International Journal of Approximate Reasoning 68 (2016) 127–152

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Bayesian network inference using marginal trees

Cory J. Butz [a,*], Jhonatan S. Oliveira [a], Anders L. Madsen [b,c]

[a] University of Regina, Department of Computer Science, Regina, S4S 0A2, Canada
[b] Aalborg University, Department of Computer Science, Aalborg, DK-9000, Denmark
[c] HUGIN EXPERT A/S, Aalborg, DK-9000, Denmark

ARTICLE INFO

Article history:
Received 31 December 2014
Received in revised form 22 July 2015
Accepted 22 July 2015
Available online 29 July 2015

Keywords:
Bayesian networks
Exact inference
Variable elimination
Join tree propagation

ABSTRACT

Variable elimination (VE) and *join tree propagation* (JTP) are two alternatives to inference in *Bayesian networks* (BNs). VE, which can be viewed as one-way propagation in a join tree, answers each query against the BN meaning that computation can be repeated. On the other hand, answering a single query with JTP involves two-way propagation, of which some computation may remain unused. In this paper, we propose *marginal tree inference* (MTI) as a new approach to exact inference in discrete BNs. MTI seeks to avoid recomputation, while at the same time ensuring that no constructed probability information remains unused. Thereby, MTI stakes out middle ground between VE and JTP. The usefulness of MTI is demonstrated in multiple probabilistic reasoning sessions.

1. Introduction

Bayesian networks (BNs) [1–4], a marriage of probability theory and graph theory, provide a rigorous foundation for uncertainty management and have been successfully applied in practice to a wide variety of problem domains. A BN consists of a *directed acyclic graph* (DAG) [1] and a set of *conditional probability tables* (CPTs) [5] corresponding to the structure of the DAG. The vertices in the DAG represent random variables in a real-world problem, while the arcs in the DAG represent probabilistic dependencies amongst the variables. More specifically, the *probabilistic conditional independencies* [6] encoded in the DAG ensure that the product of the CPTs is a joint probability distribution. Thereby, BNs continue to provide a robust framework for designing probabilistic expert systems [4]. Although Cooper [7] has shown that the complexity of exact inference in discrete BNs is NP-hard, various approaches have been developed that seem to work quite well in practice. Most of these methods centre around eliminating variables from probabilistic networks to produce posterior probability distributions and can be broadly classified into two categories.

The first category of BN inference is *join tree propagation* [8–14], which Shafer [5] states is central to the theory and practice of probabilistic expert systems. Join tree propagation first builds a secondary network, called a join tree, from the DAG of the BN and then performs inference by propagating probabilities in the join tree. In particular, following an inward-pass and an outward pass, posteriors for all non-evidence variables can be determined. The second category is *direct computation*, of which *variable elimination* (VE) [15] is the most popular. In fact, Koller and Friedman [2] introduce readers to inference in BNs using the VE algorithm. VE removes a variable by multiplying together all of the distributions involving the variable and then summing the variable out of the obtained product. VE is more specialized than join tree propagation in that it only computes the posterior probabilities for a given subset of non-evidence variables rather than all non-evidence variables.

* Corresponding author.
E-mail addresses: butz@cs.uregina.ca (C.J. Butz), oliveira@cs.uregina.ca (J.S. Oliveira), anders@hugin.com (A.L. Madsen).

http://dx.doi.org/10.1016/j.ijar.2015.07.006
0888-613X/

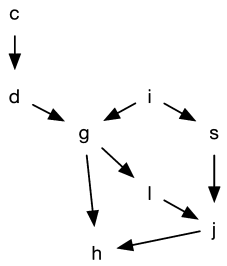

Fig. 1. The DAG of the ESBN [2].

Algorithm 1 Variable elimination.

```
1: function VARIABLE ELIMINATION(Φ, X, E, e, σ)
2:     Delete rows disagreeing with E = e from φ ∈ Φ
3:     while σ is not empty do
4:         Remove the first variable v from σ
5:         Φ = SUM-OUT(v, Φ)
6:     end while
7:     p(X, E = e) = ∏_{φ∈Φ} φ
8:     return p(X, E = e)/∑_X p(X, E = e)
9: end function
```

The question addressed in this investigation is how to determine a good way to answer a sequence of queries. If VE were applied, then computation may be repeated, since VE answers each subsequent query against the original BN. If join tree propagation is utilized, then some computation may be wasteful, since two-way propagation may build probability tables that are not required to answer the current query. Thus, there is room to stake-out middle ground between VE and JTP.

In this paper, we introduce *marginal tree inference* (MTI) as a new exact inference algorithm in discrete BNs. MTI answers the first query the same way as VE does. MTI answers each subsequent query in a two-step procedure that can readily be performed in a new secondary structure, called a *marginal tree*. First, determine whether any computation can be reused. Second, only compute what is missing to answer the query. One salient feature of MTI is that it does not involve pre-computation, meaning that every probability table built is necessarily used in answering a query. The usefulness of MTI is demonstrated in multiple probabilistic reasoning sessions.

The remainder of this paper is organized as follows. Section 2 contains definitions. Marginal trees are introduced in Section 3. Section 4 presents MTI. A diagnostic query session is given in Section 5. Empirical analysis is presented in Section 6. Conclusions are given in Section 7.

2. Definitions

2.1. Bayesian networks

Let U be a finite set of variables. Each variable $v_i \in U$ has a finite domain, denoted $dom(v_i)$. A *Bayesian network* (BN) [1] on U is a pair $(\mathcal{B}, C)$. $\mathcal{B}$ is a *directed acyclic graph* (DAG) with vertex set U and C is a set of *conditional probability tables* (CPTs) $\{p(v_i|P(v_i)) \mid v_i \in U\}$, where $P(v_i)$ denotes the parents (immediate predecessors) of $v_i \in \mathcal{B}$. For example, Fig. 1 depicts the *extended student Bayesian network* (ESBN) [2], where the CPTs are not illustrated, and all variables are binary. The product of the CPTs in C is a joint probability distribution $p(U)$. For $X \subseteq U$, the *marginal distribution* $p(X)$ is $\sum_{U-X} p(U)$. Each element $x \in dom(X)$ is called a *row* (configuration) of X. We call $\mathcal{B}$ a BN, if no confusion arises, and $X \cup Y$ may be written as XY. A *potential* on V is a function ϕ such that $\phi(v) \geq 0$ for each $v \in V$, and at least one $\phi(v) > 0$. A *leaf* is a variable in $\mathcal{B}$ without children (descendants).

2.2. Variable elimination

Variable elimination (VE) [15] computes $p(X|E = e)$, where X and E are disjoint subsets of U, and E is observed taking value e. In VE (given as Algorithm 1), Φ is the set of CPTs for $\mathcal{B}$, X is a list of query variables, E is a list of observed variables, e is the corresponding list of observed values, and σ is an *elimination ordering* [16] for variables $U - (XE)$. All elimination orderings in this paper are determined using *weighted-min-fill* (WMF), which tends to be one of the best heuristics in practice [2]. Evidence may not be denoted for simplified notation.

VE calls the sum-out algorithm, which eliminates v from a set Φ of potentials by multiplying together all potentials involving v and then summing v out of the product.