systems. In: Proceedings of the 15th National Conference on Artificial Intelligence (AAAI). pp. 580–587 (1998)
12. Lauritzen, S., Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their applications to expert systems. Journal of the Royal Statistical Society 50(2), 157–224 (1988)
13. Li, W., van Beek, P., Poupart, P.: Performing incremental Bayesian inference by dynamic model counting. In: AAAI. pp. 1173–1179. AAAI Press (2006)
14. Lin, Y., Druzdzel, M.J.: Relevance-based sequential evidence processing in Bayesian networks. In: Proceedings of the Eleventh International Florida Artificial Intelligence Research Society Conference, May 18-20, 1998, Sanibel Island, Florida, USA. pp. 446–450 (1998)
15. Madsen, A.L., Jensen, F.V.: Lazy propagation: A junction tree inference algorithm based on lazy evaluation. Artificial Intelligence 113(12), 203 – 245 (1999)
16. Murphy, K.P.: Dynamic Bayesian networks: Representation, inference and learning (2002)
17. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)
18. Pfeffer, A.J.: Probabilistic Reasoning for Complex Systems. Ph.D. thesis, Stanford, CA, USA (2000)
19. Robinson, J., Hartemink, A.: Non-stationary dynamic Bayesian networks, pp. 1369–1376 (2009)
20. Shenoy, P., Shafer, G.: Axioms for probability and belief-function propagation. In: Uncertainty in Artificial Intelligence. vol. 4, pp. 169–198 (1990)
21. Torti, L., Gonzales, C., Wuillemin, P.H.: Speeding-up structured probabilistic inference using pattern mining. International Journal of Approximate Reasoning 54(7), 900–918 (2013)